INTERNATIONAL STANDARD

ISO 8825

First edition 1987-11-15

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION
МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ

Information processing systems — Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)

Reference number
ISO 8825 : 1987 (E)

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work.

Draft International Standards adopted by the technical committees are circulated to the member bodies for approval before their acceptance as International Standards by the ISO Council. They are approved in accordance with ISO procedures requiring at least 75 % approval by the member bodies voting.

International Standard ISO 8825 was prepared by Technical Committee ISO/TC 97, *Information processing systems.*

Users should note that all International Standards undergo revision from time to time and that any reference made herein to any other International Standard implies its latest edition, unless otherwise stated.

Contents

Annexes

Figures

Tables

Information processing systems — Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)

0 Introduction

ISO 8824 (Specification of Abstract Syntax Notation One) specifies a notation for the definition of abstract syntaxes, enabling application layer standards to define the types of information they need to transfer using the presentation service. It also specifies a notation for the specification of values of a defined type.

This International Standard defines a set of encoding rules that may be applied to values of types defined using the notation specified in ISO 8824. Application of these encoding rules produces a transfer syntax for such values. It is implicit in the specification of these encoding rules that they are also to be used for decoding.

There may be more than one set of encoding rules that can be applied to values of types that are defined using the notation of ISO 8824. This International Standard defines one set of encoding rules, called **basic encoding rules**.

This International Standard is technically aligned with the relevant parts of CCITT Recommendation X.409 (1984).

Annex A gives examples of the application of the encoding rules. It is not part of this International Standard.

Annex B summarises the assignment of object identifier values made in this International Standard and is not part of this International Standard.

1 Scope and field of application

This International Standard specifies a set of basic encoding rules that may be used to derive the specification of a transfer syntax for values of types defined using the notation specified in ISO 8824. These basic encoding rules are also to be applied for decoding such a transfer syntax in order to identify the data values being transferred.

These basic encoding rules are used at the time of communication (by the presentation service provider when required by a presentation context).

2 References

ISO 2022, *Information processing - ISO 7-bit and 8-bit coded character sets - Code extension techniques.*

ISO 2375, *Data processing - Procedure for registration of escape sequences.*

ISO 7498, *Information processing systems - Open Systems Interconnection - Basic Reference Model.*

ISO 8823, *Information processing systems - Open Systems Interconnection - Connection-oriented presentation protocol specification.*

ISO 8824, *Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1).*

CCITT X.409 (1984), *Message handling systems: presentation transfer syntax and notation.*

3 Definitions

The definitions of ISO 8824 are used in this International Standard.

3.1 dynamic conformance: A statement of the requirement for an implementation to adhere to the behaviour prescribed by this International Standard in an instance of communication.

3.2 static conformance: A statement of the requirement for support by an implementation of a valid set of features from among those defined by this International Standard.

3.3 data value: Information specified as the value of a type; the type and the value are defined using ASN.1.

3.4 encoding (of a data value): The complete sequence of octets used to represent the data value.

NOTE — Some CCITT Recommendations use the term "data element" for this sequence of octets, but the term is not used in this International Standard, as other International Standards use it to mean "data value".

3.5 identifier octets: Part of a data value encoding which is used to identify the type of the value.

3.6 length octets: Part of a data value encoding following the identifier octets which is used to determine the end of the encoding.

3.7 end-of-contents octets: Part of a data value encoding, occurring at its end, which is used to determine the end of the encoding.

NOTE — Not all encodings require end-of-contents octets.

3.8 contents octets: That part of a data value encoding which represents a particular value, to distinguish it from other values of the same type.

3.9 primitive encoding: A data value encoding in which the contents octets directly represent the value.

3.10 constructed encoding: A data value encoding in which the contents octets are the complete encoding of one or more other data values.

3.11 sender: An implementation encoding a data value for transfer.

3.12 receiver: An implementation decoding the octets produced by a sender, in order to identify the datavalue which was encoded.

4 Abbreviations and notation

4.1 Abbreviations

ASN.1	Abstract Syntax Notation One

4.2 Notation

4.2.1 This International Standard references the notation defined by ISO 8824.

4.2.2 This International Standard specifies the value of each octet in an encoding by use of the terms "most significant bit" and "least significant bit".

NOTE — Lower layer standards use the same notation to define the order of bit transmission on a serial line, or the assignment of bits to parallel channels.

