INTERNATIONAL STANDARD

ISO/IEC 8825-1

Fifth edition
2015-11-15

Information technology — ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)

Technologies de l'information — Règles de codage ASN.1: Spécification des règles de codage de base (BER), des règles de codage canoniques (CER) et des règles de codage distinctives (DER)

Reference number
ISO/IEC 8825-1:2015(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

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.

This fifth edition cancels and replaces the fourth edition of ISO/IEC 8825-1:2008 which has been technically revised. It also incorporates ISO/IEC 8825-1:2008/Cor.1:2012 and ISO/IEC 8825-5:2008.Cor.2:2014.

ISO/IEC 8825-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*, in collaboration with ITU-T. The identical text is published as ITU-T X.690 (08/2015).

International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.690

(08/2015)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

OSI networking and system aspects – Abstract Syntax Notation One (ASN.1)

Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)

Recommendation ITU-T X.690

International Telecommunication Union

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	
Services and facilities	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalling and switching	X.50–X.89
Network aspects	X.90–X.149
Maintenance	X.150–X.179
Administrative arrangements	X.180–X.199
OPEN SYSTEMS INTERCONNECTION	
Model and notation	X.200–X.209
Service definitions	X.210–X.219
Connection-mode protocol specifications	X.220–X.229
Connectionless-mode protocol specifications	X.230–X.239
PICS proformas	X.240–X.259
Protocol Identification	X.260–X.269
Security Protocols	X.270–X.279
Layer Managed Objects	X.280–X.289
Conformance testing	X.290–X.299
INTERWORKING BETWEEN NETWORKS	
General	X.300–X.349
Satellite data transmission systems	X.350–X.369
IP-based networks	X.370–X.379
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	
Networking	X.600–X.629
Efficiency	X.630–X.639
Quality of service	X.640–X.649
Naming, Addressing and Registration	X.650–X.679
Abstract Syntax Notation One (ASN.1)	**X.680–X.699**
OSI MANAGEMENT	
Systems management framework and architecture	X.700–X.709
Management communication service and protocol	X.710–X.719
Structure of management information	X.720–X.729
Management functions and ODMA functions	X.730–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	
Commitment, concurrency and recovery	X.850–X.859
Transaction processing	X.860–X.879
Remote operations	X.880–X.889
Generic applications of ASN.1	X.890–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
INFORMATION AND NETWORK SECURITY	X.1000–X.1099
SECURE APPLICATIONS AND SERVICES	X.1100–X.1199
CYBERSPACE SECURITY	X.1200–X.1299
SECURE APPLICATIONS AND SERVICES	X.1300–X.1399
CYBERSECURITY INFORMATION EXCHANGE	X.1500–X.1599
CLOUD COMPUTING SECURITY	X.1600–X.1699

For further details, please refer to the list of ITU-T Recommendations.

INTERNATIONAL STANDARD ISO/IEC 8825-1

RECOMMENDATION ITU-T X.690

Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)

Summary

Recommendation ITU-T X.690 | ISO/IEC 8825-1 defines a set of Basic Encoding Rules (BER) that may be applied to values of types defined using the ASN.1 notation. 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 used for decoding. This Recommendation | International Standard defines also a set of Distinguished Encoding Rules (DER) and a set of Canonical Encoding Rules (CER) both of which provide constraints on the Basic Encoding Rules (BER). The key difference between them is that DER uses the definite length form of encoding while CER uses the indefinite length form. DER is more suitable for the small encoded values, while CER is more suitable for the large ones. It is implicit in the specification of these encoding rules that they are also used for decoding.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T X.690	1994-07-01	7	11.1002/1000/3046
1.1	ITU-T X.690 (1994) Technical Cor. 1	1995-11-21	7	11.1002/1000/3283
1.2	ITU-T X.690 (1994) Technical Cor. 2	1997-12-12	7	11.1002/1000/4182
1.3	ITU-T X.690 (1994) Technical Cor. 3	1997-12-12	7	11.1002/1000/4183
2.0	ITU-T X.690	1997-12-12	7	11.1002/1000/4447
2.1	ITU-T X.690 (1997) Technical Cor. 1	1999-06-18	7	11.1002/1000/4705
2.2	ITU-T X.690 (1997) Amd. 1	1999-06-18	7	11.1002/1000/4704
2.3	ITU-T X.690 (1997) Technical Cor. 2	2001-02-02	7	11.1002/1000/5335
3.0	ITU-T X.690	2002-07-14	17	11.1002/1000/6089
3.1	ITU-T X.690 (2002) Amd. 1	2003-10-29	17	11.1002/1000/7021
3.2	ITU-T X.690 (2002) Amd. 2	2006-06-13	17	11.1002/1000/8838
3.3	ITU-T X.690 (2002) Technical Cor. 1	2007-05-29	17	11.1002/1000/9108
4.0	ITU-T X.690	2008-11-13	17	11.1002/1000/9608
4.1	ITU-T X.690 (2008) Cor. 1	2011-10-14	17	11.1002/1000/11378
4.2	ITU-T X.690 (2008) Cor. 2	2014-03-01	17	11.1002/1000/12147
5.0	ITU-T X.690	2015-08-13	17	11.1002/1000/12483

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

CONTENTS

Introduction

Rec. ITU-T X.680 | ISO/IEC 8824-1, Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3, Rec. ITU-T X.683 | ISO/IEC 8824-4 (Abstract Syntax Notation One or ASN.1) together specify a notation for the definition of abstract syntaxes, enabling application standards to define the types of information they need to transfer. It also specifies a notation for the specification of values of a defined type.

This Recommendation | International Standard defines encoding rules that may be applied to values of types defined using the ASN.1 notation. 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 ASN.1 notation. This Recommendation | International Standard defines three sets of encoding rules, called *basic encoding rules*, *canonical encoding rules* and *distinguished encoding rules*. Whereas the basic encoding rules give the sender of an encoding various choices as to how data values may be encoded, the canonical and distinguished encoding rules select just one encoding from those allowed by the basic encoding rules, eliminating all of the sender's options. The canonical and distinguished encoding rules differ from each other in the set of restrictions that they place on the basic encoding rules.

The distinguished encoding rules is more suitable than the canonical encoding rules if the encoded value is small enough to fit into the available memory and there is a need to rapidly skip over some nested values. The canonical encoding rules is more suitable than the distinguished encoding rules if there is a need to encode values that are so large that they cannot readily fit into the available memory or it is necessary to encode and transmit a part of a value before the entire value is available. The basic encoding rules is more suitable than the canonical or distinguished encoding rules if the encoding contains a set value or set-of value and there is no need for the restrictions that the canonical and distinguished encoding rules impose. This is due to the memory and CPU overhead that the latter encoding rules exact in order to guarantee that set values and set-of values have just one possible encoding.

