INTERNATIONAL STANDARD

ISO/IEC 11404

Second edition
2007-12-15

Information technology — General-Purpose Datatypes (GPD)

Technologies de l'information — Types de données

Reference number
ISO/IEC 11404:2007(E)

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

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

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

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

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 11404 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

This second edition cancels and replaces the first edition (ISO/IEC 11404:1996), which has been technically revised.

0 Introduction

0.1 Introduction to the second edition

This second edition of ISO/IEC 11404 incorporates recent technologies and improvements since the first edition (ISO/IEC 11404:1996). The following improvements have been incorporated into the second edition.

— *Title change to reflect actual usage*. The use of ISO/IEC 11404 is no longer simply a tool for communicating among programming languages (old title: *Language-independent datatypes*). ISO/IEC 11404 is used for formal description of conceptual datatypes in binding (or binding-independent) standards and used as formalization of metadata for data elements, data element concepts, and value domains (see ISO/IEC 11179-3). The old title was potentially misleading because readers might believe that ISO/IEC 11404 is only useful for programming languages. The new title, *General-Purpose Datatypes* captures the essence of ISO/IEC 11404 and its use.

— *Incorporation of latest technologies*. Provide enhancements to the use of ISO/IEC 11404 as a datatype nomenclature reference for current programming languages, interface languages and data representation languages, specifically Java, IDL, Express, and XML.

— *Support for semi-structured and unstructured data aggregates*. Semi-structured data and unstructured data includes aggregates where datatyping and navigation may be unknown or unspecified in advance. For example, some systems permit “discovery” (or “introspection”) of data. In some cases, the datatype may be unknown in advance (e.g. at compilation time), but may be discovered and processed at runtime (e.g. via datatype libraries or metadata registries).

— *Support for data longevity, versioning, and migration*. There is a need to support, from a datatyping perspective, obsolete and reserved features, such as data elements and permissible values (enumerations and states). Marking features as “obsolete” allows processing, compilation, and runtime systems to “flag” or diagnose old (deprecated) features, while still maintaining compatibility, so that it is possible to support transitions from past to present. Similarly, marking features as “reserved” allows processing, compilation, and runtime systems to “flag” or diagnose potential incompatibilities with future systems, so that it is possible to support transitions from present to future.

— *Extensibility of datatypes and value spaces*. There is a need to support some kind of extensibility concept. For example: (1) a GPD specification of an aggregate contains the elements A and B. (2) An application creates an aggregate with the elements A, B, and C. (3) Are the application's “extensions” of the aggregate acceptable/in conformity with the GPD specification in (1)? The answer to (3) is dependent upon the intent and design of the specification in (1): in some cases extensions are permitted, in some cases extensions are not permitted. The extensibility concept would allow the user of GPD datatypes to describe the kind of extensions permitted. This feature is particularly important in (a) data conformance and (b) application runtime environments that permit “discovery” or “introspection”. This feature is available via the “provision()” capability.

Features that are not incorporated within GPD include the following:

— *Namespace capability*. Given the larger number of declarations, a namespace capability is necessary.

— *Data representation*. Although these features are a part of GPD annotations, there is no standardization of data representation in these annotations. This step is an important link for data interoperability.

0.2 Introduction to the first edition (ISO/IEC 11404:1996)

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 JTC 1, *Information technology* assigned to SC 22 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 — General-Purpose Datatypes (GPD)

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 General-Purpose Datatypes (GPD). 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 "general-purpose" in the sense that the datatypes specified are classes of datatype of which the actual datatypes used in programming languages and other entities requiring the concept "datatype" are particular instances. These datatypes are general in nature; thus, they serve a wide variety of information processing applications.

This International Standard expressly distinguishes three notions of datatype:

— 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 GPDs defined herein.

This International Standard defines a partial terminology for implementation notions of datatypes and provides for the use of this terminology in the definition of datatypes. The primary purpose of this terminology is to identify common implementation notions associated with datatypes and to distinguish them from conceptual notions.

This International Standard specifies the required elements of mappings between the GPDs 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 Normative references

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

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

ISO/IEC 8824 (all parts), *Information technology — Abstract Syntax Notation One (ASN.1)*

ISO/IEC 10646, *Information technology — Universal Multiple-Octet Coded Character Set (UCS)*

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

IETF RFC 2396, *Uniform Resource Identifiers (URI): Generic Syntax*

3 Terms and definitions

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

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

3.1
actual parametric datatype
datatype appearing as a parametric datatype in a use of a datatype generator, in contrast to the formal-parametric-types appearing in the definition of the datatype generator

3.2
actual parametric value
value appearing as a parametric value in a reference to a datatype family or datatype generator, in contrast to the formal-parametric-values appearing in the corresponding definitions

3.3
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

3.4
annotation
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

3.5
approximate
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

3.6
bounded
property of a datatype, meaning both bounded above and bounded below

3.7
bounded above
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, $s \leq U$

3.8
bounded below
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, $s \geq L$

