
**Information technology — Test methods
for measuring conformance to Open
Systems Interconnection (OSI) abstract
data manipulation C language interfaces —
Binding for Application Program Interface
(API)**

*Technologies de l'information — Méthodes d'essai pour mesurer la
conformité aux interfaces de langage C pour la manipulation de données
abstraites en interconnexion de systèmes ouverts (OSI) — Liant pour
l'interface de programme d'application (API)*



Contents

	PAGE
Section 1: General	1
1.1 Scope	1
1.2 Normative References	2
1.3 Conformance	2
Section 2: Terminology and General Requirements	3
2.1 Conventions	3
2.2 Definitions	5
Section 3: Test Assertions—General	13
3.1 Scope	13
3.2 Normative References	13
3.3 Conformance	13
3.4 Test Methods	13
Section 4: Test Assertions—Terminology and General Requirements	15
4.1 Conventions	15
4.2 Definitions	16
Section 5: Test Assertions—Service Interface	17
5.1 Introduction	17
5.2 Datatypes	17
5.2.1 Intermediate Datatypes	18
5.2.2 Boolean	18
5.2.3 Descriptor	18
5.2.4 Enumeration	21
5.2.5 Exclusions	21
5.2.6 Integer	22
5.2.7 Object	22
5.2.8 Object Identifier	22
5.2.9 Private Object	23

© ISO/IEC 1996

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

5.2.10	Public Object	23
5.2.11	Real	24
5.2.12	Return Code	24
5.2.13	String	24
5.2.14	String Length	25
5.2.15	Syntax	25
5.2.16	Type	25
5.2.17	Type List	26
5.2.18	Value	26
5.2.19	Value Length	27
5.2.20	Value Position	27
5.2.21	Workspace	27
5.3	Functions	27
5.3.1	Copy	28
5.3.2	Copy Value	30
5.3.3	Create	30
5.3.4	Decode	33
5.3.5	Delete	35
5.3.6	Encode	36
5.3.7	Get	38
5.3.8	Instance	41
5.3.9	Put	41
5.3.10	Read	41
5.3.11	Remove	42
5.3.12	Write	42
5.4	Return Codes	42
5.5	Header File	42
Section 6:	Test Assertions—Workspace Interface	47
6.1	Introduction	47
6.2	Representation of Objects	47
6.3	Types and Macros	48
6.3.1	Standard Internal Representation of an Object	48
6.3.2	Standard Internal Representation of a Workspace	48
6.3.3	Useful Macros	50
6.4	Dispatcher Macros	51
Section 7:	Test Assertions—Definitions of Constants	55
7.1	Exclusions	55
7.2	Modification	55
7.3	Object Identifiers	56
7.4	Return Code	56
7.5	String (Elements Component)	57
7.6	Syntax	58
7.7	Type	58
Annex A	(informative) Bibliography	61
Alphabetic Topical Index		63

TABLES

Table 2-1 – Naming Conventions 4

Table 4-1 – Naming Conventions 15

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. 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 14366 was prepared by IEEE (as IEEE Std 1328-1993) and was adopted, under a special “fast-track procedure”, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

Annex A of this International Standard is for information only.

1 Introduction

(This introduction is not a normative part of ISO/IEC 14366, Information technology—Test methods for measuring conformance to Open Systems Interconnection (OSI) abstract data manipulation C language interfaces—Binding for Application Program Interface (API), but is included for information only.)

The purpose of this International Standard is to specify the requirements for test methods for measuring conformance to ISO/IEC 14364 {4}, which defines a C programming language binding to the general-purpose OSI Abstract Data Manipulation (OM) Application Programming Interface (API).

10 Related Standards

ISO/IEC 14364 {4} is a C programming language binding for the programming language independent specification contained in ISO/IEC 14360 {2}. The requirements for test methods for measuring conformance to ISO/IEC 14360 {2} are contained in ISO/IEC 14362 {3}. The requirements specified in ISO/IEC 14362 {3} apply to test methods for measuring conformance to ISO/IEC 14364 {4}, in addition to the requirements specified in this International Standard.