4.2.3 For the purposes of this International Standard only, the bits of an octet are numbered from 8 to 1, where bit 8 is the "most significant bit", and bit 1 is the "least significant bit".

5 Conformance

5.1 Dynamic conformance is specified by clause 6 to clause 21 inclusive.

5.2 Static conformance is specified by those standards which specify the application of these basic encoding rules.

5.3 Alternative encodings are permitted by this International Standard as a sender's option. Conforming receivers shall support all alternatives.

NOTE — Examples of such alternative encodings appear in 6.3.2 b) and table 2.

6 General rules for encoding

6.1 Structure of an encoding

6.1.1 The encoding of a data value shall consist of four components which shall appear in the following order:

a) identifier octets (see 6.2);

b) length octets (see 6.3);

c) contents octets (see 6.4);

d) end-of-contents octets (see 6.5).

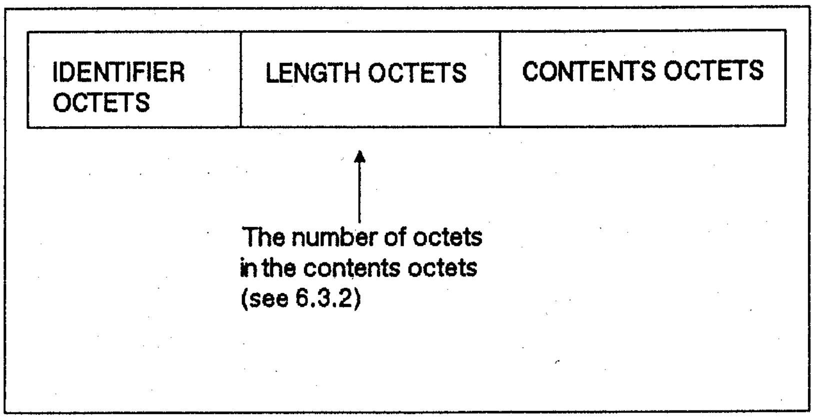

Figure 1 — Structure of an encoding

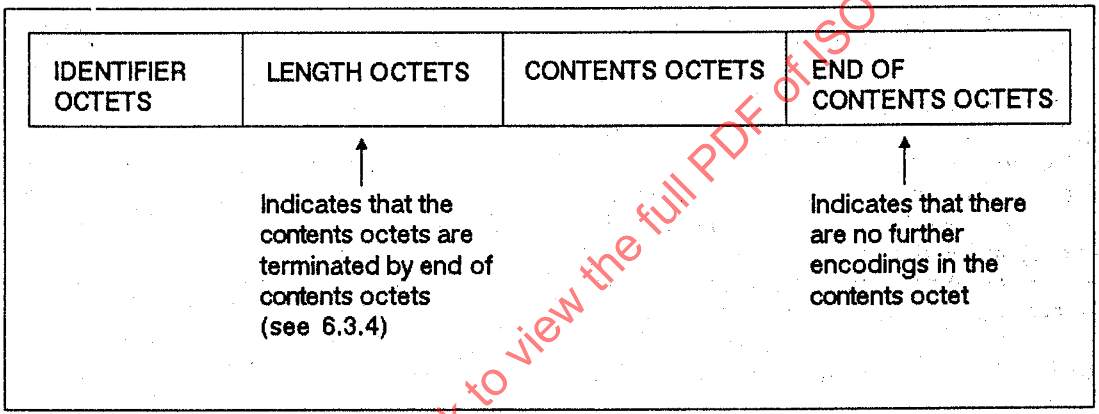
IDENTIFIER OCTETS
LENGTH OCTETS
CONTENTS OCTETS
END OF CONTENTS OCTETS
Indicates that the contents octets are terminated by end of contents octets (see 6.3.4)
Indicates that there are no further encodings in the contents octet

Figure 2 — An alternative constructed encoding

6.1.2 The end-of-contents octets shall not be present unless the value of the length octets requires them to be present (see 6.3).

6.1.3 Figure 1 illustrates the structure of an encoding (primitive or constructed). Figure 2 illustrates an alternative constructed encoding.

6.2 Identifier octets

6.2.1 The identifier octets shall encode the ASN.1 tag (class and number) of the type of the data value.

6.2.2 For tags with a number ranging from zero to 30 (inclusive), the identifier octets shall comprise a single octet encoded as follows:

a) bits 8 and 7 shall be encoded to represent the class of the tag as specified in table 1.

