PS3.5

DICOM PS3.5 ~~2015c~~2016a - Data Structures and Encoding

PS3.5: DICOM PS3.5 ~~2015c~~2016a - Data Structures and Encoding

Table of Contents

List of Figures

List of Tables

Notice and Disclaimer

The information in this publication was considered technically sound by the consensus of persons engaged in the development and approval of the document at the time it was developed. Consensus does not necessarily mean that there is unanimous agreement among every person participating in the development of this document.

NEMA standards and guideline publications, of which the document contained herein is one, are developed through a voluntary consensus standards development process. This process brings together volunteers and/or seeks out the views of persons who have an interest in the topic covered by this publication. While NEMA administers the process and establishes rules to promote fairness in the development of consensus, it does not write the document and it does not independently test, evaluate, or verify the accuracy or completeness of any information or the soundness of any judgments contained in its standards and guideline publications.

NEMA disclaims liability for any personal injury, property, or other damages of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, application, or reliance on this document. NEMA disclaims and makes no guaranty or warranty, expressed or implied, as to the accuracy or completeness of any information published herein, and disclaims and makes no warranty that the information in this document will fulfill any of your particular purposes or needs. NEMA does not undertake to guarantee the performance of any individual manufacturer or seller's products or services by virtue of this standard or guide.

In publishing and making this document available, NEMA is not undertaking to render professional or other services for or on behalf of any person or entity, nor is NEMA undertaking to perform any duty owed by any person or entity to someone else. Anyone using this document should rely on his or her own independent judgment or, as appropriate, seek the advice of a competent professional in determining the exercise of reasonable care in any given circumstances. Information and other standards on the topic covered by this publication may be available from other sources, which the user may wish to consult for additional views or information not covered by this publication.

NEMA has no power, nor does it undertake to police or enforce compliance with the contents of this document. NEMA does not certify, test, or inspect products, designs, or installations for safety or health purposes. Any certification or other statement of compliance with any health or safety-related information in this document shall not be attributable to NEMA and is solely the responsibility of the certifier or maker of the statement.

Foreword

This DICOM Standard was developed according to the procedures of the DICOM Standards Committee.

The DICOM Standard is structured as a multi-part document using the guidelines established in [ISO/IEC Directives, Part 3].

1 Scope and Field of Application

In this part of the standard the structure and encoding of the Data Set is specified. In the context of Application Entities communicating over a network (see PS3.7), a Data Set is that portion of a DICOM Message that conveys information about real world objects being managed over the network. A Data Set may have other contexts in other applications of this standard; e.g., in media exchange the Data Set translates to file content structure.

This part of the DICOM Standard specifies:

a. the encoding of Values

b. the structure and usage of a Data Set

c. Data Element usage and relationships to other elements

d. the construction and usage of Nested Data Sets

e. the construction and usage of Data Sets containing Pixel Data

f. how to uniquely identify information

g. the specification of the standard DICOM Transfer Syntaxes

This part of the DICOM Standard does not specify:

a. the structure and syntax of a message (this is specified in PS3.7)

b. the structure and usage of a command set (this is specified in PS3.7)

c. how an application service functions or is classified (this is specified in PS3.3 and PS3.4)

d. how Data Sets relate to network communication, media storage, or other services

2 Normative References

The following standards contain provisions that, through references in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibilities of applying the most recent editions of the standards indicated below.

[ANSI HISPP MSDS] ANSI. 1993. *Healthcare Informatics Standards Planning Panel Message Standard Developers Subcommittee Proposal on Common Data Types*. http://www.meb.uni-bonn.de/standards/HISPP/MSDS/CommonDataType1102.ps .

[ANSI X3.4] ANSI. 1986. *Coded Character Set - 7-Bit American National Standard Code for Information Interchange*.

[ANSI X3.9] ANSI. 1978. *Programming Language FORTRAN*. http://www.fortran.com/F77_std/rjcnf-0.html .

[ASTM E-1238-91] ASTM. 1991. *Standard Specification for Transferring Clinical Observations Between Independent Computer Systems; Draft Revision 4.2.1*.

[BDRWP 2.B] Blu-ray Disc™ Association. March 2005. *White Paper Blu-ray Disc™ Format 2.B Audio Visual Application Format Specifications for BD-ROM*.

[ETSI TS 102 366] ETSI. Feb. 2005. *Audio Compression (AC-3, Enhanced AC-3) Standard*.

[IEEE 754] IEEE. 1985. *32-bit and 64-bit Floating Point Number Representations*.

[ISO/IEC Directives, Part 3] ISO/IEC. 1989. *Drafting and presentation of International Standards*.

[ISO 646] ISO. 1990. *Information Processing - ISO 7-bit coded character set for information interchange*.

[ISO/IEC 2022] ISO/IEC. 1994. *Information technology - Character code structure and extension techniques*.

[ISO 2375] ISO. 1986. *Data Processing - Procedure for the registration of escape sequences*.

[ISO/IEC 6429] ISO/IEC. 1990. *Information Processing - Control functions for 7-bit and 8-bit coded character sets*.

[ISO 6523] ISO. 1984. *Data interchange - Structures for identification of organizations*.

[ISO 7498] ISO. 1984. *Information processing systems - Open System Interconnection - Basic Reference Model*.

[ISO 7498-4] ISO. 1989. *Information processing systems - Open Systems Interconnection - Part 4: Management Framework*.

[ISO 8649] ISO. 1988. *Information processing systems - Open Systems Interconnection - Service definition for the Association Control Service Element (ACSE)*.

[ISO 8822] ISO. 1988. *Information processing systems - Open Systems Interconnection - Connection oriented presentation service definition*.

[ISO/IEC 8824] ISO/IEC. 1990. *Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)*.

[ISO/IEC 8859-1] ISO/IEC. 1987. *Information processing - 8-bit single-byte coded graphic character sets - Part 1: Latin alphabet No. 1*.

[ISO/IEC 8859-2] ISO/IEC. 1987. *Information processing - 8-bit single-byte coded graphic character sets - Part 2: Latin alphabet No. 2*.

[ISO/IEC 8859-3] ISO/IEC. 1988. *Information processing - 8-bit single-byte coded graphic character sets - Part 3: Latin alphabet No. 3*.

[ISO/IEC 8859-4] ISO/IEC. 1988. *Information processing - 8-bit single-byte coded graphic character sets - Part 4: Latin alphabet No. 4*.

[ISO/IEC 8859-5] ISO/IEC. 1988. *Information processing - 8-bit single-byte coded graphic character sets - Part 5: Latin/Cyrillic alphabet*.

[ISO/IEC 8859-6] ISO/IEC. 1987. *Information processing - 8-bit single-byte coded graphic character sets - Part 6: Latin/Arabic alphabet*.

[ISO/IEC 8859-7] ISO/IEC. 1987. *Information processing - 8-bit single-byte coded graphic character sets - Part 7: Latin/Greek alphabet*.

[ISO/IEC 8859-8] ISO/IEC. 1988. *Information processing - 8-bit single-byte coded graphic character sets - Part 8: Latin/Hebrew alphabet*.

[ISO/IEC 8859-9] ISO/IEC. 1989. *Information processing - 8-bit single-byte coded graphic character sets - Part 9: Latin alphabet No. 5*.

[ISO/IEC 9834-1] ISO/IEC. 2005. *Information technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: General procedures and top arcs of the ASN.1 Object Identifier tree*.

[ISO/IEC 10918-1] ISO/IEC. 1994. *JPEG Standard for digital compression and encoding of continuous-tone still images. Part 1 - Requirements and implementation guidelines*.

[ISO/IEC 10918-2] ISO/IEC. 1995. *JPEG Standard for digital compression and encoding of continuous-tone still images. Part 2 - Testing*.

[ISO/IEC 11172-3] ISO/IEC. 1993. *Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 3: Audio*.

[ISO/IEC 13818-1] ISO/IEC. 2000. *Information technology -- Generic coding of moving pictures and associated audio information - Part 1: Systems*.

[ISO/IEC 13818-2] ISO/IEC. 2000. *Information technology -- Generic coding of moving pictures and associated audio information - Part 2: Video*.

[ISO/IEC 13818-3] ISO/IEC. 1998. *Information technology -- Generic coding of moving pictures and associated audio information - Part 3: Audio*.

[ISO/IEC 13818-4] ISO/IEC. 1998. *Information technology -- Generic coding of moving pictures and associated audio information - Part 4: Conformance testing*.

[ISO/IEC 13818-7] ISO/IEC. 1997. *Information technology -- Generic coding of moving pictures and associated audio information - Part 7: Advanced Audio Coding (AAC)*.

[ISO/IEC 14495-1] ISO/IEC. 1997. *Lossless and near-lossless coding of continuous tone still images (JPEG-LS)*.

[ISO/IEC 14496-3] ISO/IEC. 2009. *Information technology - Coding of audio-visual objects - Part 3: Audio*.

[ISO/IEC 14496-10] ISO/IEC. 2009. *Information technology - Coding of audio-visual objects - Part 10: Advanced Video Coding*.

[ISO/IEC 14496-12] ISO/IEC. 2003. *Information technology - Coding of audio-visual objects - Part 12: ISO base media file format*.

[ISO/IEC 14496-14] ISO/IEC. 2003. *Information technology - Coding of audio-visual objects - Part 14: MP4 file format*.

[ISO/IEC 15444-1] ISO/IEC. 2004. *JPEG 2000 Image Coding System*.

[ISO/IEC 15444-2] ISO/IEC. 2004. *JPEG 2000 Image Coding System: Extensions*.

[ISO/IEC 15444-9] ISO/IEC. 2005. *Information technology - JPEG 2000 image coding system: Interactivity tools, APIs and protocols*.

[ENV 41 503] ENV. 1990. *Information systems interconnection - European graphic character repertoires and their coding*.

[ENV 41 508] ENV. 1990. *Information systems interconnection - East European graphic character repertoires and their coding*.

[JIS X 0201] JIS. 1976. *Code for Information Interchange*.

[JIS X 0208] JIS. 1990. *Code for the Japanese Graphic Character set for information interchange*.

[JIS X 0212] JIS. 1990. *Code of the supplementary Japanese Graphic Character set for information interchange*.

[KS X 1001] KS. 1997. *Code for Information Interchange (Hangul and Hanja)*.

[RFC 1468] IETF. *Japanese Character Encoding for Internet Messages*. http://tools.ietf.org/html/rfc1468 .

[RFC 1554] IETF. *ISO-2022-JP-2: Multilingual Extension of ISO-2022-JP*. http://tools.ietf.org/html/rfc1554 .

[RFC 1951] IETF. *DEFLATE Compressed Data Format Specification version 1.3*. http://tools.ietf.org/html/rfc1951 .

[RFC 3986] IETF. *Uniform Resource Identifiers (URI) : Generic Syntax*. http://tools.ietf.org/html/rfc3986 .

[RFC 3987] IETF. *Internationalized Resource Identifiers (IRIs)*. http://tools.ietf.org/html/rfc3987 .

[RFC 5890] IETF. *Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework*. http://tools.ietf.org/html/rfc5890 .

[RFC 5891] IETF. *Internationalized Domain Names in Applications (IDNA): Protocol*. http://tools.ietf.org/html/rfc5891 .

3 Definitions

For the purposes of this standard, the following definitions apply.

3.1 Reference Model Definitions

This part of the standard makes use of the following terms defined in ISO 7498:

a) Application Entities

b) OSI Presentation Protocol

3.2 ACSE Service Definitions

This part of the standard makes use of the following terms defined in ISO 8649:

a) Association

3.3 Presentation Service Definitions

This part of the standard makes use of the following terms defined in ISO 8822:

a) Presentation Context

b) Presentation Data Value (PDV)

c) Transfer Syntax

d) Transfer Syntax Name

3.4 Object Identification Definitions

This part of the standard makes use of the following terms defined in ISO 8824:

a) OSI Object Identification

3.5 DICOM Introduction and Overview Definitions

This part of the standard makes use of the following terms defined in PS3.1:

a) Attribute

b) Command Element

c) Data Dictionary

3.6 DICOM Conformance Definitions

This part of the standard makes use of the following terms defined in PS3.2:

a) Conformance Statement

3.7 DICOM Information Object Definitions

This part of the standard makes use of the following terms defined in PS3.3:

a) Attribute Tag

b) Information Entity

c) Information Object Definition (IOD)

d) Multi-Frame Image

3.8 DICOM Service Class Specifications Definitions

This part of the standard makes use of the following terms defined in PS3.4:

a) Service-Object Pair (SOP) Class

3.9 DICOM Network Communication Support For Message Exchange Definitions

This part of the standard makes use of the following terms defined in PS3.8:

a) DICOM Upper Layer Service

3.10 DICOM Data Structures and Encoding Definitions

The following definitions are commonly used in this Standard:

Basic Offset Table: A table of pointers to individual frames of an encapsulated multi-frame image.

Big Endian: A form of byte ordering where multiple byte binary values are encoded with the most significant byte encoded first, and the remaining bytes encoded in decreasing order of significance.

Character Repertoire: A finite set of different characters that is considered to be complete for a given purpose and is specified independently of their encoding (also referred to as a character set).

Data Element: A unit of information as defined by a single entry in the data dictionary. An encoded Information Object Definition (IOD) Attribute that is composed of, at a minimum, three fields: a Data Element Tag, a Value Length, and a Value Field. For some specific Transfer Syntaxes, a Data Element also contains a VR Field where the Value Representation of that Data Element is specified explicitly.

Data Element Tag: A unique identifier for a Data Element composed of an ordered pair of numbers (a Group Number followed by an Element Number).

Data Element Type: Used to specify whether an Attribute of an Information Object Definition or an Attribute of a SOP Class Definition is mandatory, mandatory only under certain conditions, or optional. This translates to whether a Data Element of a Data Set is mandatory, mandatory only under certain conditions, or optional.

Data Set: Exchanged information consisting of a structured set of Attribute values directly or indirectly related to Information Objects. The value of each Attribute in a Data Set is expressed as a Data Element. A collection of Data Elements ordered by increasing Data Element Tag number that is an encoding of the values of Attributes of a real world object.

Defined Term: The Value of a Data Element is a Defined Term when the Value of the element may be one of an explicitly specified set of standard values, and these values may be extended by implementers.

Element Number: The second number in the ordered pair of numbers that makes up a Data Element Tag.

Enumerated Value: The Value of a Data Element is an Enumerated Value when the value of the element must be one of an explicitly specified set of standard values, and these values shall not be extended by implementers.

Group Number: The first number in the ordered pair of numbers that makes up a Data Element Tag.

Item: A component of the Value of a Data Element that is of Value Representation Sequence of Items. An Item contains a Data Set.

Item Delimitation Data Element: Used to mark the end of an Item of Undefined Length in a Sequence of Items. This is the last Data Element in an Item of Undefined Length.

Little Endian: A form of byte ordering where multiple byte binary values are encoded with the least significant byte encoded first; and the remaining bytes encoded in increasing order of significance.

Nested Data Set: A Data Set contained within a Data Element of another Data Set. Data Sets can be nested recursively. Only Data Elements with Value Representation Sequence of Items may, themselves, contain Data Sets.

Pixel Cell: The container for a single Pixel Sample Value that may include unused bits. The size of a Pixel Cell shall be specified by the Bits Allocated (0028, 0100) Data Element.

Pixel Data: Graphical data (e.g., images) of variable pixel-depth encoded in the Pixel Data, Float Pixel Data or Double Float Pixel Data Element.

Pixel Sample Value: A value associated with an individual pixel. An individual pixel consists of one or more Pixel Sample Values (e.g., color images).

Private Data Element: Additional Data Element, defined by an implementer, to communicate information that is not contained in Standard Data Elements. Private Data elements have odd Group Numbers.

Repeating Group: Standard Data Elements within a particular range of Group Numbers where elements that have identical Element Numbers have the same meaning within each Group (and the same VR, VM, and Data Element Type). Repeating Groups shall only exist for Curves and Overlay Planes (Group Numbers (50xx,eeee) and (60xx,eeee), respectively) and are a remnant of versions of this standard prior to V3.0.

Retired Data Element: A Data Element that is unsupported beginning with Version 3.0 of this standard. Implementations may continue to support Retired Data Elements for the purpose of backward compatibility with versions prior to V3.0, but this is not a requirement of this version of the standard.

Sequence Delimitation Item: Item used to mark the end of a Sequence of Items of Undefined Length. This Item is the last Item in a Sequence of Items of Undefined Length.

Sequence of Items (Value Representation SQ): A Value Representation for Data Elements that contain a sequence of Data Sets. Sequence of Items allows for Nested Data Sets.

Standard Data Element: A Data Element defined in the DICOM Standard, and therefore listed in the DICOM Data Element Dictionary in PS3.6.

Transfer Syntax (Standard and Private): A set of encoding rules that allow Application Entities to unambiguously negotiate the encoding techniques (e.g., Data Element structure, byte ordering, compression) they are able to support, thereby allowing these Application Entities to communicate.

Undefined Length: The ability to specify an unknown length for a Data Element Value (of Value Representation SQ, UN, OW, or OB) or Item. Data Elements and Items of Undefined Length are delimited with Sequence Delimitation Items and Item Delimiter Data Elements, respectively.

Unique Identifier (UID): A string of characters that uniquely identifies a wide variety of items; guaranteeing uniqueness across multiple countries, sites, vendors and equipment.

Value: A component of a Value Field. A Value Field may consist of one or more of these components.

Value Field: The field within a Data Element that contains the Value(s) of that Data Element.

Value Length: The field within a Data Element that contains the length of the Value Field of the Data Element.

Value Multiplicity (VM): Specifies the number of Values contained in the Value Field of a Data Element.

Value Representation (VR): Specifies the data type and format of the Value(s) contained in the Value Field of a Data Element.

Value Representation Field: The field where the Value Representation of a Data Element is stored in the encoding of a Data Element structure with explicit VR.

3.11 Character Handling Definitions

This part of the standard makes use of the following terms defined in ISO/IEC 2022:1994

a) Coded Character Set; Code

b) Code Extension

c) Control Character

d) To Designate

e) Escape Sequence

f) Graphic Character

g) To Invoke

4 Symbols and Abbreviations

The following symbols and abbreviations are used in this Standard.

ACR American College of Radiology

AE Application Entity

ANSI American National Standards Institute

CEN TC251 Comite Europeen de Normalisation - Technical Committee 251 - Healthcare Informatics

DICOM Digital Imaging and Communications in Medicine

HISPP Healthcare Information Standards Planning Panel

HL7 Healthcare Industry Level 7 Interface Standards

IEEE Institute of Electrical and Electronics Engineers

IOD Information Object Definition

ISO International Standards Organization

JPEG Joint Photographic Experts Group

JIRA Japan Medical Imaging and Radiological Systems Industries Association

MPEG Moving Picture Experts Group

MSDS Healthcare Message Standard Developers Sub-Committee

NEMA National Electrical Manufacturers Association

OSI Open Systems Interconnection

RLE Run Length Encoding

TCP/IP Transmission Control Protocol/Internet Protocol

UID Unique Identifier

SOP Service-Object Pair

UTC Coordinated Universal Time

URI/URL Uniform Resource Identifier/Locator

VM Value Multiplicity

VR Value Representation

5 Conventions

Word(s) are capitalized in this document (not headings) to help the reader understand that these word(s) have been previously defined in Section 3 of this document and are to be interpreted with that meaning.

The Data Element Tag is represented as (gggg,eeee), where gggg equates to the Group Number and eeee equates to the Element Number within that Group. The Data Element Tag is represented in hexadecimal notation as specified for each Data Element in PS3.6.

The notation XXXXH, where XXXX is one or more hexadecimal digits, and "H" is used to signify a hexadecimal number.

6 Value Encoding

A Data Set is constructed by encoding the values of Attributes specified in the Information Object Definition (IOD) of a Real-World Object. The specific content and semantics of these Attributes are specified in Information Object Definitions (see PS3.3). The range of possible data types of these values and their encoding are specified in this section. The structure of a Data Set, which is composed of Data Elements containing these values, is specified in Section 7.

Throughout this part, as well as other parts of the DICOM Standard, Tags are used to identify both specific Attributes and their corresponding Data Elements.

6.1 Support of Character Repertoires

Values that are text or character strings can be composed of Graphic and Control Characters. The Graphic Character set, independent of its encoding, is referred to as a Character Repertoire. Depending on the native language context in which Application Entities wish to exchange data using the DICOM Standard, different Character Repertoires will be used. The Character Repertoires supported by DICOM are:

- ISO 8859
- JIS X 0201-1976 Code for Information Interchange
- JIS X 0208-1990 Code for the Japanese Graphic Character set for information interchange
- JIS X 0212-1990 Code of the supplementary Japanese Graphic Character set for information interchange
- KS X 1001 (registered as ISO-IR 149) for Korean Language
- TIS 620-2533 (1990) Thai Characters Code for Information Interchange
- ISO 10646-1, 10646-2, and their associated supplements and extensions for Unicode character set
- GB 18030
- GB2312
- GBK

Note

1. The ISO 10646-1, 10646-2, and their associated supplements and extensions correspond to the Unicode version 3.2 character set. The ISO IR 192 corresponds to the use of the UTF-8 encoding for this character set.
2. The GB 18030 character set is harmonized with the Unicode character set on a regular basis, to reflect updates from both the Chinese language and from Unicode extensions to support other languages.
3. The issue of font selection is not addressed by the DICOM standard. Issues such as proper display of words like "bone" in Chinese or Japanese usage are managed through font selection. Similarly, other user interface issues like bidirectional character display and text orientation are not addressed by the DICOM standard. The Unicode documents provide extensive documentation on these issues.
4. The GBK character set is an extension of the GB 2312-1980 character set and supports the Chinese characters in GB 13000.1-93 that is the Chinese adaptation of Unicode 1.1. The GBK is code point backward compatible to GB2312-1980. The GB 18030 character set is an extension of the GBK character set for support of Unicode 3.2, and provides backward code point compatibility.

6.1.1 Representation of Encoded Character Values

As defined in the ISO Standards referenced in this section, byte values used for encoded representations of characters are represented in this section as two decimal numbers in the form column/row.

This means that the value can be calculated as (column * 16) + row, e.g., 01/11 corresponds to the value 27 (1BH).

Note

Two digit hex notation will be used throughout the remainder of this standard to represent character encoding. The column/row notation is used only within Section 6.1 to simplify any cross referencing with applicable ISO standards.

The byte encoding space is divided into four ranges of values:

- CL bytes from 00/00 to 01/15
- GL bytes from 02/00 to 07/15
- CR bytes from 08/00 to 09/15
- GR bytes from 10/00 to 15/15

Note

ISO 8859 does not differentiate between a code element, e.g., G0, and the area in the code table, e.g., GL, where it is invoked. The term "G0" specifies the code element as well as the area in the code table. In ISO/IEC 2022 there is a clear distinction between the code elements (G0, G1, G2, and G3) and the areas in which the code elements are invoked (GL or GR). In this Standard the nomenclature of ISO/IEC 2022 is used.

The Control Character set C0 shall be invoked in CL and the Graphic Character sets G0 and G1 in GL and GR respectively. Only some Control Characters from the C0 set are used in DICOM (see Section 6.1.3), and characters from the C1 set shall not be used.

6.1.2 Graphic Characters

A Character Repertoire, or character set, is a collection of Graphic Characters specified independently of their encoding.

6.1.2.1 Default Character Repertoire

The default repertoire for character strings in DICOM shall be the Basic G0 Set of the International Reference Version of ISO 646:1990 (ISO-IR 6). See Annex E for a table of the DICOM default repertoire and its encoding.

Note

This Basic G0 Set is identical with the common character set of ISO 8859.

6.1.2.2 Extension or Replacement of the Default Character Repertoire

DICOM Application Entities (AEs) that extend or replace the default repertoire convey this information in the Specific Character Set (0008,0005) Attribute.

Note

The Attribute Specific Character Set (0008,0005) is encoded using a subset of characters from ISO-IR 6. See the definition for the Value Representation (VR) of Code String (CS) in Table 6.2-1.

For Data Elements with Value Representations of SH (Short String), LO (Long String), UC (Unlimited Characters), ST (Short Text), LT (Long Text), UT (Unlimited Text) or PN (Person Name) the default character repertoire may be extended or replaced (these Value Representations are described in more detail in Section 6.2). If such an extension or replacement is used, the relevant "Specific Character Set" shall be defined as an attribute of the SOP Common Module (0008,0005) (see PS3.3) and shall be stated in the Conformance Statement. PS3.2 gives conformance guidelines.

Note

1. Preferred repertoires as defined in ENV 41 503 and ENV 41 508 for the use in Western and Eastern Europe, respectively, are: ISO-IR 100, ISO-IR 101, ISO-IR 144, ISO-IR 126. See Section 6.1.2.3.
2. Information Object Definitions using different character sets cannot rely per se on lexical ordering or string comparison of data elements represented as character strings. These operations can only be carried out within a given character repertoire and not across repertoire boundaries.

6.1.2.3 Encoding of Character Repertoires

The 7-bit default character repertoire can be replaced for use in Value Representations SH, LO, ST, LT, PN, UC and UT with one of the single-byte codes defined in PS3.3.

Note

This replacement character repertoire does not apply to other textual Value Representations (AE and CS).

The replacement character repertoire shall be specified in value 1 of the Attribute Specific Character Set (0008,0005). Defined Terms for the Attribute Specific Character Set are specified in PS3.3.

Note

1. The code table is split into the GL area, which supports a 94 character set only (bit combinations 02/01 to 07/14) plus SPACE in 02/00, and the GR area, which supports either a 94 or 96 character set (bit combinations 10/01 to 15/14 or 10/00 to 15/15). The default character set (ISO-IR 6) is always invoked in the GL area.

2. All character sets specified in ISO 8859 include ISO-IR 6. This set will always be invoked in the GL area of the code table and is the equivalent of ASCII (ANSI X3.4:1986), whereas the various extension repertoires are mapped onto the GR area of the code table.

3. The 8-bit code table of JIS X 0201 includes ISO-IR 14 (romaji alphanumeric characters) as the G0 code element and ISO-IR 13 (katakana phonetic characters) as the G1 code element. ISO-IR 14 is identical to ISO-IR 6, except that bit combination 05/12 represents a "¥"(YEN SIGN) and bit combination 07/14 represents an over-line.

Two character codes of the single-byte character sets invoked in the GL area of the code table, 02/00 and 05/12, have special significance in the DICOM Standard. The character SPACE, represented by bit combination 02/00, shall be used for the padding of Data Element Values that are character strings. The Graphic Character represented by the bit combination 05/12, "\" (BACKSLASH) in the repertoire ISO-IR 6, shall only be used in character strings with Value Representations of UT, ST and LT (see Section 6.2). Otherwise the character code 05/12 is used as a separator for multiple valued Data Elements (see Section 6.4).

Note

When the value of the Attribute Specific Character Set (0008,0005) is either "ISO_IR 13" or "ISO 2022 IR 13", the graphic character represented by the bit combination 05/12 is a "¥" (YEN SIGN) in the character set of ISO-IR 14.

The character DELETE (bit combination 07/15) shall not be used in DICOM character strings.

The replacement Character Repertoire specified in value 1 of the Attribute Specific Character Set (0008,0005) (or the default Character Repertoire if value 1 is empty) may be further extended with additional Coded Character Sets, if needed and permitted by the replacement Character Repertoire. The additional Coded Character Sets and extension mechanism shall be specified in additional values of the Attribute Specific Character Set. If Attribute Specific Character Set (0008,0005) has a single value, the DICOM SOP Instance supports only one code table and no Code Extension techniques. If Attribute Specific Character Set (0008,0005) has multiple values, the DICOM SOP Instance supports Code Extension techniques as described in ISO/IEC 2022:1994.

The Character Repertoires that prohibit extension are identified in Part 3.

Note

1. Considerations on the Handling of Unsupported Character Sets:

   In DICOM, character sets are not negotiated between Application Entities but are indicated by a conditional attribute of the SOP Common Module. Therefore, implementations may be confronted with character sets that are unknown to them.

   The Unicode Standard includes a substantial discussion of the recommended means for display and print for characters that lack font support. These same recommendations may apply to the mechanisms for unsupported character sets.

   The machine should print or display such characters by replacing all unknown characters with the four characters "\nnn", where "nnn" is the three digit octal representation of each byte.

An example of this for an ASCII based machine would be as follows:

Character String: Günther Encoded representation: 04/07 15/12 06/14 07/04 06/08 06/05 07/02ASCII based machine: G\374nther

Implementations may also encounter Control Characters that they have no means to print or display. The machine may print or display such Control Characters by replacing the Control Character with the four characters "\nnn", where "nnn" is the three digit octal representation of each byte.

2. Considerations for missing fonts

   The Unicode standard and the GB18030 standard define mechanisms for print and display of characters that are missing from the available fonts. If GBK is specified in Specific Character Set (0008,0005), the GB 18030 rules of print and display of characters shall apply. The DICOM standard does not specify user interface behavior since it does not affect network or media data exchange.

3. The Unicode and GB18030 standards have distinct Yen symbol, backslash, and several forms of reverse solidus. The separator for multi-valued data elements in DICOM is the character valued 05/12 regardless of what glyph is used to enter or display this character. The other reverse solidus characters that have a very similar appearance are not separators. The choice of font can affect the appearance of 05/12 significantly. Multi-byte encoding systems, such as GB18030, GBK and ISO 2022, may generate encodings that contain a byte valued 05/12. Only the character that encodes as a single byte valued 05/12 is a delimiter.

   For multi-valued Data Elements, existing implementations that are expecting only single-byte replacement character sets may misinterpret the Value Multiplicity of the Data Element as a consequence of interpreting 05/12 bytes in multi-byte characters or ISO 2022 escape sequences as delimiters, and this may affect the integrity of store-and-forward operations. Applications that do not explicitly state support for GB18030, GBK or ISO 2022 in their conformance statement, might exhibit such behavior.

6.1.2.4 Code Extension Techniques

For Data Elements with Value Representations of SH (Short String), LO (Long String), UC (Unlimited Characters), ST (Short Text), LT (Long Text), UT (Unlimited Text) or PN (Person Name), the default character repertoire or the character repertoire specified by value 1 of Attribute Specific Character Set (0008,0005), may be extended using the Code Extension techniques specified by ISO/IEC 2022:1994.

If such Code Extension techniques are used, the related Specific Character Set or Sets shall be specified by value 2 to value n of the Attribute Specific Character Set (0008,0005) of the SOP Common Module (see PS3.3), and shall be stated in the Conformance Statement.

Note

1. Defined Terms for Specific Character Set (0008,0005) are defined in PS3.3.

2. Support for Japanese kanji (ideographic), hiragana (phonetic), katakana (phonetic), Korean (Hangul phonetic and Hanja ideographic) and Chinese characters is defined in PS3.3.

3. The Chinese Character Set (GB18030) and Unicode (ISO 10646-1, 10646-2) do not allow the use of Code Extension Techniques. If either of these character sets is used, no other character set may be specified in the Specific Character Set (0008,0005) attribute, that is, it may have only one value.

6.1.2.5 Usage of Code Extension

DICOM supports Code Extension techniques if the Attribute Specific Character Set (0008,0005) is multi-valued. The method employed for Code Extension in DICOM is as described in ISO/IEC 2022:1994. The following assumptions shall be made and the following restrictions shall apply:

6.1.2.5.1 Assumed Initial States

- Code element G0 and code element G1 (in 8-bit mode only) are always invoked in the GL and GR areas of the code table respectively. Designated character sets for these code elements are immediately in use. Code elements G2 and G3 are not used.

- The primary set of Control Characters shall always be designated as the C0 code element and this shall be invoked in the CL area of the code table. The C1 code element shall not be used.

6.1.2.5.2 Restrictions for Code Extension

- As code elements G0 and G1 always have shift status, Locking Shifts (SI, SO) are not required and shall not be used.
- As code elements G2 and G3 are not used, Single Shifts (SS2 and SS3) cannot be used.
- Only the ESC sequences specified in PS3.3 shall be used to activate Code Elements.

6.1.2.5.3 Requirements

The character set specified by value 1 of the Attribute Specific Character Set (0008,0005), or the default character repertoire if value 1 is missing, shall be active at the beginning of each textual Data Element value, and at the beginning of each line (i.e., after a CR and/or LF) or page (i.e., after an FF).

If within a textual value a character set other than the one specified in value 1 of the Attribute Specific Character Set (0008,0005), or the default character repertoire if value 1 is missing, has been invoked, the character set specified in the value 1, or the default character repertoire if value 1 is missing, shall be active in the following instances:

- before the end of line (i.e., before the CR and/or LF)
- before the end of a page (i.e., before the FF)
- before any other Control Character other than ESC (e.g., before any TAB)
- before the end of a Data Element value (e.g., before the 05/12 character code that separates multiple textual Data Element Values - 05/12 corresponds to "\" (BACKSLASH) in the case of default repertoire IR-6 or "¥" (YEN SIGN) in the case of IR-14).
- before the "^" and "=" delimiters separating name components and name component groups in Data Elements with a VR of PN.

If within a textual value a character set other than the one specified in value 1 of the Attribute Specific Character Set (0008,0005), or the default character repertoire if value 1 is missing, is used, the Escape Sequence of this character set must be inserted explicitly in the following instances:

- before the first use of the character set in the line
- before the first use of the character set in the page
- before the first use of the character set in the Data Element value
- before the first use of the character set in the name component and name component group in Data Element with a VR of PN

  Note

  These requirements allow an application to skip lines, values, or components in a textual data element and start the new line with a defined character set without the need to track the character set changes in the text skipped. A similar restriction appears in the RFCs describing the use of multi-byte character sets over the Internet. An Escape Sequence switching to the value 1 or default Specific Character Set is not needed within a line, value, or component if no Code Extensions are present. Nor is a switch needed to the value 1 or default Specific Character Set if this character set has only the G0 Code Element defined, and the G0 Code Element is still active.

6.1.2.5.4 Levels of Implementation and Initial Designation

a. Attribute Specific Character Set (0008,0005) not present:
   - 7-bit code
   - Implementation level: ISO 2022 Level 1 - Elementary 7-bit code (code-level identifier 1)
   - Initial designation: ISO-IR 6 (ASCII) as G0.

