INTERNATIONAL STANDARD

ISO/IEC 8824-1

Second edition
1998-12-15

AMENDMENT 2
2000-10-15

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

AMENDMENT 2: ASN.1 Semantic Model

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

AMENDEMENT 2: Modèle sémantique ASN.1

Reference number
ISO/IEC 8824-1:1998/Amd.2:2000(E)

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

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.

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

Amendment 2 to International Standard ISO/IEC 8824-1:1998 was prepared by ITU-T (as ITU-T Recommendation X.680/Amd.2) and was adopted as common text, 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.

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

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

AMENDMENT 2
ASN.1 semantic model

1) Subclause 3.8

Add the definitions 3.8.39 bis and 3.8.71 bis to subclause 3.8 and replace the definition of **governing; governor** *with the new 3.8.39 below:*

3.8.39 governing; governor (type): A type definition or reference which affects the interpretation of a part of the ASN.1 syntax, requiring that part of the ASN.1 syntax to reference values in the governor type.

3.8.39 *bis* identical type definitions: Two instances of the ASN.1 "Type" production (see clause 16) are defined as identical type definitions if, after performing the transformations specified in Annex F, they are identical ordered lists of ASN.1 items (see clause 11).

3.8.71 *bis* value mapping: A 1-1 relationship between values in two types that enables a reference to one of those values to be used as a reference to the other value. This can, for example, be used in specifying subtypes and default values (see Annex F).

Append to clause 3.8.50:

, and which governs the subtype notation.

2) New subclause 5.9

Add a new subclause 5.9 as follows:

5.9 Value references and the typing of values

5.9.1 ASN.1 defines a value assignment notation which enables a name to be given to a value of a specified type. This name can be used wherever a reference to that value is needed. Annex F describes and specifies the **value mapping** mechanism which allows a value reference name for a value of one type to identify a value of a second type. Thus, a reference to the first value can be used where a reference to a value in the second type is required.

5.9.2 In the body of the ASN.1 standards, normal English text is used to specify legality (or otherwise) of constructs where more than one type is involved, but where the two types must be "compatible". For example, the type used in defining a value reference is required to be "compatible with" the governor type when the value reference is used. The normative Annex F uses the value mapping concept to give a precise statement about whether any given ASN.1 construct is legal or not.

3) New subclause 13.6

Add a new subclause 13.6 as follows:

13.6 Where a "DefinedType" is used as part of notation governed by a "Type" (for example, in a "SubtypeElementSpec"), then the "DefinedType" shall be compatible with the governing "Type" as specified in F.6.2.

4) New subclause 13.7

Add a new subclause 13.7 as follows:

13.7 Every occurrence within an ASN.1 specification of a "DefinedValue" is governed by a "Type", and that "DefinedValue" shall reference a value of a type that is compatible with the governing "Type" as specified in F.6.2.

5) Subclause 15.2

In 15.2, change the sentence after the BNF by inserting immediately before "shall" the text:

is governed by "Type" and

6) Subclause 36.7

Add the following to the Note 3 example in 36.7:

An alternative unambiguous definition of "mystring" would be:

```
mystring MyAlphabet(BasicLatin) ::= "HOPE"
```

Formally, "mystring" is a value reference to a value of a subset of "MyAlphabet", but it can, by the value mapping rules of Annex F, be used wherever a value reference is needed to this value within "MyAlphabet".

7) Subclause 48.3.2

Replace 48.3.2 with the following text:

48.3.2 A "ContainedSubtype" specifies all of the values in the parent type that are also in "Type". "Type" is required to be compatible with the parent type as specified in F.6.3.

8) Subclause 48.5.2

In 48.5.2, delete the text ", or types formed from any of those types by tagging".

9) Subclause 48.8.2

In 48.8.2, delete the text ", or types formed from any of those types by tagging".

10) New Annex F

Add a new Annex F as follows:

Annex F

Rules for Type and Value Compatibility

(This annex forms an integral part of this Recommendation | International Standard)

This annex is expected to be mainly of use to tool builders to ensure that they interpret the language identically. It is present in order to clearly specify what is legal ASN.1 and what is not, and to be able to specify the precise value that any value reference name identifies, and the precise set of values that any type or value set reference name identifies. It is not intended to provide a definition of valid transformations of ASN.1 notations for any purpose other than those stated above.

F.1 The need for the value mapping concept (Tutorial introduction)

F.1.1 Consider the following ASN.1 definitions:

```
A ::= INTEGER
B ::= [1] INTEGER
C ::= [2] INTEGER (0..6,...)
D ::= [2] INTEGER (0..6,...,7)
E ::= INTEGER (7..20)
F ::= INTEGER {red(0), white(1), blue(2), green(3), purple(4)}
a A ::= 3
b B ::= 4
c C ::= 5
d D ::= 6
e E ::= 7
f F ::= green
```


F.1.2 It is clear that the value references a, b, c, d, e, and f can be used in value notation governed by A, B, C, D, E, and F, respectively. For example:

```
W ::= SEQUENCE {w1   A   DEFAULT a}
```

and:

```
x A ::= a
```

and:

```
Y ::= A(1..a)
```

are all valid given the definitions in F.1.1. If, however, A above were replaced by B, or C, or D, or E, or F, would the resulting statements be illegal? Similarly, if the value reference a above were replaced in each of these cases by b, or c, or d, or e, or f, are the resulting statements legal?