Annex A gives an example of the application of the basic encoding rules. It does not form an integral part of this Recommendation | International Standard.

Annex B summarizes the assignment of object identifier and OID internationalized resource identifier values made in this Recommendation | International Standard. It does not form an integral part of this Recommendation | International Standard.

Annex C gives examples of applying the basic encoding rules for encoding reals. It does not form an integral part of this Recommendation | International Standard.

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)

1 Scope

This Recommendation | 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 Rec. ITU-T X.680 | ISO/IEC 8824-1, Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3, and Rec. ITU-T X.683 | ISO/IEC 8824-4, collectively referred to as Abstract Syntax Notation One or ASN.1. These basic encoding rules are also to be applied for decoding such a transfer syntax in order to identify the data values being transferred. It also specifies a set of canonical and distinguished encoding rules that restrict the encoding of values to just one of the alternatives provided by the basic encoding rules.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

NOTE – This Recommendation | International Standard is based on ISO/IEC 10646:2003. It cannot be applied using later versions of this standard.

2.1 Identical Recommendations | International Standards

- Recommendation ITU-T X.680 (2015) | ISO/IEC 8824-1:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*
- Recommendation ITU-T X.681 (2015) | ISO/IEC 8824-2:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- Recommendation ITU-T X.682 (2015) | ISO/IEC 8824-3:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- Recommendation ITU-T X.683 (2015) | ISO/IEC 8824-4:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications.*

2.2 Additional references

- ISO *International Register of Coded Character Sets to be used with Escape Sequences.*
- ISO/IEC 2022:1994, *Information technology – Character code structure and extension techniques.*
- ISO/IEC 2375:2003, *Information technology – Procedure for registration of escape sequences and coded character sets*.
- ISO 6093:1985, *Information processing – Representation of numerical values in character strings for information interchange.*
- ISO/IEC 6429:1992, *Information technology – Control functions for coded character sets.*
- ISO/IEC 10646:2003, *Information technology – Universal Multiple-Octet Coded Character Set (UCS).*

3 Definitions

For the purposes of this Recommendation | International Standard, the definitions of Rec. ITU-T X.200 | ISO/IEC 7498-1 and Rec. ITU-T X.680 | ISO/IEC 8824-1 and the following definitions apply.

3.1 **canonical encoding**: A complete encoding of an abstract value obtained by the application of encoding rules that have no implementation-dependent options. Such rules result in the definition of a 1-1 mapping between unambiguous and unique encodings and values in the abstract syntax.

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

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

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

3.5 **dynamic conformance**: A statement of the requirement for an implementation to adhere to the prescribed behaviour in an instance of communication.

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

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

NOTE – Some ITU-T Recommendations use the term "data element" for this sequence of octets, but the term is not used in this Recommendation | International Standard, as other Recommendations | International Standards use it to mean "data value".

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

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

3.11 **receiver**: An implementation decoding the octets produced by a sender, in order to identify the data value which was encoded.

3.12 **sender**: An implementation encoding a data value for transfer.

3.13 **static conformance**: A statement of the requirement for support by an implementation of a valid set of features from among the defined features.

3.14 **trailing 0 bit**: A 0 in the last position of a bitstring value.

NOTE – The 0 in a bitstring value consisting of a single 0 bit is a trailing 0 bit. Its removal produces an empty bitstring.

4 Abbreviations

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

ASN.1 Abstract Syntax Notation One

BER Basic Encoding Rules of ASN.1

CER Canonical Encoding Rules of ASN.1

DER Distinguished Encoding Rules of ASN.1

ULA Upper Layer Architecture

UTF8 Universal Transformation Function 8-bit (see ISO/IEC 10646, Annex D)

5 Notation

This Recommendation | International Standard references the notation defined by Rec. ITU-T X.680 | ISO/IEC 8824-1.

6 Convention

6.1 This Recommendation | 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 specifications use the same notation to define the order of bit transmission on a serial line, or the assignment of bits to parallel channels.

6.2 For the purposes of this Recommendation | 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".

6.3 For the purpose of this Recommendation | International Standard, two octet strings can be compared. One octet string is equal to another if they are of the same length and are the same at each octet position. An octet string, S_1, is greater than another, S_2, if and only if either:

a) S_1 and S_2 have identical octets in every position up to and including the final octet in S_2, but S_1 is longer; or

b) S_1 and S_2 have different octets in one or more positions, and in the first such position, the octet in S_1 is greater than that in S_2, considering the octets as unsigned binary numbers whose bit n has weight 2^{n-1}.

7 Conformance

7.1 Dynamic conformance is specified by clauses 8 to 12 inclusive.

7.2 Static conformance is specified by those standards which specify the application of one or more of these encoding rules.

7.3 Alternative encodings are permitted by the basic encoding rules as a sender's option. Receivers who claim conformance to the basic encoding rules shall support all alternatives.

NOTE – Examples of such alternative encodings appear in 8.1.3.2 b) and Table 3.

7.4 No alternative encodings are permitted by the Canonical Encoding Rules or Distinguished Encoding Rules.

8 Basic encoding rules

8.1 General rules for encoding

8.1.1 Structure of an encoding

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

a) identifier octets (see 8.1.2);

b) length octets (see 8.1.3);

c) contents octets (see 8.1.4);

d) end-of-contents octets (see 8.1.5).

8.1.1.2 The end-of-contents octets shall not be present unless the value of the length octets requires them to be present (see 8.1.3).