- Code Extension shall not be used.

b. Attribute Specific Character Set (0008,0005) single value other than "ISO_IR 192", "GB18030" or "GBK":

- 8-bit code
- Implementation level: ISO 2022 Level 1 - Elementary 8-bit code (code-level identifier 11)
- Initial designation: One of the ISO 8859-defined character sets, or the 8-bit code table of JIS X 0201 specified by value 1 of the Attribute Specific Character Set (0008,0005), as G0 and G1.
- Code Extension shall not be used.

c. Attribute Specific Character Set (0008,0005) multi-valued:

- 8-bit code
- Implementation level: ISO 2022 Level 4 - Redesignation of Graphic Character Sets within a Code (code-level identifier 14)
- Initial designation: One of the ISO 8859-defined character sets, or the 8-bit code table of JIS X 0201 specified by value 1 of the Attribute Specific Character Set (0008,0005), as G0 and G1. If value 1 of the Attribute Specific Character Set (0008,0005) is empty, ISO-IR 6 (ASCII) is assumed as G0, and G1 is undefined.
- All character sets specified in the various values of Attribute Specific Character Set (0008,0005), including value 1, may participate in Code Extension.

d. Attribute Specific Character Set (0008,0005) single value "ISO_IR 192", "GB18030" or "GBK":

- variable length code
- Implementation level: not specified (not compatible with ISO 2022)
- Initial designation: as specified by value 1 of the Attribute Specific Character Set (0008,0005)
- Code Extension shall not be used.

6.1.3 Control Characters

Textual data that is interchanged may require some formatting information. Control Characters are used to indicate formatting, but their use in DICOM is kept to a minimum since some machines may handle them inappropriately. ISO 646:1990 and ISO 6429:1990 define Control Characters. As shown in Table 6.1-1 below, only a subset of five Control Characters from the C0 set shall be used in DICOM for the encoding of Control Characters in text strings.

Table 6.1-1. DICOM Control Characters and Their Encoding

Acronym	Name	Coded Value
LF	Line Feed	00/10
FF	Form Feed	00/12
CR	Carriage Return	00/13
ESC	Escape	01/11
TAB	Horizontal Tab	00/09

The ESC character shall be used only for ISO 2022 character set control sequences, in accordance with Section 6.1.2.5.

In text strings (value representation ST, LT, or UT) a new line shall be represented as CR LF.

Note

1. Some machines (such as UNIX based machines) may interpret LF (00/10) as a new line. In such cases, it is expected that the DICOM format is converted to the correct internal representation for that machine.

2. In previous editions of the standard (see PS3.5 2015a), the TAB character was not listed as a Control Character.

6.2 Value Representation (VR)

The Value Representation of a Data Element describes the data type and format of that Data Element's Value(s). PS3.6 lists the VR of each Data Element by Data Element Tag.

Values with VRs constructed of character strings, except in the case of the VR UI, shall be padded with SPACE characters (20H, in the Default Character Repertoire) when necessary to achieve even length. Values with a VR of UI shall be padded with a single trailing NULL (00H) character when necessary to achieve even length. Values with a VR of OB shall be padded with a single trailing NULL byte value (00H) when necessary to achieve even length.

All new VRs defined in future versions of DICOM shall be of the same Data Element Structure as defined in Section 7.1.2 (i.e., following the format for VRs such as OB, OD, OF, OL, OW, SQ and UN).

Note

Since all new VRs will be defined as specified in Section 7.1.2, an implementation may choose to ignore VRs not recognized by applying the rules stated in Section 7.1.2.

An individual Value, including padding, shall not exceed the Length of Value, except in the case of the last Value of a multi-valued field as specified in Section 6.4.

Note

The lengths of Value Representations for which the Character Repertoire can be extended or replaced are expressly specified in characters rather than bytes in Table 6.2-1. This is because the mapping from a character to the number of bytes used for that character's encoding may be dependent on the character set used.

Escape Sequences used for Code Extension shall not be included in the count of characters.

Table 6.2-1. DICOM Value Representations

VR Name	Definition	Character Repertoire	Length of Value
AE Application Entity	A string of characters that identifies an Application Entity with leading and trailing spaces (20H) being non-significant. A value consisting solely of spaces shall not be used.	Default Character Repertoire excluding character code 5CH (the BACKSLASH "\" in ISO-IR 6), and all control characters.	16 bytes maximum
AS Age String	A string of characters with one of the following formats -- nnnD, nnnW, nnnM, nnnY; where nnn shall contain the number of days for D, weeks for W, months for M, or years for Y. Example: "018M" would represent an age of 18 months.	"0"-"9", "D", "W", "M", "Y" of Default Character Repertoire	4 bytes fixed
AT Attribute Tag	Ordered pair of 16-bit unsigned integers that is the value of a Data Element Tag. Example: A Data Element Tag of (0018,00FF) would be encoded as a series of 4 bytes in a Little-Endian Transfer Syntax as 18H,00H,FFH,00H and in a Big-Endian Transfer Syntax as 00H,18H,00H,FFH. Note The encoding of an AT value is exactly the same as the encoding of a Data Element Tag as defined in Section 7.	not applicable	4 bytes fixed

VR Name	Definition	Character Repertoire	Length of Value
CS Code String	A string of characters with leading or trailing spaces (20H) being non-significant.	Uppercase characters, "0"-"9", the SPACE character, and underscore "_", of the Default Character Repertoire	16 bytes maximum
DA Date	A string of characters of the format YYYYMMDD; where YYYY shall contain year, MM shall contain the month, and DD shall contain the day, interpreted as a date of the Gregorian calendar system. Example: "19930822" would represent August 22, 1993. Note 1. The ACR-NEMA Standard 300 (predecessor to DICOM) supported a string of characters of the format YYYY.MM.DD for this VR. Use of this format is not compliant. 2. See also DT VR in this table.	"0"-"9" of Default Character Repertoire In the context of a Query with range matching (see PS3.4), the character "-" is allowed, and a trailing SPACE character is allowed for padding.	8 bytes fixed In the context of a Query with range matching (see PS3.4), the length is 18 bytes maximum.
DS Decimal String	A string of characters representing either a fixed point number or a floating point number. A fixed point number shall contain only the characters 0-9 with an optional leading "+" or "-" and an optional "." to mark the decimal point. A floating point number shall be conveyed as defined in ANSI X3.9, with an "E" or "e" to indicate the start of the exponent. Decimal Strings may be padded with leading or trailing spaces. Embedded spaces are not allowed. Note Data Elements with multiple values using this VR may not be properly encoded if Explicit-VR transfer syntax is used and the VL of this attribute exceeds 65534 bytes.	"0"-"9", "+", "-", "E", "e", "." and the SPACE character of Default Character Repertoire	16 bytes maximum

VR Name	Definition	Character Repertoire	Length of Value
DT Date Time		"0"-"9", "+", "-", "." and the SPACE character of Default Character Repertoire	26 bytes maximum In the context of a Query with range matching (see PS3.4), the length is 54 bytes maximum.

VR Name	Definition	Character Repertoire	Length of Value
	A concatenated date-time character string in the format: YYYYMMDDHHMMSS.FFFFFF&ZZXX The components of this string, from left to right, are YYYY = Year, MM = Month, DD = Day, HH = Hour (range "00" - "23"), MM = Minute (range "00" - "59"), SS = Second (range "00" - "60"). FFFFFF = Fractional Second contains a fractional part of a second as small as 1 millionth of a second (range "000000" - "999999"). &ZZXX is an optional suffix for offset from Coordinated Universal Time (UTC), where & = "+" or "-", and ZZ = Hours and XX = Minutes of offset. The year, month, and day shall be interpreted as a date of the Gregorian calendar system. A 24-hour clock is used. Midnight shall be represented by only "0000" since "2400" would violate the hour range. The Fractional Second component, if present, shall contain 1 to 6 digits. If Fractional Second is unspecified the preceding "." shall not be included. The offset suffix, if present, shall contain 4 digits. The string may be padded with trailing SPACE characters. Leading and embedded spaces are not allowed. A component that is omitted from the string is termed a null component. Trailing null components of Date Time indicate that the value is not precise to the precision of those components. The YYYY component shall not be null. Non-trailing null components are prohibited. The optional suffix is not considered as a component. A Date Time value without the optional suffix is interpreted to be in the local time zone of the application creating the Data Element, unless explicitly specified by the Timezone Offset From UTC (0008,0201). UTC offsets are calculated as "local time minus UTC". The offset for a Date Time value in UTC shall be +0000. Note 1. The range of the offset is -1200 to +1400. The offset for United States Eastern Standard Time is -0500. The offset for Japan Standard Time is +0900. 2. The RFC 2822 use of -0000 as an offset to indicate local time is not allowed. 3. A Date Time value of 195308 means August 1953, not specific to particular day. A Date Time value of 19530827111300.0 means August 27, 1953, 11;13 a.m. accurate to 1/10th second. 4. The Second component may have a value of 60 only for a leap second.		

VR Name	Definition	Character Repertoire	Length of Value
	5. The offset may be included regardless of null components; e.g., 2007-0500 is a legal value.		
FL Floating Point Single	Single precision binary floating point number represented in IEEE 754:1985 32-bit Floating Point Number Format.	not applicable	4 bytes fixed
FD Floating Point Double	Double precision binary floating point number represented in IEEE 754:1985 64-bit Floating Point Number Format.	not applicable	8 bytes fixed
IS Integer String	A string of characters representing an Integer in base-10 (decimal), shall contain only the characters 0 - 9, with an optional leading "+" or "-". It may be padded with leading and/or trailing spaces. Embedded spaces are not allowed. The integer, n, represented shall be in the range: $-2^{31} <= n <= (2^{31}-1)$.	"0"-"9", "+", "-" and the SPACE character of Default Character Repertoire	12 bytes maximum
LO Long String	A character string that may be padded with leading and/or trailing spaces. The character code 5CH (the BACKSLASH "\" in ISO-IR 6) shall not be present, as it is used as the delimiter between values in multiple valued data elements. The string shall not have Control Characters except for ESC.	Default Character Repertoire and/or as defined by (0008,0005) excluding character code 5CH (the BACKSLASH "\" in ISO-IR 6), and all Control Characters except ESC when used for ISO 2022 escape sequences.	64 chars maximum (see Note in Section 6.2)
LT Long Text	A character string that may contain one or more paragraphs. It may contain the Graphic Character set and the Control Characters, CR, LF, FF, and ESC. It may be padded with trailing spaces, which may be ignored, but leading spaces are considered to be significant. Data Elements with this VR shall not be multi-valued and therefore character code 5CH (the BACKSLASH "\" in ISO-IR 6) may be used.	Default Character Repertoire and/or as defined by (0008,0005) excluding Control Characters except TAB, LF, FF, CR (and ESC when used for ISO 2022 escape sequences).	10240 chars maximum (see Note in Section 6.2)
OB Other Byte ~~String~~	A string of bytes where the encoding of the contents is specified by the negotiated Transfer Syntax. OB is a VR that is insensitive to Little/Big Endian byte ordering (see Section 7.3). The string of bytes shall be padded with a single trailing NULL byte value (00H) when necessary to achieve even length.	not applicable	see Transfer Syntax definition
OD Other Double ~~String~~	A string of 64-bit IEEE 754:1985 floating point words. OD is a VR that requires byte swapping within each 64-bit word when changing between Little Endian and Big Endian byte ordering (see Section 7.3).	not applicable	2^{32}-8 bytes maximum
OF Other Float ~~String~~	A string of 32-bit IEEE 754:1985 floating point words. OF is a VR that requires byte swapping within each 32-bit word when changing between Little Endian and Big Endian byte ordering (see Section 7.3).	not applicable	2^{32}-4 bytes maximum
OL Other Long	A string of 32-bit words where the encoding of the contents is specified by the negotiated Transfer Syntax. OL is a VR that requires byte swapping within each word when changing between Little Endian and Big Endian byte ordering (see Section 7.3).	not applicable	see Transfer Syntax definition
OW Other Word ~~String~~	A string of 16-bit words where the encoding of the contents is specified by the negotiated Transfer Syntax. OW is a VR that requires byte swapping within each word when changing between Little Endian and Big Endian byte ordering (see Section 7.3).	not applicable	see Transfer Syntax definition

VR Name	Definition	Character Repertoire	Length of Value
PN Person Name	A character string encoded using a 5 component convention. The character code 5CH (the BACKSLASH "\" in ISO-IR 6) shall not be present, as it is used as the delimiter between values in multiple valued data elements. The string may be padded with trailing spaces. For human use, the five components in their order of occurrence are: family name complex, given name complex, middle name, name prefix, name suffix. Note HL7 prohibits leading spaces within a component; DICOM allows leading and trailing spaces and considers them insignificant. Any of the five components may be an empty string. The component delimiter shall be the caret "^" character (5EH). Delimiters are required for interior null components. Trailing null components and their delimiters may be omitted. Multiple entries are permitted in each component and are encoded as natural text strings, in the format preferred by the named person. For veterinary use, the first two of the five components in their order of occurrence are: responsible party family name or responsible organization name, patient name. The remaining components are not used and shall not be present. This group of five components is referred to as a Person Name component group. For the purpose of writing names in ideographic characters and in phonetic characters, up to 3 groups of components (see Annexes H, I and J) may be used. The delimiter for component groups shall be the equals character "=" (3DH). The three component groups of components in their order of occurrence are: an alphabetic representation, an ideographic representation, and a phonetic representation. Any component group may be absent, including the first component group. In this case, the person name may start with one or more "=" delimiters. Delimiters are required for interior null component groups. Trailing null component groups and their delimiters may be omitted. Precise semantics are defined for each component group. See Section 6.2.1.2. For examples and notes, see Section 6.2.1.1.	Default Character Repertoire and/or as defined by (0008,0005) excluding character code 5CH (the BACKSLASH "\" in ISO-IR 6) and all Control Characters except ESC when used for ISO 2022 escape sequences.	64 chars maximum per component group (see Note in Section 6.2)
SH Short String	A character string that may be padded with leading and/or trailing spaces. The character code 05CH (the BACKSLASH "\" in ISO-IR 6) shall not be present, as it is used as the delimiter between values for multiple data elements. The string shall not have Control Characters except ESC.	Default Character Repertoire and/or as defined by (0008,0005) excluding character code 5CH (the BACKSLASH "\" in ISO-IR 6) and all Control Characters except ESC when used for ISO 2022 escape sequences.	16 chars maximum (see Note in Section 6.2)

VR Name	Definition	Character Repertoire	Length of Value
SL Signed Long	Signed binary integer 32 bits long in 2's complement form. Represents an integer, n, in the range: $-2^{31} <= n <= 2^{31}-1$.	not applicable	4 bytes fixed
SQ Sequence of Items	Value is a Sequence of zero or more Items, as defined in Section 7.5.	not applicable (see Section 7.5)	not applicable (see Section 7.5)
SS Signed Short	Signed binary integer 16 bits long in 2's complement form. Represents an integer n in the range: $-2^{15} <= n <= 2^{15}-1$.	not applicable	2 bytes fixed
ST Short Text	A character string that may contain one or more paragraphs. It may contain the Graphic Character set and the Control Characters, CR, LF, FF, and ESC. It may be padded with trailing spaces, which may be ignored, but leading spaces are considered to be significant. Data Elements with this VR shall not be multi-valued and therefore character code 5CH (the BACKSLASH "\" in ISO-IR 6) may be used.	Default Character Repertoire and/or as defined by (0008,0005) excluding Control Characters except TAB, LF, FF, CR (and ESC when used for ISO 2022 escape sequences).	1024 chars maximum (see Note in Section 6.2)
TM Time	A string of characters of the format HHMMSS.FFFFFF; where HH contains hours (range "00" - "23"), MM contains minutes (range "00" - "59"), SS contains seconds (range "00" - "60"), and FFFFFF contains a fractional part of a second as small as 1 millionth of a second (range "000000" - "999999"). A 24-hour clock is used. Midnight shall be represented by only "0000" since "2400" would violate the hour range. The string may be padded with trailing spaces. Leading and embedded spaces are not allowed. One or more of the components MM, SS, or FFFFFF may be unspecified as long as every component to the right of an unspecified component is also unspecified, which indicates that the value is not precise to the precision of those unspecified components. The FFFFFF component, if present, shall contain 1 to 6 digits. If FFFFFF is unspecified the preceding "." shall not be included. Examples: 1. "070907.0705 " represents a time of 7 hours, 9 minutes and 7.0705 seconds. 2. "1010" represents a time of 10 hours, and 10 minutes. 3. "021 " is an invalid value. Note 1. The ACR-NEMA Standard 300 (predecessor to DICOM) supported a string of characters of the format HH:MM:SS.frac for this VR. Use of this format is not compliant. 2. See also DT VR in this table. 3. The SS component may have a value of 60 only for a leap second.	"0"-"9", "." and the SPACE character of Default Character Repertoire In the context of a Query with range matching (see PS3.4), the character "-" is allowed.	14 bytes maximum In the context of a Query with range matching (see PS3.4), the length is 28 bytes maximum.

VR Name	Definition	Character Repertoire	Length of Value
UC Unlimited Characters	A character string that may be of unlimited length that may be padded with trailing spaces. The character code 5CH (the BACKSLASH "\" in ISO-IR 6) shall not be present, as it is used as the delimiter between values in multiple valued data elements. The string shall not have Control Characters except for ESC.	Default Character Repertoire and/or as defined by (0008,0005) excluding character code 5CH (the BACKSLASH "\" in ISO-IR 6), and all Control Characters except ESC when used for ISO 2022 escape sequences.	2^{32}-2 bytes maximum See Note 2
UI Unique Identifier (UID)	A character string containing a UID that is used to uniquely identify a wide variety of items. The UID is a series of numeric components separated by the period "." character. If a Value Field containing one or more UIDs is an odd number of bytes in length, the Value Field shall be padded with a single trailing NULL (00H) character to ensure that the Value Field is an even number of bytes in length. See Section 9 and Annex B for a complete specification and examples.	"0"-"9", "." of Default Character Repertoire	64 bytes maximum
UL Unsigned Long	Unsigned binary integer 32 bits long. Represents an integer n in the range: $0 <= n < 2^{32}$.	not applicable	4 bytes fixed
UN Unknown	A string of bytes where the encoding of the contents is unknown (see Section 6.2.2).	not applicable	Any length valid for any of the other DICOM Value Representations
UR Universal Resource Identifier or Universal Resource Locator (URI/URL)	A string of characters that identifies a URI or a URL as defined in [RFC 3986]. Leading spaces are not allowed. Trailing spaces shall be ignored. Data Elements with this VR shall not be multi-valued. Note Both absolute and relative URIs are permitted. If the URI is relative, then it is relative to the base URI of the object within which it is contained.	The subset of the Default Character Repertoire required for the URI as defined in IETF RFC3986 Section 2, plus the space (20H) character permitted only as trailing padding. Characters outside the permitted character set must be "percent encoded". Note The Backslash (5CH) character is among those disallowed in URIs.	2^{32}-2 bytes maximum. See Note 2
US Unsigned Short	Unsigned binary integer 16 bits long. Represents integer n in the range: $0 <= n < 2^{16}$.	not applicable	2 bytes fixed
UT Unlimited Text	A character string that may contain one or more paragraphs. It may contain the Graphic Character set and the Control Characters, CR, LF, FF, and ESC. It may be padded with trailing spaces, which may be ignored, but leading spaces are considered to be significant. Data Elements with this VR shall not be multi-valued and therefore character code 5CH (the BACKSLASH "\" in ISO-IR 6) may be used.	Default Character Repertoire and/or as defined by (0008,0005) excluding Control Characters except TAB, LF, FF, CR (and ESC when used for ISO 2022 escape sequences).	2^{32}-2 bytes maximum See Note 2

Note

1. For attributes that were present in ACR-NEMA 1.0 and 2.0 and that have been retired, the specifications of Value Representation and Value Multiplicity provided are recommendations for the purpose of interpreting their values in objects created in accordance with earlier versions of this standard. These recommendations are suggested as most appropriate for a particular attribute; however, there is no guarantee that historical objects will not violate some requirements or specified VR and/or VM.

2. The length of the value of UC, UR and UT VRs is limited only by the size of the maximum unsigned integer representable in a 32 bit VL field minus two, since FFFFFFFFH is reserved and lengths are required to be even.

3. In previous editions of the standard (see PS3.5 2015a), the TAB character was not listed as permitted for the ST, LT and UT VRs. It has been added for the convenience of formatting and the encoding of XML text.

6.2.1 Person Name (PN) Value Representation

6.2.1.1 Examples of PN VR and Notes

Examples:

- Rev. John Robert Quincy Adams, B.A. M.Div.

  "Adams^John Robert Quincy^^Rev.^B.A. M.Div."

  [One family name; three given names; no middle name; one prefix; two suffixes.]

- Susan Morrison-Jones, Ph.D., Chief Executive Officer

  "Morrison-Jones^Susan^^^Ph.D., Chief Executive Officer"

  [Two family names; one given name; no middle name; no prefix; two suffixes.]

- John Doe

  "Doe^John"

  [One family name; one given name; no middle name, prefix, or suffix. Delimiters have been omitted for the three trailing null components.]

- (for examples of the encoding of Person Names using multi-byte character sets see Annex H)

- "Smith^Fluffy"

  [A cat, rather than a human, whose responsible party family name is Smith, and whose own name is Fluffy]

- "ABC Farms^Running on Water"

  [A horse whose responsible organization is named ABC Farms, and whose name is "Running On Water"]

Note

1. A similar multiple component convention is also used by the HL7 v2 XPN data type. However, the XPN data type places the suffix component before the prefix, and has a sixth component "degree" that DICOM subsumes in the name suffix. There are also differences in the manner in which name representation is identified.

2. In typical American and European usage the first occurrence of "given name" would represent the "first name". The second and subsequent occurrences of the "given name" would typically be treated as a middle name(s). The "middle name" component is retained for the purpose of backward compatibility with existing standards.

3. The implementer should remain mindful of earlier usage forms that represented "given names" as "first" and "middle" and that translations to and from this previous typical usage may be required.

4. For reasons of backward compatibility with versions of this standard prior to V3.0, person names might be considered a single family name complex (single component without "^" delimiters).

6.2.1.2 Ideographic and Phonetic Characters in Data Elements with VR of PN

Character strings representing person names are encoded using a convention for PN value representations based on component groups with 5 components.

For the purpose of writing names in ideographic characters and in phonetic characters, up to 3 component groups may be used. The delimiter of the component group shall be the equals character "=" (3DH). The three component groups in their order of occurrence are: an alphabetic representation, an ideographic representation, and a phonetic representation.

Any component group may be absent, including the first component group. In this case, the person name may start with one or more "=" delimiters. Delimiters are also required for interior null component groups. Trailing null component groups and their delimiters may be omitted.

The first component group (identified by DICOM as "alphabetic") shall be encoded using the character set specified by the Attribute Specific Character Set (0008,0005), value 1. If Attribute Specific Character Set (0008,0005) is not present, the default Character Repertoire ISO-IR 6 shall be used. ISO 2022 escapes for Code Extension shall not be used in this component group. When Specific Character Set (0008,0005) value 1 specifies a multi-byte character set without Code Extension (i.e., Unicode in UTF-8, GB18030 or GBK), the characters of this component group may be encoded with multiple bytes, but shall be drawn from the code points U+~~0000~~0020 through U+1FFF of ISO/IEC 10646~~.~~, or the following ISO/IEC 10646 code points:

U+3001, U+3002, U+300C, U+300D, U+3099 through U+309C, and U+30A0 through U+30FF

The second group shall be used for ideographic characters. The character sets used will usually be those from Attribute Specific Character Set (0008,0005), value 2 through n, and may use ISO 2022 escapes.

The third group shall be used for phonetic characters. The character sets used shall be those from Attribute Specific Character Set (0008,0005), value 1 through n, and may use ISO 2022 escapes.

Delimiter characters "^" and "=" are taken from the character set specified by value 1 of the Attribute Specific Character Set (0008,0005). If Attribute Specific Character Set (0008,0005), value 1 is not present, the default Character Repertoire ISO-IR 6 shall be used.

At the beginning of the value of the Person Name data element, the following initial condition is assumed: if Attribute Specific Character Set (0008,0005), value 1 is not present, the default Character Repertoire ISO-IR 6 is invoked, and if the Attribute Specific Character Set (0008,0005), value 1 is present, the character set specified by value 1 of the Attribute is invoked.

At the end of the value of the Person Name data element, and before the component delimiters "^" and "=", the character set shall be switched to the default character repertoire ISO-IR 6, if value 1 of the Attribute Specific Character Set (0008,0005) is not present. If value 1 of the Attribute Specific Character Set (0008,0005) is present, the character set shall be switched to that specified by value 1 of the Attribute.

The value length of each component group is 64 characters maximum, including the delimiter for the component group. Each combining character (e.g., diacritics or vowel marks) shall be considered a separate character for this maximum length, regardless of how an application may display such combining characters (i.e., combined into the glyph for the base character, or rendered separately).

6.2.2 Unknown (UN) Value Representation

The Unknown (UN) VR shall only be used for Private Attribute Data Elements and Standard Data Elements previously encoded as some DICOM VR other than UN using the DICOM Default Transfer Syntax (Implicit VR Little Endian), and whose Value Representation is currently unknown, or whose known Value Representation is none of OB, OD, OF, OL, OW, SQ, UC, UR or UT and whose value length exceeds 65534 (2^{16}-2) and therefore cannot be encoded as a 16-bit unsigned integer in the Value Length Field defined for the known Value Representation (see Section 6.2.1). As long as the VR is unknown the Value Field is insensitive to Little/Big Endian byte ordering and shall not be 'byte-swapped' (see Section 7.3). In the case of undefined length sequences, the value shall remain in implicit VR form. See Section 7.8 for a description of Private Data Attribute Elements and section 10 and Annex A for a discussion of Transfer Syntaxes.

The UN VR shall not be used for Private Creator Data Elements (i.e., the VR is equal to LO, see Section 7.8.1).

The UN VR shall not be used for File Meta Information Data Elements (any Tag (0002,xxxx), see PS3.10).

Note

1. All other (non-default) DICOM Transfer Syntaxes employ explicit VR in their encoding, and therefore any Private and/or Standard Data Element Value Field Attribute value encoded and decoded using any Transfer Syntax other than the default, and not having been translated to the DICOM Default Transfer Syntax default in the interim, will have a known VR.

2. If at some point an application knows the actual VR for an Attribute of VR UN (e.g., has its own applicable data dictionary), it can assume that the Value Field of the Attribute is encoded in Little Endian byte ordering with implicit VR encoding, irrespective of the current Transfer Syntax.

3. This VR of UN is needed when an explicit VR must be given to a Data Element whose Value Representation is unknown (e.g., store and forward). ~~UN is a means to explicitly indicate that the Value Representation of a Data Element is unknown.~~

4. This VR of UN is also needed for the encoding of Data Elements with explicit VR whose value length exceeds 65534 (2^{16}-2) (FFFEH, the largest even length unsigned 16 bit number) but which are defined to have a 16 bit explicit VR length field.

5. The length field of the Value Representation of UN may contain the value of ~~"unknown length"~~Undefined Length, in which case the contents can be assumed to be encoded with implicit VR. See Section 7.5.1 to determine how to parse Data Elements with an ~~unknown length~~Undefined Length.

6. An example of a Standard Data Element using a UN VR is a Type 3 or Type U Standard Attribute added to an SOP Class definition. An existing application that does not support that new Attribute (and encounters it) could convert the VR to UN.

6.2.3 URI/URL (UR) Value Representation

The URI/URL (UR) VR uses a subset of the Default Character Repertoire as defined in [RFC 3986], and shall not use any code extension or replacement techniques. URI/URL domain name components that in their original form use characters outside the permitted character set shall use the Internationalized Domain Names for Applications encoding in accordance with IETF RFC5890 and RFC5891. Other URI/URL content that uses characters outside the permitted character set shall use the Internationalized Resource Identifiers encoding mechanism of IETF RFC 3987, representing the content string in UTF-8 and percent encoding characters as required.

Note

For example, the use of a patient name in a URI/URL string may require use of the [RFC 3987] technique.

6.3 Enumerated Values and Defined Terms

The value of certain Data Elements may be chosen among a set of explicit Values satisfying its VR. These explicit Values are either Enumerated Values or Defined Terms and are specified in PS3.3 and PS3.4.

Enumerated Values are used when the specified explicit Values are the only Values allowed for a Data Element. A Data Element with Enumerated Values that does not have a Value equivalent to one of the Values specified in this standard has an invalid value within the scope of a specific Information Object/SOP Class definition.

Note

1. Patient Sex (0010, 0040) is an example of a Data Element having Enumerated Values. It is defined to have a Value that is either "M", "F", or "O" (see PS3.3). No other Value shall be given to this Data Element.

2. Future modifications of this standard may add to the set of allowed values for Data Elements with Enumerated Values. Such additions by themselves may or may not require a change in SOP Class UIDs, depending on the semantics of the Data Element.

Defined Terms are used when the specified explicit Values may be extended by implementers to include additional new Values. These new Values shall be specified in the Conformance Statement (see PS3.2) and shall not have the same meaning as currently defined Values in this standard. A Data Element with Defined Terms that does not contain a Value equivalent to one of the Values currently specified in this standard shall not be considered to have an invalid value. An empty (zero length) value is not a valid new Value for a Defined Term; empty values shall be considered invalid unless the standard specifically permits empty values. New Values shall

not have a meaning of unknown, since that concept, if permitted by the standard, shall be conveyed explicitly either by allowing the Data Element to be zero length or by provision of a standard Defined Term with such a meaning.

Note

1. Reporting Priority (0040,1009) is an example of a Data Element having Defined Terms. It is defined to have a Value that may be one of the set of standard Values; HIGH, ROUTINE, MEDIUM, or LOW (see PS3.3). Because this Data Element has Defined Terms other reporting priorities may be defined by the implementer.

2. The validity of empty values is usually specified by the attribute being defined as Type 2 (see Section 7.4.3). However, in the context of a required Type 1 attribute with multiple values, some (but not all) values may be allowed to be empty (see Section 7.4.1); in this case the standard explicitly specifies the validity of empty values in the list of Defined Terms for each value. Specific Character Set (0008,0005) is an example of a Data Element for which the standard specifically permits the first value to be empty when multiple values are present. Image Type (0008,0008) is an example of a Data Element that in some IODs defined in PS3.3 is required to be present with multiple values, but if an empty value is not explicitly listed in the Defined Terms for Value 3 by an IOD an empty value is invalid.

The Value Representation may affect the interpretation of Defined Terms and Enumerated Values for numeric values. For binary Value Representations, the textual representation of the Value in the standard does not affect the interpretation. For string Value Representations (IS and DS), the meaning of the Value in the standard shall be used, not the literal string.

Note

For example, an Enumerated Value of "1" expressed in the text of the standard matches an IS or DS value encoded as "001", or a DS value encoded as "1.0" or "1." or "1.0000E+00" or any permitted encoding. Leading and trailing spaces are defined in Table 6.2-1 not to be significant and hence do not affect the interpretation.

6.4 Value Multiplicity (VM) and Delimitation

The Value Multiplicity of a Data Element specifies the number of Values that can be encoded in the Value Field of that Data Element. The VM of each Data Element is specified explicitly in PS3.6. If the number of Values that may be encoded in an element is variable, it shall be represented by two numbers separated by a dash; e.g., "1-10" means that there may be 1 to 10 Values in the element.

Note

Elements having a multiplicity of "S", which represented "single", in versions of this standard preceding V3.0, will have a multiplicity of "1" in this version of this standard.

When a Data Element has multiple Values, those Values shall be delimited as follows:

- For character strings, the character 5CH (BACKSLASH "\" in the case of the repertoire ISO IR-6) shall be used as a delimiter between Values.

  Note

  BACKSLASH ("\") is used as a delimiter between character string Values that are of fixed length as well as variable length.

- Multiple binary Values of fixed length shall be a series of concatenated Values without any delimiter.

Each string Value in a multiple valued character string may be of even or odd length, but the length of the entire Value Field (including "\" delimiters) shall be of even length. If padding is required to make the Value Field of even length, a single padding character shall be applied to the end of the Value Field (to the last Value), in which case the length of the last Value may exceed the Length of Value by 1.

Note

A padding character may need to be appended to a fixed length character string value in the above case.

Only the last UID Value in a multiple valued Data Element with a VR of UI shall be padded with a single trailing NULL (00H) character when necessary to ensure that the entire Value Field (including "\" delimiters) is of even length.

Data Elements with a VR of ~~SQ, OF, OD~~OB, OD, OF, OL, OW, OB, SQ, UN or UR shall always have a Value Multiplicity of one.

7 The Data Set

A Data Set represents an instance of a real world Information Object. A Data Set is constructed of Data Elements. Data Elements contain the encoded Values of Attributes of that object. The specific content and semantics of these Attributes are specified in Information Object Definitions (see PS3.3).

The construction, characteristics, and encoding of a Data Set and its Data Elements are discussed in this section. Pixel Data, Overlays, and Curves are Data Elements whose interpretation depends on other related elements.

7.1 Data Elements

A Data Element is uniquely identified by a Data Element Tag. The Data Elements in a Data Set shall be ordered by increasing Data Element Tag Number and shall occur at most once in a Data Set.

Note

A Data Element Tag may occur again within Nested Data Sets (see Section 7.5).

Two types of Data Elements are defined:

- Standard Data Elements have an even Group Number that is not (0000,eeee), (0002,eeee), (0004,eeee), or (0006,eeee).

  Note

  Usage of these groups is reserved for DIMSE Commands (see PS3.7) and DICOM File Formats.

- Private Data Elements have an odd Group Number that is not (0001,eeee), (0003,eeee), (0005,eeee), (0007,eeee), or (FFFF,eeee). Private Data Elements are discussed further in Section 7.8.

  Note

  Although similar or related Data Elements often have the same Group Number; a Data Group does not convey any semantic meaning beginning with DICOM Version 3.0.

A Data Element shall have one of three structures. Two of these structures contain the VR of the Data Element (Explicit VR) but differ in the way their lengths are expressed, while the other structure does not contain the VR (Implicit VR). All three structures contain the Data Element Tag, Value Length and Value for the Data Element. See Figure 7.1-1.

Implicit and Explicit VR Data Elements shall not coexist in a Data Set and Data Sets nested within it (see Section 7.5). Whether a Data Set uses Explicit or Implicit VR, among other characteristics, is determined by the negotiated Transfer Syntax (see Section 10 and Annex A).

Note

VRs are not contained in Data Elements when using DICOM Default Transfer Syntax (DICOM Implicit VR Little Endian Transfer Syntax).