Table 1 — Encoding of class of tag

Class	Bit 8	Bit 7
Universal	0	0
Application	0	1
Context – specific	1	0
Private	1	1

b) bit 6 shall be a zero or a one according to the rules of 6.2.5;

c) bits 5 to 1 shall encode the number of the tag as a binary integer with bit 5 as the most significant bit.

6.2.3 Figure 3 illustrates the form of an identifier octet for a type with a tag whose number is in the range zero to 30 (inclusive).

6.2.4 For tags with a number greater than or equal to 31, the identifier shall comprise a leading octet followed by one or more subsequent octets.

6.2.4.1 The leading octet shall be encoded as follows:

a) bits 8 and 7 shall be encoded to represent the class of the tag as listed in table 1;

b) bit 6 shall be a zero or a one according to the rules of 6.2.5;

c) bits 5 to 1 shall be encoded as 11111_2

6.2.4.2 The subsequent octets shall encode the number of the tag as follows:

a) bit 8 of each octet shall be set to one unless it is the last octet of the identifier octets;

b) bits 7 to 1 of the first subsequent octet, followed by bits 7 to 1 of the second subsequent octet, followed in turn by bits 7 to 1 of each further octet, up to and including the last subsequent octet in the identifier octets shall be the encoding of an unsigned binary integer equal to the tag number, with bit 7 of the first subsequent octet as the most significant bit;

c) bits 7 to 1 of the first subsequent octet shall not all be zero.

6.2.4.3 Figure 4 illustrates the form of the identifier octets for a type with a tag whose number is greater than 30.

6.2.5 Bit 6 shall be set to zero if the encoding is primitive, and shall be set to one if the encoding is constructed.

NOTE — Subsequent clauses specify whether the encoding is primitive or constructed for each type.

6.2.6 ISO 8824 specifies that the tag of a type defined using the "CHOICE" keyword takes the value of the tag of the type from which the chosen data value is taken.

6.2.7 ISO 8824 specifies that the tag of a type defined using "ANY" is indeterminate. The "ANY" type is subsequently defined to be an ASN.1 type, and the complete encoding is then identical to that of a value of the assigned type (including the identifier octets).

6.3 Length octets

6.3.1 Two forms of length octets are specified. These are

a) the definite form (see 6.3.3); and

b) the indefinite form (see 6.3.4).

6.3.2 A sender shall

a) use the definite form (6.3.3) if the encoding is primitive;

b) use either the definite form (6.3.3) or the indefinite form (6.3.4), a sender's option, if the encoding is constructed and all immediately available;

c) use the indefinite form (6.3.4) if the encoding is constructed and is not all immediately available.

6.3.3 For the definite form, the length octets shall consist of one or more octets, and shall represent the number of octets in the contents octets using either the short form (6.3.3.1) or the long form (6.3.3.2) as a sender's option.

NOTE — The short form can only be used if the number of octets in the contents octets is less than or equal to 127.

6.3.3.1 In the short form, the length octets shall consist of a single octet in which bit 8 is zero and bits 7 to 1 encode the number of octets in the contents octets (which may be zero), as an unsigned binary integer with bit 7 as the most significant bit.

EXAMPLE

L = 38 can be encoded as 00100110_2

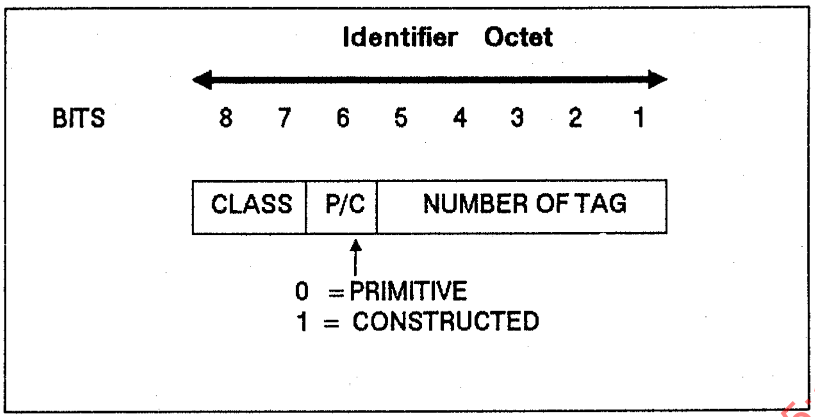

Figure 3 — Identifier octet (low tag number)