ISO/IEC 14364 {4} is intended to be used in conjunction with C programming language bindings of application-specific APIs that provide OSI services, such as the X.400-based messaging API defined in ISO/IEC 14365 {B5} and the API to directory services defined in ISO/IEC 14394 {B6}. The requirements in this International Standard may also apply to test methods for measuring conformance to standards for such application-specific APIs.

The test assertions in this International Standard were produced in accordance with IEEE Std 1003.3-1991, IEEE Standard for Information Technology—Test Methods for Measuring Conformance to POSIX {5}.

26 Overview

This International Standard contains a section corresponding to each section of ISO/IEC 14364 {4}. Each such section contains test assertions for that section of ISO/IEC 14364 {4}. A set of test methods for measuring conformance to ISO/IEC 14364 {4} is required to test these assertions in conformance with the requirements of IEEE Std 1003.3-1991 {5}.

Related Standards Activities

The following areas are under active consideration at this time, or are expected to become active in the near future, concerning standards for application APIs that use the mechanism defined in this International Standard. Similar efforts can be anticipated in the future:¹⁾

- (1) X.400-based message handling
- (2) Directory services
- (3) FTAM API
- (4) Verification testing methods
- (5) Network interface facilities
- (6) System administration.

This International Standard is based on IEEE Std 1328-1993 {B7}, which was prepared by the P1224 Working Group, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society.

1) A *Standards Status Report* that lists all current IEEE Computer Society standards projects is available from the IEEE Computer Society, 1730 Massachusetts Avenue NW, Washington, DC 20036-1903, USA; Telephone: +1 202 371-0101; FAX: +1 202 728-9614.

Information technology—Test methods for measuring conformance to Open Systems Interconnection (OSI) abstract data manipulation C language interfaces—Binding for Application Program Interface (API)

Section 1: General

1.1 Scope

This International Standard defines requirements for test methods for measuring conformance to ISO/IEC 14364 {4}.¹⁾

ISO/IEC 14364 {4} contains a programming language binding specification for ISO/IEC 14360 {2}, using the C programming language. ISO/IEC 14362 {3} contains language-independent requirements for test methods for measuring conformance to programming language binding specifications for ISO/IEC 14360 {2}, such as that contained in ISO/IEC 14364 {4}. This International Standard contains C language specific requirements for the test methods. Taken in conjunction with the requirements imposed by ISO/IEC 14362 {3}, they constitute the requirements that shall be satisfied by test methods used for measuring conformance to ISO/IEC 14364 {4}.

1) The numbers in curly brackets correspond to those of the references in 1.2.

1.2 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 currently valid International Standards.

- {1} ISO/IEC 9899: 1990,²⁾ *Programming languages—C*.
- {2} ISO/IEC 14360: 1996, *Information technology—Open Systems Interconnection (OSI) abstract data manipulation—Application Program Interface (API) [Language independent]*.
- {3} ISO/IEC 14362: 1996, *Information technology—Test methods for measuring conformance to Open Systems Interconnection (OSI) abstract data manipulation—Application Program Interface (API) [Language independent]*.
- {4} ISO/IEC 14364: 1996, *Information technology—Open Systems Interconnection (OSI) abstract data manipulation C language interfaces—Binding for Application Program Interface (API)*.
- {5} IEEE Std 1003.3-1991,³⁾ *IEEE Standard for Information Technology—Test Methods for Measuring Conformance to POSIX*.

1.3 Conformance

A set of test methods that conforms to this International Standard shall conform to IEEE Std 1003.3-1991 {5}, with references to the “POSIX.n test method specification” being interpreted as references to this International Standard, and references to “the POSIX standard for which conformance is being measured” being interpreted as references to ISO/IEC 14364 {4}.

In addition to meeting the conformance criteria defined in IEEE Std 1003.3-1991 {5}, a set of test methods that conforms to this International Standard shall test all documentation assertions defined in this International Standard.

NOTE: Conformance to IEEE Std 1003.3-1991 {5} implies that the test methods will test all other assertions defined in this International Standard.

A set of test methods that conforms to this International Standard shall also conform to ISO/IEC 14362 {3}.

2) ISO/IEC documents can be obtained from the ISO Central Secretariat, 1 Rue de Varembé, Case Postale 56, CH-1211, Genève 20, Switzerland/Suisse.

3) IEEE documents can be obtained from the IEEE Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ, 08855-1331, USA.