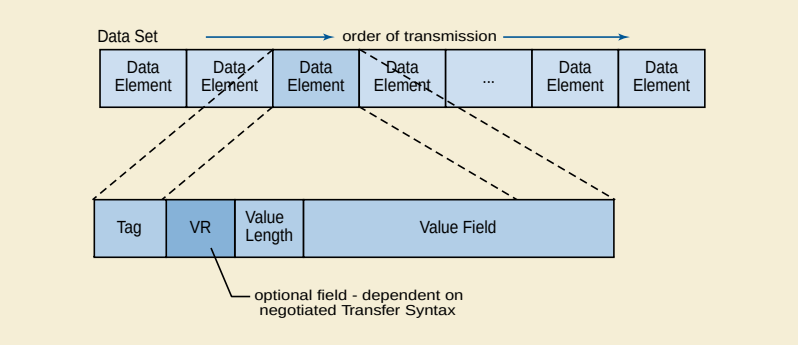

Figure 7.1-1. DICOM Data Set and Data Element Structures

7.1.1 Data Element Fields

A Data Element is made up of fields. Three fields are common to all three Data Element structures; these are the Data Element Tag, Value Length, and Value Field. A fourth field, Value Representation, is only present in the two Explicit VR Data Element structures. The Data Element structures are defined in Section 7.1.2 and Section 7.1.3. The definitions of the fields are:

Data Element Tag An ordered pair of 16-bit unsigned integers representing the Group Number followed by Element Number.

Value Representation A two-byte character string containing the VR of the Data Element. The VR for a given Data Element Tag shall be as defined by the Data Dictionary as specified in PS3.6. The two character VR shall be encoded using characters from the DICOM default character set.

Value Length Either:

- a 16 or 32-bit (dependent on VR and whether VR is explicit or implicit) unsigned integer containing the Explicit Length of the Value Field as the number of bytes (even) that make up the Value. It does not include the length of the Data Element Tag, Value Representation, and Value Length Fields.
- a 32-bit Length Field set to Undefined Length (FFFFFFFFH). Undefined Lengths may be used for Data Elements having the Value Representation (VR) Sequence of Items (SQ) and Unknown (UN). For Data Elements with Value Representation OW or OB Undefined Length may be used depending on the negotiated Transfer Syntax (see Section 10 and Annex A).

  Note

  The decoder of a Data Set should support both Explicit and Undefined Lengths for VRs of SQ and UN and, when applicable, for VRs of OW and OB.

Value Field An even number of bytes containing the Value(s) of the Data Element.

The data type of Value(s) stored in this field is specified by the Data Element's VR. The VR for a given Data Element Tag can be determined using the Data Dictionary in PS3.6, or using the VR Field if it is contained explicitly within the Data Element. The VR of Standard Data Elements shall agree with those specified in the Data Dictionary.

The Value Multiplicity specifies how many Values with this VR can be placed in the Value Field. If the VM is greater than one, multiple values shall be delimited within the Value Field as defined previously in Section 6.4. The VMs of Standard Data Elements are specified in the Data Dictionary in PS3.6.

Value Fields with Undefined Length are delimited through the use of Sequence Delimitation Items and Item Delimitation Data Elements, which are described further in Section 7.5.

7.1.2 Data Element Structure with Explicit VR

When using the Explicit VR structures, the Data Element shall be constructed of four consecutive fields: Data Element Tag, VR, Value Length, and Value. Depending on the VR of the Data Element, the Data Element will be structured in one of two ways:

- for VRs of OB, ~~OW~~OD, OF, ~~OD~~,OL, OW, SQ and UN the 16 bits following the two character VR Field are reserved for use by later versions of the DICOM Standard. These reserved bytes shall be set to 0000H and shall not be used or decoded (Table 7.1-1). The Value Length Field is a 32-bit unsigned integer. If the Value Field has an Explicit Length, then the Value Length Field shall contain a value equal to the length (in bytes) of the Value Field. Otherwise, the Value Field has an Undefined Length and a Sequence Delimitation Item marks the end of the Value Field.
- for VRs of UC, UR and UT the 16 bits following the two character VR Field are reserved for use by later versions of the DICOM Standard. These reserved bytes shall be set to 0000H and shall not be used or decoded. The Value Length Field is a 32-bit unsigned integer. The Value Field is required to have an Explicit Length, that is the Value Length Field shall contain a value equal to the length (in bytes) of the Value Field.

  Note

  VRs of UC, UR and UT may not have an Undefined Length, i.e.,a Value Length of FFFFFFFFH.
- for all other VRs the Value Length Field is the 16-bit unsigned integer following the two character VR Field (Table 7.1-2). The value of the Value Length Field shall equal the length of the Value Field.

Table 7.1-1. Data Element with Explicit VR of OB, OD, OF, OL, OW, SQ, UC, UR, UT or UN

Tag		VR		Value Length	Value
Group Number (16-bit unsigned integer)	Element Number (16-bit unsigned integer)	VR (2 byte character string) of "OB", "~~OW~~OD", "OF", "~~OD~~OL", "OW", "SQ", "UC", "UR", "UT" or "UN"	Reserved (2 bytes) set to a value of 0000H	32-bit unsigned integer	Even number of bytes containing the Data Element Value(s) encoded according to the VR and negotiated Transfer Syntax. Delimited with Sequence Delimitation Item if of Undefined Length.
2 bytes	2 bytes	2 bytes	2 bytes	4 bytes	'Value Length' bytes if of Explicit Length

Table 7.1-2. Data Element with Explicit VR other than as shown in Table 7.1-1

Tag		VR	Value Length	Value
Group Number (16-bit unsigned integer)	Element Number (16-bit unsigned integer)	VR (2 byte character string)	(16-bit unsigned integer)	Even number of bytes containing the Data Element Value(s) encoded according to the VR and negotiated Transfer Syntax.
2 bytes	2 bytes	2 bytes	2 bytes	'Value Length' bytes

7.1.3 Data Element Structure with Implicit VR

When using the Implicit VR structure the Data Element shall be constructed of three consecutive fields: Data Element Tag, Value Length, and Value (see Table 7.1-3). If the Value Field has an Explicit Length then the Value Length Field shall contain a value equal to the length (in bytes) of the Value Field. Otherwise, the Value Field has an Undefined Length and a Sequence Delimitation Item marks the end of the Value Field.

Table 7.1-3. Data Element with Implicit VR

Tag		Value Length	Value
Group Number (16-bit unsigned integer)	Element Number (16-bit unsigned integer)	32-bit unsigned integer	Even number of bytes containing the Data Elements Value encoded according to the VR specified in PS3.6 and the negotiated Transfer Syntax. Delimited with Sequence Delimitation Item if of Undefined Length.
2 bytes	2 bytes	4 bytes	'Value Length' bytes or Undefined Length

7.2 Group Length

Group Length (gggg,0000) Standard Data Elements have been retired. See PS3.5-2007.

All implementations shall be able to parse Group Length elements, and may discard and not insert or re-insert them; if present they shall be consistent with the encoding of the Data Set even if the transfer syntax is changed resulting in a change in the actual length of a group of elements. No implementation shall require the presence of Group Length elements.

Note

1. Elements in groups 0, 2, 4 and 6 are not Standard Data Elements. Mandatory requirements for Group Length for groups 0 and 2 are specified elsewhere in the standard.

2. It is recommended that Group Length elements be removed during storage or transfer in order to avoid the risk of inconsistencies arising during coercion of data element values and changes in transfer syntax.

7.3 Big Endian Versus Little Endian Byte Ordering

Another component of the encoding of a Data Set that shall be agreed upon by communicating Application Entities is the Byte Ordering.

Little Endian byte ordering is defined as follows:

- In a binary number consisting of multiple bytes (e.g., a 32-bit unsigned integer value, the Group Number, the Element Number, etc.), the least significant byte shall be encoded first; with the remaining bytes encoded in increasing order of significance.
- In a character string consisting of multiple 8-bit single byte codes, the characters will be encoded in the order of occurrence in the string (left to right).

Big Endian byte ordering is defined as follows:

- In a binary number consisting of multiple bytes, the most significant byte shall be encoded first; with the remaining bytes encoded in decreasing order of significance.
- In a character string consisting of multiple 8-bit single byte codes, the characters will be encoded in the order of occurrence in the string (left to right).

Note

The packing of bits within values of OB, OL or OW Value representation for Pixel Data and Overlay Data is described in Section 8.

Byte ordering is a component of an agreed upon Transfer Syntax (see Section 10). The default DICOM Transfer Syntax, which shall be supported by all AEs, uses Little Endian encoding and is specified in Section A.1. Alternate Transfer Syntaxes, some of which use Big Endian encoding, are also specified in Annex A.

Note

The Command Set structure as specified in PS3.7 is encoded using the Little Endian Implicit VR Transfer Syntax.

In the default case of Little Endian encoding, Big Endian Machines interpreting Data Sets shall do 'byte swapping' before interpreting or operating on certain Data Elements. The Data Elements affected are all those having VRs that are multiple byte Values and that are not a character string of 8-bit single byte codes. VRs constructed of a string of characters of 8-bit single byte codes are really constructed of a string of individual bytes, and are therefore not affected by byte ordering. The VRs that are not a string of characters and consist of multiple bytes are:

- 2-byte US, SS, OW and each component of AT
- 4-byte OF, OL, UL, SL, and FL
- 8 byte OD, FD

Note

For the above VRs, the multiple bytes are presented in increasing order of significance when in Little Endian format. For example, an 8-byte Data Element with VR of FD, might be written in hexadecimal as 68AF4B2CH, but encoded in Little Endian would be 2C4BAF68H.

7.4 Data Element Type

An attribute, encoded as a Data Element, may or may not be required in a Data Set, depending on that Attribute's Data Element Type.

The Data Element Type of an Attribute of an Information Object Definition or an Attribute of a SOP Class Definition is used to specify whether that Attribute is mandatory or optional. The Data Element Type also indicates if an Attribute is conditional (only mandatory under certain conditions). The Data Element Types of Attributes of Composite IODs are specified in PS3.3. The Data Element Types of Attributes of Normalized IODs are specified as Attributes of SOP Classes in PS3.4.

7.4.1 Type 1 Required Data Elements

IODs and SOP Classes define Type 1 Data Elements that shall be included and are mandatory elements. The Value Field shall contain valid data as defined by the elements VR and VM as specified in PS3.6. The Length of the Value Field shall not be zero. Absence of a valid Value in a Type 1 Data Element is a protocol violation.

Note

1. For data elements with a string (CS, SH, LO) rather than binary, text or sequence Value Representation, and for which multiple Values are allowed, the presence of a single Value is sufficient to satisfy the Type 1 requirement, unless specified otherwise in the Attribute description, and other Values may be empty, unless otherwise specified by the IOD. The presence of one or more delimiter (BACKSLASH) characters alone, without any Values, is not sufficient to satisfy the Type 1 requirement, since even though the Value Length is greater than zero, there is no valid Value present.

2. A Type 1 Sequence Data Element will contain one or more Items, as defined by the IOD (irrespective of the VM of the Sequence, which is always one (Section 7.5)). Whether or not those Items may be empty (contain no Data Elements) depends on the IOD definition of the Data Set for each Item.

7.4.2 Type 1C Conditional Data Elements

IODs and SOP Classes define Data Elements that shall be included under certain specified conditions. Type 1C elements have the same requirements as Type 1 elements under these conditions. It is a protocol violation if the specified conditions are met and the Data Element is not included.

When the specified conditions are not met, Type 1C elements shall not be included in the Data Set.

7.4.3 Type 2 Required Data Elements

IODs and SOP Classes define Type 2 Data Elements that shall be included and are mandatory Data Elements. However, it is permissible that if a Value for a Type 2 element is unknown it can be encoded with zero Value Length and no Value. If the Value is known the Value Field shall contain that value as defined by the elements VR and VM as specified in PS3.6. These Data Elements shall be included in the Data Set and their absence is a protocol violation.

Note

1. The intent of Type 2 Data Elements is to allow a zero length to be conveyed when the operator or application does not know its value or has a specific reason for not specifying its value. It is the intent that the device should support these Data Elements.

2. A Type 2 Sequence Data Element will contain zero or more Items, as defined by the IOD (irrespective of the VM of the Sequence, which is always one (Section 7.5)). An empty Type 2 Sequence is one with no Items, as opposed to an Item that is present but empty. Whether or not Items may be empty (contain no Data Elements) depends on the IOD definition of the Data Set for each Item, rather than the Type of the enclosing Sequence Data Element.

7.4.4 Type 2C Conditional Data Elements

IODs and SOP Classes define Type 2C elements that have the same requirements as Type 2 elements under certain specified conditions. It is a protocol violation if the specified conditions are met and the Data Element is not included.

When the specified conditions are not met, Type 2C elements shall not be included in the Data Set.

Note

An example of a Type 2C Data Element is Inversion Time (0018,0082). For several SOP Class Definitions, this Data Element is required only if the Scanning Sequence (0018,0020) has the Value "IR." It is not required otherwise. See PS3.3.

7.4.5 Type 3 Optional Data Elements

IODs and SOP Classes define Type 3 Data Elements that are optional Data Elements. Absence of a Type 3 element from a Data Set does not convey any significance and is not a protocol violation. Type 3 elements may also be encoded with zero length and no Value. The meaning of a zero length Type 3 Data Element shall be precisely the same as that element being absent from the Data Set.

7.4.6 Data Element Types Within A Sequence

When an IOD defines a Sequence Data Element (see Section 7.5), the Type of the Sequence attribute defines whether the Sequence attribute itself must be present, and the Attribute Description of the Sequence attribute may define whether and how many Items shall be present in the Sequence. The Types of the attributes of the Data Set included in the Sequence, including any conditionality, are specified within the scope of each Data Set, i.e., for each Item present in the Sequence.

Note

1. The Type and Attribute Description of the Sequence determines whether Items are present; conditionality constraints on Data Elements of the Items cannot force an Item to be present.

2. Historically, many IODs declared Type 1 and Type 2 Data Elements of the Sequence to be Type 1C and Type 2C, respectively, with the condition that an Item is present. This is exactly the same as simply defining them as Type 1 and Type 2.

3. In particular, the conditionality constraint "Required if Sequence is sent" on the Type 1C or Type 2C Data Elements subsidiary to a Type 2 or 3 Sequence attribute does not imply that an Item must be present in the Sequence. These conditions are meant to be equivalent to "Required if a Sequence Item is present", and the conditionality is not strictly necessary. Any Type 2 or Type 3 Sequence attribute may be sent with zero length.

4. In particular, the conditionality constraint "Required if <name-of-parent-sequence-attribute> is sent" on the Type 1C or Type 2C Data Elements subsidiary to a Type 2 or 3 Sequence attribute does not imply that an Item must be present in the Sequence. These conditions are meant to be equivalent to "Required if a Sequence Item is present", and the conditionality is not strictly necessary. Any Type 2 or Type 3 Sequence attribute may be sent with zero length.

7.5 Nesting of Data Sets

The VR identified "SQ" shall be used for Data Elements with a Value consisting of a Sequence of zero or more Items, where each Item contains a set of Data Elements. SQ provides a flexible encoding scheme that may be used for simple structures of repeating sets of Data Elements, or the encoding of more complex Information Object Definitions often called folders. SQ Data Elements can also be used recursively to contain multi-level nested structures.

Items present in an SQ Data Element shall be an ordered set where each Item may be referenced by its ordinal position. Each Item shall be implicitly assigned an ordinal position starting with the value 1 for the first Item in the Sequence, and incremented by 1 with each subsequent Item. The last Item in the Sequence shall have an ordinal position equal to the number of Items in the Sequence.

Note

1. This clause implies that item ordering is preserved during transfer and storage.

2. An IOD or Module Definition may choose to not use this ordering property of a Data Element with VR of SQ. This is simply done by not specifying any specific semantics to the ordering of Items, or by not specifying usage of the referencing of Items by ordering position.

The definition of the Data Elements encapsulated in each Item is provided by the specification of the Data Element (or associated Attribute) of Value Representation SQ. Items in a sequence of Items may or may not contain the same set of Data Elements. Data Elements with a VR of SQ may contain multiple Items but shall always have a Value Multiplicity of one (i.e., a single Sequence).

There are three special SQ related Data Elements that are not ruled by the VR encoding rules conveyed by the Transfer Syntax. They shall be encoded as Implicit VR. These special Data Elements are Item (FFFE,E000), Item Delimitation Item (FFFE,E00D), and Sequence Delimitation Item (FFFE,E0DD). However, the Data Set within the Value Field of the Data Element Item (FFFE,E000) shall be encoded according to the rules conveyed by the Transfer Syntax.

7.5.1 Item Encoding Rules

Each Item of a Data Element of Value Representation SQ shall be encoded as a DICOM Standard Data Element with a specific Data Element Tag of Value (FFFE,E000). The Item Tag is followed by a 4 byte Item Length field encoded in one of the following two ways:

a. Explicit Length: The number of bytes (even) contained in the Sequence Item Value (following but not including the Item Length Field) is encoded as a 32-bit unsigned integer value (see Section 7.1). This length shall include the total length of all Data Elements conveyed by this Item. This Item Length shall be equal to 00000000H if the Item contains no Data Set.

b. Undefined Length: The Item Length Field shall contain the value FFFFFFFFH to indicate an undefined Item length. It shall be used in conjunction with an Item Delimitation Data Element. This Item Delimitation Data Element has a Data Element Tag of (FFFE,E00D) and shall follow the Data Elements encapsulated in the Item. No Value shall be present in the Item Delimitation Data Element and its Length shall be 00000000H. An Item containing no Data Set is encoded by an Item Delimitation Data Element only.

The encoder of a Data Set may choose either one of the two ways of encoding. Both ways of encoding shall be supported by decoders of Data Sets. Data Element Tags (FFFF,eeee) are reserved by this standard and shall not be used.

Each Item Value shall contain a DICOM Data Set composed of Data Elements. Within the context of each Item, these Data Elements shall be ordered by increasing Data Element Tag value and appear only once (as Data Set is defined in Section 7.1). There is no relationship between the ordering of the Data Elements contained within an Item and the ordering of the Data Element Tag of SQ Value Representation that contains that Item. One or more Data Elements in an Item may be of Value Representation SQ, thus allowing for recursion.

Data Elements with a group of 0000, 0002 and 0006 shall not be present within Sequence Items.

Note

The use of Transfer Syntax UID (0002,0010) in particular is forbidden, since were it to differ from the Transfer Syntax of the enclosing Data Set then a change in encoding would be implied, which is not allowed.

Section 7.8 specifies rules for incorporating Private Data Elements into Sequence Items.

7.5.2 Delimitation of The Sequence of Items

Delimitation of the last Item of a Sequence of Items, encapsulated in a Data Element of Value Representation SQ, shall be in one of the two following ways:

a. Explicit Length: The number of bytes (even) contained in the Data Element Value (following but not including the Data Element Length Field) is encoded as a 32-bit unsigned integer value (see Section 7.1). This length shall include the total length resulting from the sequence of zero or more items conveyed by this Data Element. This Data Element Length shall be equal to 00000000H if the sequence of Items contains zero Items.

b. Undefined Length: The Data Element Length Field shall contain a Value FFFFFFFFH to indicate an Undefined Sequence length. It shall be used in conjunction with a Sequence Delimitation Item. A Sequence Delimitation Item shall be included after the last Item in the sequence. Its Item Tag shall be (FFFE,E0DD) with an Item Length of 00000000H. No Value shall be present. A Sequence containing zero Items is encoded by a Sequence Delimitation Item only.

The encoder of a Sequence of Items may choose either one of the two ways of encoding. Both ways of encoding shall be supported by decoders of the Sequence of Items.

Note

The Sequence Delimitation Item Tag (FFFE,E0DD) is different from the Item Delimitation Tag (FFFE,E00D) introduced above in that it indicates the end of a Sequence of Items whose Length was left undefined. If an undefined length Item is the last Item of a Sequence of Items of undefined length, then an Item Delimitation Tag will be followed by a Sequence Delimitation Tag.

For an example of an SQ Data Element of Explicit Length encapsulating Items of Explicit Length see Table 7.5-1.

For an example of an SQ Data Element of Undefined Length encapsulating Items of Explicit Length see Table 7.5-2.

For an example of an SQ Data Element of Undefined Length encapsulating Items of both Explicit and Undefined Length see Table 7.5-3.

Table 7.5-1. Example of a Data Element with Implicit VR Defined as a Sequence of Items (VR = SQ) with Three Items of Explicit Length

Data Element Tag	**Data Element Length**	**Data Element Value**								
(gggg, eeee) with VR of SQ	0000 0F00H	**First Item**			**Second Item**			**Third Item**		
		Item Tag (FFFE, E000)	Item Length 0000 04F8H	Item Value Data Set	Item Tag (FFFE, E000)	Item Length 0000 04F8H	Item Value Data Set	Item Tag (FFFE, E000)	Item Length 0000 04F8H	Item Value Data Set
4 bytes	4 bytes	4 bytes	4 bytes	04F8H bytes	4 bytes	4 bytes	04F8H bytes	4 bytes	4 bytes	04F8H bytes

Table 7.5-2. Example of a Data Element with Explicit VR Defined as a Sequence of Items (VR = SQ) of Undefined Length, Containing Two Items of Explicit Length

Data Element Tag	**Value Representation**		**Data Element Length**	**Data Element Value**							
(gggg, eeee) with VR of SQ	SQ	0000H Reserved	FFFF FFFFH undefined length	**First Item**			**Second Item**			**Sequence Delimitation Item**	
				Item Tag (FFFE, E000)	Item Length 98A5 2C68H	Item Value Data Set	Item Tag (FFFE, E000)	Item Length B321 762CH	Item Value Data Set	Seq. Delim. Tag (FFFE, E0DD)	Item Length 0000 0000H
4 bytes	2 bytes	2 bytes	4 bytes	4 bytes	4 bytes	98A5 2C68H bytes	4 bytes	4 bytes	B321 762CH bytes	4 bytes	4 bytes

Note

The Data Set within the Item Values in Table 7.5-2 have VRs Explicitly defined.

Table 7.5-3. Example of a Data Element with Implicit VR Defined as a Sequence of Items (VR = SQ) of Undefined Length, Containing Two Items Where One Item is of Explicit Length and the Other Item is of Undefined Length

Data Element Tag	Data Element Length	*Data Element Value*									
(gggg, eeee) with VR of SQ	FFFF FFFFH undefined length	**First Item**			**Second Item**					**Sequence Delimitation Item**	
		Item Tag (FFFE, E000)	Item Length 0000 17B6H	Item Value Data Set	Item Tag (FFFE, E000)	Item Length FFFF FFFFH undefined length	Item Value Data Set	Item Delim. Tag (FFFE, E00D)	Length 0000 0000H	Seq. Delim. Tag (FFFE, E0DD)	Item Length 0000 0000H
4 bytes	4 bytes	4 bytes	4 bytes	17B6H bytes	4 bytes	4 bytes	undefined length	4 bytes	4 bytes	4 bytes	4 bytes

7.5.3 Sequence Inheritance

An encapsulated Data Set shall only include the Specific Character Set (0008,0005) data element if the Attribute Specific Character Set is defined in the IOD for that sequence of items.

Note

An encapsulated Data Set does not include the Specific Character Set data element unless the Specific Character Set Attribute is defined as part of the IOD for that sequence.

If an encapsulated Data Set includes the Specific Character Set Attribute, it shall apply only to the encapsulated Data Set. If the Attribute Specific Character Set is not explicitly included in an encapsulated Data Set, then the Specific Character Set value of the encapsulating Data Set applies.

7.6 Repeating Groups

Multiple Overlay Planes and Curves are often associated with a single Image (see PS3.3). Standard Data Elements with even Group Numbers (5000-501E,eeee) represent Curves, while elements with even Group Numbers (6000-601E,eeee) represent Overlay Planes. Both of these ranges of Group numbers are known as Repeating Groups. This use of group numbers is a remnant of versions of this standard prior to V3.0 that associated a semantic meaning with particular Groups.

In each of these ranges of Group Numbers, Standard Data Elements that have identical Element Numbers have the same meaning within each Group (and the same VR, VM, and Data Element Type). The notation (50xx,eeee) and (60xx,eeee) are used for the Group Number in Data Element Tags when referring to a common Data Element across these groups (see PS3.6). Groups (50xx,eeee) and (60xx,eeee) are called Repeating Groups because of these characteristics.

Repeating Groups shall only be allowed in the even Groups (6000-601E,eeee) and even Groups (5000-501E,eeee) cases. In the future, Data Elements with VRs of SQ shall be used to serve a similar purpose.

Note

Private Groups in the odd Groups (5001-501F,eeee) and (6001-601F,eeee) may still be used, but there is no implication of repeating semantics, nor any implied shadowing of the standard repeating groups.

7.7 Retired Data Elements

Certain Data Elements are no longer supported beginning with Version 3.0 of this standard. These Data Elements are retired and are denoted as such (RET) in the VR column in PS3.6. Implementations may continue to support these Data Elements for the purpose of backward compatibility with versions of this standard prior to V3.0, but this is not a requirement of this standard. If a retired Data Element is used it must contain valid data as specified in versions of this standard prior to V3.0. Any other use of a retired Data Element,

and its associated Data Element Tag, is reserved by this standard. Retired Data Element Tags shall not be redefined in later versions of this standard.

7.8 Private Data Elements

Implementations may require communication of information that cannot be contained in Standard Data Elements. Private Data Elements are intended to be used to contain such information. Such Private Data Elements shall not change the semantics of the Information Object Definition or SOP Class Definition.

Private Data Elements have the same structure as Standard Data Elements specified earlier in Section 7.1 (i.e., Data Element Tag field, optional VR field, length field, and value field). The Group Number used in the Element Tag of Private Data Elements shall be an odd number. Private Data Elements shall be contained in the Data Set in increasing numeric order of Data Element Tag. The Value Field of a Private data element shall have one of the VRs specified by this standard in Section 6.2.

For each Information Object Definition or SOP Class Definition, certain Data Elements are required (Data Element Type 1, 1C, 2, or 2C) as specified in PS3.3 and PS3.4. Private Data Elements shall not be used in place of required Standard Data Elements.

7.8.1 Private Data Element Tags

It is possible that multiple implementers may define Private Elements with the same (odd) group number. To avoid conflicts, Private Elements shall be assigned Private Data Element Tags according to the following rules.

a. Private Creator Data Elements numbered (gggg,0010-00FF) (gggg is odd) shall be used to reserve a block of Elements with Group Number gggg for use by an individual implementer. The implementer shall insert an identification code in the first unused (unassigned) Element in this series to reserve a block of Private Elements. The VR of the private identification code shall be LO (Long String) and the VM shall be equal to 1.

b. Private Creator Data Element (gggg,0010), is a Type 1 Data Element that identifies the implementer reserving element (gggg,1000-10FF), Private Creator Data Element (gggg,0011) identifies the implementer reserving elements (gggg,1100-11FF), and so on, until Private Creator Data Element (gggg,00FF) identifies the implementer reserving elements (gggg,FF00-FFFF).

c. Encoders of Private Data Elements shall be able to dynamically assign private data to any available (unreserved) block(s) within the Private group, and specify this assignment through the blocks corresponding Private Creator Data Element(s). Decoders of Private Data shall be able to accept reserved blocks with a given Private Creator identification code at any position within the Private group specified by the blocks corresponding Private Creator Data Element.

   Note

   1. The versions of this standard prior to V3.0 described shadow groups. These were groups with a group number one greater than the standard groups. Elimination of conflicts in Private Data Element Tags have made this distinction obsolete and this terminology has been retired.

   2. The versions of this standard prior to V3.0 specified private group element numbers (gggg,10FF-7FFF) reserved for manufacturers and private group element numbers (gggg, 8100-FFFF) reserved for users. Elimination of conflicts in Private Data Element Tags has made this distinction obsolete and this specification has been retired.

   3. The requirements of this section do not allow any use of elements in the ranges (gggg,0001-000F) and (gggg,0100-0FFF) where gggg is odd.

d. Elements with Tags (0001,xxxx), (0003,xxxx), (0005,xxxx), and (0007,xxxx) shall not be used.

e. Whether or not Private Data Elements contain identifying information related to de-identification is defined by the Private Data Element Characteristics Sequence (0008,0300). See PS3.3 Section C.12.1.

Since each Item within a sequence is a self contained Data Set (see Section 7.5 on the nesting of Data Sets via Sequences of Items), any Item that contains Private Data Elements shall also have Private Creator Data Elements reserving blocks of Elements for those Private Data Elements. The scope of the reservation is just within the Item. Items do not inherit the Private Data Element reservations made by Private Creator Data Elements in the Data Set in which the Item is nested.

Note

1. If a sequence is itself a Private Data Element and the Items within the sequence also have Private Data Elements, then there will be Private Creator Data Elements both outside the sequence and within the sequence Items.

2. Different Items may reserve the same block of Private Data Elements for different private creators. This is necessary to allow the nesting of Data Sets collected from multiple sources into folders.

7.8.2 Encoding of Private Elements

The Value Representations used for Private Data Elements shall be the same as those VRs specified for Standard Data Elements in Section 6.2. The encoding shall conform to the requirements for those VRs and shall be in accordance with the negotiated Transfer Syntax. A Private Data Element with SQ VR (a Private Data Sequence) may include Items with both Standard and Private Data Elements. Standard Data Elements used within a Private Data Sequence shall use the VRs as defined in PS3.6 for those data elements.

The semantics of Standard Data Elements within a Private Data Sequence, and the definition of Attribute Values, are implementation dependent.

For a Standard Extended SOP Class the Attributes Pixel Data (07FE,0010), Float Pixel Data (7FE0,0008), Double Float Pixel Data (7FE0,0009), Waveform Data (5400,1010) and Overlay Data (60xx,3000) shall not be included within a Private Sequence Item, nor within a standard Sequence Item nested directly or indirectly within a Private Sequence Item.

8 Encoding of Pixel, Overlay and Waveform Data

8.1 Pixel and Overlay Data, and Related Data Elements

Pixel Data (7FE0,0010), Float Pixel Data (7FE0,0008), Double Float Pixel Data (7FE0,0009) and Overlay Data (60xx,3000) shall be used for the exchange of encoded graphical image data. These elements along with additional Data Elements, specified as Attributes of the Image Information Entities defined in PS3.3, shall be used to describe the way in which the Pixel Data and Overlay Data are encoded and shall be interpreted. Finally, depending on the negotiated Transfer Syntax (see Section 10 and Annex A), Pixel Data may be compressed.

Pixel Data (7FE0,0010) and Overlay Data (60xx,3000) have a VR of OW or OB, depending on the negotiated Transfer Syntax (see Annex A). The only difference between OW and OB being that OB, a string of bytes, shall be unaffected by Byte Ordering (see Section 7.3).

Float Pixel Data (7FE0,0008) has a Value Representation of OF.

Double Float Pixel Data (7FE0,0009) has a Value Representation of OD.

For Pixel Data values encoded in OF and OD, any value that is permitted by the IEEE 754:1985 may be used, including NaN, +ve Infinity and -ve Infinity. See Table 6.2-1

Note

Float and double float pixel data values are not arbitrarily constrained to finite numbers, since it may be important for the application to signal that the result of a calculation that produced a pixel is an infinite value or not a number.

8.1.1 Pixel Data Encoding of Related Data Elements

Encoded Pixel Data of various bit depths shall be accommodated. The following three Data Elements shall define the Pixel structure:

- Bits Allocated (0028,0100)
- Bits Stored (0028,0101)
- High Bit (0028,0102)

Each Pixel Cell shall contain a single Pixel Sample Value. The size of the Pixel Cell shall be specified by Bits Allocated (0028,0100). Bits Stored (0028,0101) defines the total number of these allocated bits that will be used to represent a Pixel Sample Value. Bits Stored (0028,0101) shall never be larger than Bits Allocated (0028,0100). High Bit (0028,0102) specifies where the high order bit of the Bits Stored (0028,0101) is to be placed with respect to the Bits Allocated (0028,0100) specification. Bits Allocated (0028,0100) shall either be 1, or a multiple of 8. High Bit (0028,0102) shall be one less than Bits Stored (0028,0101).

Note

1. For example, in Pixel Data with 16 bits (2 bytes) allocated, 12 bits stored, and bit 11 specified as the high bit, one pixel sample is encoded in each 16-bit word, with the 4 most significant bits of each word not containing Pixel Data. See Annex D for other examples of the basic encoding schemes.
2. Formerly, bits not used for Pixel Sample Values were described as being usable for overlay planes, but this usage has been retired. See PS3.5-2004.
3. Formerly, High Bit (0028,0102) was not restricted to being be one less than Bits Stored (0028,0101) in this Part, or in the general case, though almost all Information Object Definitions in PS3.3 imposed such a restriction. See PS3.5 2014c.
4. Receiving applications may not assume anything about the contents of unused bits, and in particular may not assume that they are zero, or that they contain sign extension bits.

Additional restrictions that are placed on acceptable Values for Bits Allocated (0028,0100), Bits Stored (0028,0101), and High Bit (0028,0102) for Pixel Data (7FE0,0010) are specified in the Information Object Definitions in PS3.3.

Restrictions are placed on acceptable Values for Bits Allocated (0028,0100) for Float Pixel Data (7FE0,0008) and Double Float Pixel Data (7FE0,0009), such that only a single Pixel Cell entirely occupies the allocated bits specified by Bits Allocated (0028,0100), hence Bits Stored (0028,0101) and High Bit (0028,0102) are not sent.

Also, the Value Field containing Pixel Data, like all other Value Fields in DICOM, shall be an even number of bytes in length. This means that the Value Field may need to be padded with data that is not part of the image and shall not be considered significant. If needed, the padding bits shall be appended to the end of the Value Field, and shall be used only to extend the data to the next even byte increment of length.

In a multi-frame object that is transmitted in Native Format, the individual frames are not padded. The individual frames shall be concatenated and padding bits (if necessary) apply to the complete Value Field.

Note

Receiving applications should be aware that some older applications may send Pixel Data with excess padding, which was not explicitly prohibited in earlier versions of the Standard. Applications should be prepared to accept such Pixel Data elements, but may delete the excess padding. In no case should a sending application place private data in the padding data.

The field of bits representing the value of a Pixel Sample shall be a binary 2's complement integer or an unsigned integer, as specified by the Data Element Pixel Representation (0028,0103). The sign bit shall be the High Bit in a Pixel Sample Value that is a 2's complement integer. The minimum actual Pixel Sample Value encountered in the Pixel Data is specified by Smallest Image Pixel Value (0028,0106) while the maximum value is specified by Largest Image Pixel Value (0028,0107).