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

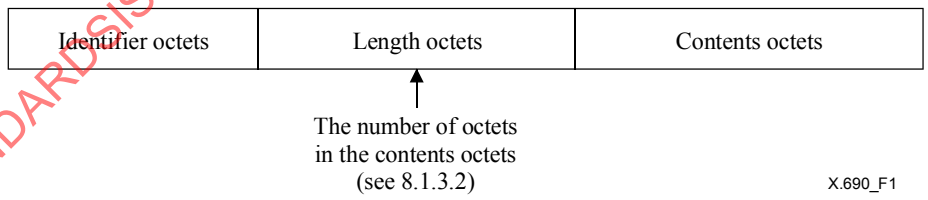

Figure 1 – Structure of an encoding

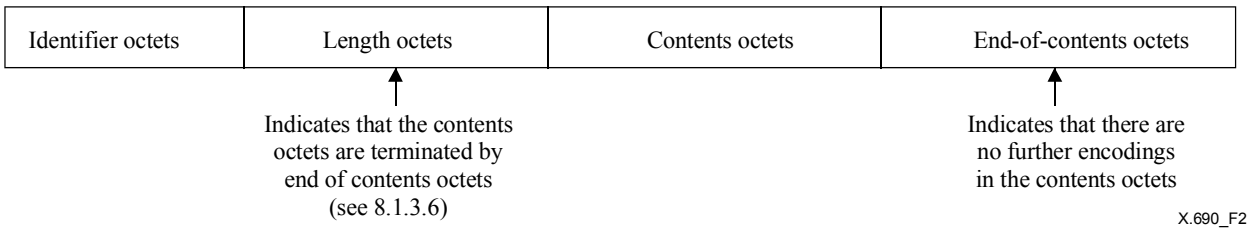

Figure 2 – An alternative constructed encoding

8.1.1.4 Encodings specified in this Recommendation | International Standard are not affected by either the ASN.1 subtype notation or the ASN.1 type extensibility notation.

NOTE – This means that all constraint notation is ignored when determining encodings, and all extensibility markers in **CHOICE**, **SEQUENCE** and **SET** are ignored, with the extensions treated as if they were in the extension root of the type.

8.1.1.5 There are no encoding instructions (see Rec. ITU-T X.680 | ISO/IEC 8824-1, 3.8.27) defined for the encoding rules specified in this Recommendation | International Standard.

8.1.2 Identifier octets

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

8.1.2.2 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;

b) bit 6 shall be a zero or a one according to the rules of 8.1.2.5;

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

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

8.1.2.3 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).

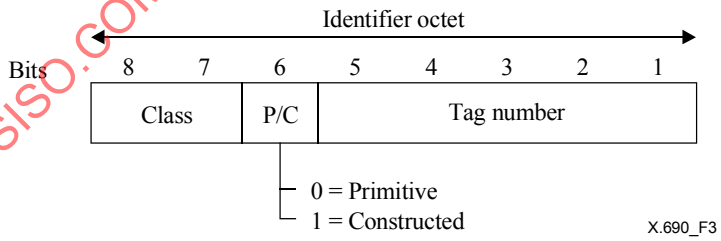

Figure 3 – Identifier octet (low tag number)

8.1.2.4 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.

8.1.2.4.1 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 8.1.2.5;

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

8.1.2.4.2 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.

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

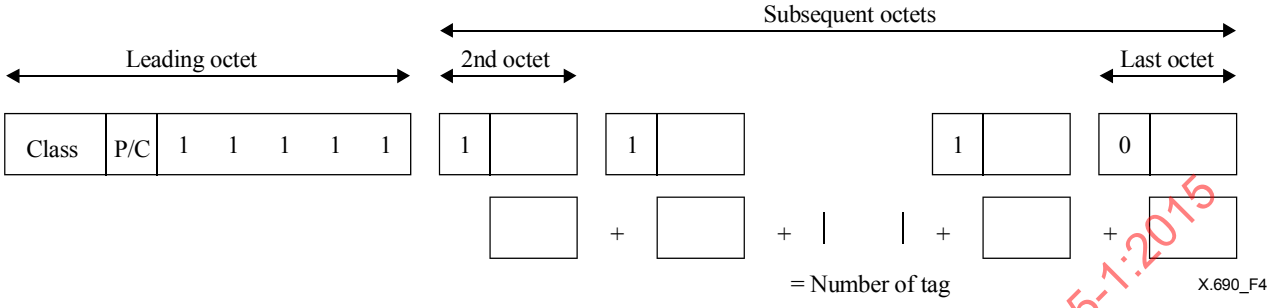

Figure 4 – Identifier octets (high tag number)

8.1.2.5 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 subclauses specify whether the encoding is primitive or constructed for each type.

8.1.2.6 Rec. ITU-T X.680 | ISO/IEC 8824-1 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.

8.1.2.7 Rec. ITU-T X.681 | ISO/IEC 8824-2, 14.2 and 14.4, specifies that the tag of a type defined using "ObjectClassFieldType" is indeterminate if it is a type field, a variable-type value field, or a variable-type value set field. This 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).

8.1.3 Length octets

8.1.3.1 Two forms of length octets are specified. These are:

a) the definite form (see 8.1.3.3); and

b) the indefinite form (see 8.1.3.6).

8.1.3.2 A sender shall:

a) use the definite form (see 8.1.3.3) if the encoding is primitive;

b) use either the definite form (see 8.1.3.3) or the indefinite form (see 8.1.3.6), a sender's option, if the encoding is constructed and all immediately available;

c) use the indefinite form (see 8.1.3.6) if the encoding is constructed and is not all immediately available.

8.1.3.3 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 (see 8.1.3.4) or the long form (see 8.1.3.5) 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.

8.1.3.4 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

8.1.3.5 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 1 – This restriction is introduced for 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 2 – In the long form, it is a sender's option whether to use more length octets than the minimum necessary.

8.1.3.6 For the indefinite form, the length octets indicate that the contents octets are terminated by end-of-contents octets (see 8.1.5), and shall consist of a single octet.

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

8.1.3.6.2 If this form of length is used, then end-of-contents octets (see 8.1.5) shall be present in the encoding following the contents octets.

8.1.4 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.

8.1.5 End-of-contents octets

The end-of-contents octets shall be present if the length is encoded as specified in 8.1.3.6, 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 are absent, thus:

End-of-contents	Length	Contents
00_{16}	00_{16}	Absent

8.2 Encoding of a boolean value

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

8.2.2 If the boolean value is:

FALSE

the octet shall be zero.

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
01_{16}	01_{16}	FF_{16}

8.3 Encoding of an integer value

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

8.3.2 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.

8.3.3 The contents octets shall be a two's 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.

8.4 Encoding of an enumerated value

The encoding of an enumerated value shall be that of the integer value with which it is associated.

NOTE – It is primitive.

8.5 Encoding of a real value

8.5.1 The encoding of a real value shall be primitive.

8.5.2 If the real value is the value plus zero, there shall be no contents octets in the encoding.

8.5.3 If the real value is the value minus zero, then it shall be encoded as specified in 8.5.9.

8.5.4 For a non-zero real value, if the base of the abstract value is 10, then the base of the encoded value shall be 10, and if the base of the abstract value is 2 the base of the encoded value shall be 2, 8 or 16 as a sender's option.