Section 2: Terminology and General Requirements

2.1 Conventions

2.1.1 General and Typographic Conventions

When presented in **Synopsis** sections or in extended code fragments, C language source code is printed using `fixed width font`. Where C language names or code fragments appear in the text of this International Standard the following conventions apply:

- A C language function, datatype, or code fragment appears in *italic font*. A function name is indicated by following parentheses, for example, *om_get()*.
- A symbolic constant, other than an error name, is surrounded by braces, for example, {OM_ARBITRARY_ENCODING}.
- A symbolic constant for an error name is surrounded by brackets, for example, [OM_FUNCTION_DECLINED].

2.1.2 C Naming Conventions

The C language binding specifies C identifiers for all the elements of the interface. These elements include function names, typedef names, and constants. All the C identifiers are mechanically derived from the language-independent names.

The interface uses part of the C public namespace for its facilities. All identifiers that start with the letters *om*, *omP*, *OM*, or *OMP* are reserved. It also reserves all identifiers starting with the letters *omX* or *OMX* for extensions of the interface. A list of the conventions used is given in Table 2-1.

Table 2-1 – Naming Conventions

Item	Prefix
Reserved for implementors	<i>omP</i>
Reserved for interface extensions	<i>omX</i>
Reserved for implementors	<i>OMP</i>
Reserved for interface extensions	<i>OMX</i>
Functions	<i>om_</i>
OM class names	<i>OM_C_</i>
Other constants	<i>OM_</i>

All identifiers used (except those beginning *omP*, *omX*, *OMX*, or *OMP*) are defined in ISO/IEC 14364 {4}. No implementation of the interface shall use any other public identifiers. A *public identifier* is any name except those reserved in the ISO C Standard (ISO/IEC 9899 {1}), and the *public namespace* is the set of all possible public identifiers.

The process for deriving C names from those of the language-independent specification is as follows:

- The C function names are identical to the language-independent operation names.
Thus,
om_copy_val
becomes
om_copy_val()
— C function parameters are identical to the corresponding language-independent argument names.
included_types
— The names of constants identifying OM classes are derived from the class names by converting lowercase to uppercase, converting hyphens to underscores and the prefix “OM_C_” is added.
Thus,
Object
becomes
{OM_C_OBJECT}
— For enumeration constants, OM attributes, errors, and all other constants, lowercase is converted to uppercase, hyphens are converted to underscores and the prefix “OM_” is added.
Thus,
Arbitrary-Encoding
becomes
{OM_ARBITRARY_ENCODING}
— The sequence of octets for each object identifier is also assigned an identifier, for internal use by certain OM macros. These identifiers are all uppercase and have “OMP_O_” as the prefix. See ISO/IEC 14360 {2} for further details on the use of object identifiers.

2.1.3 Language-Independent Conventions

The language-independent conventions defined in ISO/IEC 14360 {2} shall apply in this International Standard.

2.1.4 Test Methods Conventions

The assertion classification and labelling schemes defined in IEEE Std 1003.3-1991 {5} are used in this International Standard, with the following addition.

This International Standard contains an assertion for each statement that is derived from the standard to which conformance is being measured, that pertains to the documentation associated with the implementation being tested, and that is true for a conforming implementation. These assertions are called *documentation assertions*. Each documentation assertion is labelled *Dnn*, where *nn* is a two digit number that is unique within this International Standard.

2.2 Definitions

2.2.1 Terminology

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

2.2.1.1 conformance document: A document provided by an implementor that contains implementation details as described in ISO/IEC 14364 {4}.

2.2.1.2 implementation defined: An indication that the implementation shall define and document the requirements for correct program constructs and correct data of a value or behavior [ISO/IEC 9945-1 {B4}].

2.2.1.3 may: An indication of an optional feature.

With respect to implementations, the word *may* is to be interpreted as an optional feature that is not required in this International Standard, but can be provided. With respect to Strictly Conforming Applications, the word *may* means that the optional feature shall not be used [ISO/IEC 9945-1 {B4}].

2.2.1.4 shall: An indication of a requirement on the implementation or on Strictly Conforming Applications, where appropriate [ISO 9945-1 {B4}].