8.1.2 Overlay Data Encoding of Related Data Elements

Encoded Overlay Planes always have a bit depth of 1, and are encoded separately from the Pixel Data in Overlay Data (60xx,3000). The following two Data Elements shall define the Overlay Plane structure:

- Overlay Bits Allocated (60xx,0100)
- Overlay Bit Position (60xx,0102)

Note

1. There is no Data Element analogous to Bits Stored (0028,0101) since Overlay Planes always have a bit depth of 1.
2. Restrictions on the allowed values for these Data Elements are defined in PS3.3. Formerly overlay data stored in unused bits of Pixel Data (7FE0,0010) was described, and these attributes had meaningful values but this usage has been retired. See PS3.5-2004. For overlays encoded in Overlay Data Element (60xx,3000), Overlay Bits Allocated (60xx,0100) is always 1 and Overlay Bit Position (60xx,0102) is always 0.

For Overlay Data Element (60xx,3000), the Value Representation OW is most often required. The Value Representation OB may also be used for Overlay Data in cases where the Value Representation is explicitly conveyed (see Annex A).

Note

The DICOM default Transfer Syntax (Implicit VR Little Endian) does not explicitly convey Value Representation and therefore the VR of OB may not be used for Pixel Data when using the default Transfer Syntax.

Overlay Data is encoded as the direct concatenation of the bits of a single Overlay Plane, where the first bit of an Overlay Plane is encoded in the least significant bit, immediately followed by the next bit of the Overlay Plane in the next most significant bit. When the Overlay Data crosses a word boundary in the OW case, or a byte boundary in the OB case, it shall continue to be encoded, least significant bit to most significant bit, in the next word, or byte, respectively (see Annex D). For Overlay Data encoded with the Value Representation OW, the byte ordering of the resulting 2-byte words is defined by the Little Endian or Big Endian Transfer Syntaxes negotiated at the Association Establishment (see Annex A).

Note

For Overlay Data encoded with the Value Representation OB, the Overlay Data encoding is unaffected by Little Endian or Big Endian byte ordering.

8.2 Native or Encapsulated Format Encoding

Pixel data conveyed in the Pixel Data Element (7FE0,0010) may be sent either in a Native (uncompressed) Format or in an Encapsulated Format (e.g., compressed) defined outside the DICOM standard.

Pixel Data conveyed in the Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009) shall be in a Native (uncompressed) Format if encoded in a Standard Transfer Syntax.

Note

1. In future, if Standard Transfer Syntaxes are defined for compression of Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009), this constraint may be relaxed and Encapsulated Format permitted.

2. This constraint does not apply to Private Transfer Syntaxes.

If Pixel Data (7FE0,0010) is sent in a Native Format, the Value Representation OW is most often required. The Value Representation OB may also be used for Pixel Data (7FE0,0010) in cases where Bits Allocated has a value less than or equal to 8, but only with Transfer Syntaxes where the Value Representation is explicitly conveyed (see Annex A).

Note

The DICOM default Transfer Syntax (Implicit VR Little Endian) does not explicitly convey Value Representation and therefore the VR of OB may not be used for Pixel Data (7FE0,0010) when using the default Transfer Syntax.

Float Pixel Data (7FE0,0008) is sent in Native Format; the Value Representation shall be OF, Bits Allocated (0028,0100) shall be 32, Bits Stored (0028,0101), High Bit (0028,0102) and Pixel Representation (0028,0103) shall not be present.

Double Float Pixel Data (7FE0,0009) is sent in Native Format; the Value Representation shall be OD, Bits Allocated (0028,0100) shall be 64, Bits Stored (0028,0101) and High Bit (0028,0102) and Pixel Representation (0028,0103) shall not be present.

It is not permitted to have more than one of Pixel Data Provider URL (0028,7FE0), Pixel Data (7FE0,0010), Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009) in the top level Data Set.

Note

Pixel Data encoded in Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009) can be considered as consisting of Pixel Cells that entirely occupy the allocated bits, and therefore do not cross word boundaries.

Native format Pixel Cells are encoded as the direct concatenation of the bits of each Pixel Cell, the least significant bit of each Pixel Cell is encoded in the least significant bit of the encoded word or byte, immediately followed by the next most significant bit of each Pixel Cell in the next most significant bit of the encoded word or byte, successively until all bits of the Pixel Cell have been encoded, then immediately followed by the least significant bit of the next Pixel Cell in the next most significant bit of the encoded word or byte. The number of bits of each Pixel Cell is defined by the Bits Allocated (0028,0100) Data Element Value. When a Pixel Cell crosses a word boundary in the OW case, or a byte boundary in the OB case, it shall continue to be encoded, least significant bit to most significant bit, in the next word, or byte, respectively (see Annex D). For Pixel Data (7FE0,0010) encoded with the Value Representation OW, the byte ordering of the resulting 2-byte words is defined by the Little Endian or Big Endian Transfer Syntaxes negotiated at the Association Establishment (see Annex A).

Note

1. For Pixel Data (7FE0,0010) encoded with the Value Representation OB, the Pixel Data (7FE0,0010) encoding is unaffected by Little Endian or Big Endian byte ordering.

2. If encoding Pixel Data (7FE0,0010) with a Value for Bits Allocated (0028,0100) not equal to 16 be sure to read and understand Annex D.

If sent in an Encapsulated Format (i.e., other than the Native Format) the Value Representation OB is used. The Pixel Cells are encoded according to the encoding process defined by one of the negotiated Transfer Syntaxes (see Annex A). The encapsulated pixel stream of encoded pixel data is segmented into one or more Fragments, each of which conveys its own explicit length. The sequence of Fragments of the encapsulated pixel stream is terminated by a delimiter, thus allowing the support of encoding processes where the resulting length of the entire pixel stream is not known until it is entirely encoded. This Encapsulated Format supports both Single-Frame and Multi-Frame images (as defined in PS3.3).

Note

Depending on the transfer syntax, a frame may be entirely contained within a single fragment, or may span multiple fragments to support buffering during compression or to avoid exceeding the maximum size of a fixed length fragment. A recipient can detect fragmentation of frames by comparing the number of fragments (the number of Items minus one for the Basic Offset Table) with the number of frames. Some performance optimizations may be available to a recipient in the absence of fragmentation of frames, but an implementation that fails to support such fragmentation does not conform to the Standard.

8.2.1 JPEG Image Compression

DICOM provides a mechanism for supporting the use of JPEG Image Compression through the Encapsulated Format (see PS3.3). Annex A defines a number of Transfer Syntaxes that reference the JPEG Standard and provide a number of lossless (bit preserving) and lossy compression schemes.

Note

The context where the usage of lossy compression of medical images is clinically acceptable is beyond the scope of the DICOM Standard. The policies associated with the selection of appropriate compression parameters (e.g., compression ratio) for JPEG lossy compression is also beyond the scope of this standard.

In order to facilitate interoperability of implementations conforming to the DICOM Standard that elect to use one or more of the Transfer Syntaxes for JPEG Image Compression, the following policy is specified:

- Any implementation that conforms to the DICOM Standard and has elected to support any one of the Transfer Syntaxes for lossless JPEG Image Compression, shall support the following lossless compression: The subset (first-order horizontal prediction [Selection Value 1) of JPEG Process 14 (DPCM, non-hierarchical with Huffman coding) (see Annex F).
- Any implementation that conforms to the DICOM Standard and has elected to support any one of the Transfer Syntaxes for 8-bit lossy JPEG Image Compression, shall support the JPEG Baseline Compression (coding Process 1).
- Any implementation that conforms to the DICOM Standard and has elected to support any one of the Transfer Syntaxes for 12-bit lossy JPEG Image Compression, shall support the JPEG Compression Process 4.

Note

The DICOM conformance statement shall differentiate whether or not the implementation is capable of simply receiving or receiving and processing JPEG encoded images (see PS3.2).

The use of the DICOM Encapsulated Format to support JPEG Compressed Pixel Data requires that the Data Elements that are related to the Pixel Data encoding (e.g., Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the characteristics of the compressed data stream. The Pixel Data characteristics included in the JPEG Interchange Format shall be used to decode the compressed data stream.

Note

1. These requirements were formerly specified in terms of the "uncompressed pixel data from which the compressed data stream was derived". However, since the form of the "original" uncompressed data stream could vary between different implementations, this requirement is now specified in terms of consistency with what is encapsulated.

   When decompressing, should the characteristics explicitly specified in the compressed data stream (e.g., spatial subsampling or number of components or planar configuration) be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed Data Set might be encoded.

2. Those characteristics not explicitly specified in the compressed data stream (e.g., the color space of the compressed components, which is not specified in the JPEG Interchange Format), or implied by the definition of the compression scheme (e.g., always unsigned in JPEG), can therefore be determined from the DICOM Data Element in the enclosing Data Set. For example a Photometric Interpretation of "YBR_FULL_422" would describe the color space that is commonly used to lossy compress images using JPEG. It is unusual to use an RGB color space for lossy compression, since no advantage is taken of correlation between the red, green and blue components (e.g., of luminance), and poor compression is achieved.

3. The JPEG Interchange Format is distinct from the JPEG File Interchange Format (JFIF). The JPEG Interchange Format is defined in [ISO/IEC 10918-1] section 4.9.1, and refers to the inclusion of decoding tables, as distinct from the "abbreviated format" in which these tables are not sent (and the decoder is assumed to already have them). The JPEG Interchange Format does NOT specify the color space. The JPEG File Interchange Format, not part of the original JPEG standard, but defined in ECMA TR-098, and under development as ISO 101918-5, is often used to store JPEG bit streams in consumer format files, and does include the ability to specify the color space of the components. THE JFIF APP0 marker segment is NOT required to be present in DICOM encapsulated JPEG bit streams, and should not be relied upon to recognize the color space. Its presence is not forbidden (unlike the JP2 information for JPEG 2000 Transfer Syntaxes), but it is recommended that it be absent.

4. Should the compression process be incapable of encoding a particular form of pixel data representation (e.g., JPEG cannot encode signed integers, only unsigned integers), then ideally only the appropriate form should be "fed" into the compression process. However, for certain characteristics described in DICOM Data Elements but not explicitly described in the compressed data stream (such as Pixel Representation), then the DICOM Data Element should be considered to describe what has been compressed (e.g., the pixel data really is to be interpreted as signed if Pixel Representation so specifies).

5. DICOM Data Elements should not describe characteristics that are beyond the capability of the compression scheme used. For example, JPEG lossy processes are limited to 12 bits, hence the value of Bits Stored should be 12 or less. Bits Allocated is irrelevant, and is likely to be constrained by the Information Object Definition in PS3.3 to values of 8 or 16. Also, JPEG compressed data streams are always color-by-pixel and should be specified as such (a decoder can essentially ignore this element however as the value for JPEG compressed data is already known).

8.2.2 Run Length Encoding Compression

DICOM provides a mechanism for supporting the use of Run Length Encoding (RLE) Compression, which is a byte oriented lossless compression scheme through the encapsulated Format (see PS3.3 of this Standard). Annex G defines RLE Compression and its Transfer Syntax.

Note

The RLE Compression algorithm described in Annex G is the compression used in the TIFF 6.0 specification known as the "PackBits" scheme.

The use of the DICOM Encapsulated Format to support RLE Compressed Pixel Data requires that the Data Elements that are related to the Pixel Data encoding (e.g., Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the compressed data.

Note

1. These requirements were formerly specified in terms of the "uncompressed pixel data from which the compressed data was derived". However, since the form of the "original" uncompressed data stream could vary between different implementations, this requirement is now specified in terms of consistency with what is encapsulated.

2. Those characteristics not implied by the definition of the compression scheme (e.g., always color-by-plane in RLE), can therefore be determined from the DICOM Data Element in the enclosing Data Set. For example a Photometric Interpretation of "~~YBR_FULL~~YBR_FULL" would describe the color space that is commonly used to losslessly compress images using RLE. It is unusual to use an RGB color space for RLE compression, since no advantage is taken of correlation between the red, green and blue components (e.g., of luminance), and poor compression is achieved (note however that the conversion from RGB to ~~YBR_FULL~~YBR_FULL is itself lossy. A new photometric interpretation may be proposed in the future that allows lossless conversion from RGB and also results in better RLE compression ratios).

3. DICOM Data Elements should not describe characteristics that are beyond the capability of the compression scheme used. For example, RLE compressed data streams (using the algorithm mandated in the DICOM Standard) are always color-by-plane.

8.2.3 JPEG-LS Image Compression

DICOM provides a mechanism for supporting the use of JPEG-LS Image Compression through the Encapsulated Format (see PS3.3). Annex A defines a number of Transfer Syntaxes that reference the JPEG-LS Standard and provide a number of lossless (bit preserving) and lossy (near-lossless) compression schemes.

Note

The context where the usage of lossy (near-lossless) compression of medical images is clinically acceptable is beyond the scope of the DICOM Standard. The policies associated with the selection of appropriate compression parameters (e.g., compression ratio) for JPEG-LS lossy (near-lossless) compression is also beyond the scope of this standard.

The use of the DICOM Encapsulated Format to support JPEG-LS Compressed Pixel Data requires that the Data Elements that are related to the Pixel Data encoding (e.g., Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the characteristics of the compressed data stream. The Pixel Data characteristics included in the JPEG-LS Interchange Format shall be used to decode the compressed data stream.

Note

See also the notes in Section 8.2.1.

8.2.4 JPEG 2000 Image Compression

DICOM provides a mechanism for supporting the use of JPEG 2000 Image Compression through the Encapsulated Format (see PS3.3). Annex A defines a number of Transfer Syntaxes that reference the JPEG 2000 Standard and provide lossless (bit preserving) and lossy compression schemes.

Note

The context where the usage of lossy compression of medical images is clinically acceptable is beyond the scope of the DICOM Standard. The policies associated with the selection of appropriate compression parameters (e.g., compression ratio) for JPEG 2000 lossy compression are also beyond the scope of this standard.

The use of the DICOM Encapsulated Format to support JPEG 2000 Compressed Pixel Data requires that the Data Elements that are related to the Pixel Data encoding (e.g., Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the characteristics of the compressed data stream. The Pixel Data characteristics included in the JPEG 2000 bit stream shall be used to decode the compressed data stream.

Note

These requirements are specified in terms of consistency with what is encapsulated, rather than in terms of the uncompressed pixel data from which the compressed data stream may have been derived.

When decompressing, should the characteristics explicitly specified in the compressed data stream be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed Data Set might be encoded.

The JPEG 2000 bit stream specifies whether or not a reversible or irreversible multi-component (color) transformation, if any, has been applied. If no multi-component transformation has been applied, then the components shall correspond to those specified by the DICOM Attribute Photometric Interpretation (0028,0004). If the JPEG 2000 Part 1 reversible multi-component transformation has been applied then the DICOM Attribute Photometric Interpretation (0028,0004) shall be YBR_RCT. If the JPEG 2000 Part 1 irreversible multi-component transformation has been applied then the DICOM Attribute Photometric Interpretation (0028,0004) shall be YBR_ICT.

Note

1. For example, single component may be present, and the Photometric Interpretation (0028,0004) may be MONOCHROME2.

2. Though it would be unusual, would not take advantage of correlation between the red, green and blue components, and would not achieve effective compression, a Photometric Interpretation of RGB could be specified as long as no multi-component transformation was specified by the JPEG 2000 bit stream.

3. Despite the application of a multi-component color transformation and its reflection in the Photometric Interpretation attribute, the "color space" remains undefined. There is currently no means of conveying "standard color spaces" either by fixed values (such as sRGB) or by ICC profiles. Note in particular that the JP2 file header is not sent in the JPEG 2000 bitstream that is encapsulated in DICOM.

The JPEG 2000 bitstream is capable of encoding both signed and unsigned pixel values, hence the value of Pixel Representation (0028,0103) may be either 0 or 1 depending on what has been encoded (as specified in the SIZ marker segment in the precision and sign of component parameter).

The value of Planar Configuration (0028,0006) is irrelevant since the manner of encoding components is specified in the JPEG 2000 standard, hence it shall be set to 0.

8.2.5 MPEG2 MP@ML Image Compression

DICOM provides a mechanism for supporting the use of MPEG2 MP@ML Image Compression through the Encapsulated Format (see PS3.3). Annex A defines a Transfer Syntax that references the MPEG2 MP@ML Standard.

Note

MPEG2 compression is inherently lossy. The context where the usage of lossy compression of medical images is clinically acceptable is beyond the scope of the DICOM Standard. The policies associated with the selection of appropriate compression parameters (e.g., compression ratio) for MPEG2 MP@ML are also beyond the scope of this standard.

The use of the DICOM Encapsulated Format to support MPEG2 MP@ML compressed pixel data requires that the Data Elements that are related to the Pixel Data encoding (e.g., Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the characteristics of the compressed data stream, with some specific exceptions noted here. The Pixel Data characteristics included in the MPEG2 MP@ML bit stream shall be used to decode the compressed data stream.

Note

These requirements are specified in terms of consistency with what is encapsulated, rather than in terms of the uncompressed pixel data from which the compressed data stream may have been derived.

When decompressing, should the characteristics explicitly specified in the compressed data stream be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed Data Set might be encoded.

The MPEG2 MP@ML bit stream specifies whether or not a reversible or irreversible multi-component (color) transformation, if any, has been applied. If no multi-component transformation has been applied, then the components shall correspond to those specified by the DICOM Attribute Photometric Interpretation (0028,0004). MPEG2 MP@ML applies an irreversible multi-component transformation, so DICOM Attribute Photometric Interpretation (0028,0004) shall be YBR_PARTIAL_420 in the case of multi-component data, and MONOCHROME2 in the case of single component data (even though the MPEG2 bit stream itself is always encoded as three components, one luminance and two chrominance).

Note

MPEG2 proposes some video formats. Each of the standards specified is used in a different market, including: ITU-R BT.470-2 System M for SD NTSC and ITU-R BT.470-2 System B/G for SD PAL/SECAM. A PAL based system should therefore be based on ITU-BT.470 System B for each of Color Primaries, Transfer Characteristic (gamma) and matrix coefficients and should take a value of 5 as defined in [ISO/IEC 13818-2].

The value of Planar Configuration (0028,0006) is irrelevant since the manner of encoding components is specified in the MPEG2 MP@ML standard, hence it shall be set to 0.

In summary:

- Samples per Pixel (0028,0002) shall be 3
- Photometric Interpretation (0028,0004) shall be YBR_PARTIAL_420
- Bits Allocated (0028,0100) shall be 8
- Bits Stored (0028,0101) shall be 8
- High Bit (0028,0102) shall be 7
- Pixel Representation (0028,0103) shall be 0
- Planar Configuration (0028,0006) shall be 0
- Rows (0028,0010), Columns (0028,0011), Cine Rate (0018,0040) and Frame Time (0018,1063) or Frame Time Vector (0018,1065) shall be consistent with the limitations of MP@ML, as specified in Table 8-1.

Table 8-1. MPEG2 MP@ML Image Transfer Syntax Rows and Columns Attributes

Video Type	Spatial resolution	Frame Rate (see Note 4)	Frame Time (see Note 5)	Maximum Rows	Maximum Columns
525-line NTSC	Full	30	33.33 ms	480	720
625-line PAL	Full	25	40.0 ms	576	720

Note

1. Although different combinations of values for Rows and Columns values are possible while respecting the maximum values listed above, it is recommended that the typical 4:3 ratio of image width to height be maintained in order to avoid image deformation by MPEG2 decoders. A common way to maintain the ratio of width to height is to pad the image with black areas on either side.
2. "Half" definition of pictures (240x352 and 288x352 for NTSC and PAL, respectively) are always supported by decoders.
3. MP@ML allows for various different display and pixel aspect ratios, including the use of square pixels, and the use of non-square pixels with display aspect ratios of 4:3 and 16:9. DICOM specifies no additional restrictions beyond what is provided for in MP@ML. All permutations allowed by MP@ML are valid and are require to be supported by all DICOM decoders.
4. The actual frame rate for NTSC MPEG2 is approximately 29.97 frames/sec.
5. The nominal Frame Time is supplied for the purpose of inclusion on the DICOM Cine Module Attributes, and should be calculated from the actual frame rate.

One fragment shall contain the whole MPEG2 stream.

Note

1. If a video stream exceeds the maximum length of one fragment, it may be sent as multiple SOP Instances, but each SOP Instance will contain an independent and playable bit stream, and not depend on the encoded bit stream in other (previous) instances. The manner in which such separate instances are related is not specified in the standard, but mechanisms such as grouping into the same Series, and references to earlier instances using Referenced Image Sequence may be used.
2. This constraint limits the length of the compressed bit stream to no longer than 2^{32}-2 bytes.

The Basic Offset Table shall be empty (present but zero length).

Note

The Basic Offset Table is not used because MPEG2 contains its own mechanism for describing navigation of frames. To enable decoding of only a part of the sequence, MPEG2 manages a header in any group of pictures (GOP) containing a time_code - a 25-bit integer containing the following: drop_frame_flag, time_code_hours, time_code_minutes, marker_bit, time_code_seconds and time_code_pictures.

The container format for the video bitstream shall be MPEG-2 Transport Stream, a.k.a. MPEG-TS (see [ISO/IEC 13818-1]) or MPEG-4, a.k.a. MP4 container (see [ISO/IEC 14496-12] and [ISO/IEC 14496-14]).

Any audio components present within the MPEG bit stream shall comply with the following restrictions:

- CBR MPEG-1 LAYER III (MP3) Audio Standard
- up to 24 bits
- 32 kHz, 44.1 kHz or 48 kHz for the main channel (the complementary channels can be sampled at the half rate, as defined in the Standard)
- one main mono or stereo channel, and optionally one or more complementary channel(s)

Note

1. MPEG-1 Layer III is standardized in Part 3 of the MPEG-1 standard (see [ISO/IEC 11172-3]).
2. Although MPEG describes each channel as including up to 5 signals (e.g., for surround effects), it is recommended to limit each of the two channels to 2 signals each one (stereo).

8.2.6 MPEG2 MP@HL Image Compression

MPEG2 Main Profile at High Level (MP@HL) corresponds to what is commonly known as HDTV ('High Definition Television'). DICOM provides a mechanism for supporting the use of MPEG2 MP@HL Image Compression through the Encapsulated Format (see PS3.3). Annex A defines a Transfer Syntax that references the MPEG2 MP@HL Standard.

Note

MPEG2 compression is inherently lossy. The context where the usage of lossy compression of medical images is clinically acceptable is beyond the scope of the DICOM Standard. The policies associated with the selection of appropriate compression parameters (e.g., compression ratio) for MPEG2 MP@HL are also beyond the scope of this standard.

The use of the DICOM Encapsulated Format to support MPEG2 MP@HL compressed pixel data requires that the Data Elements that are related to the Pixel Data encoding (e.g., Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the characteristics of the compressed data stream, with some specific exceptions noted here. The Pixel Data characteristics included in the MPEG2 MP@HL bit stream shall be used to decode the compressed data stream.

Note

These requirements are specified in terms of consistency with what is encapsulated, rather than in terms of the uncompressed pixel data from which the compressed data stream may have been derived.

When decompressing, should the characteristics explicitly specified in the compressed data stream be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed Data Set might be encoded.

The requirements are:

- Planar Configuration (0028,0006) shall be 0

Note

The value of Planar Configuration (0028,0006) is irrelevant since the manner of encoding components is specified in the MPEG2 standard, hence it is set to 0.

- Samples per Pixel (0028,0002) shall be 3
- Photometric Interpretation (0028,0004) shall be YBR_PARTIAL_420 or MONOCHROME2
- Bits Allocated (0028,0100) shall be 8
- Bits Stored (0028,0101) shall be 8
- High Bit (0028,0102) shall be 7
- Pixel Representation (0028,0103) shall be 0
- Rows (0028,0010) shall be either 720 or 1080
- Columns (0028,0011) shall be 1280 if Rows is 720, or shall be 1920 if Rows is 1080.
- The value of MPEG2 aspect_ratio_information shall be 0011 in the encapsulated MPEG2 data stream corresponding to a 'Display Aspect Ratio' (DAR) of 16:9.
- The DICOM attribute Pixel Aspect Ratio (0028,0034) shall be absent. This corresponds to a 'Sampling Aspect Ratio' (SAR) of 1:1.
- Cine Rate (0018,0040) and Frame Time (0018,1063) or Frame Time Vector (0018,1065) shall be consistent with the limitations of MP@HL, as specified in Table 8-2.

Table 8-2. MPEG2 MP@HL Image Transfer Syntax Frame Rate Attributes

Video Type	Spatial resolution layer	Frame Rate (see Note 2)	Frame Time (see Note 3)
30 Hz HD	Single level, Enhancement	30	33.33 ms
25 Hz HD	Single level, Enhancement	25	40.0 ms
60 Hz HD	Single level, Enhancement	60	16.17 ms
50 Hz HD	Single level, Enhancement	50	20.00 ms

Note

1. The requirements on rows and columns are to maximize interoperability between software environments and commonly available hardware MPEG2 encoder/decoder implementations. Should the source picture have a lower value, it should be re-formatted accordingly by scaling and/or pixel padding prior to MPEG2 encoding.
2. The frame rate of the acquiring camera for '30 Hz HD' MPEG2 may be either 30 or 30/1.001 (approximately 29.97) frames/sec. Similarly, the frame rate in the case of 60 Hz may be either 60 or 60/1.001 (approximately 59.94) frames/sec This may lead to small inconsistencies between the video timebase and real time.
3. The Frame Time (0018,1063) may be calculated from the frame rate of the acquiring camera. A frame time of 33.367 ms corresponds to 29.97 frames per second.
4. The value of chroma_format for this profile and level is defined by MPEG as 4:2:0.
5. Examples of screen resolutions supported by MPEG2 MP@HL are shown in Table 8-y. Frame rates of 50 Hz and 60 Hz (progressive) at the maximum resolution of 1080 by 1920 are not supported by MP@HL. Interlace at the maximum resolution is supported at a field rate of 50 Hz or 60 Hz, which corresponds to a frame rate of 25 Hz or 30 Hz respectively as described in Table 8-y.
6. An MPEG2 MP@HL decoder is able to decode bit streams conforming to lower levels. These include the 1080 by 1440 bit streams of MP@H-14, and the Main Level bit streams used in the existing MPEG2 MP@ML transfer syntax in the Visible Light IOD.

7. MP@H-14 is not supported by this transfer syntax.

8. The restriction of DAR to 16:9 is required to ensure interoperability because of limitations in commonly available hardware chip set implementations for MPEG2 MP@HL.

Table 8-3. Examples of MPEG2 MP@HL Screen Resolution

Rows	Columns	Frame rate	Video Type	Progressive or Interlace
1080	1920	25	25 Hz HD	P
1080	1920	29.97, 30	30 Hz HD	P
1080	1920	25	25 Hz HD	I
1080	1920	29.97, 30	30 Hz HD	I
720	1280	25	25 Hz HD	P
720	1280	29.97, 30,	30 Hz HD	P
720	1280	50	50 Hz HD	P
720	1280	59.94, 60	60 Hz HD	P

One fragment shall contain the whole MPEG2 bit stream.

Note

1. If a video stream exceeds the maximum length of one fragment (approximately 4 GB), it may be sent as multiple SOP Instances, but each SOP Instance will contain an independent and playable bit stream, and not depend on the encoded bit stream in other (previous) instances. The manner in which such separate instances are related is not specified in the standard, but mechanisms such as grouping into the same Series, and references to earlier instances using Referenced Image Sequence may be used.

2. This constraint limits the length of the compressed bit stream to no longer than 2^{32}-2 bytes.

The Basic Offset Table in the Pixel Data (7FE0,0010) shall be empty (present but zero length).

Note

The Basic Offset Table is not used because MPEG2 contains its own mechanism for describing navigation of frames. To enable decoding of only a part of the sequence, MPEG2 manages a header in any group of pictures (GOP) containing a time_code - a 25-bit integer containing the following: drop_frame_flag, time_code_hours, time_code_minutes, marker_bit, time_code_seconds and time_code_pictures.

The container format for the video bitstream shall be MPEG-2 Transport Stream, a.k.a. MPEG-TS (see [ISO/IEC 13818-1]) or MPEG-4, a.k.a. MP4 container (see [ISO/IEC 14496-12] and [ISO/IEC 14496-14]).

Any audio components present within the MPEG2 MP@HL bit stream shall comply with the restrictions as for MPEG2 MP@ML as stated in Section 8.2.5.

8.2.7 MPEG-4 AVC/H.264 HiP@Level4.1 Video Compression

MPEG-4 AVC/H.264 High Profile / Level 4.1 corresponds to what is commonly known as HDTV ('High Definition Television'). DICOM provides a mechanism for supporting the use of MPEG-4 AVC/H.264 Image Compression through the Encapsulated Format (see PS3.3). Annex A defines a Transfer Syntax that references the MPEG-4 AVC/H.264 Standard.

Note

MPEG-4 AVC/H.264 compression @ High Profile compression is inherently lossy. The context where the usage of lossy compression of medical images is clinically acceptable is beyond the scope of the DICOM Standard. The policies associated with the selection of appropriate compression parameters (e.g., compression ratio) for MPEG-4 AVC/H.264 HiP@Level4.1 are also beyond the scope of this standard.

The use of the DICOM Encapsulated Format to support MPEG-4 AVC/H.264 compressed pixel data requires that the Data Elements that are related to the Pixel Data encoding (e.g., Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the characteristics of the compressed data stream, with some specific exceptions noted here. The Pixel Data characteristics included in the MPEG-4 AVC/H.264 bit stream shall be used to decode the compressed data stream.

Note

~~1. These requirements are specified in terms of consistency with what is encapsulated, rather than in terms of the uncompressed pixel data from which the compressed data stream may have been derived.~~

~~2. When decompressing, should the characteristics explicitly specified in the compressed data stream be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed Data Set might be encoded.~~

These requirements are specified in terms of consistency with what is encapsulated, rather than in terms of the uncompressed pixel data from which the compressed data stream may have been derived.

When decompressing, should the characteristics explicitly specified in the compressed data stream be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed Data Set might be encoded.

The requirements are:

- Planar Configuration (0028,0006) shall be 0
- Samples per Pixel (0028,0002) shall be 3
- Photometric Interpretation (0028,0004) shall be YBR_PARTIAL_420
- Bits Allocated (0028,0100) shall be 8
- Bits Stored (0028,0101) shall be 8
- High Bit (0028,0102) shall be 7
- Pixel Representation (0028,0103) shall be 0
- The value of MPEG-4 AVC/H.264 sample aspect_ratio_idc shall be 1 in the encapsulated MPEG-4 AVC/H.264 bit stream if aspect_ratio_info_present_flag is 1.
- Pixel Aspect Ratio (0028,0034) shall be absent. This corresponds to a 'Sampling Aspect Ratio' (SAR) of 1:1.
- The possible values for Rows (0028,0010), Columns (0028,0011), Cine Rate (0018,0040), and Frame Time (0018,1063) or Frame Time Vector (0018,1065) depend on the used transfer syntax.
  - For MPEG-4 AVC/H.264 High Profile / Level 4.1 transfer syntax, the values for these data elements shall be compliant with the High Profile / Level 4.1 of the MPEG-4 AVC/H.264 standard ([ISO/IEC 14496-10]) and restricted to a square pixel aspect ratio.
  - For MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1 transfer syntax, the values for these data elements shall be as specified in Table 8-4.

Table 8-4. Values Permitted for MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1

Rows	Columns	Frame rate	Video Type	Progressive or Interlace
1080	1920	25	25 Hz HD	I
1080	1920	29.97	30 Hz HD	I
1080	1920	24	24 Hz HD	P

Rows	Columns	Frame rate	Video Type	Progressive or Interlace
1080	1920	23.976	24 Hz HD	P
720	1280	50	50 Hz HD	P
720	1280	59.94,	60 Hz HD	P
720	1280	24	24 Hz HD	P
720	1280	23.976	24 Hz HD	P

Note

1. The value of Planar Configuration (0028,0006) is irrelevant since the manner of encoding components is specified in the MPEG-4 AVC/H.264 standard, hence it is set to 0.

2. The limitation on rows and columns are to maximize interoperability between software environments and commonly available hardware MPEG-4 AVC/H.264 encoder/decoder implementations. Source pictures that have a lower value should be re-formatted by scaling and/or pixel padding prior to MPEG-4 AVC/H.264 encoding.

3. The frame rate of the acquiring camera for '30 Hz HD' MPEG-4 AVC/H.264 may be either 30 or 30/1.001 (approximately 29.97) frames/sec. Similarly, the frame rate in the case of 60 Hz may be either 60 or 60/1.001 (approximately 59.94) frames/sec. This may lead to small inconsistencies between the video timebase and real time. The relationship between frame rate and frame time is shown in Table 8-5.

4. The Frame Time (0018,1063) may be calculated from the frame rate of the acquiring camera. A frame rate of 29.97 frames per second corresponds to a frame time of 33.367 ms.

5. The value of chroma_format for this profile and level is defined by MPEG as 4:2:0.

6. Example screen resolutions supported by MPEG-4 AVC/H.264 High Profile / Level 4.1 can be taken from Table 8-4. Frame rates of 50 Hz and 60 Hz (progressive) at the maximum resolution of 1080 by 1920 are not supported by MPEG-4 AVC/H.264 High Profile / Level 4.1. Interlace at the maximum resolution is supported at a field rate of 50 Hz or 60 Hz, which corresponds to a frame rate of 25 Hz or 30 Hz respectively. Smaller resolutions may be used as long as they comply with the square pixel aspect ratio. An example is XGA resolution with an image resolution of 768 by 1024 pixels. For smaller resolutions there are higher frame rates possible. For example it may be up to 80 Hz for XGA.

7. The display aspect ratio is defined implicitly by the pixel resolution of the video picture. Only square pixel aspect ratio is allowed. MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1 will only support resolutions that result in a 16:9 display aspect ratio

8. The permitted screen resolutions for the MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1 are listed in Table 8-4. Only HD resolutions and no progressive frame rates for 25 or 29.97 frames per seconds are supported. Frame rates of 50 Hz and 60 Hz (progressive) at the maximum resolution of 1080 by 1920 are not supported.