8.5.5 If the real value is non-zero, then the base used for the encoding shall be B' as specified in 8.5.4. If B' is 2, 8 or 16, a binary encoding, specified in 8.5.7, shall be used. If B' is 10, a character encoding, specified in 8.5.8, shall be used.

8.5.6 Bit 8 of the first contents octet shall be set as follows:

a) if bit 8 = 1, then the binary encoding specified in 8.5.7 applies;

b) if bit 8 = 0 and bit 7 = 0, then the decimal encoding specified in 8.5.8 applies;

c) if bit 8 = 0 and bit 7 = 1, then either a "SpecialRealValue" (see Rec. ITU-T X.680 | ISO/IEC 8824-1) or the value minus zero is encoded as specified in 8.5.9.

8.5.7 When binary encoding is used (bit 8 = 1), then if the mantissa M is non-zero, it shall be represented by a sign S, a positive integer value N and a binary scaling factor F, such that:

$$M = S \times N \times 2^F$$

$$0 \leq F < 4$$

$$S = +1 \text{ or } -1$$

NOTE – The binary scaling factor F is required under certain circumstances in order to align the implied point of the mantissa to the position required by the encoding rules of this subclause. This alignment cannot always be achieved by modification of the exponent E. If the base B' used for encoding is 8 or 16, the implied point can only be moved in steps of 3 or 4 bits, respectively, by changing the component E. Therefore, values of the binary scaling factor F other than zero may be required in order to move the implied point to the required position.

8.5.7.1 Bit 7 of the first contents octets shall be 1 if S is –1 and 0 otherwise.

8.5.7.2 Bits 6 to 5 of the first contents octets shall encode the value of the base B' as follows:

Bits 6 to 5	*Base*
00	base 2
01	base 8
10	base 16
11	Reserved for further editions of this Recommendation | International Standard.

8.5.7.3 Bits 4 to 3 of the first contents octet shall encode the value of the binary scaling factor F as an unsigned binary integer.

8.5.7.4 Bits 2 to 1 of the first contents octet shall encode the format of the exponent as follows:

a) if bits 2 to 1 are 00, then the second contents octet encodes the value of the exponent as a two's complement binary number;

b) if bits 2 to 1 are 01, then the second and third contents octets encode the value of the exponent as a two's complement binary number;

c) if bits 2 to 1 are 10, then the second, third and fourth contents octets encode the value of the exponent as a two's complement binary number;

d) if bits 2 to 1 are 11, then the second contents octet encodes the number of octets, X say, (as an unsigned binary number) used to encode the value of the exponent, and the third up to the (X plus 3)th (inclusive) contents octets encode the value of the exponent as a two's complement binary number; the value of X shall be at least one; the first nine bits of the transmitted exponent shall not be all zeros or all ones.

8.5.7.5 The remaining contents octets encode the value of the integer N (see 8.5.7) as an unsigned binary number.

NOTE 1 – For non-canonical BER there is no requirement for floating point normalization of the mantissa. This allows an implementer to transmit octets containing the mantissa without performing shift functions on the mantissa in memory. In the Canonical Encoding Rules and the Distinguished Encoding Rules normalization is specified and the mantissa (unless it is 0) needs to be repeatedly shifted until the least significant bit is a 1.

NOTE 2 – This representation of real numbers is very different from the formats normally used in floating point hardware, but has been designed to be easily converted to and from such formats (see Annex C).

8.5.8 When decimal encoding is used (bits 8 to 7 = 00), all the contents octets following the first contents octet form a field, as the term is used in ISO 6093, of a length chosen by the sender, and encoded according to ISO 6093. The choice of ISO 6093 number representation is specified by bits 6 to 1 of the first contents octet as follows:

Bits 6 to 1	*Number representation*
00 0001	ISO 6093 NR1 form
00 0010	ISO 6093 NR2 form
00 0011	ISO 6093 NR3 form

The remaining values of bits 6 to 1 are reserved for further editions of this Recommendation | International Standard.

There shall be no use of scaling factors specified in accompanying documentation (see ISO 6093).

NOTE 1 – The recommendations in ISO 6093 concerning the use of at least one digit to the left of the decimal mark are also recommended in this Recommendation | International Standard, but are not mandatory.

NOTE 2 – Use of the normalized form (see ISO 6093) is a sender's option, and has no significance.

8.5.9 When "SpecialRealValues" or minus zero are to be encoded (bits 8 to 7 = 01), there shall be only one contents octet, with values as follows:

01000000	Value is **`PLUS-INFINITY`**
01000001	Value is **`MINUS-INFINITY`**
01000010	Value is **`NOT-A-NUMBER`**
01000011	Value is minus zero

All other values having bits 8 and 7 equal to 0 and 1 respectively are reserved for addenda to this Recommendation | International Standard.

8.6 Encoding of a bitstring value

8.6.1 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.

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

8.6.2.1 The bits in the bitstring value, commencing with the leading 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 terms "leading bit" and "trailing bit" are defined in Rec. ITU-T X.680 | ISO/IEC 8824-1, 22.2.

8.6.2.2 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.

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

8.6.2.4 Where Rec. ITU-T X.680 | ISO/IEC 8824-1, 22.7, applies a BER encoder/decoder can add or remove trailing 0 bits from the value.

NOTE – If a bitstring value has no 1 bits, then an encoder (as a sender's option) may encode the value with a length of 1 and with an initial octet set to 0 or may encode it as a bit string with one or more 0 bits following the initial octet.

8.6.3 The contents octets for the constructed encoding shall consist of zero, one, or more nested encodings.

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

8.6.4 To encode a bitstring value in this way, it is segmented. Each segment shall consist of a series of consecutive bits of the value, and with the possible exception of the last, shall contain a number of bits which is a multiple of eight. Each bit in the overall value shall be in precisely one segment, but there shall be no significance placed on the segment boundaries.

NOTE – A segment may be of size zero, i.e. contain no bits.

8.6.4.1 Each encoding in the contents octets shall represent a segment of the overall bitstring, the encoding arising from a recursive application of this subclause. In this recursive application, each segment is treated as if it were a bitstring value. The encodings of the segments shall appear in the contents octets in the order in which their bits appear in the overall value.

NOTE 1 – As a consequence of this recursion, each encoding in the contents octets may itself be primitive or constructed. However, such encodings will usually be primitive.

NOTE 2 – In particular, the tags in the contents octets are always universal class, number 3.

8.6.4.2 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}

8.7 Encoding of an octetstring value

8.7.1 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.

8.7.2 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.