F.1.3 A more sophisticated question would be to consider in each case replacement of the type reference by the explicit text to the right of its assignment. Consider for example:

```
f INTEGER {red(0), white(1), blue(2), green(3), purple(4)} ::= green
W ::= SEQUENCE {
        w1   INTEGER {red(0), white(1), blue(2), green(3),  purple(4)}
             DEFAULT f}
x INTEGER {red(0), white(1), blue(2), green(3),  purple(4)} ::= f
Y ::= INTEGER {red(0), white(1), blue(2), green(3),  purple(4)}(1..f)
```

Would the above be legal ASN.1?

F.1.4 Some of the above examples are cases which, even if legal (as most of them are – see later text), users would be ill-advised to write similar text, as they are at the least obscure and at worst confusing. However, there are frequent uses of a value reference to a value of some type (not necessarily just an INTEGER type) as the default value for that type with tagging or subtyping applied in the governor. The **value mapping** concept is introduced in order to provide a clear and precise means of determining which constructs such as the above are legal.

F.1.5 Again, consider:

```
C ::= [2] INTEGER (0..6,...)

E ::= INTEGER (7..20)

F ::= INTEGER {red(0), white(1), blue(2), green(3), purple(4)}
```

In each case a new type is being created. For F we can clearly identify a 1-1 correspondence between the values in it and the values in the universal type "INTEGER". In the case of C and E, we can clearly identify a 1-1 correspondence between the values in them and a subset of the values in the universal type "INTEGER". We call this relationship a **value mapping** between values in the two types. Moreover, because values in F, C, and E all have (1-1) mappings to values of "INTEGER", we can use these mappings to provide mappings between the values of F, C, and E themselves. This is illustrated for F and C in Figure F.1.

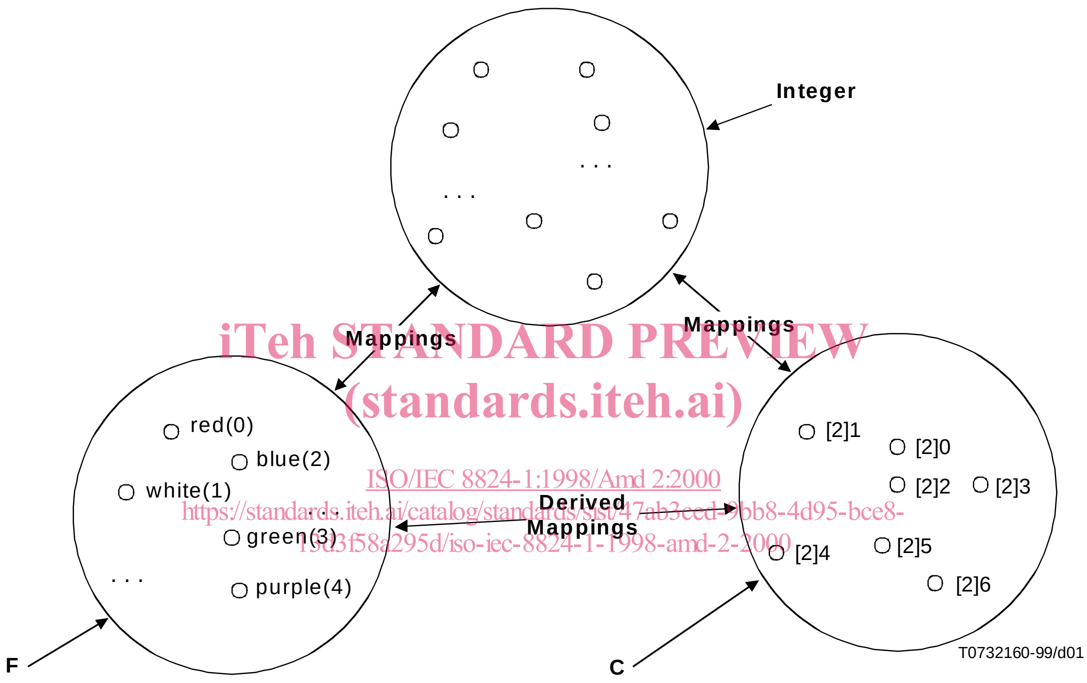

Figure F.1

F.1.6 Now when we have a value reference such as:

```
c C ::= 5
```

to a value in "C" which is required in some context to identify a value in "F", then, provided a value mapping exists between that value in "C" and a (single) value in "F", we can (and do) define "c" to be a legal reference to the value in "F". This is illustrated in Figure F.2, where the value reference "c" is used to identify a value in "F", and can be used in place of a direct reference f1 where we would otherwise have to define:

```
f1 F ::= 5
```

F.1.7 It should be noted that in some cases there will be values in one type (7 to 20 in A of F.1.1 for example) that have value mappings to values in another type (7 to 20 in E of F.1.1 for example), but other values (21 upwards of A) that have no such mapping. A reference to such values in A would not provide a valid reference to a value in E. (In this example, the whole of E has a value mapping to a subset of A. In the general case, there may be a subset of values in both types that have mappings, with other values in both types that are unmapped.)

