INTERNATIONAL STANDARD

ISO/IEC 11404

First edition
1996-12-15

Information technology — Programming languages, their environments and system software interfaces — Language-independent datatypes

Technologies de l'information — Langages de programmation, leur environnement et interfaces du logiciel système — Types de données indépendants du langage

Reference number
ISO/IEC 11404:1996(E)

Contents

Foreword v
Introduction vi

1 Scope 1

2 Conformance 1
2.1 Direct conformance 2
2.2 Indirect conformance 2
2.3 Conformance of a mapping standard 2

3 Normative References 3

4 Definitions 3

5 Conventions Used in this International Standard 5
5.1 Formal syntax 5
5.2 Text conventions 6

6 Fundamental Notions 6
6.1 Datatype 6
6.2 Value space 7
6.3 Datatype properties 7
6.3.1 Equality 7
6.3.2 Order 7
6.3.3 Bound 8
6.3.4 Cardinality 8
6.3.5 Exact and approximate 8
6.3.6 Numeric 9
6.4 Primitive and non-primitive datatypes 9
6.5 Datatype generator 9
6.6 Characterizing operations 9
6.7 Datatype families 10
6.8 Aggregate datatypes 10
6.8.1 Homogeneity 11

6.8.2 Size ... 11
6.8.3 Uniqueness ... 11
6.8.4 (Aggregate-imposed) ordering ... 11
6.8.5 Access method ... 11
6.8.6 Recursive structure ... 12

7 Elements of the Datatype Specification Language ... 12
7.1 IDN character-set ... 12
7.2 Whitespace ... 13
7.3 Lexical objects ... 13
7.3.1 Identifiers ... 13
7.3.2 Digit-string ... 14
7.3.3 Character-literal and string-literal ... 14
7.3.4 Keywords ... 14
7.4 Annotations ... 14
7.5 Values ... 15
7.5.1 Independent values ... 15
7.5.2 Dependent values ... 16

8 Datatypes ... 17
8.1 Primitive datatypes ... 17
8.1.1 Boolean ... 18
8.1.2 State ... 19
8.1.3 Enumerated ... 19
8.1.4 Character ... 20
8.1.5 Ordinal ... 21
8.1.6 Date-and-Time ... 21
8.1.7 Integer ... 22
8.1.8 Rational ... 23
8.1.9 Scaled ... 23
8.1.10 Real ... 24
8.1.11 Complex ... 26
8.1.12 Void ... 27
8.2 Subtypes and extended types ... 27
8.2.1 Range ... 28
8.2.2 Selecting ... 28
8.2.3 Excluding ... 28
8.2.4 Size ... 29
8.2.5 Explicit subtypes ... 29
8.2.6 Extended ... 30
8.3 Generated datatypes ... 30
8.3.1 Choice ... 31
8.3.2 Pointer ... 33
8.3.3 Procedure ... 34
8.4 Aggregate Datatypes ... 36
8.4.1 Record ... 37
8.4.2 Set ... 38
8.4.3 Bag ... 39
8.4.4 Sequence ... 40
8.4.5 Array ... 41