8.7.3 The contents octets for the constructed encoding shall consist of zero, one, or more encodings.

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

8.7.3.1 To encode an octetstring value in this way, it is segmented. Each segment shall consist of a series of consecutive octets of the value. There shall be no significance placed on the segment boundaries.

NOTE – A segment may be of size zero, i.e. contain no octets.

8.7.3.2 Each encoding in the contents octets shall represent a segment of the overall octetstring, the encoding arising from a recursive application of this subclause. In this recursive application, each segment is treated as if it were

an octetstring value. The encodings of the segments shall appear in the contents octets in the order in which their octets appear in the overall value.

NOTE 1 – As a consequence of this recursion, each encoding in the contents octets may itself be primitive or constructed. However, such encodings will usually be primitive.

NOTE 2 – In particular, the tags in the contents octets are always universal class, number 4.

8.8 Encoding of a null value

8.8.1 The encoding of a null value shall be primitive.

8.8.2 The contents octets shall not contain any octets.

NOTE – The length octet is zero.

EXAMPLE

If of type **NULL**, the **NULL** value can be encoded as:

Null	*Length*
05_{16}	00_{16}

8.9 Encoding of a sequence value

8.9.1 The encoding of a sequence value shall be constructed.

8.9.2 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**.

8.9.3 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		
30_{16}	$0A_{16}$			
		IA5String	Length	Contents
		16_{16}	05_{16}	"Smith"
		Boolean	Length	Contents
		01_{16}	01_{16}	FF_{16}

8.10 Encoding of a sequence-of value

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

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

8.10.3 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.

8.11 Encoding of a set value

8.11.1 The encoding of a set value shall be constructed.

8.11.2 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**.

8.11.3 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.

8.12 Encoding of a set-of value

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

8.12.2 The text of 8.10.2 applies.

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

8.13 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.

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

NOTE 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.

8.14 Encoding of a value of a prefixed type

8.14.1 If the prefixed type is an "EncodingPrefixedType", then the encoding is that of the "Type" in the "EncodingPrefixedType". If the prefixed type is a "TaggedType", then the following subclauses apply.

8.14.2 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 8.14.3 and 8.14.4.

8.14.3 If implicit tagging (see Rec. ITU-T X.680 | ISO/IEC 8824-1, 31.2.7) was not used in the definition of the type, the encoding shall be constructed and the contents octets shall be the complete base encoding.

8.14.4 If implicit tagging 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 (in an explicit tagging environment) 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}	$4A6F6E6573_{16}$

8.15 Encoding of an open type

The value of an open type is also a value of some (other) ASN.1 type. The encoding of such a value shall be the complete encoding herein specified for the value considered as being of that other type.

8.16 Encoding of an instance-of value

8.16.1 The encoding of the instance-of type shall be the BER encoding of the following sequence type with the value as specified in 8.16.2:

```
[UNIVERSAL 8] IMPLICIT SEQUENCE {
                type-id     <DefinedObjectClass>.&id,
                value [0] EXPLICIT <DefinedObjectClass>.&Type
            }
```

where "<DefinedObjectClass>" is replaced by the particular "DefinedObjectClass" used in the "InstanceOfType" notation.

NOTE – When the value is a value of a single ASN.1 type and BER encoding is used for it, the encoding of this type is identical to an encoding of a corresponding value of the external type, where the **syntax** alternative is in use for representing the abstract value.

8.16.2 The value of the components of the sequence type in 8.16.1 shall be the same as the values of the corresponding components of the associated type in Rec. ITU-T X.681 | ISO/IEC 8824-2, C.7.

8.17 Encoding of a value of the embedded-pdv type

8.17.1 The encoding of a value of the embedded-pdv type shall be the BER encoding of the type as defined in 36.5 of Rec. ITU-T X.680 | ISO/IEC 8824-1.

8.17.2 The contents of the **data-value OCTET STRING** shall be the encoding of the abstract data value of the embedded-pdv type [see 36.3 a) in Rec. ITU-T X.680 | ISO/IEC 8824-1] using the identified transfer syntax, and the value of all other fields shall be the same as the values appearing in the abstract value.

8.18 Encoding of a value of the external type

8.18.1 The encoding of a value of the external type shall be the BER encoding of the following sequence type, assumed to be defined in an environment of **EXPLICIT TAGS**, with a value as specified in the subclauses below:

```
[UNIVERSAL 8] IMPLICIT SEQUENCE {
                direct-reference            OBJECT IDENTIFIER OPTIONAL,
                indirect-reference          INTEGER OPTIONAL,
                data-value-descriptor       ObjectDescriptor OPTIONAL,
                encoding                    CHOICE {
                single-ASN1-type                  [0] ABSTRACT-SYNTAX.&Type,
                octet-aligned                      [1] IMPLICIT OCTET STRING,
                arbitrary                          [2] IMPLICIT BIT STRING } }
```

NOTE – This sequence type differs from that in Rec. ITU-T X.680 | ISO/IEC 8824-1 for historical reasons.

8.18.2 The value of the fields depends on the abstract value being transmitted, which is a value of the type specified in 36.5 of Rec. ITU-T X.680 | ISO/IEC 8824-1.

8.18.3 The **data-value-descriptor** above shall be present if and only if the **data-value-descriptor** is present in the abstract value, and shall have the same value.

8.18.4 Values of **direct-reference** and **indirect-reference** above shall be present or absent in accordance with Table 2. Table 2 maps the external type alternatives of **identification** defined in Rec. ITU-T X.680 | ISO/IEC 8824-1, 36.5, to the external type components **direct-reference** and **indirect-reference** defined in 8.18.1.

Table 2 – Alternative encodings for "identification"

identification	direct-reference	indirect-reference
syntaxes	*** CANNOT OCCUR ***	*** CANNOT OCCUR ***
syntax	syntax	ABSENT
presentation-context-id	ABSENT	presentation-context-id
context-negotiation	transfer-syntax	presentation-context-id
transfer-syntax	*** CANNOT OCCUR ***	*** CANNOT OCCUR ***
fixed	*** CANNOT OCCUR ***	*** CANNOT OCCUR ***

8.18.5 The data value shall be encoded according to the transfer syntax identified by the encoding, and shall be placed in an alternative of the **`encoding`** choice as specified below.

8.18.6 If the data value is the value of a single ASN.1 data type, and if the encoding rules for this data value are one of those specified in this Recommendation | International Standard, then the sending implementation shall use any of the **`encoding`** choices:

– **`single-ASN1-type`**;

– **`octet-aligned`**;

– **`arbitrary`**.

as an implementation option.