Table 8-5. MPEG-4 AVC/H.264 High Profile / Level 4.1 Image Transfer Syntax Frame Rate Attributes

Video Type	Spatial resolution layer	Frame Rate (see Note 2)	Frame Time (see Note 3)
30 Hz HD	Single level, Enhancement	30	33.33 ms
25 Hz HD	Single level, Enhancement	25	40.0 ms
60 Hz HD	Single level, Enhancement	60	16.17 ms
50 Hz HD	Single level, Enhancement	50	20.00 ms

One fragment shall contain the whole MPEG-4 AVC/H.264 bit stream.

Note

If a video stream exceeds the maximum length of one fragment (approximately 4 GB), it may be sent as multiple SOP Instances, but each SOP Instance will contain an independent and playable bit stream, and not depend on the encoded bit

stream in other (previous) instances. The manner in which such separate instances are related is not specified in the standard, but mechanisms such as grouping into the same Series, and references to earlier instances using Referenced Image Sequence may be used.

The container format for the video bitstream shall be MPEG-2 Transport Stream, a.k.a. MPEG-TS (see [ISO/IEC 13818-1]) or MPEG-4, a.k.a. MP4 container (see [ISO/IEC 14496-12] and [ISO/IEC 14496-14]). The PTS/DTS of the transport stream shall be used in the MPEG coding. Any audio components present within the bit stream shall be interleaved in either LPCM, AC-3, AAC, MP3 or MPEG-1 Layer II audio format and shall comply with the following restrictions:

Table 8-6. Allowed Audio Formats

Audio Format	MPEG-2 TS Container	MP4 Container
LPCM	Allowed	-
AC3	Allowed	-
AAC	Allowed	Allowed
MP3	Allowed	Allowed
MPEG-1 Audio Layer II	Allowed	Allowed

- LPCM
  - Maximum bit rate: 4.608 Mbps
  - Sampling frequency: 48, 96 kHz
  - Bits per sample: 16, 20 or 24 bits
  - Number of channels: 2 channels

    Note

    If LPCM is used for Audio components, the container format shall be MPEG-2 TS.
- AC-3
  - Maximum bit rate: 640kbps
  - Sampling frequency: 48kHz
  - Bits per sample: 16 bits
  - Number of channels: 2 or 5.1 channels

    Note

    1. AC-3 is standardized in [ETSI TS 102 366]
    2. If AC-3 is used for Audio components, the container format shall be MPEG-2 TS.
- AAC
  - Maximum bit rate: 640kbps
  - Sampling frequency: 48kHz
  - Bits per sample: 16, 20 or 24 bits
  - Number of channels: 2 or 5.1 channels

Note

AAC is standardized in Part 7 of the MPEG-2 standard (see [ISO/IEC 13818-7], and Subpart 4 in Part 3 of the MPEG-4 standard (see [ISO/IEC 14496-3]).

- CBR MPEG-1 LAYER III (MP3) Audio Standard
  - Maximum bit rate: 320kbps
  - Sampling frequency: 32 kHz, 44.1 kHz or 48 kHz for the main channel (the complementary channels can be sampled at the half rate, as defined in the Standard)
  - Bits per sample: up to 24 bits
  - Number of channels: one main mono or stereo channel, and optionally one or more complementary channel(s)

    Note

    1. MPEG-1 Layer III is standardized in Part 3 of the MPEG-1 standard (see [ISO/IEC 11172-3]).
    2. Although MPEG describes each channel as including up to 5 signals (e.g. for surround effects), it is recommended to limit each of the two channels to 2 signals each one (stereo).
- MPEG-1 LAYER II (MP2)
  - Maximum bit rate: 384kbps
  - Sampling frequency: 32 kHz, 44.1 kHz or 48 kHz
  - Bits per sample: up to 24 bits
  - Number of channels: 2

    Note

    MPEG-1 Layer II is standardized in Part 3 of the MPEG-1 standard (see [ISO/IEC 11172-3]).

8.2.8 MPEG-4 AVC/H.264 HiP@Level4.2 Video Compression

DICOM provides a mechanism for supporting the use of MPEG-4 AVC/H.264 Image Compression through the Encapsulated Format (see PS3.3). Annex A defines Transfer Syntaxes that reference the MPEG-4 AVC/H.264 Standard.

Note

MPEG-4 AVC/H.264 compression @ High Profile compression is inherently lossy. The context where the usage of lossy compression of medical images is clinically acceptable is beyond the scope of the DICOM Standard. The policies associated with the selection of appropriate compression parameters (e.g. compression ratio) for MPEG-4 AVC/H.264 HiP@Level4.2 are also beyond the scope of this standard.

The use of the DICOM Encapsulated Format to support MPEG-4 AVC/H.264 compressed pixel data requires that the Data Elements that are related to the Pixel Data encoding (e.g. Photometric Interpretation, Samples per Pixel, Planar Configuration, Bits Allocated, Bits Stored, High Bit, Pixel Representation, Rows, Columns, etc.) shall contain values that are consistent with the characteristics of the compressed data stream, with some specific exceptions noted here. The Pixel Data characteristics included in the MPEG-4 AVC/H.264 bit stream shall be used to decode the compressed data stream.

Note

1. ~~These requirements are specified in terms of consistency with what is encapsulated, rather than in terms of the uncompressed pixel data from which the compressed data stream may have been derived.~~
2. ~~When decompressing, should the characteristics explicitly specified in the compressed data stream be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used~~

~~to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed data set might be encoded.~~

These requirements are specified in terms of consistency with what is encapsulated, rather than in terms of the uncompressed pixel data from which the compressed data stream may have been derived.

When decompressing, should the characteristics explicitly specified in the compressed data stream be inconsistent with those specified in the DICOM Data Elements, those explicitly specified in the compressed data stream should be used to control the decompression. The DICOM data elements, if inconsistent, can be regarded as suggestions as to the form in which an uncompressed data set might be encoded.

The requirements are:

- Planar Configuration (0028,0006) shall be 0
- Samples per Pixel (0028,0002) shall be 3
- Photometric Interpretation (0028,0004) shall be YBR_PARTIAL_420
- Bits Allocated (0028,0100) shall be 8
- Bits Stored (0028,0101) shall be 8
- High Bit (0028,0102) shall be 7
- Pixel Representation (0028,0103) shall be 0
- The value of MPEG-4 AVC/H.264 sample aspect_ratio_idc shall be 1 in the encapsulated MPEG-4 AVC/H.264 bit stream if aspect_ratio_info_present_flag is 1.
- Pixel Aspect Ratio (0028,0034) shall be absent. This corresponds to a 'Sampling Aspect Ratio' (SAR) of 1:1.
- The values for Rows (0028,0010), Columns (0028,0011), Cine Rate (0018,0040), and Frame Time (0018,1063) or Frame Time Vector (0018,1065) shall be compliant with the High Profile / Level 4.2 of the MPEG-4 AVC/H.264 standard ([ISO/IEC 14496-10]) and restricted to a square pixel aspect ratio.

Note

1. The value of Planar Configuration (0028,0006) is irrelevant since the manner of encoding components is specified in the MPEG-4 AVC/H.264 standard, hence it is set to 0.
2. The frame rate of the acquiring camera for '30 Hz HD' MPEG-4 AVC/H.264 may be either 30 or 30/1.001 (approximately 29.97) frames/sec. Similarly, the frame rate in the case of 60 Hz may be either 60 or 60/1.001 (approximately 59.94) frames/sec. This may lead to small inconsistencies between the video timebase and real time. The relationship between frame rate and frame time is shown in Table 8-7.
3. The Frame Time (0018,1063) may be calculated from the frame rate of the acquiring camera. A frame rate of 29.97 frames per second corresponds to a frame time of 33.367 ms.
4. The value of chroma_format for this profile and level is defined by MPEG as 4:2:0.

Table 8-7. MPEG-4 AVC/H.264 High Profile / Level 4.2 Image Transfer Syntax Frame Rate Attributes

Video Type	Frame Rate (see Note 2)	Frame Time (see Note 3)
30 Hz HD	30	33.33 ms
25 Hz HD	25	40.0 ms
60 Hz HD	60	16.17 ms
50 Hz HD	50	20.00 ms

Stereo Pairs Present (0022,0028) shall be YES if stereoscopic pairs are present, otherwise shall be NO or absent.

Table 8-8. MPEG-4 AVC/H.264 High Profile / Level 4.2 Image Transfer Syntax Stereo Attributes

Transfer Syntax	Stereo Pairs Present	Stereo Frame Packing Format
MPEG-4 AVC/H.264 High Profile / Level 4.2 for 2D Image Compression	NO or absent	absent
MPEG-4 AVC/H.264 High Profile / Level 4.2 for 3D Image Compression	YES	present

One fragment shall contain the whole MPEG-4 AVC/H.264 bit stream.

Note

If a video stream exceeds the maximum length of one fragment (approximately 4 GB), it may be sent as multiple SOP Instances, but each SOP Instance will contain an independent and playable bit stream, and not depend on the encoded bit stream in other (previous) instances. The manner in which such separate instances are related is not specified in the standard, but mechanisms such as grouping into the same Series, and references to earlier instances using Referenced Image Sequence may be used.

The container format for the video bit stream shall be MPEG-2 Transport Stream, a.k.a. MPEG-TS (see [ISO/IEC 13818-1]) or MPEG-4, a.k.a. MP4 container (see [ISO/IEC 14496-12] and [ISO/IEC 14496-14]). The PTS/DTS of the transport stream shall be used in the MPEG coding. Any audio components present within the bit stream shall be interleaved as defined for MPEG-4 AVC/H.264 High Profile Level 4.1 (see Section 8.2.7).

8.2.9 MPEG-4 AVC/H.264 Stereo HiP@Level4.2 Video Compression

DICOM provides a mechanism for supporting the use of MPEG-4 AVC/H.264 Image Compression through the Encapsulated Format (see PS3.3). Annex A defines a Transfer Syntax that references the MPEG-4 AVC/H.264 Standard.

MPEG-4 AVC/H.264 Stereo High Profile can achieve better compression by additionally making use of prediction between the base and dependent stereoscopic views. The base view frames make use of intra and inter prediction as in MPEG-4 AVC/H.264 High Profile. This makes it possible for decoders which do not know how to decode the stereoscopic data to decode only the base view. The dependent view is encoded to make use of redundancy due to prediction based upon similarities between the base and the dependent views.

MPEG-4 AVC/H.264 Stereo High Profile makes use of the Level table A-1 of the MPEG-4 specification to set through-put limits. The properties required by the MPEG-4 AVC/H.264 Stereo High Profile Compression are identical to the properties defined in Section 8.2.8, except that Stereo Pairs Present (0022,0028) shall always be YES.

The container format for the video bitstream shall be MPEG-2 Transport Stream, a.k.a. MPEG-TS (see [ISO/IEC 13818-1]) or MPEG-4, a.k.a. MP4 container (see [ISO/IEC 14496-12] and [ISO/IEC 14496-14]). The PTS/DTS of the transport stream shall be used in the MPEG coding. Any audio components present within the bit stream shall be interleaved as defined for MPEG-4 AVC/H.264 High Profile Level 4.1 (see Section 8.2.7).

8.3 Waveform Data and Related Data Elements

The DICOM protocol provides for the exchange of encoded time-based signals, or waveforms, encoded in the Waveform Data Element (5400,1010).

Note

Per Section 7.6, an IOD supporting multiple sets of Waveform Data will encapsulate Data Element (5400,1010) within a Sequence.

Encoded Waveform Data of various bit depths is accommodated through the Waveform Bits Allocated (5400,1004) Data Element. This element defines the size of each waveform data sample within the Waveform Data (5400,1010). Allowed values are 8 and 16 bits.

The Value Representation of the Waveform Data (5400,1010) shall be OW; OB shall be used in cases where Waveform Bits Allocated has a value of 8, but only with Transfer Syntaxes where the Value Representation is explicitly conveyed.

Note

1. Under the Default Transfer Syntax, OB and OW VRs have the identical byte transfer order.
2. Conversion of a SOP Instance from the Default Transfer Syntax to an Explicit VR Transfer Syntax (uncompressed) requires the interpretation of the Waveform Bits Allocated (5400,1004) Data Element, to determine the proper VR of the Waveform Data.

The following data elements related to Waveform Data shall be encoded with the same VR as Waveform Data: Channel Minimum Value (5400,0110), Channel Maximum Value (5400,0112) and Waveform Padding Value (5400,100A).

8.4 Pixel Data Provider Service

Specific Transfer Syntaxes allow for the pixel data of the message to be replaced with a reference to a pixel data provider service. The pixel data provider service that is referenced supplies the pixel data using a network protocol that is defined outside DICOM.

Note

The Pixel Data Provider Service is not applicable to Pixel Data encoded as Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009).

8.4.1 JPIP Referenced Pixel Data

DICOM provides a mechanism for supporting the use of JPEG 2000 Interactive Protocol through the inclusion of a URL reference to a pixel data provider service. Annex A defines two Transfer Syntaxes that utilize URL references to a JPIP pixel data provider service.

The use of the these Transfer Syntaxes requires that the Pixel Data Provider URL specify a URL that will represent the JPIP request including the specific target information. Additional parameters required by the application may be appended to the URL when accessing the pixel data provider.

Note

For example, a JPIP request for a 200 by 200 pixel rendition of the entire image can be constructed from the Pixel Data Provider URL as follows:

Pixel Data Provider URL (0028,7FE0) = http://server.xxx/jpipserver.cgi?target=imgxyz.jp2

URL Generated by the application = http://server.xxx/jpipserver.cgi?target=imgxyz.jp2&fsiz=200,200

The JPIP client shall only request a JPEG 2000 bit stream.

The JPIP server shall return a Content-type of image/jp2, image/jpp-stream or image/jpt-stream, all of which shall be supported by the JPIP client.

The Number of Frames (0028,0008) attribute, if present in the Data Set, identifies the number of frames available for this image. Each frame is accessible as a separate JPIP code stream. Code streams referenced in the URL Target shall be sequentially numbered starting with stream 1.

Note

For example, a JPIP request for a 200 by 200 pixel rendition of frame 17 of a multi-frame image can be constructed from Pixel Data Provider URL as follows:

Pixel Data Provider URL (0028,7FE0) = http://server.xxx/multiframeimage.jp2

URL Generated by the application = http://server.xxx/multiframeimage.jp2?fsiz=200,200&stream=17

A valid stream query parameter value is always less than or equal to the value in the Number of Frames (0028,0008).

The syntax of the Pixel Data Provider URL (0028,7FE0) is defined in [ISO/IEC 15444-9] Annex C (Client Request). That standard respects the URI recommendations [RFC 3986]. The transport protocol shall be HTTP or HTTPS.

Note

1. According to [ISO/IEC 15444-9], "Each JPIP request is directed to a specific representation of a specific original named resource or a specific portion of that resource. That resource may be a physically stored file or object, or may be something that is created virtually by the server upon request."

   "The Target request field specifies the original named resource to which the request is directed. It is specified using a PATH, which could be a simple string or a URI. If the Target field is not specified and the request is carried over HTTP, then the JPIP request shall be directed to the resource specified through the path component of the JPIP request URL."

2. Transport over UDP or other protocols is not supported.

9 Unique Identifiers (UIDs)

Unique Identifiers (UIDs) provide the capability to uniquely identify a wide variety of items. They guarantee uniqueness across multiple countries, sites, vendors and equipment. Different classes of objects, instance of objects and information entities can be distinguished from one another across the DICOM universe of discourse irrespective of any semantic context.

Note

For example the same UID value cannot be used to identify both a study instance (Study Instance UID) and a series instance (Series Instance UID) within that study or a different study. Implementers also need to be cautioned against building new UID values by derivation (for example by adding a suffix) from a UID assigned by another implementation.

The UID identification scheme is based on the OSI Object Identification (numeric form) as defined by the ISO 8824 standard. All Unique Identifiers, used within the context of the DICOM Standard, are registered values as defined by ISO 9834-1 to ensure global uniqueness. The uses of such UIDs are defined in the various Parts of the DICOM Standard.

Each UID is composed of two parts, an <org root> and a <suffix>:

UID = <org root>.<suffix>

The <org root> portion of the UID uniquely identifies an organization, (i.e., manufacturer, research organization, NEMA, etc.), and is composed of a number of numeric components as defined by ISO 8824. The <suffix> portion of the UID is also composed of a number of numeric components, and shall be unique within the scope of the <org root>. This implies that the organization identified in the <org root> is responsible for guaranteeing <suffix> uniqueness by providing registration policies. These policies shall guarantee <suffix> uniqueness for all UIDs created by that organization. Unlike the <org root>, which may be common for UIDs in an organization, the <suffix> shall take different unique values between different UIDs that identify different objects.

The <org root> "1.2.840.10008" is reserved for DICOM defined items (such as DICOM Transfer Syntaxes) and shall not be used for privately defined items (such as an Image Instance).

Although a specific implementation may choose some particular structure for its generated UIDs, it should never assume that a UID carries any semantics. Thus, a UID shall not be "parsed" to find a particular value or component. Component definition (for the suffix) is implementation specific and may change as long as uniqueness is maintained. Parsing UIDs may jeopardize the ability to inter-operate as implementations evolve.

Example of UID structure is given in Annex C.

9.1 UID Encoding Rules

The DICOM UID encoding rules are defined as follows:

- Each component of a UID is a number and shall consist of one or more digits. The first digit of each component shall not be zero unless the component is a single digit.

  Note

  Registration authorities may distribute components with non-significant leading zeroes. The leading zeroes should be ignored when being encoded (i.e.,"00029" would be encoded "29").

- Each component numeric value shall be encoded using the characters 0-9 of the Basic G0 Set of the International Reference Version of ISO 646:1990 (the DICOM default character repertoire).
- Components shall be separated by the character "." (2EH).
- If ending on an odd byte boundary, except when used for network negotiation (see PS3.8), one trailing NULL (00H), as a padding character, shall follow the last component in order to align the UID on an even byte boundary.
- UIDs, shall not exceed 64 total characters, including the digits of each component, separators between components, and the NULL (00H) padding character if needed.

9.2 Unique Identifier Registration

Each UID used in DICOM shall be defined and registered in one of the following two ways:

- DICOM defined and registered UIDs
- Privately defined and registered UIDs

Both UIDs use the same encoding rules as defined in Section 9.1. See Annex C for a more detailed description of the UID registration process.

9.2.1 DICOM Defined and Registered Unique Identifiers

A limited number of registered DICOM Defined UIDs are used within the DICOM Standard. The organization responsible for the definition and registration of such DICOM UIDs is NEMA.

The registration process will rely on the publication of the DICOM Registered UIDs in PS3.6.

9.2.2 Privately Defined Unique Identifiers

Privately Defined UIDs are commonly used within DICOM. However, such UIDs will not be registered by NEMA. Organizations that define private UIDs are responsible for properly registering their UIDs (at least obtain a registered <Org Root>) as defined for OSI Object Identifiers (ISO 9834-1). The private organization defining the UID shall accept the responsibility of ensuring its uniqueness.

10 Transfer Syntax

A Transfer Syntax is a set of encoding rules able to unambiguously represent one or more Abstract Syntaxes. In particular, it allows communicating Application Entities to negotiate common encoding techniques they both support (e.g., byte ordering, compression, etc.). A Transfer Syntax is an attribute of a Presentation Context, one or more of which are negotiated at the establishment of an Association between DICOM Application Entities. This Association negotiation is specified in PS3.8 and discussed in PS3.7.

The selection of a Transfer Syntax applies to the encoding rules for the Data Set portion of a DICOM Message only. All DICOM Standard and Private Transfer Syntaxes implicitly specify a fixed encoding for the Command Set portion of a DICOM Message as specified in PS3.7.

This part of the DICOM Standard defines standard DICOM Transfer Syntaxes and assigns a unique Transfer Syntax Name to each one. The standard DICOM Transfer Syntaxes are specified in Annex A. The DICOM notation for Transfer Syntax names is the notation used for UIDs (see Section 9).

The organization responsible for the definition and registration of DICOM Transfer Syntaxes is NEMA. NEMA guarantees uniqueness for all DICOM Transfer Syntax Names.

Privately defined Transfer Syntax Names may also be used; however, they will not be registered by NEMA. Organizations that define private Transfer Syntax Names shall follow the registration process defined in Section 9.2.

10.1 DICOM Default Transfer Syntax

DICOM defines a default Transfer Syntax, the DICOM Implicit VR Little Endian Transfer Syntax (UID = "1.2.840.10008.1.2 "), which shall be supported by every conformant DICOM Implementation. This implies that:

a. If an Application Entity issues an A-ASSOCIATE request, it shall offer the DICOM Implicit VR Little Endian Transfer Syntax in at least one of the Presentation Contexts associated with each offered Abstract Syntax.

   Note

   Offering Abstract Syntax (AS1) in two Presentation Contexts with Transfer Syntaxes (TS1) and (TS2) is not valid, but offering AS1-TS1, AS1-TS2 and AS1-TSD is valid because the DICOM Default Little Endian Transfer Syntax (TSD) is present in at least one of the Presentation Contexts that are based on Abstract Syntax (AS1).

b. If an Application Entity receives an A-ASSOCIATE indication corresponding to a request that follows the requirements specified in Section 10.1 (a), every Presentation Context related to a given Abstract Syntax cannot be rejected in an A-ASSOCIATE response for the reason that none of the Transfer Syntaxes are supported.

Both of these requirements, (a) and (b), are waived when the Application Entity sending the pixel data has only access to the pixel data in lossy compressed form and a Transfer Syntax that uses a pixel data reference is not offered.

Requirement (b) to accept the default Transfer Syntax is waived if a Transfer Syntax that uses a pixel data reference is offered.

Note

In other words, every sending AE is required to be able to convert any Data Set it is going to transmit into the default Transfer Syntax, regardless of the form in which it originally received or stored the Data Set, except in the single case of when it received it in a lossy compressed form. In that exceptional case, the sending AE is permitted to propose only the lossy compressed Transfer Syntax appropriate to the lossy form that was received.

In particular, this waiver does not apply to Data Sets received in a lossless compressed form, which means that any AE receiving a Data Set in a lossless compressed Transfer Syntax that needs to re-send the Data Set is required to be able to decompress it in order to support (at least) the default Transfer Syntax.

10.2 Transfer Syntax for a DICOM Default of Lossless JPEG Compression

DICOM defines a default for lossless JPEG Image Compression, which uses a subset of coding Process 14 with a first-order prediction (Selection Value 1). It is identified by Transfer Syntax UID = "1.2.840.10008.1.2.4.70" and shall be supported by every DICOM implementation that chooses to support one or more of the lossless JPEG compression processes. This implies that:

a. If an Application Entity issues an A-ASSOCIATE request where any offered Abstract Syntaxes is associated in one or more Presentation Context with a JPEG lossless compression Transfer Syntax, at least one of the Presentation Contexts that include this Abstract Syntax, shall include the DICOM Default Lossless JPEG Compression Transfer Syntax and the DICOM Default Transfer Syntax (uncompressed).

Note

Offering Abstract Syntax (AS1) in two Presentation Contexts with Transfer Syntaxes JPEG lossless (JL1) and (JL2) is not valid, but offering AS1-JL1, AS1-JL2, AS1-TSD, and AS1-JLD is valid because the DICOM Default JPEG Lossless Transfer Syntax (JLD) and the DICOM Default Transfer Syntax (TSD) are present in at least one of the Presentation Contexts that are based on Abstract Syntax (AS1).

b. If an Application Entity that supports one or more lossless JPEG Transfer Syntax receives an A-ASSOCIATE indication corresponding to a request that follows the requirements specified in Section 10.2 (a), every Presentation Context related to a given Abstract Syntax cannot be rejected in an A-ASSOCIATE response for the reason that the DICOM Default lossless JPEG Transfer Syntax is not supported.

10.3 Transfer Syntaxes for a DICOM Default of Lossy JPEG Compression

DICOM defines defaults for Lossy JPEG Image Compression, one for 8-bit images and the other for 12-bit images. JPEG coding Process 1 (identified by Transfer Syntax UID = "1.2.840.10008.1.2.4.50 ") is used for 8-bit images. JPEG coding Process 4 (identified by Transfer Syntax UID = "1.2.840.10008.1.2.4.51 ") is used for 12-bit images. This implies that:

a. If an Application Entity issues an A-ASSOCIATE request where any offered Abstract Syntaxes is associated in one or more Presentation Context(s) with a JPEG lossy compression Transfer Syntax, at least one of the Presentation Contexts that include this Abstract Syntax, shall include the appropriate DICOM Default Lossy JPEG Compression Transfer Syntax.

Note

Offering Abstract Syntax (AS1) in two Presentation Contexts with Transfer Syntaxes JPEG lossy (JL1) and (JL2) is not valid, but offering AS1-JL1, AS1-JL2 and AS1-JLD is valid because the DICOM Default JPEG Lossy Transfer Syntax (JLD) is present in at least one of the Presentation Contexts that are based on Abstract Syntax (AS1).2. The DICOM Default Transfer Syntax (uncompressed) may be offered if the sender has access to the original pixel data in an uncompressed or lossless compressed form.

b. If an Application Entity that supports one or more Lossy JPEG Transfer Syntaxes receives an A-ASSOCIATE indication corresponding to a request that follows the requirements specified in Section 10.3 (a), every Presentation Context related to a given Abstract Syntax cannot be rejected in an A-ASSOCIATE response for the reason that the DICOM Default lossy JPEG Transfer Syntax is not supported.

Note

The 12 bit default Transfer Syntax 1.2.840.10008.1.2.4.51 can also be used to encode 8 bit images, but the bit stream required is not identical to that used in the 8 bit default Transfer Syntax 1.2.840.10008.1.2.4.50 (see A.4.1).

10.4 Transfer Syntax For DICOM RLE Compression

DICOM defines the RLE Compression (see Annex G). This implies that:

a. If an Application Entity issues an A-ASSOCIATE request where any offered Abstract Syntaxes is associated in one or more Presentation Contexts(s) with RLE compression Transfer Syntax, at least one of the Presentation Contexts that include this Abstract Syntax, shall include the DICOM Default Transfer Syntax (uncompressed).

10.5 Transfer Syntax For A DICOM Default of Lossless and Lossy (Near-lossless) JPEG-LS Compression

One Transfer Syntax is specified for JPEG-LS Lossless Image Compression, and one Transfer Syntax is specified for JPEG-LS Lossy (Near-Lossless) Image Compression. The JPEG-LS Lossless Transfer Syntax shall be supported as a baseline if the JPEG-LS Lossy (Near-Lossless) Transfer Syntax is supported.

10.6 Transfer Syntax For JPEG 2000 Compression

One Transfer Syntax is specified for JPEG 2000 Image Compression (Lossless Only), and one Transfer Syntax is specified for JPEG 2000 Image Compression. Either of these may be negotiated separately and there is no default or baseline specified (other than described in Section 10.1).

Note

1. All JPEG 2000 codecs are required by [ISO/IEC 15444-1] to support both reversible and irreversible wavelet and multi-component transformations. The reason for specifying two separate Transfer Syntaxes in DICOM is to allow an application to request the transfer of images in a lossless manner when possible. The JPEG 2000 Image Compression Transfer Syntax allows for either lossless or lossy compression to be used at the sender's discretion.

2. No baseline using other compression schemes is required.

3. When the pixel data has been received in the JPEG 2000 Image Compression Transfer Syntax, since it may have been lossy compressed, the waiver of the requirement in Section 10.1 to support the DICOM default Transfer Syntax still applies.

In addition, one Transfer Syntax is specified for JPEG 2000 Multi-component Image Compression (Lossless Only) with Multi-Component Transformation Extensions, and one Transfer Syntax is specified for JPEG 2000 Multi-component Image Compression with Multi-Component Transformation Extensions. Either of these may be negotiated separately and there is no default or baseline specified (other than described in Section 10.1).

Note

JPEG 2000 codecs that support the Part 2 JPEG 2000 Multi-Component Transformation Extensions are required to support all the multi-component extensions as described in Annex J of [ISO/IEC 15444-2]. This includes both array based transformations and the 9-7 and 5-3 wavelet transformations that are also used in Part 1 of JPEG 2000. This also includes component reordering, component collections and application of more than one multi-component transformation in succession.

10.7 Transfer Syntax For MPEG2 MP@ML Image Compression

One Transfer Syntax is specified for MPEG2 MP@ML Image Compression.

10.8 Transfer Syntax For JPIP Referenced Pixel Data

Two Transfer Syntaxes are specified for JPIP Referenced Pixel Data.

The persistence of the references in objects transferred with one of these transfer syntaxes is not defined. That is, applications should make no assumptions as to the timeframe when the referenced pixel data will be available. Due to the indeterminate time that the URL remains valid, it may be inappropriate to cache the URL. Because the pixel data may not have been retrieved in its entirety or full fidelity, it may be inappropriate to use this transfer syntax for the purpose of permanent storage or to reference such instances in Storage Commitment and Performed Procedure Step service classes.

These transfer syntaxes shall not be used for media storage defined by PS3.10.

10.9 Transfer Syntax For MPEG2 MP@HL Image Compression

One Transfer Syntax is specified for MPEG2 MP@HL Image Compression.

10.10 Transfer Syntax For MPEG-4 AVC/H.264 HiP@Level4.1 Image Compression

One Transfer Syntax is specified for MPEG-4 AVC/H.264 High Profile / Level 4.1 Image Compression and one Transfer Syntax is specified for MPEG-4 AVC/H.264 BD-compliant High Profile / Level 4.1. Transfer Syntax MPEG-4 AVC/H.264 High Profile / Level 4.1 corresponds to the ITU-T H.264 standard's profile and level specifications. Transfer Syntax MPEG-4 AVC/H.264 BD-compliant High Profile / Level 4.1 corresponds to a restricted set of spatial and temporal resolutions described Table 8-4. This Transfer Syntax limits the ITU-T H.264 High Profile / Level 4.1 to HD video formats that are supported by Blu-ray™ (BDRWP 2.B).

10.11 Transfer Syntaxes for MPEG-4 AVC/H.264 HiP@Level4.2 Image Compression

One Transfer Syntax is specified for MPEG-4 AVC/H.264 High Profile / Level 4.2 for 2D Image Compression and one Transfer Syntax is specified for MPEG-4 AVC/H.264 High Profile / Level 4.2 for 3D Image Compression. Transfer Syntax MPEG-4 AVC/H.264 High Profile / Level 4.2 for 2D Image Compression corresponds to the ITU-T H.264 standard's profile and level specifications except that the use of frame packing formats for 3D video is not allowed as defined in Table 8-8. Transfer Syntax MPEG-4 AVC/H.264 High Profile / Level 4.2 for 3D Image Compression corresponds to the ITU-T H.264 standard's profile and level specifications. It should be used for transmitting stereoscopic 3D content with frame packing formats as defined in Table 8-8.

10.12 Transfer Syntax For MPEG-4 AVC/H.264 Stereo HiP@Level4.2 Image Compression

One Transfer Syntax is specified for MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2 Image Compression. Transfer Syntax MPEG-4 AVC/H.264 Stereo High Profile corresponds to the ITU-T H.264 standard's profile and level specifications.

A Transfer Syntax Specifications (Normative)

A.1 DICOM Implicit VR Little Endian Transfer Syntax

This Transfer Syntax applies to the encoding of the entire DICOM Data Set. This implies that when a DICOM Data Set is being encoded with the DICOM Implicit VR Little Endian Transfer Syntax the following requirements shall be met:

a. The Data Elements contained in the Data Set structure shall be encoded with Implicit VR (without a VR Field) as specified in Section 7.1.3.

b. The encoding of the overall Data Set structure (Data Element Tags, Value Length, and Value) shall be in Little Endian as specified in Section 7.3.

c. The encoding of the Data Elements of the Data Set shall be as follows according to their Value Representations:

- For all Value Representations defined in this part, except for the Value Representations OB and OW, the encoding shall be in Little Endian as specified in Section 7.3.
- For the Value Representations OB, OL and OW, the encoding shall meet the following specification depending on the Data Element Tag:
  - Data Element (7FE0,0010) Pixel Data has the Value Representation OW and shall be encoded in Little Endian.

    Note

    The OL Value Representation is not used for Pixel Data, even if it has a Bits Allocated (0028,0100) of 32, since OL was added to the standard after the encoding of Pixel Data had been established
  - Data Element (60xx,3000) Overlay Data has the Value Representation OW and shall be encoded in Little Endian.
  - Data Element (5400,1010) Waveform Data shall have Value Representation OW and shall be encoded in Little Endian.
  - Data Elements (0028,1201), (0028,1202), (0028,1203), (0028,1204) Red, Green, Blue, Alpha Palette Lookup Table Data have the Value Representation OW and shall be encoded in Little Endian.

    Note

    Previous versions of the Standard either did not specify the encoding of Data Elements (0028,1201), (0028,1202), (0028,1203) in this Part, but specified a VR of US or SS in PS3.6-1993, or specified OW in this Part but a VR of US, SS or OW in PS3.6-1996. The actual encoding of the values and their byte order would be identical in each case.
- Data Elements (0028,1101), (0028,1102),(0028,1103) Red, Green, Blue Palette Lookup Table Descriptor have the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Little Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.
- Data Elements (0028,1221),(0028,1222),(0028,1223) Segmented Red, Green, Blue Palette Color Lookup Table Data have the Value Representation OW and shall be encoded in Little Endian.
- Data Element (0028,3006) LUT Data has the Value Representation US or OW and shall be encoded in Little Endian.

  Note

  Previous versions of the Standard did not specify the encoding of these Data Elements in this Part, but specified a VR of US or SS in PS3.6-1998. A VR of OW has been added to support explicit VR transfer syntaxes. Moreover this element is always unsigned, therefore the VR of SS has been removed. The actual encoding of the values and their byte order would be identical in each case.
- Data Element (0028,3002) LUT Descriptor has the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Little Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.

- Data Element (0028,1408) Blending Lookup Table Data has the Value Representation OW and shall be encoded in Little Endian.
- Data ~~Elements (0066,0025) Vertex~~Element (0066,0129) Track Point Index List~~, (0066,0024) Edge Point Index List, (0066,0023) Triangle Point Index List and (0066,0029) Primitive Point Index List have~~ has the Value Representation ~~OW~~OL and shall be encoded in Little Endian and ~~are~~is always interpreted as unsigned.