8.4.6 Table ... 43
8.5 Defined Datatypes ... 44
9 Declarations ... 45
9.1 Type Declarations ... 45
9.1.1 Renaming declarations ... 46
9.1.2 New datatype declarations ... 46
9.1.3 New generator declarations ... 46
9.2 Value Declarations ... 46
9.3 Termination Declarations ... 47
10 Defined Datatypes and Generators ... 47
10.1 Defined datatypes ... 47
10.1.1 Natural number ... 47
10.1.2 Modulo ... 48
10.1.3 Bit ... 48
10.1.4 Bit string ... 48
10.1.5 Character string ... 49
10.1.6 Time interval ... 50
10.1.7 Octet ... 50
10.1.8 Octet string ... 50
10.1.9 Private ... 50
10.1.10 Object identifier ... 51
10.2 Defined generators ... 52
10.2.1 Stack ... 53
10.2.2 Tree ... 53
10.2.3 Cyclic enumerated ... 53
10.2.4 Optional ... 54
11 Mappings ... 54
11.1 Outward Mappings ... 55
11.2 Inward Mappings ... 56
11.3 Reverse Inward Mapping ... 56
11.4 Support of Datatypes ... 57
11.4.1 Support of equality ... 57
11.4.2 Support of order ... 57
11.4.3 Support of bounds ... 57
11.4.4 Support of cardinality ... 57
11.4.5 Support for the exact or approximate property ... 58
11.4.6 Support for the numeric property ... 58
Annex A. Character-Set Standards ... 59
Annex B. Recommended Placement of Annotations ... 62
Annex C. Implementation Notions of Datatypes ... 64
Annex D. Syntax for the Common Interface Definition Notation ... 67
Annex E. Example Mapping to Pascal ... 72
Annex F. Example Mapping to MUMPS ... 85
Annex G. Resolved Issues ... 89

Foreword

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

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 11404 was prepared by Joint Technical Committee ISO/IEC JTC1, Information technology, Subcommittee SC22, *Programming languages, their environments and system software interfaces.*

Annexes A to G of this International Standard are for information only.

Introduction

Many specifications of software services and applications libraries are, or are in the process of becoming, International Standards. The interfaces to these libraries are often described by defining the form of reference, e.g. the "procedure call", to each of the separate functions or services in the library, as it must appear in a user program written in some standard programming language (Fortran, COBOL, Pascal, etc.). Such an interface specification is commonly referred to as the "<language> binding of <service>", e.g. the "Fortran binding of PHIGS" (ISO/IEC 9593-1:1990, *Information processing systems — Computer Graphics — Programmer's Hierarchical Interactive Graphics System (PHIGS) language bindings — Part 1: FORTRAN*).

This approach leads directly to a situation in which the standardization of a new service library immediately requires the standardization of the interface bindings to every standard programming language whose users might reasonably be expected to use the service, and the standardization of a new programming language immediately requires the standardization of the interface binding to every standard service package which users of that language might reasonably be expected to use. To avoid this n-to-m binding problem, ISO/IEC JTC1 (Information Technology) assigned to SC22 the task of developing an International Standard for Language-Independent Procedure Calling and a parallel International Standard for Language-Independent Datatypes, which could be used to describe the parameters to such procedures.

This International Standard provides the specification for the Language-Independent Datatypes. It defines a set of datatypes, independent of any particular programming language specification or implementation, that is rich enough so that any common datatype in a standard programming language or service package can be mapped to some datatype in the set.

The purpose of this International Standard is to facilitate commonality and interchange of datatype notions, at the conceptual level, among different languages and language-related entities. Each datatype specified in this International Standard has a certain basic set of properties sufficient to set it apart from the others and to facilitate identification of the corresponding (or nearest corresponding) datatype to be found in other standards. Hence, this International Standard provides a single common reference model for all standards which use the concept *datatype*. It is expected that each programming language standard will define a mapping from the datatypes supported by that programming language into the datatypes specified herein, semantically identifying its datatypes with datatypes of the reference model, and thereby with corresponding datatypes in other programming languages.

It is further expected that each programming language standard will define a mapping from those Language-Independent (LI) Datatypes which that language can reasonably support into datatypes which may be specified in the programming language. At the same time, this International Standard will be used, among other applications, to define a "language-independent binding" of the parameters to the procedure calls constituting the principal elements of the standard interface to each of the standard services. The production of such service bindings and language mappings leads, in cooperation with the parallel Language-Independent Procedure Calling mechanism, to a situation in which no further "<language> binding of <service>" documents need to be produced: Each service interface, by defining its parameters using LI datatypes, effectively defines the binding of such parameters to any standard programming language; and each language, by its mapping from the LI datatypes into the language datatypes, effectively defines the binding to that language of parameters to any of the standard services.