8.18.7 If the encoding of the data value, using the agreed or negotiated encoding, is an integral number of octets, then the sending implementation shall use any of the **`encoding`** choices:

– **`octet-aligned`**;

– **`arbitrary`**.

as an implementation option.

NOTE – A data value which is a series of ASN.1 types, and for which the transfer syntax specifies simple concatenation of the octet strings produced by applying the ASN.1 Basic Encoding Rules to each ASN.1 type, falls into this category, not that of 8.18.6.

8.18.8 If the encoding of the data value, using the agreed or negotiated encoding, is not an integral number of octets, the **`encoding`** choice shall be:

– **`arbitrary`**.

8.18.9 If the **`encoding`** choice is chosen as **`single-ASN1-type`**, then the ASN.1 type shall replace the open type, with a value equal to the data value to be encoded.

NOTE – The range of values which might occur in the open type is determined by the registration of the object identifier value associated with the **`direct-reference`**, and/or the integer value associated with the **`indirect-reference`**.

8.18.10 If the **`encoding`** choice is chosen as **`octet-aligned`**, then the data value shall be encoded according to the agreed or negotiated transfer syntax, and the resulting octets shall form the value of the octetstring.

8.18.11 If the **`encoding`** choice is chosen as **`arbitrary`**, then the data value shall be encoded according to the agreed or negotiated transfer syntax, and the result shall form the value of the bitstring.

8.19 Encoding of an object identifier value

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

8.19.2 The contents octets shall be an (ordered) list of encodings of subidentifiers (see 8.19.3 and 8.19.4) 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 to 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_{16}.

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

8.19.4 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 recognizes 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.

8.19.5 The numerical value of the i*th* subidentifier, ($2 \le i \le N$) is that of the (i + 1)*th* object identifier component.

EXAMPLE

An **OBJECT IDENTIFIER** value of:

{joint-iso-itu-t 999 3}

which is the same as:

{2 999 3}

has a first subidentifier of 1079 and a second subidentifier of 3. The resulting encoding is:

OBJECT IDENTIFIER	Length	Contents
06_{16}	03_{16}	883703_{16}

8.20 Encoding of a relative object identifier value

NOTE – The encoding of the object identifier components in a relative object identifier is the same as the encoding of components (after the second) in an object identifier.

8.20.1 The encoding of a relative object identifier value shall be primitive.

8.20.2 The contents octets shall be an (ordered) list of encodings of sub-identifiers (see 8.20.3 and 8.20.4) concatenated together. Each sub-identifier 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 sub-identifier. 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 sub-identifier shall be encoded in the fewest possible octets, that is, the leading octet of the sub-identifier shall not have the value 80_{16}.

8.20.3 The number of sub-identifiers (N) shall be equal to the number of object identifier arcs in the relative object identifier value being encoded.

8.20.4 The numerical value of the ith sub-identifier ($1 \le i \le N$) is that of the ith object identifier arc in the relative object identifier value being encoded.

8.20.5 EXAMPLE – A relative object identifier value of:

{8571 3 2}

has sub-identifiers of 8571, 3, and 2. The resulting encoding is:

RELATIVE OID	Length	Contents
$0D_{16}$	04_{16}	$C27B0302_{16}$

8.21 Encoding of an OID internationalized resource identifier value

8.21.1 The encoding of an OID internationalized resource identifier value shall be primitive.

8.21.2 The contents octets shall be the UTF8 encoding (see ISO/IEC 10646, Annex D) of the characters in the lexical items in the XML value notation (see Rec. ITU-T X.680 | ISO/IEC 8824-1, 34.3) for the OID internationalized resource identifier type, with no white-space between the encoding of lexical items. Announcers and escape sequences shall not be used, and each character shall be encoded in the smallest number of octets available for that character.

8.22 Encoding of a relative OID internationalized resource identifier value

8.22.1 The encoding of a relative OID internationalized resource identifier value shall be primitive.

8.22.2 The contents octets shall be the UTF8 encoding for the characters of the lexical items in the XML value notation (see Rec. ITU-T X.680 | ISO/IEC 8824-1, 35.3) for the relative OID internationalized resource identifier type, with no white-space between the encoding of lexical items.

8.23 Encoding for values of the restricted character string types

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

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

8.23.3 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 Rec. ITU-T X.680 | ISO/IEC 8824-1. The value of the octet string is specified in 8.23.4 and 8.23.5.

8.23.4 Where a character string type is specified in Rec. ITU-T X.680 | ISO/IEC 8824-1 by direct reference to an enumerating table (**`NumericString`** and **`PrintableString`**), the value of the octet string shall be that specified in 8.23.5 for a **`VisibleString`** type with the same character string value.

8.23.5 For restricted character strings apart from **`UniversalString`**, **`UTF8String`** and **`BMPString`**, the octet string shall contain the octets specified in ISO/IEC 2022 for encodings in an 8-bit environment, using the escape sequence and character codings registered in accordance with ISO/IEC 2375.

8.23.5.1 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 Rec. ITU-T X.680 | ISO/IEC 8824-1.

8.23.5.2 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/IEC 2022). These are specified for each type in Table 3, together with the assumed escape sequence they imply.

Table 3 – 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`	6	None	ESC 2/8 4/2 LS0	No
`PrintableString`	6	None	ESC 2/8 4/2 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)`**	6	None	ESC 2/8 4/2 LS0	No
`IA5String`	6	1 (C0)	ESC 2/8 4/2 LS0 ESC 2/1 4/0	No
`GraphicString`	6	None	ESC 2/8 4/2 LS0	Yes
`GeneralString`	6	1 (C0)	ESC 2/8 4/2 LS0 ESC 2/1 4/0	Yes
NOTE – Many of the commonly used characters (for example, A-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 7.3).				

8.23.5.3 Certain character string types shall not contain explicit escape sequences in their encodings; in all other cases, any escape sequence allowed by 8.23.5.1 can appear at any time, including at the start of the encoding. Table 3 lists the types for which explicit escape sequences are allowed.

8.23.5.4 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
$1A_{16}$	05_{16}	$4A6F6E6573_{16}$

or (constructor form, definite length) as:

VisibleString	Length	Contents		
$3A_{16}$	09_{16}			
		OctetString	Length	Contents
		04_{16}	03_{16}	$4A6F6E_{16}$
		OctetString	Length	Contents
		04_{16}	02_{16}	6573_{16}

or (constructor form, indefinite length) as:

VisibleString	Length	Contents		
$3A_{16}$	80_{16}			
		OctetString	Length	Contents
		04_{16}	03_{16}	$4A6F6E_{16}$
		OctetString	Length	Contents
		04_{16}	02_{16}	6573_{16}
		EOC	Length	
		00_{16}	00_{16}	

8.23.6 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 7.3).