Note

~~Encoding of Curve Data and Audio Sample Data was previously defined but has been retired. See PS3.5-2004.~~

1. Encoding of Curve Data and Audio Sample Data was previously defined but has been retired. See PS3.5-2004.
2. Vertex Point Index List (0066,0025), Edge Point Index List (0066,0024), Triangle Point Index List (0066,0023) and Primitive Point Index List (0066,0029) were previously defined with a value representation of OW and always interpreted as unsigned, but have been retired. These have been replaced by corresponding OL data elements, which allow values larger than 65535 to index the full range of points that can be encoded in Point Coordinates Data (0066,0016). See PS3.5-2015c.

This DICOM Implicit VR Little Endian Transfer Syntax shall be identified by a UID of Value "1.2.840.10008.1.2".

A.2 DICOM Little Endian Transfer Syntax (Explicit VR)

This Transfer Syntax applies to the encoding of the entire DICOM Data Set. This implies that when a DICOM Data Set is being encoded with the DICOM Little Endian Transfer Syntax the following requirements shall be met:

a. The Data Elements contained in the Data Set structure shall be encoded with Explicit VR (with a VR Field) as specified in Section 7.1.2.

b. The encoding of the overall Data Set structure (Data Element Tags, Value Length, and Value) shall be in Little Endian as specified in Section 7.3.

c. The encoding of the Data Elements of the Data Set shall be as follows according to their Value Representations:

- For all Value Representations defined in this part, except for the Value Representations OB and OW, the encoding shall be in Little Endian as specified in Section 7.3.
- For the Value Representations OB, OL and OW, the encoding shall meet the following specification depending on the Data Element Tag:
  - Data Element (7FE0,0010) Pixel Data
    - where Bits Allocated (0028,0100) has a value greater than 8 shall have Value Representation OW and shall be encoded in Little Endian;
    - where Bits Allocated (0028,0100) has a value less than or equal to 8 shall have the Value Representation OB or OW and shall be encoded in Little Endian.

    Note

    The OL Value Representation is not used for Pixel Data, even if it has a Bits Allocated (0028,0100) of 32, since OL was added to the standard after the encoding of Pixel Data had been established

  - Data Element (60xx,3000) Overlay Data
    - shall have the Value Representation OB or OW and shall be encoded in Little Endian.

    Note

    Previous versions of the Standard specified that the choice of OB or OW VR was based on whether or not Overlay Bits Allocated (60xx,0100) was greater than, or less than or equal to, 8. However, since only one bit plane can be encoded in each Overlay Data Element (60xx,3000), no value of Overlay Bits Allocated other than 1 makes sense. Such a restriction is now present in PS3.3.

- Data Element (5400,1010) Waveform Data has the Value Representation specified in its Explicit VR Field. The component points shall be encoded in Little Endian.
- Data Elements (0028,1201), (0028,1202), (0028,1203), (0028,1204) Red, Green, Blue, Alpha Palette Lookup Table Data have the Value Representation OW and shall be encoded in Little Endian.

  Note

  Previous versions of the Standard either did not specify the encoding of Data Elements (0028,1201), (0028,1202), (0028,1203) in this Part, but specified a VR of US or SS in PS3.6-1993, or specified OW in this Part but a VR of US, SS or OW in PS3.6-1996. The actual encoding of the values and their byte order would be identical in each case, though the explicitly encoded VR field would be different. However, an explicit VR of US or SS cannot be used to encode a table of 2^{16} elements, since the Value Length is restricted to 16 bits.
- Data Elements (0028,1101), (0028,1102),(0028,1103) Red, Green, Blue Palette Lookup Table Descriptor have the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Little Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.
- Data Elements (0028,1221), (0028,1222), (0028,1223) Segmented Red, Green, Blue Palette Color Lookup Table Data have the Value Representation OW and shall be encoded in Little Endian.
- Data Element (0028,3006) LUT Data has the Value Representation US or OW and shall be encoded in Little Endian.

  Note

  Previous versions of the Standard did not specify the encoding of these Data Elements in this Part, but specified a VR of US or SS in PS3.6-1998. However, an explicit VR of US or SS cannot be used to encode a table of 2^{16} elements, since the Value Length is restricted to 16 bits. Hence a VR of OW has been added. Moreover this element is always unsigned, therefore the VR of SS has been removed. The actual encoding of the values and their byte order would be identical in each case, though the explicitly encoded VR field would be different.
- Data Element (0028,3002) LUT Descriptor has the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Little Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.
- Data Element (0028,1408) Blending Lookup Table Data has the Value Representation OW and shall be encoded in Little Endian.
- Data ~~Elements (0066,0025) Vertex~~Element (0066,0129) Track Point Index List~~, (0066,0024) Edge Point Index List, (0066,0023) Triangle Point Index List and (0066,0029) Primitive Point Index List have~~ has the Value Representation ~~OW~~OL and shall be encoded in Little Endian and ~~are~~is always interpreted as unsigned.

Note

1. For Data encoded with the Value Representation OB, the Data encoding is unaffected by Little Endian or Big Endian byte ordering.
2. Encoding of Curve Data and Audio Sample Data was previously defined but has been retired. See PS3.5-2004.
3. Vertex Point Index List (0066,0025), Edge Point Index List (0066,0024), Triangle Point Index List (0066,0023) and Primitive Point Index List (0066,0029) were previously defined with a value representation of OW and always interpreted as unsigned, but have been retired. These have been replaced by corresponding OL data elements, which allow values larger than 65535 to index the full range of points that can be encoded in Point Coordinates Data (0066,0016). See PS3.5-2015c.

This DICOM Explicit VR Little Endian Transfer Syntax shall be identified by a UID of Value "1.2.840.10008.1.2.1".

A.3 DICOM Big Endian Transfer Syntax (Explicit VR)

This Transfer Syntax applies to the encoding of the entire DICOM Data Set. This implies that when a DICOM Data Set is being encoded with the DICOM Big Endian Transfer Syntax the following requirements shall be met:

a. The Data Elements contained in the Data Set structure shall be encoded with Explicit VR (with a VR Field) as specified in Section 7.1.2.

b. The encoding of the overall Data Set structure (Data Element Tags, Value Length, and Value) shall be in Big Endian as specified in Section 7.3.

c. The encoding of the Data Elements of the Data Set shall be as follows according to their Value Representations:

- For all Value Representations defined in this part, except for the Value Representations OB and OW, the encoding shall be in Big Endian as specified in Section 7.3.
- For the Value Representations OB, OL and OW, the encoding shall meet the following specification depending on the Data Element Tag:
  - Data Element (7FE0,0010) Pixel Data
    - where Bits Allocated (0028,0100) has a value greater than 8 shall have Value Representation OW and shall be encoded in Big Endian;
    - where Bits Allocated (0028,0100) has a value less than or equal to 8 shall have the Value Representation OB or OW and shall be encoded in Big Endian.

      Note

      The OL Value Representation is not used for Pixel Data, even if it has a Bits Allocated (0028,0100) of 32, since OL was added to the standard after the encoding of Pixel Data had been established
  - Data Element (60xx,3000) Overlay Data
    - shall have the Value Representation OB or OW and shall be encoded in Big Endian.

      Note

      Previous versions of the Standard specified that the choice of OB or OW VR was based on whether or not Overlay Bits Allocated (60xx,0100) was greater than, or less than or equal to, 8. However, since only one bit plane can be encoded in each Overlay Data Element (60xx,3000), no value of Overlay Bits Allocated other than 1 makes sense. Such a restriction is now present in PS3.3.
  - Data Element (5400,1010) Waveform Data has the Value Representation specified in its Explicit VR Field. The component points shall be encoded in Big Endian.
  - Data Elements (0028,1201), (0028,1202), (0028,1203), (0028,1204) Red, Green, Blue, Alpha Palette Lookup Table Data have the Value Representation OW and shall be encoded in Big Endian.

    Note

    Previous versions of the Standard either did not specify the encoding of Data Elements (0028,1201), (0028,1202), (0028,1203) in this Part, but specified a VR of US or SS in PS3.6-1993, or specified OW in this Part but a VR of US, SS or OW in PS3.6-1996. The actual encoding of the values and their byte order would be identical in each case, though the explicitly encoded VR field would be different. However, an explicit VR of US or SS cannot be used to encode a table of 2^{16} elements, since the Value Length is restricted to 16 bits.
  - Data Elements (0028,1101), (0028,1102),(0028,1103) Red, Green, Blue Palette Lookup Table Descriptor have the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Big Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.
  - Data Elements (0028,1221), (0028,1222), (0028,1223) Segmented Red, Green, Blue Palette Color Lookup Table Data have the Value Representation OW and shall be encoded in Big Endian.
  - Data Element (0028,3006) LUT Data has the Value Representation US or OW and shall be encoded in Big Endian.

Note

Previous versions of the Standard did not specify the encoding of these Data Elements in this Part, but specified a VR of US or SS in PS3.6-1998. However, an explicit VR of US or SS cannot be used to encode a table of 2^{16} elements, since the Value Length is restricted to 16 bits. Hence a VR of OW has been added. Moreover this element is always unsigned, therefore the VR of SS has been removed. The actual encoding of the values and their byte order would be identical in each case, though the explicitly encoded VR field would be different.

- Data Element (0028,3002) LUT Descriptor has the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Big Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.
- Data Element (0028,1408) Blending Lookup Table Data has the Value Representation OW and shall be encoded in Big Endian.
- Data ~~Elements (0066,0025) Vertex~~Element (0066,0129) Track Point Index List~~, (0066,0024) Edge Point Index List, (0066,0023) Triangle Point Index List and (0066,0029) Primitive Point Index List have~~ has the Value Representation ~~OW~~OL and shall be encoded in ~~Little~~Big Endian and ~~are~~is always interpreted as unsigned.

Note

1. For Data encoded with the Value Representation OB, the Data encoding is unaffected by Little Endian or Big Endian byte ordering.
2. Encoding of Curve Data and Audio Sample Data was previously defined but has been retired. See PS3.5-2004.
3. Vertex Point Index List (0066,0025), Edge Point Index List (0066,0024), Triangle Point Index List (0066,0023) and Primitive Point Index List (0066,0029) were previously defined with a value representation of OW and always interpreted as unsigned, but have been retired. These have been replaced by corresponding OL data elements, which allow values larger than 65535 to index the full range of points that can be encoded in Point Coordinates Data (0066,0016). See PS3.5-2015c.

This DICOM Explicit VR Big Endian Transfer Syntax shall be identified by a UID of Value "1.2.840.10008.1.2.2".

A.4 Transfer Syntaxes For Encapsulation of Encoded Pixel Data

These Transfer Syntaxes apply to the encoding of the entire DICOM Data Set, even though the image Pixel Data (7FE0,0010) portion of the DICOM Data Set is the only portion that is encoded by an encapsulated format. These Transfer Syntaxes shall only be used when Pixel Data (7FE0,0010) is present in the top level Data Set, and hence shall not be used when Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009) are present. This implies that when a DICOM Message is being encoded according to an encapsulation Transfer Syntax the following requirements shall be met:

1. The Data Elements contained in the Data Set structure shall be encoded with Explicit VR (with a VR Field) as specified in Section 7.1.2.
2. The encoding of the overall Data Set structure (Data Element Tags, Value Length, etc.) shall be in Little Endian as specified in Section 7.3.
3. The encoding of the Data Elements of the Data Set shall be as follows according to their Value Representations:
   - For all Value Representations defined in this part of the DICOM Standard, except for the Value Representations OB and OW, the encoding shall be in Little Endian as specified in Section 7.3.
   - For the Value Representations OB, OL and OW, the encoding shall meet the following specification depending on the Data Element Tag:
     - Data Element (7FE0,0010) Pixel Data may be encapsulated or native.

       It shall be encapsulated if present in the top-level Data Set (i.e., not nested within a Sequence Data Element).

Note

The distinction between fixed value length (native) and undefined value length (encapsulated) is present so that the top level Data Set Pixel Data can be compressed (and hence encapsulated), but the Pixel Data within an Icon Image Sequence may or may not be compressed.

If native, it shall have a defined Value Length, and be encoded as follows:

- where Bits Allocated (0028,0100) has a value greater than 8 shall have Value Representation OW and shall be encoded in Little Endian;
- where Bits Allocated (0028,0100) has a value less than or equal to 8 shall have the Value Representation OB or OW and shall be encoded in Little Endian.

Note

~~That is, as if the transfer syntax were Explicit VR Little Endian.~~

a. The OL Value Representation is not used for Pixel Data, even if it has a Bits Allocated (0028,0100) of 32, since OL was added to the standard after the encoding of Pixel Data had been established

b. That is, as if the transfer syntax were Explicit VR Little Endian.

If encapsulated, it has the Value Representation OB and is a sequence of bytes resulting from one of the encoding processes. It contains the encoded pixel data stream fragmented into one or more Item(s). This Pixel Data Stream may represent a Single or Multi-frame Image. See Table A.4-1 and Table A.4-2.

- The Length of the Data Element (7FE0,0010) shall be set to the Value for Undefined Length (FFFFFFFFH).
- Each Data Stream Fragment encoded according to the specific encoding process shall be encapsulated as a DICOM Item with a specific Data Element Tag of Value (FFFE,E000). The Item Tag is followed by a 4 byte Item Length field encoding the explicit number of bytes of the Item.

Note

Whether more than one fragment per frame is permitted or not is defined per Transfer Syntax.

- All items containing an encoded fragment shall be made of an even number of bytes greater or equal to two. The last fragment of a frame may be padded, if necessary, to meet the sequence item format requirements of the DICOM Standard.

Note

1. Any necessary padding may be added in the JPEG or JPEG-LS compressed data stream as per ISO 10918-1 and ISO 14495-1 such that the End of Image (EOI) marker ends on an even byte boundary, or may be appended after the EOI marker, depending on the implementation.
2. ISO 10918-1 and ISO 14495-1 define the ability to add any number of padding bytes FFH before any marker (all of which also begin with FFH). It is strongly recommended that FFH padding bytes not be added before the Start of Image (SOI) marker.

- The first Item in the Sequence of Items before the encoded Pixel Data Stream shall be a Basic Offset Table item. The Basic Offset Table Item Value, however, is not required to be present:
  - When the Item Value is not present, the Item Length shall be zero (00000000H) (see Table A.4-1).
  - When the Item Value is present, the Basic Offset Table Item Value shall contain concatenated 32-bit unsigned integer values that are byte offsets to the first byte of the Item Tag of the first fragment for each frame in the Sequence of Items. These offsets are measured from the first byte of the first Item Tag following the Basic Offset Table item (see Table A.4-2).

Note

1. For a Multi-Frame Image containing only one frame or a Single Frame Image, the Basic Offset Table Item Value may be present or not. If present it will contain a single 00000000H value.

2. Decoders of encapsulated pixel data, whether Single Frame or Multi-Frame, need to accept both an empty Basic Offset Table (zero length) and a Basic Offset Table filled with 32 bit offset values.

  - This Sequence of Items is terminated by a Sequence Delimiter Item with the Tag (FFFE,E0DD) and an Item Length Field of Value (00000000H) (i.e., no Value Field shall be present).

- Data Element (60xx,3000) Overlay Data

  - shall have the Value Representation OB or OW and shall be encoded in Little Endian.

- Data Element (5400,1010) Waveform Data has the Value Representation specified in its Explicit VR Field. The component points shall be encoded in Little Endian.

- Data Elements (0028,1201), (0028,1202), (0028,1203), (0028,1204) Red, Green, Blue, Alpha Palette Lookup Table Data have the Value Representation OW and shall be encoded in Little Endian.

  Note

  Previous versions of the Standard either did not specify the encoding of Data Elements 0028,1201), (0028,1202), (0028,1203) in this Part, but specified a VR of US or SS in PS3.6-1993, or specified OW in this Part but a VR of US, SS or OW in PS3.6-1996. The actual encoding of the values and their byte order would be identical in each case, though the explicitly encoded VR field would be different. However, an explicit VR of US or SS cannot be used to encode a table of 2^{16} elements, since the Value Length is restricted to 16 bits.

- Data Elements (0028,1101), (0028,1102),(0028,1103) Red, Green, Blue Palette Lookup Table Descriptor have the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Little Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.

- Data Elements (0028,1221), (0028,1222), (0028,1223) Segmented Red, Green, Blue Palette Color Lookup Table Data have the Value Representation OW and shall be encoded in Little Endian.

- Data Element (0028,3006) LUT Data has the Value Representation US or OW and shall be encoded in Little Endian.

  Note

  Previous versions of the Standard did not specify the encoding of these Data Elements in this Part, but specified a VR of US or SS in PS3.6-1998. However, an explicit VR of US or SS cannot be used to encode a table of 2^{16} elements, since the Value Length is restricted to 16 bits. Hence a VR of OW has been added. Moreover this element is always unsigned, therefore the VR of SS has been removed. The actual encoding of the values and their byte order would be identical in each case, though the explicitly encoded VR field would be different.

- Data Element (0028,3002) LUT Descriptor has the Value Representation SS or US (depending on rules specified in the IOD in PS3.3), and shall be encoded in Little Endian. The first and third values are always interpreted as unsigned, regardless of the Value Representation.

- Data Element (0028,1408) Blending Lookup Table Data has the Value Representation OW and shall be encoded in Little Endian.

- Data ~~Elements (0066,0025) Vertex~~Element (0066,0129) Track Point Index List~~, (0066,0024) Edge Point Index List, (0066,0023) Triangle Point Index List and (0066,0029) Primitive Point Index List have~~ has the Value Representation ~~OW~~OL and shall be encoded in Little Endian and ~~are~~is always interpreted as unsigned.

Note

1. For Data encoded with the Value Representation OB, the Data encoding is unaffected by Little Endian or Big Endian byte ordering.

2. Encoding of Curve Data and Audio Sample Data was previously defined but has been retired. See PS3.5-2004.

3. Vertex Point Index List (0066,0025), Edge Point Index List (0066,0024), Triangle Point Index List (0066,0023) and Primitive Point Index List (0066,0029) were previously defined with a value representation of OW and always interpreted as unsigned, but have been retired. These have been replaced by corresponding OL data elements, which allow values larger than 65535 to index the full range of points that can be encoded in Point Coordinates Data (0066,0016). See PS3.5-2015c.

Table A.4-1. Example for Elements of an Encoded Single-Frame Image Defined as a Sequence of Three Fragments Without Basic Offset Table Item Value

Pixel Data Element Tag	Value Representation		Data Element Length	Data Element				
(7FE0, 0010) with VR of OB	OB	0000H Reserved	FFFF FFFFH undefined length	Basic Offset Table with NO Item Value		First Fragment (Single Frame) of Pixel Data		
				Item Tag	Item Length	Item Tag	Item Length	Item Value
				(FFFE, E000)	0000 0000H	(FFFE, E000)	0000 04C6H	Compressed Fragment
4 bytes	2 bytes	2 bytes	4 bytes	4 bytes	4 bytes	4 bytes	4 bytes	04C6H bytes

Table A.4-1b. Example for Elements of an Encoded Single-Frame Image Defined as a Sequence of Three Fragments Without Basic Offset Table Item Value (continued)

Data Element Continued							
Second Fragment (Single Frame) of Pixel Data			Third Fragment (Single Frame) of Pixel Data			Sequence Delimiter Item	
Item Tag	Item Length	Item Value	Item Tag	Item Length	Item Value	Sequence Delim. Tag	Item Length
(FFFE, E000)	0000 024AH	Compressed Fragment	(FFFE, E000)	0000 0628H	Compressed Fragment	(FFFE,E0DD)	0000 000H
4 bytes	4 bytes	024AH bytes	4 bytes	4 bytes	0628Hbytes	4 bytes	4 bytes

Table A.4-2. Examples of Elements for an Encoded Two-Frame Image Defined as a Sequence of Three Fragments with Basic Table Item Values

Pixel Data Element Tag	Value Representation		Data Element Length	Data Element					
(7FE0, 0010) with VR of OB	OB	0000H Reserved	FFFF FFFFH undefined length	Basic Offset Table with Item Value			First Fragment (Frame 1) of Pixel Data		
				Item Tag	Item Length	Item Value	Item Tag	Item Length	Item Value
				(FFFE, E000)	0000 0008H	0000 0000H 0000 0646H	(FFFE, E000)	0000 02C8H	Compressed Fragment
4 bytes	2 bytes	2 bytes	4 bytes	4 bytes	4 bytes	0008H bytes	4 bytes	4 bytes	02C8H bytes

Table A.4-2b. Examples of Elements for an Encoded Two-Frame Image Defined as a Sequence of Three Fragments with Basic Table Item Values (continued)

Data Element Continued							
Second Fragment (Frame 1) of Pixel Data			Third Fragment (Frame 2) of Pixel Data			Sequence Delimiter Item	
Item Tag	Item Length	Item Value	Item Tag	Item Length	Item Value	Sequence Delimiter Tag	Item Length
(FFFE, E000)	0000 036EH	Compressed Fragment	(FFFE, E000)	0000 0BC8H	Compressed Fragment	(FFFE, E0DD)	0000 0000H
4 bytes	4 bytes	036EH bytes	4 bytes	4 bytes	0BC8H bytes	4 bytes	4 bytes

A.4.1 JPEG Image Compression

The International Standards Organization ISO/IEC JTC1 has developed an International Standard, ISO~~/IS-~~ 10918-1 (JPEG Part 1) and an International ~~Draft~~ Standard, ISO~~/IS-~~ 10918-2 (JPEG Part 2), known as the JPEG Standard, for digital compression and coding of continuous-tone still images (see Annex F for further details).

A DICOM Transfer Syntax for JPEG Image Compression shall be identified by a UID value, appropriate to its JPEG coding process, chosen from Table A.4-3.

Table A.4-3. DICOM Transfer Syntax UIDs for JPEG

DICOM Transfer Syntax UID	JPEG coding process	JPEG description
1.2.840.10008.1.2.4.50	1	baseline
1.2.840.10008.1.2.4.51	2(8-bit),4(12-bit)	extended
1.2.840.10008.1.2.4.57	14	lossless, non-hierarchical
1.2.840.10008.1.2.4.70	14 (Selection Value 1)	lossless, non-hierarchical, first-order prediction

Note

~~DICOM identifies, to increase the likelihood of successful association, three Transfer Syntaxes for Default JPEG Compression Image processes (see Section 8.2.1 and Section 10).~~

1. DICOM identifies, to increase the likelihood of successful association, three Transfer Syntaxes for Default JPEG Compression Image processes (see Section 8.2.1 and Section 10).

2. Different JPEG processes may use different SOF marker segments. E.g., the baseline JPEG process 1 used with the 1.2.840.10008.1.2.4.50 Transfer Syntax uses the SOF0 marker, whereas the extended process 2 used with the 1.2.840.10008.1.2.4.51 Transfer Syntax uses the SOF1 marker. Accordingly, even though both bit streams encode 8 bit images using DCT and Huffman coding, the bit streams are not identical. Further, the extended process 2 may (but is not required to) use more AC and DC tables (up to 4 of each, rather than 2, per ISO 10918-1 Section F.1.3).

   It is not compliant to send bit streams with the SOF0 marker using the 1.2.840.10008.1.2.4.51 Transfer Syntax, but it is recommended that receivers of the 1.2.840.10008.1.2.4.51 Transfer Syntax be able to decode bit streams with the SOF0 marker (this asymmetry is consistent with ISO 10918-2 requirements; see A.4.1).

3. It is recommended that lossy compressed 8 bit images be encoded with the 1.2.840.10008.1.2.4.50 Transfer Syntax rather than the 1.2.840.10008.1.2.4.51 Transfer Syntax, unless the additional features of the extended process are required. Support of the 1.2.840.10008.1.2.4.50 Transfer Syntax is required for 8 bit images anyway (as described in 8.2.1) and to avoid confusion with the use of 12 bit images encoded with Process 4 in the 1.2.840.10008.1.2.4.51 Transfer Syntax (defined as a DICOM Default Transfer Syntax for 12 bit images in 10.3).

If the object allows multi-frame images in the pixel data field, then each frame shall be encoded separately. Each fragment shall contain encoded data from a single-frame image.

Note

Though a fragment may not contain encoded data from more than one frame, the encoded data from one frame may span multiple fragments. See note in Section 8.2.

For all images, including all frames of a multi-frame image, the JPEG Interchange Format shall be used (the table specification shall be included).

Note

This refers to the ISO 10918-1 "interchange format", not the ~~DIS~~ISO 10918-5 JPEG File Interchange Format (JFIF).

If images with Photometric Interpretation (0028,0004) YBR_FULL_422 or YBR_PARTIAL_422, are encoded with JPEG coding Process 1 (non hierarchical with Huffman coding), identified by DICOM Transfer Syntax UID "1.2.840.10008.1.2.4.50"the minimum compressible

unit is YYCBCR, where Y, CB, and CR are 8 by 8 blocks of pixel values. The data stream encodes two Y blocks followed by the corresponding CB and CR blocks.

A.4.2 RLE Compression

Annex G defines a RLE Compression Transfer Syntax. This transfer Syntax is identified by the UID value "1.2.840.10008.1.2.5". If the object allows multi-frame images in the pixel data field, then each frame shall be encoded separately. Each frame shall be encoded in one and only one Fragment (see Section 8.2).

A.4.3 JPEG-LS Image Compression

The International Standards Organization ISO/IEC JTC1 has developed an International Standard, ISO/IS-14495-1 (JPEG-LS Part 1), for digital compression and coding of continuous-tone still images (see Annex F for further details).

A DICOM Transfer Syntax for JPEG-LS Image Compression shall be identified by a UID value, appropriate to its JPEG-LS coding process.

Two Transfer Syntaxes are specified for JPEG-LS:

1. A Transfer Syntax with a UID of "1.2.840.10008.1.2.4.80 ", which specifies the use of the lossless mode of JPEG-LS. In this mode the absolute error between the source and reconstructed images will be zero.

2. A Transfer Syntax with a UID of "1.2.840.10008.1.2.4.81 ", which specifies the use of the near-lossless mode of JPEG-LS. In this mode, the absolute error between the source and reconstructed images will be constrained to a finite value that is conveyed in the compressed bit stream. Note that this process can, at the discretion of the encoder, be used to compress images with an error constrained to a value of zero, resulting in no loss of information.

If the object allows multi-frame images in the pixel data field, then each frame shall be encoded separately. Each fragment shall contain encoded data from a single-frame image.

Note

Though a fragment may not contain encoded data from more than one frame, the encoded data from one frame may span multiple fragments. See note in Section 8.2.

For all images, including all frames of a multi-frame image, the JPEG-LS Interchange Format shall be used (all parameter specifications shall be included).

A.4.4 JPEG 2000 Image Compression

The International Standards Organization ISO/IEC JTC1 has developed an International Standard, ISO/IS 15444-1 (JPEG 2000 Part 1), for digital compression and coding of continuous-tone still images (see Annex F for further details).

A DICOM Transfer Syntax for JPEG 2000 Image Compression shall be identified by a UID value, appropriate to the choice of JPEG 2000 coding process.

Two Transfer Syntaxes are specified for JPEG 2000 Part 1:

1. A Transfer Syntax with a UID of "1.2.840.10008.1.2.4.90 ", which specifies the use of the lossless (reversible) mode of JPEG 2000 Part 1 ([ISO/IEC 15444-1]) (i.e., the use of a reversible wavelet transformation and a reversible color component transformation, if applicable, and no quantization).

2. A Transfer Syntax with a UID of "1.2.840.10008.1.2.4.91", which specifies the use of either:

   a. the lossless (reversible) mode of JPEG 2000 Part 1 ([ISO/IEC 15444-1]) (i.e., the use of a reversible wavelet transformation and a reversible color component transformation, if applicable, and no quantization or code stream truncation), or

   b. the lossy (irreversible) mode of JPEG 2000 Part 1 ([ISO/IEC 15444-1]) (i.e., the use of an irreversible wavelet transformation and an irreversible color component transformation, if applicable, and optionally quantization, or the use of a reversible wavelet transformation and a reversible color component transformation, if applicable, followed by code stream truncation).

   The choice reversible versus irreversible is at the discretion of the sender (SCU or FSC/FSU).

Note

When using the irreversible wavelet transformation and an irreversible color component transformation, if applicable, even if no quantization is performed, some loss will always occur due to the finite precision of the calculation of the wavelet and multi-component transformations.

Only the features defined in JPEG 2000 Part 1 ([ISO/IEC 15444-1]) are permitted for these two Transfer Syntaxes. Additional features and extensions that may be defined in other parts of JPEG 2000 shall not be included in the compressed bitstream unless they can be decoded or ignored without loss of fidelity by all Part 1 compliant implementations.

If the object allows multi-frame images in the pixel data field, then for these JPEG 2000 Part 1 Transfer Syntaxes, each frame shall be encoded separately. Each fragment shall contain encoded data from a single frame.

Note

1. That is, the processes defined in [ISO/IEC 15444-1] shall be applied on a per-frame basis. The proposal for encapsulation of multiple frames in a non-DICOM manner in so-called "Motion-JPEG" or "M-JPEG" defined in 15444-3 is not used.

2. Though a fragment may not contain encoded data from more than one frame, the encoded data from one frame may span multiple fragments. See note in Section 8.2.

For all images, including all frames of a multi-frame image, the JPEG 2000 bitstream specified in [ISO/IEC 15444-1] shall be used. The optional JP2 file format header shall NOT be included.

Note

The role of the JP2 file format header is fulfilled by the non-pixel data attributes in the DICOM Data Set.

The International Standards Organization ISO/IEC JTC1 has also developed JPEG 2000 Part 2 ([ISO/IEC 15444-2]), which includes Extensions to the compression techniques described in Part 1 of the JPEG 200 Standard. Annex J of JPEG 2000 Part 2 describes extensions to the ICT and RCT multiple component transformations allowed in Part 1. Two types of multiple component transformations are defined in Annex J of Part 2 of JPEG 2000:

1. Array based multiple component transforms that form linear combinations of components to reduce the correlation between components. Array based transforms include prediction based transformations such as DPCM as well as more complicated transformations such as the KLT. These array based transformations can be implemented reversibly or irreversibly.

2. Wavelet based multiple component transformations using the same two wavelet filters as used in Part 1 of JPEG 2000 (5-3 reversible wavelet and 9-7 irreversible wavelet).

Annex J of JPEG 2000 Part 2 also describes a flexible mechanism to allow these techniques to be applied in sequence. Furthermore, it provides mechanisms that allow components to be re-ordered and grouped into component collections. Different multiple component transformation can then be applied to each component collection.

Two additional Transfer Syntaxes are specified for Part 2 JPEG 2000:

1. A Transfer Syntax with a UID of 1.2.840.10008.1.2.4.92, which specifies the use of the lossless (reversible) mode of JPEG 2000 Part 2 ([ISO/IEC 15444-2]) multiple component transformation extensions, as defined in Annex J of JPEG 2000 Part 2 (i.e., the use of a reversible wavelet transformation and a reversible multiple component transformation, and no quantization or code stream truncation).

2. A Transfer Syntax with a UID of 1.2.840.10008.1.2.4.93, which specifies the use of either:

   a. the lossless (reversible) mode of JPEG 2000 Part 2 ([ISO/IEC 15444-2]) multiple component transformation extensions, as defined in Annex J of JPEG 2000 Part 2 (i.e., the use of a reversible wavelet transformation and a reversible multiple component transformation, and no quantization), or

   b. the lossy (irreversible) mode of JPEG 2000 Part 2 ([ISO/IEC 15444-2]) multiple component transformation extensions, as defined in Annex J of JPEG 2000 Part 2 (i.e., the use of an irreversible wavelet transformation and an irreversible multiple component transformation, and optionally quantization, or the use of an reversible wavelet transformation and a reversible multiple component transformation, followed by code stream truncation).

Only the multiple component transformation extensions defined in Annex J of JPEG 2000 Part 2 ([ISO/IEC 15444-2]) are permitted for these two Transfer Syntaxes. Additional features and extensions that may be defined in other Annexes of JPEG 2000 Part 2 shall not be included in the compressed bitstream.

Note

the arbitrary wavelet transformations, as defined in Annex H of JPEG 2000 Part 2 ([ISO/IEC 15444-2]) are not allowed for these two Transfer Syntaxes. The only wavelet transformations that are allowed to be used as multiple component transformations are the reversible 5-3 wavelet transformation and the irreversible 9-7 wavelet transformation, as defined in Annex F of JPEG 2000 Part 1 ([ISO/IEC 15444-1]).

If the object allows multi-frame images in the pixel data field, then, for these JPEG 2000 Part 2 Transfer Syntaxes, the frames in the object are first processed using the multi-component transformation. After the multiple component transformation has been applied, the transformed frames are encoded using the process described in JPEG 2000 Part 1.

Optionally, the frames can be grouped into one or more component collections. The multiple component transformations are then applied to each component collection independently. The use of component collections can be used to reduce computational complexity and to improve access to specific frames on the decoder. If component collections are used, each fragment shall contain encoded data from a single component collection.

Note

1. The 3rd dimension transformations that are described in this Supplement are treated in Part 2 of JPEG 2000 as direct extensions to the color component transformations (RGB to YUV) that are described in Part 1 of JPEG 2000. For this reason, each image or frame in the sequence is called a "component". Although the term component is used as a generic term to identify an element of the 3rd dimension, no restriction is made or implied that the transformations in this Supplement apply only to multi-component (or multiple color channel) data. To compress a volumetric data set using this transfer syntax, each frame of the DICOM image is treated as a component of a multi-component image.

2. The progressive nature of the JPEG 2000 code stream allows for the decompression of the image before the complete image has been transferred. If a Storage SCP truncates the code stream by aborting the association, the instance has not been completely transferred and hence should not persist unless different UIDs are assigned (even though it may have been transiently used for display purposes).

3. It has been shown that the use of component collections does not significantly affect the compression efficiency (for details, see http://medical.nema.org/Dicom/minutes/WG-04/2004/2004-02-18/3D_compression_RSNA_2003_ver2.pdf).

4. Though a fragment may not contain encoded data from more than one component collection, the encoded data from one component collection may span multiple fragments.

A.4.5 MPEG2 Image Compression

The International Standards Organization ISO/IEC MPEG2 has developed an International Standard, [ISO/IEC 13818-2] 'Information Technology - Generic coding of moving pictures and associated audio information: video -- part 2', referred to as "MPEG-2".

A DICOM Transfer Syntax for MPEG2 Image Compression shall be identified by a UID value of either:

- 1.2.840.10008.1.2.4.100 corresponding to MPEG2 MP@ML option of the ISO/IEC MPEG2 Video standard
- 1.2.840.10008.1.2.4.101 corresponding to the MPEG2 MP@HL option of the ISO/IEC MPEG2 Video standard.