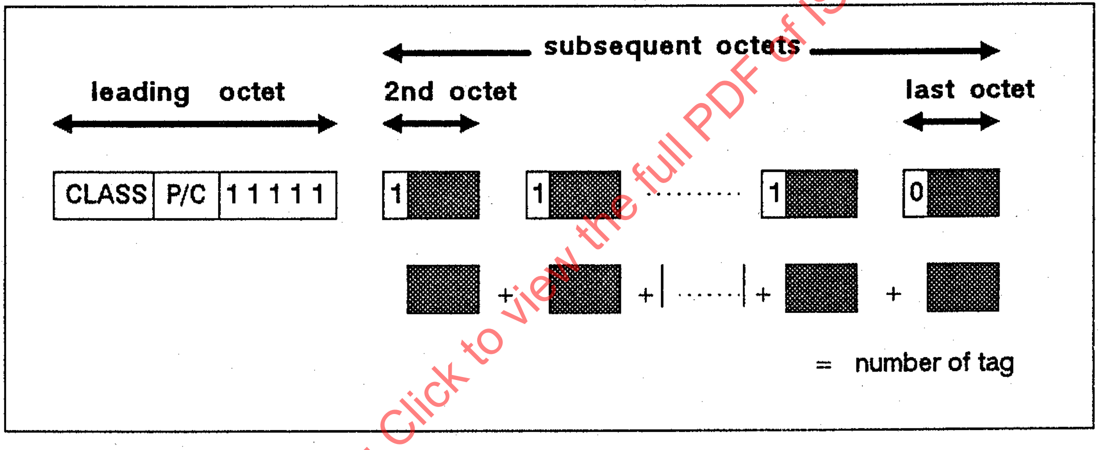

Figure 4 — Identifier octets (high tag number)

6.3.3.2 In the long form, the length octets shall consist of an initial octet and one or more subsequent octets. The initial octet shall be encoded as follows:

a) bit 8 shall be one;

b) bits 7 to 1 shall encode the number of subsequent octets in the length octets, as an unsigned binary integer with bit 7 as the most significant bit;

c) the value 11111111_2 shall not be used

NOTE — This restriction is introduced for compatibility with CCITT Recommendation X.409 and possible future extension.

Bits 8 to 1 of the first subsequent octet, followed by bits 8 to 1 of the second subsequent octet, followed in turn by bits 8 to 1 of each further octet up to and including the last subsequent octet, shall be the encoding of an unsigned binary integer equal to the number of octets in the contents octets, with bit 8 of the first subsequent octet as the most significant bit.

EXAMPLE

L=201 can be encoded as: 10000001_2
11001001_2

NOTE — In the long form, it is a sender's option whether to use more length octets than the minimum necessary.

6.3.4 For the indefinite form, the length octets indicate that the contents octets are terminated by end-of-contents octets (see 6.5), and shall consist of a single octet.

6.3.4.1 The single octet shall have bit 8 set to one, and bits 7 to 1 set to zero.

6.3.4.2 If this form of length is used, then end-of-contents octets (see 6.5) shall be present in the encoding following the contents octets.

6.4 Contents octets

The contents octets shall consist of zero, one or more octets, and shall encode the data value as specified in subsequent clauses.

NOTE — The contents octets depend on the type of the data value: subsequent clauses follow the same sequence as the definition of types in ASN.1.

6.5 End-of-contents octets

The end-of-contents octets shall be present if the length is encoded as specified in 6.3.4, otherwise they shall not be present.

The end-of-contents octets shall consist of two zero octets.

NOTE — The end-of-contents octets can be considered as the encoding of a value whose tag is universal class, whose form is primitive, whose number of the tag is zero, and whose contents is absent, thus:

```
End-of-contents Length Contents
00₁₆            00₁₆   Absent
```

7 Encoding of a boolean value

7.1 The encoding of a boolean value shall be primitive. The contents octets shall consist of a single octet.

7.2 If the boolean value is

FALSE

the octet shall be zero.

7.2.1 If the boolean value is

TRUE

the octet shall have any non-zero value, as a sender's option.