2.2.1.5 should:

- (1) With respect to implementations, an indication of an implementation recommendation, but not a requirement.
- (2) With respect to applications, an indication of a recommended programming practice for applications and a requirement for Strictly Conforming Applications [ISO/IEC 9945-1 {B4}].

2.2.1.6 supported: A condition regarding optional functionality.

Certain functionality in this International Standard is optional, but the interfaces to that functionality are always required. If the functionality is *supported*, the interfaces work as specified by this International Standard (except that they do not return the error condition indicated for the unsupported case). If the functionality is not *supported*, the interface shall always return the indication specified for this situation [ISO/IEC 9945-1 {B4}].

2.2.1.7 system documentation: All documentation provided with an implementation, except the conformance document [ISO/IEC 9945-1 {B4}].

Electronically distributed documents for an implementation are considered part of the system documentation.

2.2.1.8 undefined: An indication that this International Standard imposes no portability requirements on an application's use of an indeterminate value or its behavior with erroneous program constructs or erroneous data [ISO/IEC 9945-1 {B4}].

Implementations (or other standards) may specify the result of using that value or causing that behavior. An application using such behaviors is using extensions, as defined in ISO/IEC 14364 {4}.

2.2.1.9 unspecified: An indication that this International Standard imposes no portability requirements on applications for correct program constructs or correct data regarding a value or behavior [ISO/IEC 9945-1 {B4}].

Implementations (or other standards) may specify the result of using that value or causing that behavior. An application requiring a specific behavior, rather than tolerating any behavior when using that functionality, is using extensions, as defined in ISO/IEC 14364 {4}.

2.2.2 General Terms

121 For the purposes of this International Standard, the following definitions apply:

122 **2.2.2.1 abstract class:** A class, instances of which are forbidden unless they
123 belong to one of its concrete subclasses [ISO/IEC 14360 {2}].

124 **2.2.2.2 accessible object:** An object for which the client possesses a valid design-
125 ator or handle [ISO/IEC 14360 {2}].

126 **2.2.2.3 assertion:** A statement that is derived from the standard to which confor-
127 mance is being measured, that is true for a conforming implementation, and that
128 pertains either to functionality or behavior of a functional interface or namespace
129 allocation or to the documentation associated with the implementation being tested
130 [ISO/IEC 14360 {2}].

131 **2.2.2.4 attribute:** A component of an object, comprised of an integer denoting the
132 type of the attribute and an ordered sequence of one or more attribute values, each
133 accompanied by an integer denoting the syntax of the value [ISO/IEC 14360 {2}].

134 **2.2.2.5 base assertion:** An assertion that is required to be tested for required
135 features and for implemented conditional features [IEEE Std 1003.3-1991 {5}].

136 **2.2.2.6 bit string:** An ordered sequence of zero or more bits [ISO/IEC 14360 {2}].

137 **2.2.2.7 byte:** A unit of machine storage containing an ordered sequence of 8 b
138 [ISO/IEC 14360 {2}].

139 **2.2.2.8 C interface:** The C language binding, defined in terms of ISO/IEC
140 9899: 1990 [ISO/IEC 14364 {4}].

141 **2.2.2.9 character string:** An ordered sequence of zero or more characters
142 [ISO/IEC 14360 {2}].

143 **2.2.2.10 class:** A category into which objects are placed on the basis of both their
144 purpose and their internal structure [ISO/IEC 14360 {2}].

145 **2.2.2.11 client:** Software that uses the interface [ISO/IEC 14360 {2}].

146 **2.2.2.12 concrete class:** A class, instances of which are permitted either by
147 direct instantiation or the instantiation of its concrete subclasses [ISO/IEC 14360
148 {2}].

149 **2.2.2.13 datatype:** A collection of distinguished values, together with a collection
150 of characterizing operations on those values [ISO/IEC 14360 {2}].

2.2.2.14 descriptor: The means by which the client and service exchange an attribute value and the integers that denote its representation, type, and syntax [ISO/IEC 14364 {4}].

2.2.2.15 dispatcher: The software that implements the service interface functions using workspace interface functions [ISO/IEC 14364 {4}].

2.2.2.16 documentation assertion: An assertion that pertains to the documentation associated with the implementation being tested [ISO/IEC 14362 {3}].

2.2.2.17 element: Any of the bits of a bit string, the octets of an octet string, or the bytes by means of which the characters of a character string are represented [ISO/IEC 14360 {2}].