A.4.6 MPEG-4 AVC/H.264 HiP@Level4.1 Video Compression

The International Standards Organization ISO/IEC MPEG4 has developed an International Standard, [ISO/IEC 14496-10] (MPEG-4 Part 10), for the video compression of generic coding of moving pictures and associated audio information. This standard is jointly maintained and has identical technical content as the ITU-T H.264 standard.

A DICOM Transfer Syntax for MPEG-4 AVC/H.264 Image Compression shall be identified by a UID value of either:

- 1.2.840.10008.1.2.4.102 corresponding to the MPEG-4 AVC/H.264 High Profile / Level 4.1 of the ITU-T H.264 Video standard

- 1.2.840.10008.1.2.4.103 corresponding to the MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1 of the ITU-T H.264 Video standard with the temporal and spatial resolution restrictions defined in Table 8-4.

A.4.7 MPEG-4 AVC/H.264 HiP@Level4.2 Video Compression

The International Standards Organization ISO/IEC MPEG4 has developed an International Standard, [ISO/IEC 14496-10] (MPEG-4 Part 10), for the video compression of generic coding of moving pictures and associated audio information. This standard is jointly maintained and has identical technical content as the ITU-T H.264 standard.

A DICOM Transfer Syntax MPEG-4 AVC/H.264 High Profile / Level 4.2 for 2D Image Compression shall be identified by a UID value of:

- 1.2.840.10008.1.2.4.104 corresponding to the MPEG-4 AVC/H.264 High Profile / Level 4.2 of the ITU-T H.264 Video standard with the restriction that frame packing for stereoscopic 3D content shall not be used as defined in Table 8-8.

A DICOM Transfer Syntax MPEG-4 AVC/H.264 High Profile / Level 4.2 for 3D Image Compression shall be identified by a UID value of:

- 1.2.840.10008.1.2.4.105 corresponding to the MPEG-4 AVC/H.264 High Profile / Level 4.2 of the ITU-T H.264 Video standard. It should be used for transmitting stereoscopic 3D content with frame packing formats as defined in Table 8-8.

A.4.8 MPEG-4 AVC/H.264 Stereo HiP@Level4.2 Video Compression

The International Standards Organization ISO/IEC MPEG4 has developed an International Standard, [ISO/IEC 14496-10] (MPEG-4 Part 10), for the video compression of generic coding of moving pictures and associated audio information. This standard is jointly maintained and has identical technical content as the ITU-T H.264 standard.

A DICOM Transfer Syntax for MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2 Image Compression shall be identified by a UID value of:

- 1.2.840.10008.1.2.4.106 corresponding to the MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2 of the ITU-T H.264 Video standard.

A.5 DICOM Deflated Little Endian Transfer Syntax (Explicit VR)

This Transfer Syntax applies to the encoding of the entire DICOM Data Set.

The entire Data Set is first encoded according to the rules specified in Section A.2.

The entire byte stream is then compressed using the "Deflate" algorithm defined in Internet RFC 1951.

If the deflate algorithm produces an odd number of bytes then a single trailing NULL byte shall be added after the last byte of the deflated bit stream.

Note

1. The Pixel Data in Pixel Data (7FE0,0010), Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009) is not handled in any special manner. The pixel data is first encoded as sequential uncompressed frames without encapsulation, and then is handled as part of the byte stream fed to the "deflate" compressor in the same manner as the value of any other attribute.

2. This transfer syntax is particularly useful for compression of objects without pixel data, such as structured reports. It is not particularly effective at image compression, since any benefit obtained from compressing the non-pixel data is offset by less effective compression of the much larger pixel data.

3. A freely available reference implementation of the "deflate" compressor may be found in the zlib package, which may be downloaded from http://www.zlib.net/.

4. Although the encoded stream may be padded by a trailing NULL byte, the end of the deflated bit stream will be indicated by the delimiter that will occur before the padding.

In order to facilitate interoperability of implementations conforming to the DICOM Standard that elect to use this Transfer Syntax, the following policy is specified:

- Any implementation that has elected to support the Deflated Explicit VR Little Endian Transfer Syntax for any Abstract Syntax, shall also support the Explicit VR Little Endian Transfer for that Abstract Syntax

  Note

  1. This requirement to support the (uncompressed) Explicit VR Little Endian Transfer Syntax is in order to ensure full-fidelity exchange of VR information in the case that the Association Acceptor does not support the Deflated Explicit VR Little Endian Transfer Syntax. The requirement specified in Section 10.1 of this part, that the Default Implicit VR Little Endian Transfer Syntax be supported by all implementations except those that only have access to lossy compressed pixel data, is not waived. In other words, an implementation must support all three transfer syntaxes.

  2. There are no such "baseline" requirements on media, since such requirements are at the discretion of the Media Application Profile. Furthermore, sufficient object "management" information should be present in the DICOMDIR even if an individual application cannot decompress an instance encoded with the deflated transfer syntax.

This DICOM Deflated Explicit VR Little Endian Transfer Syntax shall be identified by a UID of Value "1.2.840.10008.1.2.1.99".

A.6 DICOM JPIP Referenced Transfer Syntax (Explicit VR)

This Transfer Syntax applies to the encoding of the entire DICOM Data Set. This Transfer Syntax shall only be used when Pixel Data (7FE0,0010) is present in the top level Data Set, and hence shall not be used when Float Pixel Data (7FE0,0008) or Double Float Pixel Data (7FE0,0009) are present. This implies that when a DICOM Data Set is being encoded with the DICOM Little Endian Transfer Syntax the following requirements shall be met:

a. The Data Elements contained in the Data Set structure shall be encoded with Explicit VR (with a VR Field) as specified in Section 7.1.2.

b. The encoding of the overall Data Set structure (Data Element Tags, Value Length, and Value) shall be in Little Endian as specified in Section 7.3.

c. The encoding of the Data Elements of the Data Set shall be as follows according to their Value Representations:

   - For all Value Representations defined in this part, except for the Value Representations OB and OW, the encoding shall be in Little Endian as specified in Section 7.3.

   - For the Value Representations OB and OW, the encoding shall meet the following specification depending on the Data Element Tag:

     - Data Element (7FE0,0010) Pixel Data shall not be present, but rather pixel data shall be referenced via Data Element (0028,7FE0) Pixel Data Provider URL

     - Overlay data, if present, shall only be encoded in the Overlay Data attribute (60xx,3000), which shall have the Value Representation OB or OW and shall be encoded in Little Endian.

     - Data Element (0028,0004) Photometric Interpretation shall be limited to the values: MONOCHROME1, MONOCHROME2, YBR_ICT and YBR_RCT.

This DICOM JPIP Referenced Transfer Syntax shall be identified by a UID of Value "1.2.840.10008.1.2.4.94 ".

A.7 DICOM JPIP Referenced Deflate Transfer Syntax (Explicit VR)

This Transfer Syntax applies to the encoding of the entire DICOM Data Set.

The entire Data Set is first encoded according to the rules specified in Section A.6.

The entire byte stream is then compressed using the "Deflate" algorithm defined in Internet RFC 1951.

This DICOM JPIP Referenced Deflate Transfer Syntax shall be identified by a UID of Value "1.2.840.10008.1.2.4.95".

B Creating a Privately Defined Unique Identifier (Informative)

Privately defined Unique Identifiers (UIDs) are used in DICOM to uniquely identify items such as Specialized or Private SOP Classes, Image SOP Instances, Study SOP Instances, etc.

B.1 Organizationally Derived UID

A UID may be formed using a registered root (see Annex C) and an organization specific suffix. The manner in which the suffix of such an organizationally derived UID is defined is not constrained by the DICOM Standard. Only the guarantee of its uniqueness by the defining organization is required by DICOM.

The following example presents a particular choice made by a specific organization in defining its suffix to guarantee uniqueness of a SOP Instance UID.

```
 "1.2.840.xxxxx.3.152.235.2.12.187636473"
  \___________/ \_____________________/
     root     .        suffix
```

In this example, the root is:

- 1 Identifies ISO
- 2 Identifies ANSI Member Body
- 840 Country code of a specific Member Body (U.S. for ANSI)
- xxxxx Identifies a specific Organization.(assigned by ANSI)

In this example the first two components of the suffix relate to the identification of the device:

- 3 Manufacturer defined device type
- 152 Manufacturer defined serial number

The remaining four components of the suffix relate to the identification of the image:

- 235 Study number
- 2 Series number
- 12 Image number
- 187636473 Encoded date and time stamp of image acquisition

In this example, the organization has chosen these components to guarantee uniqueness. Other organizations may choose an entirely different series of components to uniquely identify its images. For example it may have been perfectly valid to omit the Study Number, Series Number and Image Number if the time stamp had a sufficient precision to ensure that no two images might have the same date and time stamp.

Because of the flexibility allowed by the DICOM Standard in creating organizationally derived UIDs, implementations should not depend on any assumed structure of UIDs and should not attempt to parse UIDs to extract the semantics of some of its components.

B.2 UUID Derived UID

ISO/IEC 9834-8 / ITU-T X.667 defines a method by which a UID may be constructed from the root "2.25." followed by a decimal representation of a Universally Unique Identifier (UUID). That decimal representation treats the 128 bit UUID as an integer, and may thus be up to 39 digits long (leading zeros must be suppressed).

Note

ISO/IEC 9834-8 / ITU-T X.667 is available at http://www.itu.int/rec/T-REC-X.667/en.

A UUID derived UID may be appropriate for dynamically created UIDs, such as SOP Instance UIDs, but is usually not appropriate for UIDs determined during application software design, such as private SOP Class or Transfer Syntax UIDs, or Implementation Class UIDs.

C DICOM Unique Identifier Registration Process (Informative)

This registration process applies to a number of unique identifiers that share the same properties, structure and registration process. It applies to the following identifiers:

- The Values assigned to DICOM Data Elements of Value Representation VR = UID (see Table 6.2-1). Such Data Elements are defined in PS3.3, PS3.4, PS3.6, and PS3.7.
- The DICOM Abstract Syntaxes Names. Abstract Syntax Names are defined in PS3.4.
- The DICOM Transfer Syntax Names. Transfer Syntax Names are defined in Annex A.
- The DICOM Application Context Names. Application Context Names are defined in PS3.7

UID structure is based on the numeric form of the OSI Object Identifier as defined by ISO 8824. Values shall be registered as defined by ISO 9834-1 to ensure global uniqueness.

The DICOM Standard assigns Values to a number of such unique identifiers. The organization responsible for their registration is NEMA, which guarantees uniqueness.

For privately registered identifiers, NEMA will not act as registration authority. Related organizations shall obtain their proper registration as defined for OSI Object Identifiers by ISO 9834-1 to ensure global uniqueness. National Standards Organizations representing a number of countries (e.g., UK, France, Japan, USA, etc.) for the International Standards Organization act as a registration authority by delegation from ISO, as defined by ISO 9834-1.

Note

1. For example, in the USA ANSI assigns, for a fee, Organization Identifiers to any requesting organization. Such an identifier may be used by the identified organization as a root to which it may add a suffix made of one or more components. The identified organization accepts the responsibility to properly register these suffixes to ensure uniqueness.

2. Following are two typical examples of obtaining a UID <org root>. These examples are not intended to illustrate all the possible methods for obtaining a UID <org root>, see ISO 8824 and ISO 9834-1 for complete specifications. Organization identifiers may be obtained from various ISO member bodies (e.g., IBN in Belgium, ANSI in the United States, AFNOR in France, BSI in Great Britain, DIN in Germany, COSIRA in Canada).

   The first example shows the case of an <org root> issued by an ISO Member Body (ANSI in the USA in this example). The <org root> is composed of an identifier for ISO, a member body branch identifier, a country code and an organization ID. Note that there is no requirement that an implementation using an ANSI issued <org root> be made or located in the USA. The <org root> is made up of the following components: 1.2.840.xxxxx

   - 1 identifies ISO
   - 2 identifies the ISO member body branch
   - 840 identifies the country code of a specific ISO member body (U.S. for ANSI)
   - xxxxx identifies a specific organization as registered by the ISO member body ANSI.

   The second example shows the case of an <org root> issued by ISO (is delegated to BSI) to an international organization. It is composed of an identifier for ISO, an international organization branch identifier, and an International Code Designator. The value of the<org root> is assigned by an international registration authority that may be used by many different UIDs defined by the same international organization. The <org root> is made up of the following components: 1.3.yyyy

   - 1 identifies ISO
   - 3 identifies the international organization branch

- yyyy identifies a specific organization as registered by an International Code Designator registration authority (see ISO 6523).

3. Example components of a <suffix> for unique identification of an image could include:

   - product
   - system identifier
   - study, series and image numbers
   - study, series and image date & times.

D Examples of Various Pixel Data and Overlay Encoding Schemes (Informative)

D.1 Detailed Example of Pixel Data Encoding

As specified in PS3.3, Image Pixel Data is stored within the Value of the Pixel Data Element (7FE0,0010). The order in which Pixel Data for an image plane is encoded is from left to right and top to bottom, a row at a time (see Figure D-1).

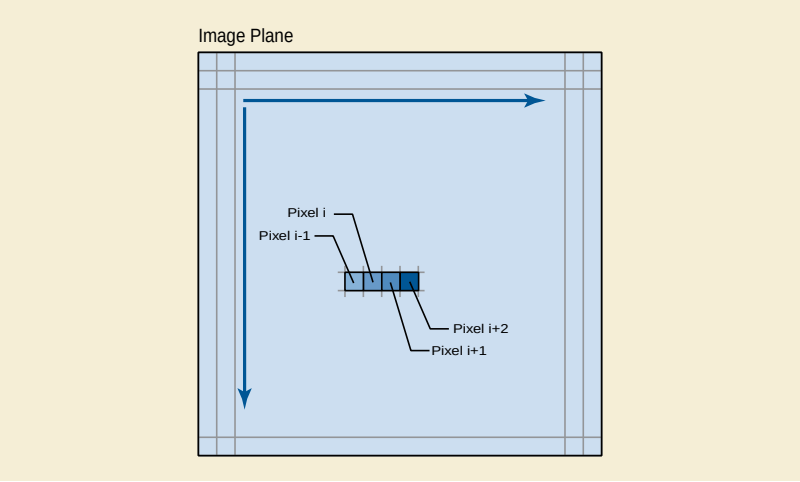

Figure D-1. An Image Pixel Plane

An individual pixel may consist of one or more Pixel Sample Values (e.g., color or multi-planar images). Each Pixel Sample Value can be expressed as either a binary 2's complement integer or a binary unsigned integer, as specified by the Pixel Representation Data Element (0028, 0103). The number of bits in each Pixel Sample Value is specified by Bits Stored (0028,0101). For 2's complement integer Pixel Samples the sign bit is the most significant bit of the Pixel Sample Value.

A Pixel Cell is the container for a Pixel Sample Value and optionally additional bits. These additional bits are used to place Pixels on certain boundaries (byte, word, etc.). A Pixel Cell exists for every individual Pixel Sample Value in the Pixel Data. The size of the Pixel Cells is specified by Bits Allocated (0028,0100) and is greater than or equal to the Bits Stored (0028,0101). The placement of the Pixel Sample Values within the Pixel Cells is specified by High Bit (0028,0102).

Any restrictions on the characteristics of a Pixel Cell and the Pixel Sample Value contained therein are specific to the Information Object Definition (e.g., Image Object) containing the Pixel Data Element (see PS3.3).

The Pixel Data Element, as specified by the DICOM Default Transfer Syntax in Section 10.1, has a Value Representation of OW (Other Word String). The Pixel Data in DICOM 3.0, as it was in ACR-NEMA 2.0, is packed, except that Bits Allocated is always either 1, or a multiple of 8 (see Figure D-2). One way to visualize this packed encoding is to imagine encoding the Pixel Cells as a concatenated stream of bits from the least significant bit of the first Pixel Cell up through the most significant bit of the last Pixel Cell. Within this stream, the most significant bit of any Pixel Cell is followed by the least significant bit of the next Pixel Cell. The Pixel Data can then be broken up into a stream of physical 16-bit words, each of which is subject to the byte ordering constraints of the Transfer Syntax.

All other (non-default) DICOM Transfer Syntaxes make use of explicit VR encoding. For these Transfer Syntaxes, all Pixel Data where Bits Allocated is less than or equal to 8 may be encoded with an explicit VR of OB (see Annex A). As in the OW case, Pixel Cells are packed together, but in this case the Pixel Data is broken up into a stream of physical 8-bit words.

Note

For Pixel Data encoded with an explicit VR of OB, the encoding of the Pixel Data is unaffected by Little Endian or Big Endian byte ordering.

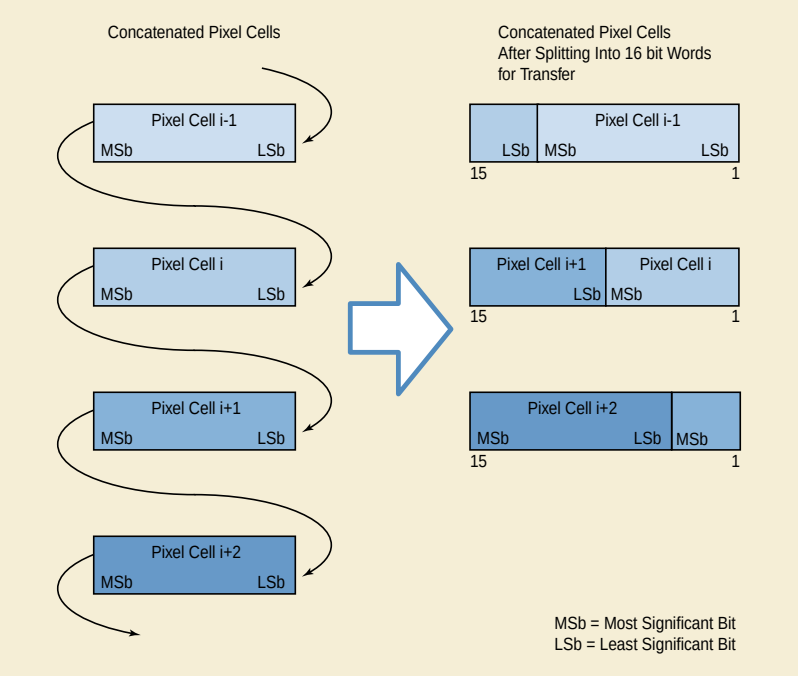

Figure D-2. Encoding (Packing) of Arbitrary Pixel Data with a VR of OW

With the exception of single bit images, Pixel Cells begin and end on byte or word boundaries and such that the Pixel Sample Value contained within also fits 'neatly' within a cell.

Figure D-3 is an example of Pixel Data encoding using the Value representation of OW for the purposes of clarification. The Example is a valid example for a CT Image Information Object.

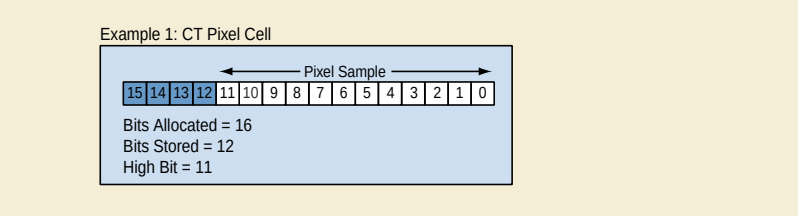

Figure D-3. Example Pixel Cells

Figure D-4 shows Pixel Data constructed of these example Pixel Cells as they are packed into a stream of 16-bit words.

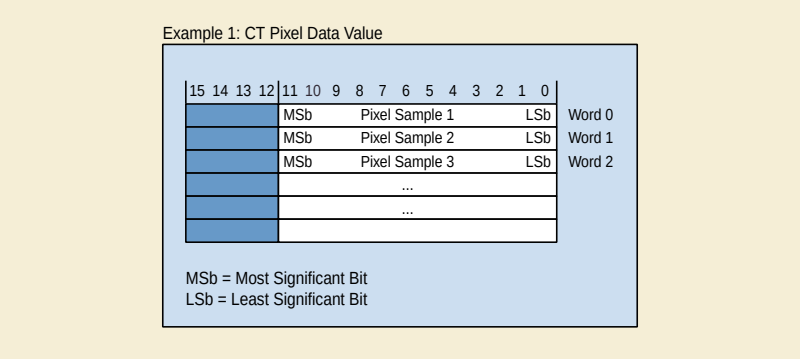

Figure D-4. Example Pixel Cells Packed into 16-bit Words (VR = OW)

Byte ordering becomes a consideration when we represent the Pixel Data physically, in memory, a file, or on a network.

In the memory of a byte-addressable Big Endian machine, the highest order byte (bits 8 - 15) in each 16-bit word has a binary address of x...x0. While in a byte-addressable Little Endian machine, the lowest order byte (bits 0 - 7) in each 16-bit word has a binary address of x...x0. Figure D-5 pictures our example Pixel Data streams as they would be addressed in the memory of both a Big Endian and a Little Endian machine.

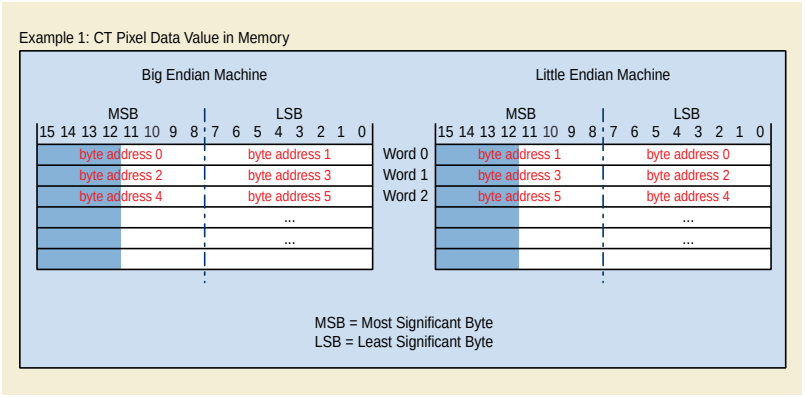

Figure D-5. Example Pixel Cells Byte Ordered in Memory (VR = OW)

Byte ordering is also specified as part of the negotiated Transfer Syntax used in the exchange of a DICOM message. Sixteen bit words are transmitted across the network (a byte at a time) least significant byte first in the case of a Little Endian Transfer Syntax and most significant byte first when using a Big Endian Transfer Syntax (see Figure D-6).

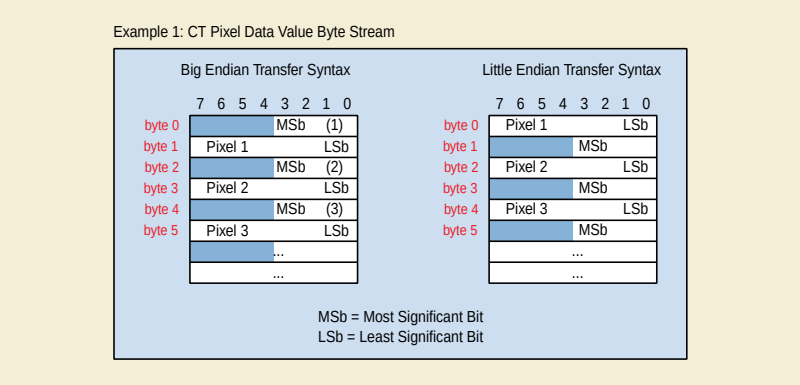

Figure D-6. Sample Pixel Data Byte Streams (VR = OW)

As a last pair of examples, for Pixel Data having the Value Representation OW and the following attributes: 8 bits allocated, 8 bits stored, and a high bit of 7; the resulting byte streams pictured in Figure D-7 are as they would be transmitted across a network and/or stored on media. For Pixel Data having the same attributes, but having the explicit Value Representation OB; the resulting byte streams are unaffected by byte ordering and are pictured in Figure D-8.

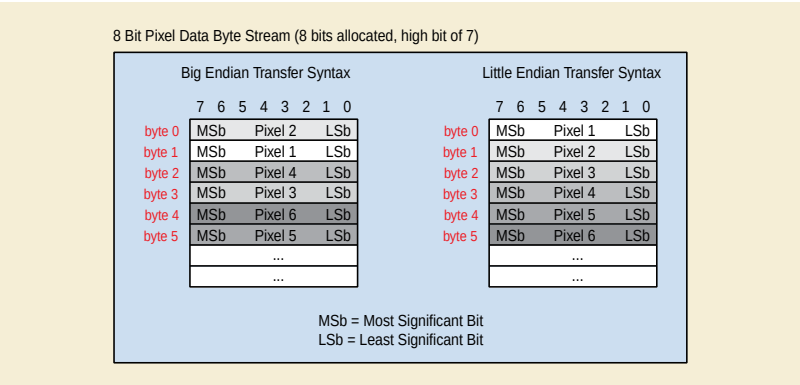

Figure D-7. Sample Pixel Data Byte Streams for 8-bits Allocated and 8-bits Stored (VR = OW)

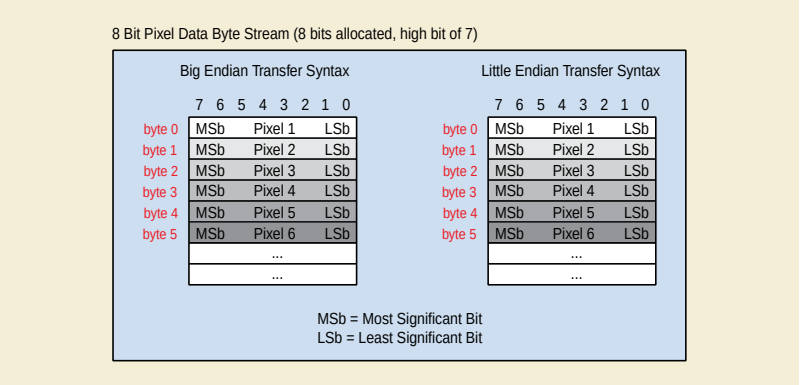

Figure D-8. Sample Pixel Data Byte Streams for 8-bits Allocated and 8-bits Stored (Explicit VR = OB)

D.2 Various Additional Examples of Pixel and Overlay Data Cells

The following examples further illustrate the use of the data elements for Bits Allocated (0028,0100), Bits Stored (0028,0101) and High Bit (0028,0102) in the encoding of Pixel and Overlay Data. All examples show sample Pixel Cells before being encoded in byte streams (and before being affected by a particular Transfer Syntax).

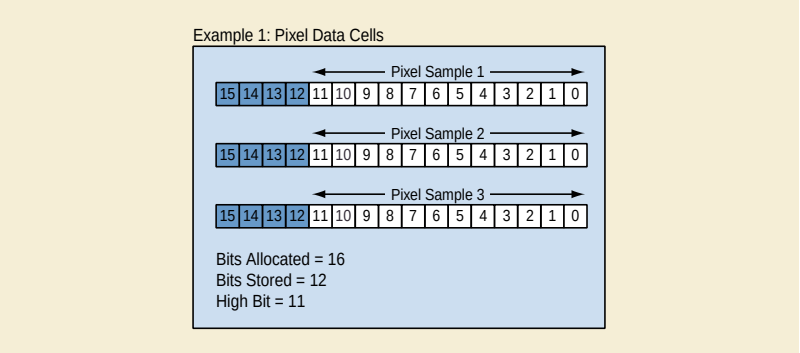

Figure D.2-1. Example 1 of Pixel and Overlay Data Cells

Figure D.2-2 Example 2 of Pixel and Overlay Data Cells has been retired. See PS3.3 2014c.

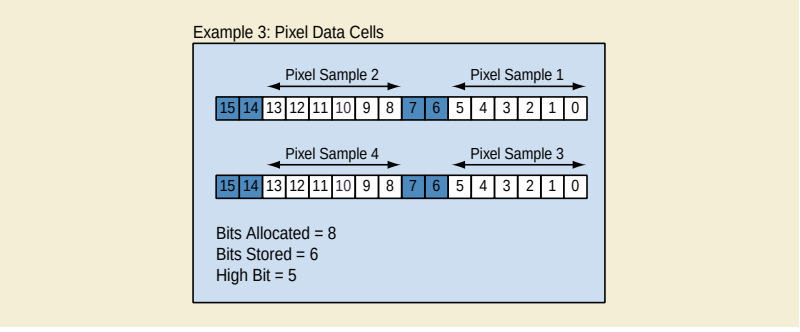

Figure D.2-3. Example 3 of Pixel and Overlay Data Cells

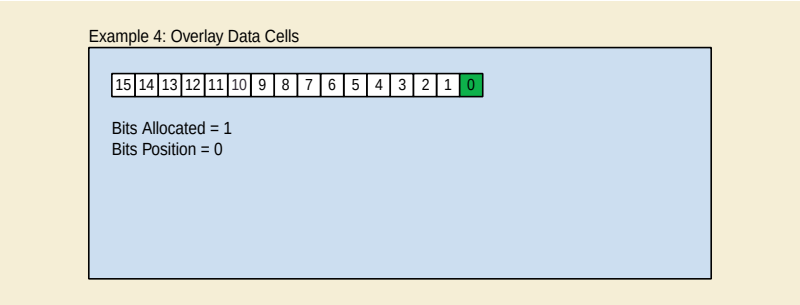

Figure D.2-4. Example 4 of Overlay Data Cells

Note

In this example, the Overlay Bits are numbered in the same manner that Pixel Cells are numbered in the other examples in this Annex. That is Overlay Bit 1 is the first bit of the Overlay Plane, encoded from left to right and top to bottom, a row at a time.

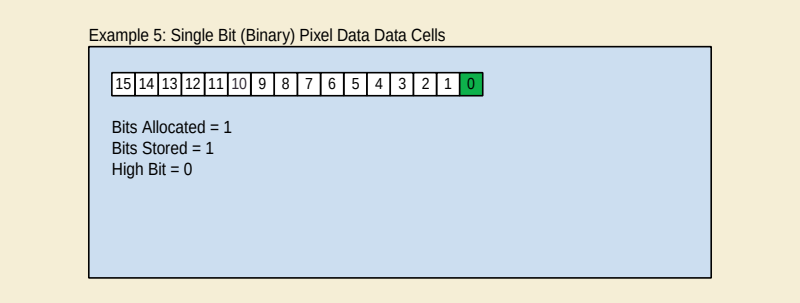

Figure D.2-5. Example 5 of Single Bit Pixel Data Cells (VR=OW)

D.3 Examples of Float and Double Float Pixel Data

Float Pixel Data having the Value Representation OF always has 32 bits allocated; the resulting byte streams pictured in Figure D.3-1 are as they would be transmitted across a network and/or stored on media.

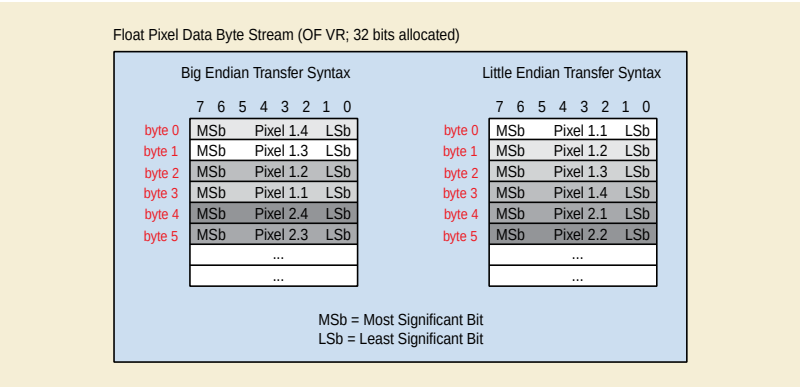

Figure D.3-1. Sample Float Pixel Data Byte Streams for VR = OF

Double Float Pixel Data having the Value Representation OD always has 64 bits allocated; the resulting byte streams pictured in Figure D.3-2 are as they would be transmitted across a network and/or stored on media.

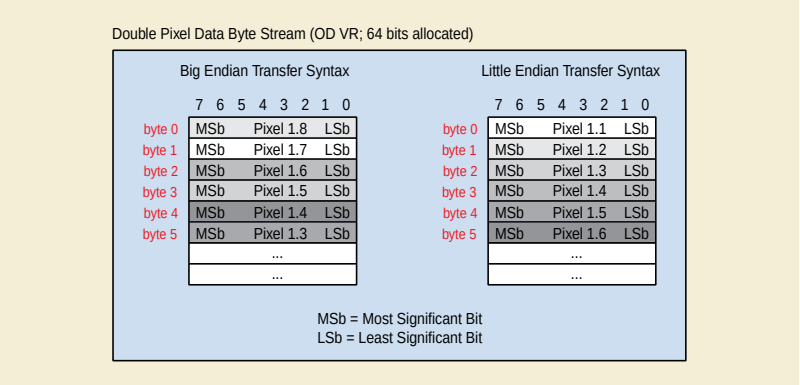

Figure D.3-2. Sample Float Pixel Data Byte Streams for VR = OD

E DICOM Default Character Repertoire (Normative)

The default repertoire for character strings in DICOM is the Basic G0 Set of the International Reference Version of ISO 646:1990 (ISO IR-6). In addition, the Control Characters LF, FF, CR, TAB and ESC are supported. These control characters are a subset of the C0 set defined in ISO 646:1990 and ISO 6429:1990.

The byte encoding of the default character repertoire is pictured in Table E-1. This table can be used to derive both ISO column/row byte values and hex values for encoded representations (see Section 6.1.1).

Table E-1. DICOM Default Character Repertoire Encoding

					b_8	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
					b_7	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
					b_6	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**1**
					b_5	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**
b_4	b_3	b_2	b_1			**00**	**01**	**02**	**03**	**04**	**05**	**06**	**07**
0	0	0	0	00				SP	0	@	P	`	p
0	0	0	1	01				!	1	A	Q	a	q
0	0	1	0	02				"	2	B	R	b	r
0	0	1	1	03				#	3	C	S	c	s
0	1	0	0	04				$	4	D	T	d	t
0	1	0	1	05				%	5	E	U	e	u
0	1	1	0	06				&	6	F	V	f	v
0	1	1	1	07				'	7	G	W	g	w
1	0	0	0	08				(	8	H	X	h	x
1	0	0	1	09		TAB		)	9	I	Y	i	y
1	0	1	0	10		LF		*	:	J	Z	j	z
1	0	1	1	11			ESC	+	;	K	[	k	{
1	1	0	0	12		FF		,	<	L	\	l	|
1	1	0	1	13		CR		-	=	M	]	m	}
1	1	1	0	14				.	>	N	^	n	~
1	1	1	1	15				/	?	O	_	o	

F Encapsulated Images As Part of A DICOM Message (Informative)

The following remarks apply generally to communicating an encoded image within a message structure according to the DICOM Standard:

a) In the course of including an encoded image in a DICOM message, the encoding is not changed. The encoded data stream is merely segmented and encapsulated according to the protocols of the DICOM Standard. After unpacking the DICOM message, the encoded data stream can be fully reconstructed at the receiving node.

b) The object definition of the DICOM Standard is always determining format and other choices that a specific encoding implementation may offer. The encoded image must be consistent with the definition of the object of which the encoded image is part. For example:

1) If the object is defined to contain 10-bit pixel data, it is assumed that the encoding process is one that accepts at least 10-bit data. Hence, there is no need for defining separate Transfer Syntaxes, e.g., for 8-bit or 12-bit implementations. Any 12-bit implementation is assumed to operate in an 8-bit process if the object is defined to contain 8-bit data.