EXAMPLE - If of type BOOLEAN, the value TRUE can be encoded as:

```
Boolean Length Contents
01₁₆    01₁₆   FF₁₆
```

8 Encoding of an integer value

8.1 The encoding of an integer value shall be primitive. The contents octets shall consist of one or more octets.

8.2 If the contents octets of an integer value encoding consist of more than one octet, then the bits of the first octet and bit 8 of the second octet

a) *shall not all be ones*; and

b) shall not all be zero.

NOTE — These rules ensure that an integer value is always encoded in the smallest possible number of octets.

8.3 The contents octets shall be a twos-complement binary number equal to the integer value, and consisting of bits 8 to 1 of the first octet, followed by bits 8 to 1 of the second octet, followed by bits 8 to 1 of each octet in turn up to and including the last octet of the contents octets.

NOTE — The value of a two's complement binary number is derived by numbering the bits in the contents octets, starting with bit 1 of the last octet as bit zero and ending the numbering with bit 8 of the first octet. Each bit is assigned a numerical value of 2^N, where N is its position in the above numbering sequence. The value of the two's complement binary number is obtained by summing the numerical values assigned to each bit for those bits which are set to one, *excluding bit 8 of the first* octet, and then reducing this value by the numerical value assigned to bit 8 of the first octet if that bit is set to one.

9 Encoding of a bitstring value

9.1 The encoding of a bitstring value shall be either primitive or constructed at the option of the sender.

NOTE — Where it is necessary to transfer part of a bit string before the entire bitstring is available, the constructed encoding is used.

9.2 The contents octets for the primitive encoding shall contain an initial octet followed by zero, one or more subsequent octets.

9.2.1 The bits in the bitstring, commencing with the first bit and proceeding to the trailing bit, shall be placed in bits 8 to 1 of the first subsequent octet, followed by bits 8 to 1 of the second subsequent octet, followed by bits 8 to 1 of each octet in turn, followed by as many bits as are needed of the final subsequent octet, commencing with bit 8.

NOTE — The notation "first bit" and "trailing bit" is specified in ISO 8824.

9.2.2 The initial octet shall encode, as an unsigned binary integer with bit 1 as the least significant bit, the number of unused bits in the final subsequent octet. The number shall be in the range zero to seven.

9.2.3 If the bitstring is empty, there shall be no subsequent octets, and the initial octet shall be zero.

9.3 The contents octets for the constructed encoding shall consist of the complete encoding of zero, one or more data values.

NOTE — Each such encoding includes identifier, length, and contents octets, and may include end-of-contents octets if it is constructed.

9.3.1 Each data value encoding in the contents octets shall be the encoding of a value of type BITSTRING.

NOTE — In particular, the tags in the contents octets are always universal class, number 3.

9.3.2 The bits in the bitstring value being encoded, commencing with the first bit and proceeding to the trailing bit, shall be placed in the first bit up to the trailing bit of the first data value encoded in the contents octets, followed by the first bit up to the trailing bit of the second data value encoded in the contents octets, followed by the first bit up to the trailing bit of each data value in turn, followed by the first bit up to the trailing bit of the last data value encoded in the contents octets.

9.3.3 Each data value encoded in the contents octets, with the exception of the last, shall consist of a number of bits which is a multiple of eight.

NOTE — A data value encoded in the contents octets may be a zero-length bitstring.

9.3.4 Where a constructed encoding is used, there shall be no significance placed on the boundary between the data values encoded in the contents octets.

9.3.5 The encoding of each data value encoded in the contents octets may be primitive or constructed.

NOTE — It is usually primitive.

EXAMPLE - If of type BIT STRING, the value '0A3B5F291CD'H can be encoded as shown below. In this example, the Bit String is represented as a primitive:

BitString	Length	Contents
03_{16}	07_{16}	$040A3B5F291CD0_{16}$

The value shown above can also be encoded as shown below. In this example, the Bit String is represented as a constructor:

BitString	Length	Contents	
23_{16}	80_{16}		
	BitString	Length	Contents
	03_{16}	03_{16}	$000A3B_{16}$
	03_{16}	05_{16}	$045F291CD0_{16}$
	EOC	Length	
	00_{16}	00_{16}	

10 Encoding of an octetstring value

10.1 The encoding of an octetstring value shall be either primitive or constructed at the option of the sender.

NOTE — Where it is necessary to transfer part of an octet string before the entire octetstring is available, the constructed encoding is used.