Information technology — Programming languages, their environments and system software interfaces — Language-independent datatypes

1 Scope

This International Standard specifies the nomenclature and shared semantics for a collection of datatypes commonly occurring in programming languages and software interfaces, referred to as the Language-Independent (LI) Datatypes. It specifies both primitive datatypes, in the sense of being defined ab initio without reference to other datatypes, and non-primitive datatypes, in the sense of being wholly or partly defined in terms of other datatypes. The specification of datatypes in this International Standard is "language-independent" in the sense that the datatypes specified are classes of datatypes of which the actual datatypes used in programming languages and other entities requiring the concept *datatype* are particular instances.

This International Standard expressly distinguishes three notions of "datatype", namely:

- the conceptual, or abstract, notion of a datatype, which characterizes the datatype by its nominal values and properties;
- the structural notion of a datatype, which characterizes the datatype as a conceptual organization of specific component datatypes with specific functionalities; and
- the implementation notion of a datatype, which characterizes the datatype by defining the rules for representation of the datatype in a given environment.

This International Standard defines the abstract notions of many commonly used primitive and non-primitive datatypes which possess the structural notion of atomicity. This International Standard does not define all atomic datatypes; it defines only those which are common in programming languages and software interfaces. This International Standard defines structural notions for the specification of other non-primitive datatypes and provides a means by which datatypes not defined herein can be defined structurally in terms of the LI datatypes defined herein.

This International Standard defines a partial vocabulary for implementation notions of datatypes and provides for, but does not require, the use of this vocabulary in the definition of datatypes. The primary purpose of this vocabulary is to identify common implementation notions associated with datatypes and to distinguish them from conceptual notions. Specifications for the use of implementation notions are deemed to be outside the scope of this International Standard, which is concerned solely with the identification and distinction of datatypes.

This International Standard specifies the required elements of mappings between the LI datatypes and the datatypes of some other language. This International Standard does not specify the precise form of a mapping, but rather the required information content of a mapping.

2 Conformance

An information processing product, system, element or other entity may conform to this International Standard either directly, by utilizing datatypes specified in this International Standard in a conforming manner (2.1), or indirectly, by means of mappings between internal datatypes used by the entity and the datatypes specified in this International Standard (2.2).

NOTE — The general term **information processing entity** is used in this clause to include anything which processes information and contains the concept of *datatype*. Information processing entities for which conformance to this International Standard may be appropriate include other standards (e.g. standards for programming languages or language-related facilities), specifications, data handling facilities and services, etc.

2.1 Direct conformance

An information processing entity which **conforms directly** to this International Standard shall:

i) specify which of the datatypes and datatype generators specified in Clauses 8 and 10 are provided by the entity and which are not, and which, if any, of the declaration mechanisms in Clause 9 it provides; and

ii) define the value spaces of the LI datatypes used by the entity to be identical to the value-spaces specified by this International Standard; and

iii) use the notation prescribed by clauses 7 through 10 of this International Standard to refer to those datatypes and to no others; and

iv) to the extent that the entity provides operations other than movement or translation of values, define operations on the LI datatypes which can be derived from, or are otherwise consistent with, the characterizing operations specified by this International Standard.

NOTES

1. This International Standard defines a syntax for the denotation of values of each datatype it defines, but, in general, requirement *(iii)* does not require conformance to that syntax. Conformance to the value-syntax for a datatype is required only in those cases in which the value appears in a *type-specifier*, that is, only where the value is part of the identification of a datatype.

2. The requirements above prohibit the use of a *type-specifier* defined in this International Standard to designate any other datatype. They make no other limitation on the definition of additional datatypes in a conforming entity, although it is recommended that either the form in Clause 8 or the form in Clause 10 be used.

