INTERNATIONAL STANDARD ISO/IEC 8824-1:1998
TECHNICAL CORRIGENDUM 2

Published 2002-02-01

Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation

TECHNICAL CORRIGENDUM 2

Technologies de l'information — Notation de syntaxe abstraite numéro un (ASN.1): Spécification de la notation de base

RECTIFICATIF TECHNIQUE 2

Technical Corrigendum 2 to International Standard ISO/IEC 8824-1:1998 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology,* Subcommittee SC 6, *Telecommunications and information exchange between systems.*

ICS 35.100.60

Ref. No. ISO/IEC 8824-1:1998/Cor.2:2002(E)

iTeh STANDARD PREVIEW

(standards.iteh.ai)

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

INFORMATION TECHNOLOGY – ABSTRACT SYNTAX NOTATION ONE (ASN.1): SPECIFICATION OF BASIC NOTATION

TECHNICAL CORRIGENDUM 2

1) Subclause 12.1 and Annex G

Change subclause 12.1 and Annex G so that "Exports" is as follows:

> **Exports ::=**
>> **EXPORTS SymbolsExported ";" |**
>> **EXPORTS ALL ";" |**
>> **empty**

2) Subclause 12.3

Replace the sentence:

This transformation is formally specified by 24.7 to 24.9, 26.3 and 28.2 regarding the notation for sequence types, set types and choice types, respectively.

by:

This transformation is formally specified by 24.7 to 24.9, 26.3 and 28.2 to 28.3 regarding the notation for sequence types, set types and choice types, respectively.

3) Subclause 12.13

Replace subclause 12.13 with the following:

12.13   When either the "empty" alternative or the "EXPORTS ALL" alternative of "Exports" is selected, every "Symbol" defined in the module or imported by the module may be referenced from other modules subject to the restriction specified in subclause 12.12 a).

4) Subclause 14.3 and Annex G

Change subclause 14.3 and Annex G so that "AbsoluteReference" is as follows:

> **AbsoluteReference ::= "@" ModuleIdentifier**
>> **"."**
>> **ItemSpec**

5) Subclause 14.4

Replace subclause 14.4 with the following:

The "ModuleIdentifier" identifies an ASN.1 module (see 12.1).

6)   New subclause 14.4 *bis*

Add a new subclause 14.4 bis as follows:

14.4 *bis*   When the first altenative of "DefinitiveIdentifier" is used as part of the "ModuleIdentifier", the "DefinitiveIdentifier" unambiguously and uniquely identifies the module from which an item is being referenced.

7)   Subclause 14.5

Replace subclause 14.5 with the following:

14.5   The "typereference" references any ASN.1 type defined in the module identified by "ModuleIdentifier".

8)   Subclause 19.1 and Annex G

Change subclause 19.1 and Annex G so that "Enumerations" is as follows:

> **Enumerations ::= RootEnumeration |**
> **RootEnumeration ","   "..." ExceptionSpec |**
> **RootEnumeration ","   "..." ExceptionSpec ","   AdditionalEnumeration**

9)   Subclause 24.5.1

Delete **"if automatic tagging has be applied"** *from the end of the last sentence.*

10)   Subclause 24.6

Replace subclause 24.6 by:

24.6   When the third or the fourth alternative of "ComponentTypeLists" is used, all "ComponentType"s in extension additions shall have tags which are distinct from the tags of the textually following "ComponentType"s up to and including the first such "ComponentType" that is not marked OPTIONAL or DEFAULT in the trailing "RootComponentTypeList", if any.

11)   Subclause 24.8

Replace subclause 24.8 by:

24.8   If automatic tagging is in effect and the "ComponentType"s in the extension root have no tags, then no "ComponentType" within the "ExtensionAdditionList" shall be a "TaggedType".

12)   Subclause 24.9

Replace the beginning of subclause 24.9 c) by:

c) the "ClassNumber" in the replacement "TaggedType" is the tag value zero for the first "ComponentType" in the "RootComponentTypeList", one for the second, and so on, proceeding with increasing tag numbers;

Replace NOTE 2 *by:*

> NOTE 2 – Once 24.7 is satisfied, the tags of the components are completely determined, and are not modified even when the sequence type is referenced in the definition of a component within another "ComponentTypeLists" for which automatic tagging transformation applies. Thus, in the following case:
>
> **T ::= SEQUENCE { a Ta, b   Tb, c   Tc }**
>
> **E ::= SEQUENCE { f1 E1, f2 T,   f3 E3 }**
>
> automatic tagging applied to the components of E never affects the tags attached to components a, b and c of T, whatever the tagging environment of T. If T is defined in an automatic tagging environment and E is not in an automatic tagging environment, automatic tagging is still applied to components a, b and c of T.