10.2 The primitive encoding contains zero, one or more contents octets equal in value to the octets in the data value, in the order they appear in the data value, and with the most significant bit of an octet of the data value aligned with the most significant bit of an octet of the contents octets.

10.3 The contents octets for the constructed encoding shall consist of the complete encoding of zero, one or more data values.

NOTE — Each such encoding includes identifier, length, and contents octets, and may include end-of-contents octets if it is constructed.

10.3.1 Each data value encoding in the contents octets shall be the encoding of a value of type octetstring.

NOTE — In particular, the tags in the contents octets are always universal class, number 4.

10.3.2 The octets in the octetstring value being encoded, commencing with the first octet and proceeding to the trailing octet, shall be placed in the first up to the trailing octet of the first data value encoded in the contents octets, followed by the first up to the trailing octet of the second data value encoded in the contents octets, followed by the first up to the trailing octet of each data value in turn, followed by the first up to the trailing octet of the last data value encoded in the contents octets.

NOTE — A data value encoded in the contents octets may be a zero length octet string.

10.3.3 Where a constructed encoding is used, there shall be no significance placed on the boundary between the data values encoded in the contents octets.

10.3.4 The encoding of each data value encoded in the contents octets may be primitive or constructed.

NOTE — It is usually primitive.

11 Encoding of a null value

11.1 The encoding of a null value shall be primitive.

11.2 The contents octets shall not contain any octets.

NOTE — The length octet is zero.

EXAMPLE - If of type NULL, the NULL can be encoded as:

```
Null Length
05₁₆ 00₁₆
```

12 Encoding of a sequence value

12.1 The encoding of a sequence value shall be constructed.

12.2 The contents octets shall consist of the complete encoding of one data value from each of the types listed in the ASN.1 definition of the sequence type, in the order of their appearance in the definition, unless the type was referenced with the keyword "OPTIONAL" or the keyword "DEFAULT".

12.3 The encoding of a data value may, but need not, be present for a type which was referenced with the keyword "OPTIONAL" or the keyword "DEFAULT". If present, it shall appear in the encoding at the point corresponding to the appearance of the type in the ASN.1 definition.

EXAMPLE - If of type

```
SEQUENCE {name IA5String, ok BOOLEAN}
```

the value

```
    {name "Smith", ok TRUE}
```

can be encoded as:

```
Sequence Length Contents
30₁₆     0A₁₆
           IA5String Length Contents
           16₁₆      05₁₆   "Smith"
           Boolean   Length Contents
           01₁₆      01₁₆   FF₁₆
```

13 Encoding of a sequence-of value

13.1 The encoding of a sequence-of value shall be constructed.

13.2 The contents octets shall consist of zero, one or more complete encodings of data values from the type listed in the ASN.1 definition.

13.3 The order of the encodings of the data values shall be the same as the order of the data values in the sequence-of value to be encoded.

14 Encoding of a set value

14.1 The encoding of a set value shall be constructed.

14.2 The contents octets shall consist of the complete encoding of a data value from each of the types listed in the ASN.1 definition of the set type, in an order chosen by the sender, unless the type was referenced with the keyword "OPTIONAL" or the keyword "DEFAULT".

14.3 The encoding of a data value may, but need not, be present for a type which was referenced with the keyword "OPTIONAL" or the keyword "DEFAULT".

NOTE — The order of data values in a set value is not significant, and places no constraints on the order during transfer.

15 Encoding of a set-of value

15.1 The encoding of a set-of value shall be constructed.

15.2 The text of 13.2 applies.

15.3 The order of data values need not be preserved by the encoding and subsequent decoding.

16 Encoding of a choice value

The encoding of a choice value shall be the same as the encoding of a value of the chosen type.

NOTES

1 The encoding may be primitive or constructed depending on the chosen type.

2 The tag used in the identifier octets is the tag of the chosen type, as specified in the ASN.1 definition of the choice type.

17 Encoding of a selection value

The encoding of a selection value shall be the same as the encoding of a value of the selected type.

NOTE — The encoding may be primitive or constructed depending on the selected type.

18 Encoding of a tagged value

18.1 The encoding of a tagged value shall be derived from the complete encoding of the corresponding data value of the type appearing in the "TaggedType" notation (called the base encoding) as specified in 18.2 and 18.3.

18.2 If the "IMPLICIT" keyword was not used in the definition of the type, the encoding shall be constructed and the contents octets shall be the complete base encoding.