3. Requirement (*iv*) does not require all characterizing operations to be supported and permits additional operations to be provided. The intention is to permit addition of semantic interpretation to the LI datatypes and generators, as long as it does not conflict with the interpretations given in this International Standard. A conflict arises only when a given characterizing operation could not be implemented or would not be meaningful, given the entity-provided operations on the datatype.

4. Examples of entities which could conform directly are language definitions or interface specifications whose datatypes, and the notation for them, are those defined herein. In addition, the verbatim support by a software tool or application package of the datatype syntax and definition facilities herein should not be precluded.

2.2 Indirect conformance

An information processing entity which **conforms indirectly** to this International Standard shall:

i) provide mappings between its internal datatypes and the LI datatypes conforming to the specifications of Clause 11 of this International Standard; and

ii) specify for which of the datatypes in Clause 8 and Clause 10 an inward mapping is provided, for which an outward mapping is provided, and for which no mapping is provided.

NOTES

1. Standards for existing programming languages are expected to provide for indirect conformance rather than direct conformance.

2. Examples of entities which could conform indirectly are language definitions and implementations, information exchange specifications and tools, software engineering tools and interface specifications, and many other entities which have a concept of datatype and an existing notation for it.

2.3 Conformance of a mapping standard

In order to conform to this International Standard, a standard for a mapping shall include in its conformance requirements the requirement to conform to this International Standard.

NOTES

1. It is envisaged that this International Standard will be accompanied by other standards specifying mappings between the internal datatypes specified in language and language-related standards and the LI datatypes. Such mapping standards are required to comply with this Interna-

tional Standard.

2. Such mapping standards may define "generic" mappings, in the sense that for a given internal datatype the standard specifies a parametrized LI datatype in which the parametric values are not derived from parametric values of the internal datatype nor specified by the standard itself, but rather are required to be specified by a "user" or "implementor" of the mapping standard. That is, instead of specifying a particular LI datatype, the mapping specifies a family of LI datatypes and requires a further user or implementor to specify which member of the family applies to a particular use of the mapping standard. This is always necessary when the internal datatypes themselves are, in the intention of the language standard, either explicitly or implicitly parametrized. For example, a programming language standard may define a datatype INTEGER with the provision that a conforming processor will implement some range of Integer; hence the mapping standard may map the internal datatype INTEGER to the LI datatype :

integer range (min..max),

and require a conforming processor to provide values for "min" and "max".

3 Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of current valid International Standards.

ISO/IEC 8601:1988, *Data elements and interchange formats — Information interchange —Representation of dates and times.*

ISO/IEC 8824:1990, *Information technology — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1).*

ISO/IEC 10646-1:1993, *Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane.*

4 Definitions

For the purposes of this International Standard, the following definitions apply.

NOTE — These definitions may not coincide with accepted mathematical or programming language definitions of the same terms.

4.1 actual parametric datatype: a datatype appearing as a parametric datatype in a use of a datatype generator, as opposed to the *formal-parametric-types* appearing in the definition of the datatype generator.

4.2 actual parametric value: a value appearing as a parametric value in a reference to a datatype family or datatype generator, as opposed to the *formal-parametric-values* appearing in the corresponding definitions.

4.3 aggregate datatype: a 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.

4.4 annotation: a descriptive information unit attached to a datatype, or a component of a datatype, or a procedure (value), to characterize some aspect of the representations, variables, or operations associated with values of the datatype which goes beyond the scope of this International Standard.

4.5 approximate: a property of a datatype indicating that there is not a 1-to-1 relationship between values of the conceptual datatype and the values of a valid computational model of the datatype.

4.6 bounded: a property of a datatype, meaning both *bounded above* and *bounded below*.

4.7 bounded above: a property of a datatype indicating that there is a value U in the value space such that, for all values s in the value space, $s \leq U$.

4.8 bounded below: a property of a datatype indicating that there is a value L in the value space such that, for all values *s* in the value space, $L \leq s$.

4.9 characterizing operations:
(of a datatype): a collection of operations on, or yielding, values of the datatype, which distinguish this datatype from