2.2.2.18 extended assertion: An assertion that is not required to be tested [IEEE Std 1003.3-1991 {5}].

2.2.2.19 generic interface: The interface, defined at a level that is independent of any particular programming language [ISO/IEC 14360 {2}].

2.2.2.20 immediate subclass: A subclass, of a class *C*, having no superclasses that are themselves subclasses of *C* [ISO/IEC 14360 {2}].

2.2.2.21 immediate subobject: One object that is a value of an attribute of another [ISO/IEC 14360 {2}].

2.2.2.22 immediate superclass: The superclass, of a class *C*, having no subclasses that are themselves superclasses of *C* [ISO/IEC 14360 {2}].

2.2.2.23 immediate superobject: One object that contains another among its attribute values [ISO/IEC 14360 {2}].

2.2.2.24 inaccessible object: An object for which the client does not possess a valid designator or handle [ISO/IEC 14360 {2}].

2.2.2.25 independent copy: A copy of the object plus independent copies of all its subobjects (applied recursively) [ISO/IEC 14360 {2}].

2.2.2.26 instance: An object in the category represented by a class [ISO/IEC 14360 {2}].

2.2.2.27 intermediate datatype: Any of the basic datatypes in terms of which the other, substantive, datatypes of the interface are defined [ISO/IEC 14360 {2}].

2.2.2.28 intermediate macro: Any of the basic macros in terms of which the other, substantive, macros used to realize the dispatcher are defined [ISO/IEC 14364 {4}].

2.2.2.29 length: The number of bytes required to represent any string [ISO/IEC 14360 {2}].

2.2.2.30 minimally consistent object: An object that satisfies various conditions set forth in the definition of its class [ISO/IEC 14360 {2}].

2.2.2.31 name: An identifier used to represent a specific value of a specific datatype.

This International Standard does not specify when a name is bound to its value: at compile time, link time, or load time. The use of a name as a parameter in a datatype specification does not imply that its value is defined at compile time [ISO/IEC 14360 {2}].

2.2.2.32 object: Any of the complex information objects created, examined, modified, or destroyed by means of the interface [ISO/IEC 14360 {2}].

2.2.2.33 octet string: A string comprised of octets [ISO/IEC 14360 {2}].

2.2.2.34 operation: A program abstraction with formal input and output parameters that are bound to objects or values in the application that invokes it [ISO/IEC 14360 {2}].

2.2.2.35 OSI abstract data manipulation application program interface: The interface that ISO/IEC 14360 {2} and ISO/IEC 14362 {3} define.

2.2.2.36 package: A group of related classes [ISO/IEC 14360 {2}].

2.2.2.37 package closure: The set of classes that need to be supported in order to be able to create all possible instances of all classes defined in the package [ISO/IEC 14360 {2}].

2.2.2.38 position (within a string): The ordinal position of one element of a string relative to another [ISO/IEC 14360 {2}].

2.2.2.39 position (within an attribute): The ordinal position of one value relative to another [ISO/IEC 14360 {2}].

2.2.2.40 primary representation: The form in which the service supplies an attribute value to the client [ISO/IEC 14360 {2}].

2.2.2.41 private object: The internal representation of the language-independent model of an object in a service, and thus unspecified [ISO/IEC 14360 {2}].

2.2.2.42 programming language binding specification: For a language-independent specification, a document that specifies, in terms of a particular programming language, the behavior that the language-independent specification specifies in language-independent terms.

It may also specify additional behavior that is relevant to the usage of the particular programming language [ISO/IEC 14360 {2}].

2.2.2.43 public object: The representation by the client of an object in a particular language binding [ISO/IEC 14360 {2}].

2.2.2.44 secondary representation: A second form, an alternative to the primary representation, in which the client may supply an attribute value to the service [ISO/IEC 14360 {2}].

2.2.2.45 segment: Zero or more contiguous elements of a string [ISO/IEC 14360 {2}].

2.2.2.46 service: Software that implements the interface [ISO/IEC 14360 {2}].

2.2.2.47 service interface: The interface as realized, for the benefit of the client, by the service as a whole [ISO/IEC 14360 {2}].