18.3 If the "IMPLICIT" keyword was used in the definition of the type, then

a) the encoding shall be constructed if the base encoding is constructed, and shall be primitive otherwise; and

b) the contents octets shall be the same as the contents octets of the base encoding.

EXAMPLE - With ASN.1 type definitions of

```
Type1 ::= VisibleString
Type2 ::= [APPLICATION 3]
              IMPLICIT Type1
Type3 ::= [2] Type2
Type4 ::= [APPLICATION 7]
              IMPLICIT Type3
Type5 ::= [2] IMPLICIT Type2
```

a value of

```
"Jones"
```

is encoded as follows:

For Type1:

VisibleString	Length	Contents
$1A_{16}$	05_{16}	$4A6F6E6573_{16}$

For Type2:

[Application 3]	Length	Contents
43_{16}	05_{16}	$4A6F6E6573_{16}$

For Type3:

[2] Length Contents
$A2_{16}$ 07_{16}
  [Application 3] Length Contents
  43_{16} 05_{16}
$4A6F6E6573_{16}$

For Type4:

[Application 7] Length Contents
67_{16} 07_{16}
  [Application 3] Length Contents
  43_{16} 05_{16}
$4A6F6E6573_{16}$

For Type5:

[2]	Length	Contents
82_{16}	05_{16}	$4A6F6E6573_{16}$

19 Encoding of a value of the ANY type

The encoding of an ANY type shall be the complete encoding specified in this International Standard for the type of the value of the ANY type.

20 Encoding of an object identifier value

20.1 The encoding of an object identifier value shall be primitive.

20.2 The contents octets shall be an (ordered) list of encodings of subidentifiers (see 20.3 and 20.4) concatenated together.

Each subidentifier is represented as a series of (one or more) octets. Bit 8 of each octet indicates whether it is the last in the series: bit 8 of the last octet is zero; bit 8 of each preceding octet is one. Bits 7-1 of the octets in the series collectively encode the subidentifier. Conceptually, these groups of bits are concatenated to form an unsigned binary number whose most significant bit is bit 7 of the first octet and whose least significant bit is bit 1 of the last octet. The subidentifier shall be encoded in the fewest possible octets, that is, the leading octet of the subidentifier shall not have the value 80 (hexadecimal).

20.3 The number of subidentifiers (N) shall be one less than the number of object identifier components in the object identifier value being encoded.

20.4 The numerical value of the first subidentifier is derived from the values of the first *two* object identifier components in the object identifier value being encoded, using the formula

```
(X*40) + Y
```

where X is the value of the first object identifier component and Y is the value of the second object identifier component.

NOTE — This packing of the first two object identifier components recognises that only three values are allocated from the root node, and at most 39 subsequent values from nodes reached by X=0 and X=1.

20.5 The numerical value of the i'th subidentifier, (2 ≤ i ≤ N) is that of the (i+1)'th object identifier component.

EXAMPLE - An OBJECT IDENTIFIER value of

```
{joint-iso-ccitt   100   3}
```

which is the same as

```
{2   100   3}
```

has a first subidentifier of 180 and a second subidentifier of 3. The resulting encoding is

```
OBJECT
IDENTIFIER Length Contents
06₁₆       03₁₆   813403₁₆
```

21 Encoding for values of the character string types

21.1 The data value consists of a string of characters from the character set specified in the ASN.1 type definition.

21.2 Each data value shall be encoded independently of other data values of the same type.

21.3 Each character string type shall be encoded as if it had been declared

```
[UNIVERSAL x] IMPLICIT OCTET STRING
```

where x is the number of the universal class tag assigned to the character string type in ISO 8824. The value of the octet string is specified in 21.4 and 21.5.

21.4 Where a character string type is specified in ISO 8824 by direct reference to an enumerating table (NumericString and PrintableString), the value of the octet string shall be that specified in 21.5 for a VisibleString type with the same character string value.

21.5 The octet string shall contain the octets specified in ISO 2022 for encodings in an 8-bit environment, using the escape sequence and character codings registered in accordance with ISO 2375.

21.5.1 An escape sequence shall not be used unless it is one of those specified by one of the registration numbers used to define the character string type in ISO 8824.

21.5.2 At the start of each string, certain registration numbers shall be assumed to be designated as G0 and/or C0 and/or C1, and invoked (using the terminology of ISO 2022). These are specified for each type in table 2, together with the assumed escape sequence they imply.