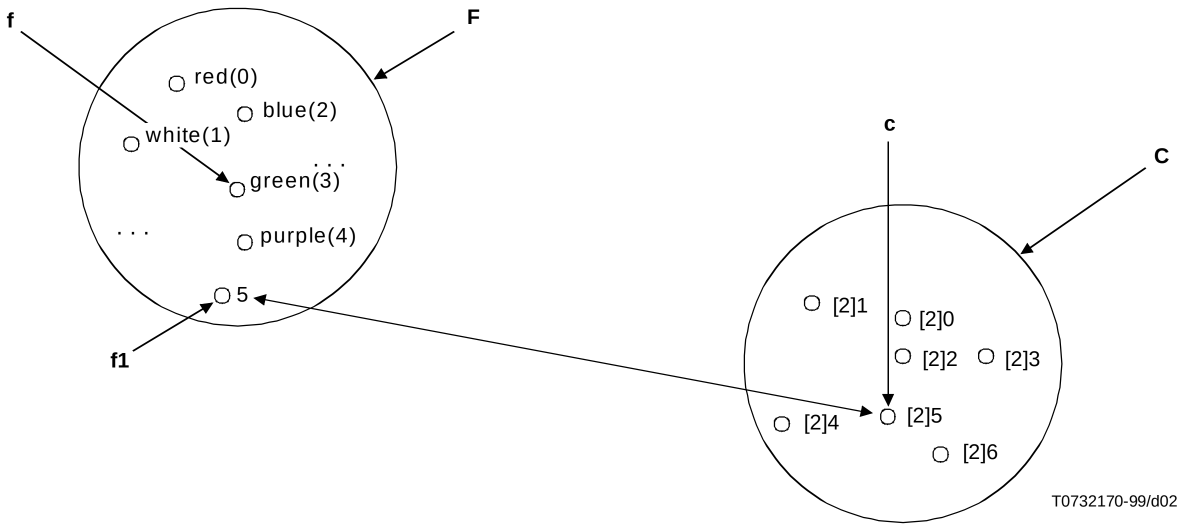

Figure F.2

F.1.8 In the body of the ASN.1 standards, normal English text is used to specify legality in the above and similar cases. Subclause F.6 gives the precise requirements for legality and should be referenced whenever there is doubt about a complex construction.

NOTE – The fact that value mappings are defined to exist between two occurrences of the "Type" construct permits the use of value references established using one "Type" construct to identify values in another "Type" construct which is sufficiently similar. It allows dummy and actual parameters to be typed using two textually separate "Type" constructs without violating the rules for compatibility of dummy and actual parameters. It also allows fields of information object classes to be specified using one "Type" construct and the corresponding value in an information object to be specified using a distinct "Type" construct which is sufficiently similar. (These examples are not intended to be exhaustive.) It is, however, recommended that advantage be taken of this freedom only for simple cases such as "SEQUENCE OF INTEGER", or "CHOICE {int INTEGER, id OBJECT IDENTIFIER}", and not for more complex "Type" constructs.

F.2 Value mappings

F.2.1 The underlying model is of types, as non-overlapping containers, that contain values, with every occurrence of the ASN.1 "Type" construct defining a distinct new type (see Figures F.1 and F.2). This annex specifies when **value mappings** exist between such types, enabling a reference to a value in one type to be used where a reference to a value in some other type is needed.

Example: Consider:

```
X ::= INTEGER

Y ::= INTEGER
```

X and Y are type reference names (pointers) to two distinct types, but value mappings exist between these types, so any value reference to a value of X can be used when governed by Y (for example, following DEFAULT).

F.2.2 In the set of all possible ASN.1 values, a value mapping relates a pair of values. The whole set of value mappings is a mathematical relation. This relation possesses the following properties: it is reflexive (each ASN.1 value is related to itself), it is symmetric (if a value mapping is defined to exist from a value x1 to a value x2, then there automatically exists a value mapping from x2 to x1), and it is transitive (if there is a value mapping from a value x1 to x2, and a value mapping from x2 to x3, then there automatically exists a value mapping from x1 to x3).

F.2.3 Furthermore, given any two types X1 and X2, seen as sets of values, the set of value mappings from values in X1 to values in X2 is a one-to-one relation, that is, for all values x1 in X1, and x2 in X2, if there is a value mapping from x1 to x2, then:

a) there is no value mapping from x1 to another value in X2 different from x2; and

b) there is no value mapping from any value in X1 (other than x1) to x2.