2.2.2.48 specific: With respect to a class, the attribute types that may appear in an instance of the class but not in an instance of its superclasses [ISO/IEC 14360 {2}].

2.2.2.49 status datatype: An abstract datatype whose values may be bound to “control” values as well as “data” values [ISO/IEC 14360 {2}].

2.2.2.50 string: An ordered sequence of zero or more bits, octets, or characters, accompanied by the length of the string [ISO/IEC 14360 {2}].

2.2.2.51 subclass: One of the classes, designated as such, whose attribute types are a superset of those of another class [ISO/IEC 14360 {2}].

2.2.2.52 subobject: An immediate subobject of an object or of one of its subobjects applied recursively [ISO/IEC 14360 {2}].

2.2.2.53 superclass: One of the classes, designated as such, whose attribute types are a subset of those of another class [ISO/IEC 14360 {2}].

2.2.2.54 superobject: The immediate superobject of an object, or one of its superobjects applied recursively [ISO/IEC 14360 {2}].

2.2.2.55 syntax: A category into which an attribute value is placed on the basis of its form [ISO/IEC 14360 {2}].

2.2.2.56 syntax template: A lexical construct containing an asterisk from which several attribute syntaxes can be derived by substituting text for the asterisk [ISO/IEC 14360 {2}].

2.2.2.57 test methods: The software, procedures, or other means specified to measure conformance to a specification [ISO/IEC 14360 {2}].

2.2.2.58 type: A category into which attribute values are placed on the basis of their purpose [ISO/IEC 14360 {2}].

2.2.2.59 value: An arbitrarily complex information item that can be viewed as a characteristic or property of an object [ISO/IEC 14360 {2}].

2.2.2.60 workspace: A repository for instances of classes in the closures of one or more packages associated with the workspace [ISO/IEC 14360 {2}].

2.2.2.61 workspace interface: The interface as realized, for the benefit of the dispatcher, by each workspace individually [ISO/IEC 14364 {4}].

2.2.3 Abbreviations

For the purposes of this International Standard, the following abbreviations apply:

2.2.3.1 API: Application Program Interface.

2.2.3.2 ASN.1: Abstract Syntax Notation One (defined in ISO/IEC 8824 {B1}).

2.2.3.3 BER: Basic Encoding Rules (defined in ISO/IEC 8825 {B2}).

2.2.3.4 C Standard: ISO/IEC 9899, *Information technology—Programming languages—C* {1}.

2.2.3.5 IA5: International Alphabet No. 5 (used in “IA5 String,” see ISO/IEC 8824: 1990 {B1}).

2.2.3.6 OM: OSI abstract data Manipulation.

2.2.3.7 OSI: Open Systems Interconnection.

2.2.3.8 UTC: Universal Coordinated Time.

Section 3: Test Assertions—General

3.1 Scope

There are no test assertions for this clause.

3.2 Normative References

There are no test assertions for this clause.

3.3 Conformance

3.3.1 Implementation Conformance

D01 The precise definitions in C of the following intermediate datatypes are documented: *OM_sint*, *OM_sint16*, *OM_sint32*, *OM_uint*, *OM_uint16*, *OM_uint32*.

3.3.2 Application Conformance

There are no test assertions for this subclause.

3.4 Test Methods

There are no test assertions for this clause.

Section 4: Test Assertions—Terminology and General Requirements

4.1 Conventions

4.1.1 General and Typographic Conventions

There are no test assertions for this subclause.

4.1.2 C Naming Conventions

001(B) The identifier for an element of the language-independent specification depends on the type of element, as specified in Table 4-1.

Table 4-1 – Naming Conventions

Element Type	Prefix	Case
Datatype	OM_	Lower
Data value	OM_	Upper
Data value (Class1)	OM_C_	Upper
Data value (Syntax)	OM_S_	Upper
Data value component (Structure member)	<i>none</i>	Lower
Function	om_	Lower
Function argument	<i>none</i>	Lower
Function result	<i>none</i>	Lower
Macro	OM_	Upper
Reserved for use by implementors	OMP	any
Reserved for use by implementors	omP	any
Reserved for proprietary extension	omX	any
Reserved for proprietary extension	OMX	any

4.1.3 Language-Independent Conventions

There are no test assertions for this subclause.

4.2 Definitions

25 There are no test assertions for this clause.