21.5.3 Certain character string types shall not contain explicit escape sequences in their encodings; in all other cases, any escape sequence allowed by 21.5.1 can appear at any time, including at the start of the encoding. Table 2 lists the types for which explicit escape sequences are allowed.

21.5.4 Announcers shall not be used unless explicitly permitted by the user of ASN.1.

NOTE — The choice of ASN.1 type provides a limited form of announcer functionality. Specific application protocols may choose to carry announcers in other protocol elements, or to specify in detail the manner of use of announcers.

EXAMPLE - With the ASN.1 type definition

```
Name ::= VisibleString
```

a value

```
"Jones"
```

can be encoded (primitive form) as

```
VisibleString Length Contents
1A₁₆          05₁₆   4A6F6E6573₁₆
```

or (constructor form, definite length) as

```
VisibleString Length Contents
3A₁₆          09₁₆
          OctetString Length Contents
          04₁₆        03₁₆   4A6F6E₁₆
          OctetString Length Contents
          04₁₆        02₁₆   6573₁₆
```

Table 2 — Use of escape sequences

Type	Assumed G0 (Registration number)	Assumed C0 & C1 (Registration number)	Assumed escape sequence(s) and locking shift (where applicable)	Explicit escape sequences allowed?
NumericString	2	None	ESC 2/8 4/0 LS0	NO
PrintableString	2	None	ESC 2/8 4/0 LS0	NO
TeletexString (T61String)	102	106 (C0) 107 (C1)	ESC 2/8 7/5 LS0 ESC 2/1 4/5 ESC 2/2 4/8	YES
VideotexString	102	1 (C0) 73 (C1)	ESC 2/8 7/5 LS0 ESC 2/1 4/0 ESC 2/2 4/1	YES
VisibleString (ISO646String)	2	None	ESC 2/8 4/0 LS0	NO
IA5String	2	1 (C0)	ESC 2/8 4/0 LS0 ESC 2/1 4/0	NO
GraphicString	2	None	ESC 2/8 4/0 LS0	YES
GeneralString	2	1 (C0)	ESC 2/8 4/0 LS0 ESC 2/1	YES

NOTE – Many of the commonly used characters (for example, A to Z) appear in a number of character repertoires with individual registration numbers and escape sequences. Where ASN.1 types allow escape sequences, a number of encodings may be possible for a particular character string. (See also 5.3)

or (constructor form, indefinite length) as

```
VisibleString Length Contents
3A16          8016
    OctetString Length Contents
    0416        0316   4A6F6E16
    OctetString Length Contents
    0416        0216   657316
    EOC         Length
    0016        0016
```

The above example illustrates three of the (many) possible forms available as a sender's option.

Receivers are required to handle all permitted forms (see 5.3).

22 Encoding for values of the ASN.1 useful types

A definition of these types using ASN.1 is provided in ISO 8824. The encoding shall be that obtained by applying the rules specified in this International Standard to that type definition.

23 Use in transfer syntax definition

23.1 The encoding rules specified in this International Standard can be referenced and applied whenever there is a need to specify an unambiguous, undivided and self-delimiting octet string representation for all of the values of a single ASN.1 type.

NOTE — All such octet strings are unambiguous within the scope of the single ASN.1 type. They would not necessarily be unambiguous if mixed with encodings of a different ASN.1 type.

23.2 The object identifier and object descriptor values

```
  {joint-iso-ccitt asn1 (1)
                     basic-encoding (1)}
```

and

```
"Basic Encoding of a single ASN.1 type"
```

are assigned to identify and describe the encoding rules specified in this International Standard.

23.3 Where an application standard defines an abstract syntax as a set of presentation data values, each of which is a value of some specifically named ASN.1 type, usually (but not necessarily) a choice type, then the object identifier value specified in 23.2 may be used with the abstract syntax name to identify that transfer syntax which results from the application of the encoding rules specified in this International Standard to the specifically named ASN.1 type used in defining the abstract syntax.

NOTE — In particular, this identification of the encoding rules can appear in the "transfer syntax name" field of the presentation protocol (ISO 8823).

23.4 The name specified in 23.2 shall not be used with an abstract syntax name to identify a transfer syntax if the conditions of 23.3 for the definition of the abstract syntax are not met.