8.23.7 For the **`UniversalString`** type, the octet string shall contain the octets specified in ISO/IEC 10646, using the 4-octet canonical form (see 13.2 of ISO/IEC 10646). Signatures shall not be used. Control functions may be used provided they satisfy the restrictions imposed by 8.23.9.

8.23.8 For the **`BMPString`** type, the octet string shall contain the octets specified in ISO/IEC 10646, using the 2-octet BMP form (see 13.1 of ISO/IEC 10646). Signatures shall not be used. Control functions may be used provided they satisfy the restrictions imposed by 8.23.9.

8.23.9 The C0 and C1 control functions of ISO/IEC 6429 may be used with the following exceptions.

NOTE 1 – The effect of this subclause is to allow the useful control functions such as LF, CR, TAB, etc., while forbidding the use of escapes to other character sets.

NOTE 2 – The C0 and C1 control functions are each encoded in two octets for BMPString and four for UniversalString.

a) Announcer escape sequences defined in ISO/IEC 2022 shall not be used.

NOTE 3 – The assumed character coding environment is ISO/IEC 10646.

b) Designating or identifying escape sequences defined in ISO/IEC 2022 shall not be used, including the identifying escape sequences permitted by ISO/IEC 10646, 17.2 and 17.4.

NOTE 4 – ASN.1 allows the use of the PermittedAlphabet subtype notation to select the set of allowed characters. PermittedAlphabet is also used to select the level of implementation of ISO/IEC 10646. **`BMPString`** is always used for the two-octet form and **`UniversalString`** for the four-octet form.

c) Invoking escape sequence or control sequences of ISO/IEC 2022 shall not be used, such as SHIFT IN (SI), SHIFT OUT (SO), or LOCKING SHIFT FOR G3 (SS3)

d) The coding shall conform to ISO/IEC 10646 and remain in that code set.

e) Control sequences for identifying subsets of graphic characters according to ISO/IEC 10646, 16.3, shall not be used.

NOTE 5 – ASN.1 applications use subtyping to indicate subsets of the graphic characters of ISO/IEC 10646 and to select the ISO/IEC 10646 cells that correspond to the control characters of ISO/IEC 6429.

f) The escape sequences of ISO/IEC 10646, 16.5, shall not be used to switch to ISO/IEC 2022 codes.

8.23.10 For the **UTF8String** type, the octet string shall contain the octets specified in ISO/IEC 10646, Annex D. Announcers and escape sequences shall not be used, and each character shall be encoded in the smallest number of octets available for that character.

8.24 Encoding for values of the unrestricted character string type

8.24.1 The encoding of a value of the unrestricted character string type shall be the BER encoding of the type as defined in 44.5 of Rec. ITU-T X.680 | ISO/IEC 8824-1.

8.24.2 The contents of the **string-value OCTET STRING** shall be the encoding of the abstract character string value of the unrestricted character string type [see 44.3 a) of Rec. ITU-T X.680 | ISO/IEC 8824-1] using the identified character transfer syntax, and the value of all other fields shall be the same as the values appearing in the abstract value.

8.25 Encoding for values of the useful types

The following "useful types" shall be encoded as if they had been replaced by their definitions given in clauses 46-48 of Rec. ITU-T X.680 | ISO/IEC 8824-1:

– generalized time;

– universal time;

– object descriptor.

8.26 Encoding for values of the TIME type and the useful time types

8.26.1 Encoding for values of the TIME type

NOTE – The defined time types are subtypes of the **TIME** type, with the same tag, and have the same encoding as the **TIME** type.

8.26.1.1 The encoding of the **TIME** type shall be primitive.

8.26.1.2 The contents octets shall be the UTF-8 encoding of the value notation, after the removal of initial and final QUOTATION MARK (34) characters.

8.26.2 Encoding for values of the DATE type

8.26.2.1 The encoding of the **DATE** type shall be primitive.

8.26.2.2 The contents octets shall be the UTF-8 encoding of the value notation, after the removal of initial and final QUOTATION MARK (34) characters and all HYPHEN-MINUS (45) characters.

8.26.3 Encoding for values of the TIME-OF-DAY type

8.26.3.1 The encoding of the **TIME-OF-DAY** type shall be primitive.

8.26.3.2 The contents octets shall be the UTF-8 encoding of the value notation, after the removal of initial and final QUOTATION MARK (34) characters and all COLON (58) characters.

8.26.4 Encoding for values of the DATE-TIME type

8.26.4.1 The encoding of the **DATE-TIME** type shall be primitive.

8.26.4.2 The contents octets shall be the UTF-8 encoding of the value notation, after the removal of initial and final QUOTATION MARK (34) characters, all HYPHEN-MINUS (45) characters, all COLON (58) characters, and the LATIN CAPITAL LETTER T character.

8.26.5 Encoding for values of the DURATION type

8.26.5.1 The encoding of the **DURATION** type shall be primitive.

8.26.5.2 The contents octets shall be the UTF-8 encoding of the value notation, after the removal of initial and final QUOTATION MARK (34) characters and the LATIN CAPITAL LETTER P character.

9 Canonical encoding rules

The encoding of a data values employed by the canonical encoding rules is the basic encoding described in clause 8, together with the following restrictions and those also listed in clause 11.

9.1 Length forms

If the encoding is constructed, it shall employ the indefinite length form. If the encoding is primitive, it shall include the fewest length octets necessary. [Contrast with 8.1.3.2 b).]

9.2 String encoding forms

Bitstring, octetstring, and restricted character string values shall be encoded with a primitive encoding if they would require no more than 1000 contents octets, and as a constructed encoding otherwise. The string fragments contained in the constructed encoding shall be encoded with a primitive encoding. The encoding of each fragment, except possibly the last, shall have 1000 contents octets. (Contrast with 8.23.6.) The last fragment shall have at least one, and no more than 1000, contents octets.

9.3 Set components

The encodings of the component values of a set value shall appear in an order determined by their tags as specified in 8.6 of Rec. ITU-T X.680 | ISO/IEC 8824-1. Additionally, for the purposes of determining the order in which components are encoded when one or more component is an untagged choice type, each untagged choice type is ordered as though it has a tag equal to that of the smallest tag in that choice type or any untagged choice types nested within.