Replace NOTE 5 *by:*

> NOTE 5 – When automatic tagging is in place, insertion of new components at any location other than the extension insertion point (see 3.8.29) may result in changes to other components due to the side effect of modifying the tags thus causing interworking problems with an older version of the specification.

13)   Subclause 28.2

Replace subclause 28.2 by:

28.2   When the "AlternativeTypeLists" production occurs within the definition of a module for which automatic tagging is selected (see 12.3), and none of the occurrences of "NamedType" in any "AlternativeTypeList" contains a "TaggedType", the automatic tagging transformation is selected for the entire "AlternativeTypeLists", otherwise it is not.

Add the following subclauses:

28.2 *bis*   The types defined in the "AlternativeTypeList" productions in an "AlternativeTypeLists" shall have distinct tags (see clause 30). If automatic tagging was selected, the requirement that tags be distinct applies only after automatic tagging has been performed, and will always be satisfied.

28.2 *ter*   If automatic tagging is in effect and the "NamedType"s in the extension root have no tags, then no "NamedType" within the "ExtensionAdditionAlternativesList" shall be a tagged type.

14)   Subclause 28.3

Replace subclause 28.3 by:

28.3   The automatic tagging transformation impacts each "NamedType" of the "AlternativeTypeLists" by replacing the "Type" originally in the "NamedType" production with a replacement "TaggedType". The replacement "TaggedType" is specified as follows:

   a)   the replacement "TaggedType" notation uses the "Tag Type" alternative;

   b)   the "Class" of the replacement "TaggedType" is empty (i.e. tagging is context-specific);

   c)   the "ClassNumber" in the replacement "TaggedType" is tag value zero for the first "NamedType" in the "RootAlternativeTypeList", one for the second, and so on, proceeding with increasing tag numbers;

   d)   the "ClassNumber" in the replacement "TaggedType" of the first "NamedType" in the "ExtensionAdditionAlternativesList" is one greater than the largest "ClassNumber" in the "RootAlternativeTypeList", with the next "NamedType" in the "ExtensionAdditionAlternativesList" having a "ClassNumber" one greater than the first, and so on, proceeding with increasing tag numbers;

   e)   the "Type" in the replacement "TaggedType" is the original "Type" being replaced.

NOTE 1 – The rules governing specification of implicit tagging or explicit tagging for replacement "TaggedType"s are provided by 30.6. Automatic tagging is always implicit tagging unless the "Type" is a choice type or an open type notation, or a "DummyReference" (see 8.3 of ITU-T Rec. X683 | ISO/IEC 8824-4), in which case it is explicit tagging.

NOTE 2 – Once automatic tagging has been applied, the tags of the components are completely determined, and are not modified even when the choice type is referenced in the definition of an alternative within another "AlternativeTypeLists" for which automatic tagging transformation applies. Thus, in the following case:

T ::= CHOICE { a Ta, b Tb, c Tc }

E ::= CHOICE {f1 E1, f2 T, f3 E3}

automatic tagging applied to the components of E never affects the tags attached to components a, b and c of T, whatever the tagging environment of T. If T is defined in an automatic tagging environment and E is not in an automatic tagging environment, automatic tagging is still applied to components a, b and c of T.

NOTE 3 – Subtyping has no impact on automatic tagging.

NOTE 4 – When automatic tagging is in place, insertion of new alternatives at any location other than the extension insertion point (see 3.8.29) may result in changes to other alternatives due to the side effect of modifying the tags thus causing interworking problems with an older version of the specification.

15)   Subclause 28.6

Replace the following:

      (see 24.5 to 24.6 and 28.2)

by:

      (see 28.2 *bis*)

16) Subclauses 30, 30.7, 47.7, 47.7.2, 47.7.3 (Note)

Replace:

> (see 24.5 to 24.6, 26.3 and 28.2).

by:

> (see 24.5 to 24.6, 26.3 and 28.2 *bis*).

17) Subclause 48.1

Add the following line to Table 6 after the line whose column 1 specifies "Object class field type":

Object Descriptor	Yes	Yes	No	Yes	Yes	No	No

and add the following line to Table 6 after the line whose column 1 specifies "Set-of":

Time Types	Yes	Yes	No	Yes	No	No	No