EXAMPLE

In the following which assumes a tagging environment of **IMPLICIT TAGS**:

```
A ::= SET
        {
        a       [3] INTEGER,
        b       [1] CHOICE
                    {
                    c       [2] INTEGER,
                    d       [4] INTEGER
                    },
        e       CHOICE
                    {
                    f       CHOICE
                            {
                            g   [5] INTEGER,
                            h   [6] INTEGER
                            },
                    i       CHOICE
                            {
                            j       [0] INTEGER
                            }
                    }
        }
```

the order in which the components of the set are encoded will always be e, b, a, since the tag **[0]** sorts lowest, then **[1]**, then **[3]**.

10 Distinguished encoding rules

The encoding of a data values employed by the distinguished encoding rules is the basic encoding described in clause 8, together with the following restrictions and those also listed in clause 11.

10.1 Length forms

The definite form of length encoding shall be used, encoded in the minimum number of octets. [Contrast with 8.1.3.2 b).]

10.2 String encoding forms

For bitstring, octetstring and restricted character string types, the constructed form of encoding shall not be used. (Contrast with 8.23.6.)

10.3 Set components

The encodings of the component values of a set value shall appear in an order determined by their tags as specified in 8.6 of Rec. ITU-T X.680 | ISO/IEC 8824-1.

NOTE – Where a component of the set is an untagged choice type, the location of that component in the ordering will depend on the tag of the choice component being encoded.

11 Restrictions on BER employed by both CER and DER

References in clause 8 and its subclauses to "shall be the BER encoding" shall be interpreted as "shall be the CER or DER encoding, as appropriate". (See 8.16.1, 8.17.1, 8.18.1 and 8.24.1.)

11.1 Boolean values

If the encoding represents the boolean value **`TRUE`**, its single contents octet shall have all eight bits set to one. (Contrast with 8.2.2.)

11.2 Unused bits

11.2.1 Each unused bit in the final octet of the encoding of a bit string value shall be set to zero.

11.2.2 Where Rec. ITU-T X.680 | ISO/IEC 8824-1, 22.7, applies, the bitstring shall have all trailing 0 bits removed before it is encoded.

NOTE 1 – In the case where a size constraint has been applied, the abstract value delivered by a decoder to the application will be one of those satisfying the size constraint and differing from the transmitted value only in the number of trailing 0 bits.

NOTE 2 – If a bitstring value has no 1 bits, then an encoder shall encode the value with a length of 1 and an initial octet set to 0.

11.3 Real values

11.3.1 If the encoding represents a real value whose base B is 2, then binary encoding employing base 2 shall be used. Before encoding, the mantissa M and exponent E are chosen so that M is either 0 or is odd.

NOTE – This is necessary because the same real value can be regarded as both {M, 2, E} and {M', 2, E'} with $M \neq M'$ if, for some non-zero integer n:

$$M' = M \times 2^{-n}$$
$$E' = E + n$$

In encoding the value, the binary scaling factor F shall be zero, and M and E shall each be represented in the fewest octets necessary.

11.3.2 If the encoding represents a real value whose base B is 10, then decimal encoding shall be used. In forming the encoding, the following applies:

11.3.2.1 The ISO 6093 NR3 form shall be used (see 8.5.8).

11.3.2.2 SPACE shall not be used within the encoding.

11.3.2.3 If the real value is negative, then it shall begin with a MINUS SIGN (–), otherwise, it shall begin with a digit.

11.3.2.4 Neither the first nor the last digit of the mantissa may be a 0.

11.3.2.5 The last digit in the mantissa shall be immediately followed by FULL STOP (.), followed by the exponent-mark "E".

11.3.2.6 If the exponent has the value 0, it shall be written "+0", otherwise the exponent's first digit shall not be zero, and PLUS SIGN shall not be used.

11.4 GeneralString values

The encoding of values of the **`GeneralString`** type (and all other restricted character string types defined by reference to the International Register of Coded Character Sets) shall generate escape sequences to designate and invoke a new register entry only when the register entry for the character is not currently designated as the G0, G1, G2, G3, C0, or C1 set. All designations and invocations shall be into the smallest numbered G or C set for which there is an escape sequence defined in the entry of the International Register of Coded Character Sets to be used with Escape Sequences.

NOTE 1 – For the purposes of the above clause, G0 is the smallest numbered G set, followed by G1, G2, and G3 in order. C0 is the smallest numbered C set, followed by C1.

NOTE 2 – Each character in a character string value is associated with a particular entry in the International Register of Coded Character Sets.

11.5 Set and sequence components with default value

The encoding of a set value or sequence value shall not include an encoding for any component value which is equal to its default value.

11.6 Set-of components

The encodings of the component values of a set-of value shall appear in ascending order, the encodings being compared as octet strings with the shorter components being padded at their trailing end with 0-octets.

NOTE – The padding octets are for comparison purposes only and do not appear in the encodings.

11.7 GeneralizedTime

11.7.1 The encoding shall terminate with a "Z", as described in the Rec. ITU-T X.680 | ISO/IEC 8824-1 clause on **`GeneralizedTime`**.

11.7.2 The seconds element shall always be present.

11.7.3 The fractional-seconds elements, if present, shall omit all trailing zeros; if the elements correspond to 0, they shall be wholly omitted, and the decimal point element also shall be omitted.

EXAMPLE

A seconds element of "26.000" shall be represented as "26"; a seconds element of "26.5200" shall be represented as "26.52".

11.7.4 The decimal point element, if present, shall be the point option "**`.`**".

11.7.5 Midnight (GMT) shall be represented in the form:

"YYYYMMDD000000Z"

where "YYYYMMDD" represents the day following the midnight in question.

EXAMPLE

Examples of valid representations:

"19920521000000Z"

"19920622123421Z"

"19920722132100.3Z"

Examples of invalid representations:

`"19920520240000Z"` (midnight represented incorrectly)

`"19920622123421.0Z"` (spurious trailing zeros)

`"19920722132100.30Z"` (spurious trailing zeros)

11.8 UTCTime

11.8.1 The encoding shall terminate with "Z", as described in the ITU-T X.680 | ISO/IEC 8824-1 clause on UTCTime.

11.8.2 The seconds element shall always be present.

11.8.3 Midnight (GMT) shall be represented in the form:

"YYMMDD000000Z"

where "YYMMDD" represents the day following the midnight in question.

11.8.4 Examples of valid representations

"920521000000Z"

"920622123421Z"

"920722132100Z"