2) If the image of an object is interleaved, the encoding process must reproduce the interleaving.

c) Specifications in the encoding file header must be consistent with the DICOM Message header, e.g., regarding the number of rows and columns.

d) The byte order specification of an encoded file is not altered in the course of encapsulating it in a DICOM message.

F.1 Encapsulated JPEG Encoded Images

The International Standards Organization (ISO/IEC JTC1/SC2/WG10) has prepared an International Standard, ISO~~/IS~~ 10918-1 (JPEG Part 1) and International Draft Standard ISO~~/IS~~ 10918-2 (JPEG Part 2), for the digital compression and coding of continuous-tone still images. This standard is collectively known as the JPEG Standard.

Part 1 of the JPEG Standard sets out requirements and implementation guidelines for the coded representation of compressed image data to be interchanged between applications. The processes and representations are intended to be generic in order to support the broad range of applications for color and grayscale still images for the purpose of communications and storage within computer systems. Part 2 of the JPEG Standard defines tests for determining whether implementations comply with the requirements of the various encoding and decoding processes specified in Part 1 of the JPEG Standard.

The JPEG Standard specifies lossy and lossless code processes. The lossy coding is based on the discrete cosine transform (DCT), permitting data compression with an adjustable compression ratio. The lossless coding employs differential pulse code modulation (DPCM).

The JPEG Standard permits a variety of coding processes for the coder and decoder. These processes differ in coding schemes for the quantified data and in sample precision. The coding processes are consecutively numbered as defined in the International Draft Standard ISO~~/IS~~ 10918-2 (JPEG Part 2), and are summarized in Table F.1-1. The simplest DCT-based coding process is referred to as Baseline Sequential with Huffman Coding for 8-bit Samples.

Table F.1-1. JPEG Modes of Image Coding

No.	Description	Lossy LY Lossless LL	Non-Hierarchical NH Hierarchical H	Sequential S Progressive P	Transform	Coding	Accepted Bits
1	Baseline	LY	NH	S	DCT	Huffman	8
2	Extended	LY	NH	S	DCT	Huffman	8
4	Extended	LY	NH	S	DCT	Huffman	12
14	Lossless	LL	NH	S	DPCM	Huffman	2-16

The different coding processes specified in the JPEG Standard are closely related. By extending the capability of an implementation, increasingly more 'lower level' processes can also be executed by the implementation. This is shown in Table F.1-2 for Huffman Coding.

Inclusion of a JPEG-coded image in a DICOM message is facilitated by the use of specific Transfer Syntaxes that are defined in Annex A. Independent of the JPEG coding processes, the same syntax applies. The only distinction for different processes in the syntax (apart from different SOF marker segments in the JPEG bit stream) is the UID value. Table F.1-5 lists the UID values in the Transfer Syntax for the various JPEG coding processes for reference.

Table F.1-2. Relationship Between the Lossy JPEG Huffman Coding Processes

Process	1	2	4
1	*	*	*
2		*	*
4			*

* Coding process of column can execute coding process of row

Table F.1-5. Identification of JPEG Coding Processes in DICOM

DICOM Transfer Syntax UID	JPEG process	JPEG description	capable of ~~performing~~decoding
1.2.840.10008.1.2.4.50	1	baseline	1
1.2.840.10008.1.2.4.51	2,4	extended	1,2,4 (see Note)
1.2.840.10008.1.2.4.57	14	lossless NH	14
1.2.840.10008.1.2.4.70	14 Selection Value 1	lossless NH, first-order prediction	

Note

Though the coding processes (2, 4) described in ISO 10918-1 are capable of decoding the other listed process (1), the bit stream uses different SOF marker segments. I.e., the baseline JPEG process 1 used with the 1.2.840.10008.1.2.4.50 Transfer Syntax uses the SOF0 marker, whereas the extended process 2 used with the 1.2.840.10008.1.2.4.51 Transfer Syntax uses the SOF1 marker. Accordingly, even though both bit streams encode 8 bit images using DCT and Huffman coding, the bit streams are not identical.

ISO 10918-2 describes compliance tests for decoders, and requires that implementations of specific extended processes (such as 2 and 4) be capable of decoding bit streams of related baseline processes (such as 1) (ISO 10918-2 Section 7.4 Compliance tests for DCT-based sequential mode decoding processes). The converse is not true for encoders however,

and the presence of SOF marker segments not defined by the specific process is not compliant (ISO 10918-2 Section 5.1.1 Non-hierarchical coding processes syntax compliance test Tables 1 and 2).

F.2 Encapsulated JPEG-LS Encoded Images

The International Standards Organization (ISO/IEC JTC1/SC2/WG10) has prepared an International Standard, ISO/IS-14495-1 (JPEG-LS Part 1), for the digital compression and coding of continuous-tone still images. This standard is known as the JPEG-LS Standard.

Part 1 of the JPEG-LS Standard sets out requirements and implementation guidelines for the coded representation of compressed image data to be interchanged between applications. The processes and representations are intended to be generic in order to support the broad range of applications for color and grayscale still images for the purpose of communications and storage within computer systems.

The JPEG-LS Standard specifies a single lossy (near-lossless) code process that can achieve lossless compression by constraining the absolute error value during encoding to zero. The lossless and lossy (near-lossless) coding is based on a predictive scheme with statistical modeling, in which differences between pixels and their surround are computed and their context modeled prior to coding, with a run-length escape mechanism. This scheme achieves consistently better compression in lossless mode than the lossless processes of JPEG defined in ISO 10918-1, with less complexity.

Though a different coding process from those specified in ISO 10918-1 is used, the syntax of the encoded bit stream is closely related.

A single JPEG-LS process is used for bit depths up to 16 bits.

Inclusion of a JPEG-LS coded image in a DICOM message is facilitated by the use of specific Transfer Syntaxes that are defined in Annex A.

F.3 Encapsulated JPEG 2000 Encoded Images

The International Standards Organization (ISO/IEC JTC1/SC2/WG10) has prepared an International Standard, ISO/IEC-15444 (JPEG 2000), for the digital compression and coding of continuous-tone still images. This standard is known as the JPEG 2000 Standard.

The JPEG 2000 Standard sets out requirements and implementation guidelines for the coded representation of compressed image data to be interchanged between applications. The processes and representations are intended to be generic in order to support the broad range of applications for color and grayscale still images for the purpose of communications and storage within computer systems.

Though a different coding process from those specified in ISO 10918-1 is used, the syntax of the encoded bit stream is closely related.

A single JPEG 2000 process is used for bit depths up to 16 bits.

Inclusion of a JPEG 2000 coded image in a DICOM message is facilitated by the use of specific Transfer Syntaxes that are defined in Annex A.

G Encapsulated RLE Compressed Images (Normative)

G.1 Summary

This annex describes how to apply RLE Compression to an image or an individual frame of a multi-frame image. This method can be used for any image, independent of the values of the data elements that describe the image (i.e., Photometric Interpretation (0028,0004) and Bits Stored (0028,0101)).

RLE Compression consists of the following steps:

1. The image is converted to a sequence of Composite Pixel Codes (see PS3.3).
2. The Composite Pixel Codes are used to generate a set of Byte Segments (see Section G.2).
3. Each Byte Segment is RLE compressed to produce a RLE Segment (see Section G.4).
4. The RLE Header is appended in front of the concatenated RLE Segments (see Section G.5).

G.2 Byte Segments

A Byte Segment is a series of bytes generated by decomposing the Composite Pixel Code (see PS3.3).

If the Composite Pixel Code is not an integral number of bytes in size, sufficient Most Significant zero bits are added to make it an integral byte size. This is known as the Padded Composite Pixel Code.

The first Segment is generated by stripping off the most significant byte of each Padded Composite Pixel Code and ordering these bytes sequentially. The second Segment is generated by repeating this process on the stripped Padded Composite Pixel Code continuing until the last Pixel Segment is generated by ordering the least significant byte of each Padded Component Pixel Code sequentially.

Note

If Photometric Interpretation (0028, 0004) equals RGB and Bits Stored equals 8, then three Segments are generated. The first one holds all the Red values, the second all the Green values, and the third all the Blue values.

G.3 The RLE Algorithm

The RLE algorithm described in this section is used to compress Byte Segments into RLE Segments. There is a one-to-one correspondence between Byte Segments and RLE Segments. Each RLE segment must be an even number of bytes or padded at its end with zero to make it even.

G.3.1 The RLE Encoder

A sequence of identical bytes (Replicate Run) is encoded as a two-byte code:

< -count + 1 > <byte value>, where

count = the number of bytes in the run, and

2 <= count <= 128

and a non-repetitive sequence of bytes (Literal Run) is encoded as:

< count - 1 > <Literal sequence of bytes>, where

count = number of bytes in the sequence, and

1 <= count <= 128.

The value of -128 may not be used to prefix a byte value.

Note

It is common to encode a 2-byte repeat run as a Replicate Run except when preceded and followed by a Literal Run, in which case it's best to merge the three runs into a Literal Run.

Three-byte repeats shall be encoded as Replicate Runs. Each row of the image shall be encoded separately and not cross a row boundary.

G.3.2 The RLE Decoder

Pseudo code for the RLE decoder is shown below:

```
Loop until the number of output bytes equals the uncompressed segment size
  Read the next source byte into n
  If n> =0 and n <= 127 then
    output the next n+1 bytes literally
  Elseif n <= - 1 and n >= -127 then
    output the next byte -n+1 times
  Elseif n = - 128 then
    output nothing
  Endif
Endloop
```

G.4 Organization of RLE Compressed Frame

The RLE Segments are ordered as described in Section G.2. They are preceded by the RLE Header, which contains offsets to the start of each RLE Segment. The RLE Header is described in G.5.

The first RLE Segment immediately follows the RLE Header and the remaining RLE Segments immediately follow each other. This is illustrated in the diagram below.

Table G.4-1. Organization of RLE Compressed Frame

Header
RLE Segment 1
RLE Segment 2
. . .
. . .
RLE Segment n

G.5 RLE Header Format

The RLE Header contains the number of RLE Segments for the image, and the starting offset of each of the RLE Segments. Each of these numbers is represented by a UL (unsigned long) value stored in little-endian format. The RLE Header is 16 long words in length. This allows it to describe a compressed image with up to 15 RLE Segments. All unused segments offsets shall be set to zero.

Each of the starting locations for the RLE Segments are byte offsets relative to the beginning of the RLE Header. Since the RLE Header is 16 unsigned longs or 64 bytes, the offset of RLE Segment One is 64.

The following diagram illustrates the ordering of the offsets within the RLE Header.

Table G.5-1. Ordering of the Offsets Within the RLE Header

number of RLE Segments
offset of RLE Segment 1 = 64
offset of RLE Segment 2
. . .
. . .
offset of RLE Segment n
0
0
0

G.6 Example of Elements For An Encoded YCbCr RLE Three-frame Image with Basic Offset Table

Table G.6-1 is an example of encoding of RLE Compressed Frames (described in Section G.4) with the basic offset table. Table G.6-2 is an example of Item Value data for one frame.

Table G.6-1. Example of Elements for an Encoded YCbCr RLE Three-Frame Image with Basic Offset Table

Pixel Data Element Tag	Value Representation		Data Element Length	Data Element					
				Basic Offset Table with Item Value			First Fragment (Frame 1) of Pixel Data		
				Item Tag	Item Length	Item Value	Item Tag	Item Length	Item Value
(7FE0,0010) with VR of OB	OB	0000H	FFFF FFFFH undefined length	(FFFE,E000)	0000 000CH	0000 0000H 0000 02D0H 0000 0642H	(FFFE,E000)	0000 02C8H	RLE Compressed Frame
4 bytes	2 bytes	2 bytes	4 bytes	4 bytes	4 bytes	000CH bytes	4 bytes	4 bytes	02C8H bytes

Table G.6-1b. Example of Elements for an Encoded YCbCr RLE Three-Frame Image with Basic Offset Table (continued)

Data Element Continued							
Second Fragment (Frame 2) of Pixel Data			Third Fragment (Frame 3) of Pixel Data			Sequence Delimiter Item	
Item Tag	Item Length	Item Value	Item Tag	Item Length	Item Value	Sequence Delimiter Tag	Item Length
(FFFE,E000)	0000 036AH	RLE Compressed Frame	(FFFE,E000)	0000 0BC8H	RLE Compressed Frame	(FFFE,E0DD)	0000 0000H
4 bytes	2 bytes	036AH bytes	4 bytes	4 bytes	0BC8H bytes	4 bytes	4 bytes

Table G.6-2. Example of Encoded YCbCr RLE Compressed Frame Item Value

Offset	Data	Description of Data	
0000 0000H	0000 0003H	number of RLE Segments	(Header)
	0000 0040H	location of RLE Segment 1 (Y component)	
	0000 0140H	location of RLE Segment 2 (C_B component)	
	0000 01C0H	location of RLE Segment 3 (C_R component)	
	0000 0000H		
	…..		
	…..		
	0000 0000H		
0000 0040H	Y - RLE Segment Data		(DATA)
0000 0140H	C_B - RLE Segment Data		(DATA)
0000 01C0H	C_R - RLE Segment Data		(DATA)

H Character Sets and Person Name Value Representation in the Japanese Language (Informative)

H.1 Character Sets for the Japanese Language

The purpose of this section is to explain the character sets for the Japanese language.

H.1.1 JIS X 0201

JIS X 0201 has the following code elements:

- ISO-IR 13 Japanese katakana (phonetic) characters (94 characters)
- ISO-IR 14 Japanese romaji (alphanumeric) characters (94 characters)

JIS X 0201 defines a 7-bit romaji code table (ISO-IR 14), a 7-bit katakana code table (ISO-IR 13), and the combination of romaji and katakana as an 8-bit code table (ISO-IR 14 as G0, ISO-IR 13 as G1).

The 7-bit romaji (ISO-IR 14) is identical to ASCII (ISO-IR 6) except that bit combination 05/12 represents a yen sign and bit combination 07/14 represents an over-line. These are national Graphic Character allocations in ISO 646.

The Escape Sequence for ISO/IEC 2022 is shown for reference in Table H.1-1 (for the Defined Terms, see PS3.3).

Table H.1-1. ISO/IEC 2022 Escape Sequence for ISO-IR 13 and ISO-IR 14

	ISO-IR 14	ISO-IR 13
G0 set	**ESC 02/08 04/10**	ESC 02/08 04/09
G1 set	ESC 02/09 04/10	**ESC 02/09 04/09**

Note

1. Table H.1-1 does not include the G2 and G3 sets that are not used in DICOM. See Section 6.1.2.5.1.
2. Defined Terms ISO_IR 13 and ISO 2022 IR 13 for the value of the Specific Character Set (0008,0005) support the G0 set for ISO-IR 14 and G1 set for ISO-IR 13. See PS3.3.

H.1.2 JIS X 0208

JIS X 0208 has the following code element:

- ISO-IR 87: Japanese kanji (ideographic), hiragana (phonetic), and katakana (phonetic) characters (94^2 characters, 2-byte).

H.1.3 JIS X 0212

JIS X 0212 has the following code element:

- ISO-IR 159: Supplementary Japanese kanji (ideographic) characters (94^2 characters, 2-byte)

The Escape Sequence for ISO/IEC 2022 is shown for reference in Table H.1-2 (for the Defined Terms, see PS3.3)

Table H.1-2. ISO/IEC 2022 Escape Sequence for ISO-IR 87 and ISO-IR 159

	ISO-IR 87	ISO-IR 159
G0 set	**ESC 02/04 04/02**	**ESC 02/04 02/08 04/04**
G1 set	ESC 02/04 02/09 04/02	ESC 02/04 02/09 04/04

Note

1. The Escape Sequence for the designation function G0-DESIGNATE 94-SET, has first I byte 02/04 and second I byte 02/08. There is an exception to this: The second I byte 02/08 is omitted if the Final Byte is 04/00, 04/01 or 04/02. See ISO/IEC 2022.
2. The table does not include the G2 and G3 sets that are not used in DICOM. See Section 6.1.2.5.1.
3. Defined Term ISO 2022 IR 87 for the value of the Specific Character Set (0008,0005) supports the G0 set for ISO-IR 87, and Defined Term ISO 2022 IR 159 supports the G0 set for ISO-IR 159. See PS3.3.

H.2 Internet Practice

DICOM has adopted an encoding method for Japanese character sets that is similar to the method for Internet practice.

The major protocols for the Internet such as SMTP, NNTP and HTTP adopt the encoding method for Japanese characters called "ISO-2022-JP" as described in RFC 1468, Japanese Character Encoding for Internet Messages. There is also a less commonly used Internet practice called "ISO-2022-JP-2" described in RFC 1554, which supports a larger repertoire of character sets and additionally requires an escape to a single-byte character set before encoding a SPACE (unlike DICOM and ISO-2022-JP).

The character sets supported for the Japanese language in DICOM and Internet practice are shown in Table H.2-1.

Table H.2-1. Character Sets for the Japanese language in DICOM and Internet practice

DICOM	ISO-2022-JP	ISO-2022-JP-2
ASCII (ISO-IR 6)	ASCII (ISO-IR 6)	ASCII (ISO-IR 6)
JIS X 0201 Katakana (ISO-IR 13)	JIS-X 0201 Romaji (ISO-IR 14)	ISO8859-1 (ISO-IR 100)
JIS X 0201 Romaji (ISO-IR 14)	JIS X 0208-1978 Kanji (ISO-IR 42)	ISO8859-7 Greek (ISO-IR 126)
JIS X 0208 Kanji (ISO-IR 87)	JIS-X 0208-1983 Kanji (ISO-IR 87)	JIS X 0201 Romaji (ISO-IR 14)
JIS X 0212 Kanji (ISO-IR 159)		JIS X 0208-1978 Kanji (ISO-IR 42)
		JIS X 0208-1983 Kanji (ISO-IR 87)
		JIS X 0212-1990 Kanji (ISO-IR 159)
		GB2312-1980 (ISO-IR 58)
		KSC5601-1987 (ISO-IR 149)

The Control Characters supported in DICOM and Internet practice are shown in Table H.2-2.

Table H.2-2. Control Characters Supported in DICOM and Internet practice

DICOM	ISO-2022-JP and ISO-2022-JP-2
LF (00/10)	LF (00/10)
FF (00/12)	CR (00/13)
CR (00/13)	SO (00/14)
ESC (01/11)	SI (00/15)
	ESC (01/11)

H.3 Example of Person Name Value Representation in the Japanese Language

Character strings representing person names are encoded using a convention for PN value representations based on component groups with 5 components.

For languages that use ideographic characters, it is sometimes necessary to write names both in ideographic characters and in phonetic characters. Ideographic characters may be required for official purposes, while phonetic characters may be needed for pronunciation and data processing purposes.

For the purpose of writing names in ideographic characters and in phonetic characters, up to 3 component groups may be used. The delimiter of the component group shall be the equals character "=" (3DH). The three component groups in their order of occurrence are: an alphabetic representation, an ideographic representation, and a phonetic representation.

H.3.1 Value 1 of Attribute Specific Character Set (0008,0005) is Not Present.

Example H.3-1. Value 1 of Attribute Specific Character Set (0008,0005) is Not Present

In this case, ISO-IR 6 is used by default in Specific Character Set:

(0008,0005) \ISO 2022 IR 87

Character String:

Yamada^Tarou=山田^太郎=やまだ^たろう

Yamada^Tarou= ESC 02/04 04/02 山田 ESC 02/08 04/02 ^ ESC 02/04 04/02 太郎 ESC 02/08 04/02 = ESC 02/04 04/02 やまだ ESC 02/08 04/02 ^ ESC 02/04 04/02 たろう ESC 02/08 04/02

Encoded representation:

05/09 06/01 06/13 06/01 06/04 06/01 5/14 05/04 06/01 07/02 06/15 07/05 03/13 01/11 02/04 04/02 03/11 03/03 04/05 04/04 01/11 02/08 04/02 05/14 01/11 02/04 04/02 04/02 04/00 04/15 03/10 01/11 02/08 04/02 03/13 01/11 02/04 04/02 02/04 06/04 02/04 05/14 02/04 04/00 01/11 02/08 04/02 05/14 01/11 02/04 04/02 02/04 03/15 02/04 06/13 02/04 02/06 01/11 02/08 04/02

An example of what might be displayed or printed by an ASCII based machine that displays or prints the Control Character ESC (01/11) using \033:

Yamada^Tarou=\033$B;3ED\033(B^\033$BB@O:\033(B=\033Bd$^$@\033(B^\033B?m&\033(B

Table H.3-1. Character Sets and Escape Sequences Used in Example 1

Character Set Description	Component Group	Value of (0008,0005) Defined Term	ISO Registration Number	Standard for Code Extension	ESC Sequence	Code Element	Character Set: Purpose of Use
Japanese	First: Single-byte	Value 1: none	ISO-IR 6			GL	ISO 646:
	Second: Ideographic	Value 2: ISO 2022 IR 87	ISO-IR 87	ISO 2022	ESC 02/04 04/02	GL	JIS X 0208: Japanese kanji, hiragana, katakana
		Value 1: none	ISO-IR 6	ISO 2022	ESC 02/08 04/02	GL	ISO 646: for delimiters
	Third: Phonetic	Value 2: ISO 2022 IR 87	ISO-IR 87	ISO 2022	ESC 02/04 04/02	GL	JIS X 0208: Japanese hiragana, and katakana
		Value 1: none	ISO-IR 6	ISO 2022	ESC 02/08 04/02	GL	ISO 646: for delimiters

H.3.2 Value 1 of Attribute Specific Character Set (0008,0005) is ISO 2022 IR 13.

Example H.3-2. Value 1 of Attribute Specific Character Set (0008,0005) is ISO 2022 IR 13

Specific Character Set:

(0008,0005) ISO 2022 IR 13\ISO 2022 IR 87

Character String:

ヤマダ^タロウ=山田^太郎=やまだ^たろう

ヤマダ^タロウ= ESC 02/04 04/02 山田 ESC 02/08 04/10 ^ ESC 02/04 04/02 太郎 ESC 02/08 04/10 = ESC 02/04 04/02 やまだ ESC 02/08 04/10 ^ ESC 02/04 04/02 たろう ESC 02/08 04/10

Encoded representation:

13/04 12/15 12/00 13/14 05/14 12/00 13/11 11/03 03/13 01/11 02/04 04/02 03/11 03/03 04/05 04/04 01/11 02/08 04/10 05/14 01/11 02/04 04/02 04/02 04/00 04/15 03/10 01/11 02/08 04/10 03/13 01/11 02/04 04/02 02/04 06/04 02/04 05/14 02/04 04/00 01/11 02/08 04/10 05/14 01/11 02/04 04/02 02/04 03/15 02/04 06/13 02/04 02/06 01/11 02/08 04/10

An example of what might be displayed or printed by an ASCII based machine that displays or prints the Control Character ESC (01/11) using \033:

\324\3I7\300\336^\300\333\263=\033$B;3ED\033(J^\033$BB@O:\033(J=\033Bd$^$@\033(J^\033B?m&\033(J

Table H.3-2. Character Sets and Escape Sequences Used in Example 2

Character Set Description	Component Group	Value of (0008,0005) Defined Term	ISO Registration Number	Standard for Code Extension	ESC Sequence	Code Element	Character Set: Purpose of Use
Japanese	First: Single-byte	Value 1: ISO 2022 IR 13	ISO-IR 13	ISO 2022	ESC 02/09 04/09	GR	JIS X 0201: Japanese katakana
			ISO-IR 14	ISO 2022	ESC 02/08 04/10	GL	JIS X 0201: Japanese romaji for delimiters
	Second: Ideographic	Value 2: ISO 2022 IR 87	ISO-IR 87	ISO 2022	ESC 02/04 04/02	GL	JIS X 0208: Japanese kanji, hiragana, katakana
		Value 1: ISO 2022 IR 13	ISO-IR 14	ISO 2022	ESC 02/08 04/10	GL	JIS X 0201: Japanese romaji for delimiters
	Third: Phonetic	Value 2: ISO 2022 IR 87	ISO-IR 87	ISO 2022	ESC 02/04 04/02	GL	JIS X 0208: Japanese hiragana, and katakana
		Value 1: ISO 2022 IR 13	ISO-IR 14	ISO 2022	ESC 02/08 04/10	GL	JIS X 0201: Japanese romaji for delimiters

I Character Sets and Person Name Value Representation in the Korean Language (Informative)

I.1 Character Sets For The Korean Language in DICOM

KS X 1001 (registered as ISO-IR 149) is used as a Korean character set in DICOM. This character set is the one most broadly used for the representation of Korean characters. It can be encoded by ISO 2022 code extension techniques, and is registered in ISO 2375.

The Escape Sequence is shown for reference in Table I.1-1 (see PS3.3)

Table I.1-1. ISO/IEC 2022 Escape Sequence for ISO-IR 149

	ISO-IR 149
G0 set	ESC 02/04 02/08 04/03
G1 set	**ESC 02/04 02/09 04/03**

Note

1. ISO-IR 149 is only used as a G1 set in DICOM.
2. The Korean character set (ISO IR 149) is invoked to the G1 area. This is different from the Japanese multi-byte character sets (ISO 2022 IR 87 and ISO 2022 IR 159), which use the G0 code area. Japan's choice of G0 is due to the adoption of an encoding method based on "ISO-2022-JP". ISO-2022-JP, the most familiar encoding method in Japan, and uses only the G0 code area. In Korea, most operating systems adopt an encoding method that invokes the Hangul character set (KS X 1001) in the G1 code area. So, the difference between code areas of Korean and Japanese character originates in convention, not a technical problem. Invocation of multi-byte character sets to the G1 area does not change the current DICOM normative requirements.

I.2 Example of Person Name Value Representation in the Korean Language

Example I.2-1. Example of Person Name Value Representation in the Korean Language

Person names in the Korean language may be written in Hangul (phonetic characters), Hanja (ideographic characters), or Latin (alphabetic characters). The three component groups should be written in the order of alphabetic, ideographic, and phonetic (see Table 6.2-1).

Specific Character Set:

(0008,0005) \ISO 2022 IR 149

Character String:

Hong^Gildong=洪^吉洞=홍^길동

Hong^Gildong= ESC 02/04 02/09 04/03 洪 ^ ESC 02/04 02/09 04/03 吉洞 = ESC 02/04 02/09 04/03 홍 ^ ESC 02/04 02/09 04/03 길동

Encoded representation:

04/08 06/15 06/14 06/07 05/14 04/07 06/09 06/12 06/04 06/15 06/14 06/07 03/13 01/11 02/04 02/09 04/03 15/11 15/03 05/14 01/11 02/04 02/09 04/03 13/01 12/14 13/04 13/07 03/13 01/11 02/04 02/09 04/03 12/08 10/11 05/14 01/11 02/04 02/09 04/03 11/01 14/06 11/05 11/15

An example of what might be displayed or printed by an ASCII based machine that displays or prints the Control Character ESC (01/11) using \033:

Hong^Gildong=\033$) C\373\363^\033$)C\321\316\324\327=\033$)C\310\253^\033$)C\261\346\265\277

Note

1. The multi-byte character set (ISO-IR 149) and single-byte character set (ISO 646) can be used intermixed without any explicit escape sequence after the initial escape sequence. Once ISO 646 has been designated to the GL area and ISO-IR 149 to the GR area, each character set has different code area, thus can be used intermixed. The decoder will check the most significant bit of a character to know whether it is a two byte character in the GR area (high bit one) or a one byte character in the GL area (high bit zero).

2. In the above example of person name representation, explicit escape sequences precede each Hangul and Hanja string. These escape sequences are to meet the requirements of the code extension technique that specifies a switch to the default character repertoire before delimiters. In the previous example, it is assumed that the default character repertoire (ISO-646) is invoked to G0 code area and no character set to G1 area after delimiters ("^" and "=" signs). See Section 6.1.2.5.3.

I.3 Example of Long Text Value Representation in the Korean Language Without Explicit Escape Sequences Between Character Sets

Example I.3-1. Example of Long Text Value Representation in the Korean Language Without Explicit Escape Sequences Between Character Sets

Hangul (ISO IR 149) and ASCII (ISO 646) character sets can be used intermingled without explicit escape sequences between them. The Hangul character set ISO IR 149 is invoked to the G1 area, so this invocation doesn't affect the G0 area to which the ASCII character set has been invoked. The following is an example of a Long Text value representation that includes ASCII and Hangul character set.

Specific Character Set:

(0008,0005) \ISO 2022 IR 149

Character String:

The first line includes 한글.

The second line includes 한글, too.

The third line

Encoded String:

ESC 02/04 02/09 04/03 The first line includes 한글.

ESC 02/04 02/09 04/03 The second line includes 한글, too.

The third line

Once having invoked the ISO IR 149 character set to G1 area by the escape sequence in the head of line, one can use Hangul and ASCII intermixed in that line.

Table I.3-1. Character Sets and Escape Sequences Used in the Examples

Character Set Description	Component Group	Value of (0008,0005) Defined Term	ISO Registration Number	Standard for Code Extension	ESC Sequence	Code Element	Character Set: Purpose of Use
Korean	First: Single-byte	Value 1: none	ISO-IR 6			GL	ISO 646:
	Second: Ideographic	Value 1: none	ISO-IR 6			GL	ISO 646: For delimiters
		Value 2: ISO 2022 IR 149	ISO-IR 149	ISO 2022	ESC 02/04 02/09 04/03	GR	KS X 1001: Hangul and Hanja
	Third: Phonetic	Value 1: none	ISO-IR 6			GL	ISO 646: For delimiters
		Value 2: ISO 2022 IR 149	ISO-IR 149	ISO 2022	ESC 02/04 02/09 04/03	GR	KS X 1001: Hangul and Hanja

J Character Sets and Person Name Value ~~Representationusing~~Representation using Unicode UTF-8, GB18030 and GBK (Informative)

The Unicode UTF-8 character set and the GB18030 character set may be used for multiple languages. Some of these languages may also be encoded using other character sets that are defined elsewhere in the DICOM standard. As Unicode UTF-8 and GB18030 encodings do not allow ISO 2022 character set replacement, these must be used for all strings in a single SOP Instance. This may have implications for the character set selected for the encoding of the SOP Instance.

Since the GBK character set is fully code point compatible to the larger character set of GB 18030, and the specific examples of GB 18030 encoding this in Annex (J.3 and J.4) include only the Chinese characters falling in the common coding area between the two standards, these examples are used to demonstrate the person name and text encoding in both standards. Examples specific to GBK are not necessary.

J.1 Example of Person Name Value Representation in the Chinese Language Using Unicode

Example J.1-1. Example of Person Name Value Representation in the Chinese Language Using Unicode

Person names in the Chinese language may be written in Hanzi (ideographic characters), and/or Latin (alphabetic characters). The Latin representation may be derived using pinyin or another Romanization method, or may be a chosen "westernized" name. The two component groups should be written in the order of alphabetic, then ideographic; the phonetic component group is typically not used (see Table 6.2-1). In this example the traditional script is used.

Note

1. Some healthcare information systems may encode a "westernized" name with other patient aliases in a separate attribute, e.g., Other Patient Names (0010,1091).
2. Some environments using Chinese language may use the third name component, e.g., for the Yi or Mongolian script, with or without the first name component. This would be similar to the Japanese and Korean name component usage.

In the example below, the Specific Character Set attribute (0008,0005) would contain:

(0008,0005) ISO_IR 192

Text string:

Wang^XiaoDong=王^小東=

Character encoded representation is:

0x57 0x61 0x6e 0x67 0x5e 0x58 0x69 0x61 0x6f 0x44 0x6f 0x6e 0x67 0x3d 0xe7 0x8e 0x8b 0x5e 0xe5 0xb0 0x8f 0xe6 0x9d 0xb1 0x3d

Note

The underlined bytes correspond to the Unicode code points for the Chinese characters:

王 (U+738B)

小 (U+5C0F)

東 (U+6771)

and the corresponding UTF-8 encodings are:

UTF-8 (U+738b) = 0xe7 0x8e 0x8b

UTF-8 (U+5c0f U+6771) = 0xe5 0xb0 0x8f 0xe6 0x9d 0xb1

J.2 Example of Long Text Value Representation in the Chinese Language Using Unicode

Example J.2-1. Example of Long Text Value Representation in the Chinese Language Using Unicode

The following is an example of a Long Text value representation that includes ASCII and ISO 10646 character set.

Specific Character Set:

(0008,0005) ISO_IR 192

Text string:

The first line includes 中文.

The second line includes 中文, too.

The third line.

Character encoded representation is:

0x54 0x68 0x65 0x20 0x66 0x69 0x72 0x73 0x74 0x20 0x6c 0x69 0x6e 0x65 0x20 0x69 0x6e 0x63 0x6c 0x75 0x64 0x65 0x73 0xe4 0xb8 0xad 0xe6 0x96 0x87 0x2e 0x0d 0x0a 0x54 0x68 0x65 0x20 0x73 0x65 0x63 0x6f 0x6e 0x64 0x20 0x6c 0x69 0x6e 0x65 0x20 0x69 0x6e 0x63 0x6c 0x75 0x64 0x65 0x73 0xe4 0xb8 0xad 0xe6 0x96 0x87 0x2c 0x20 0x74 0x6f 0x6f 0x2e 0x0d 0x0a 0x54 0x68 0x65 0x20 0x74 0x68 0x69 0x72 0x64 0x20 0x6c 0x69 0x6e 0x65 0x2e 0x0d 0x0a

Note

The underlined byte codes correspond to the Unicode code points for the Chinese characters:

中 (U+4E2D) 0xe4 0xb8 0xad

文 (U+6587) 0xe6 0x96 0x87

J.3 Example of Person Name Value Representation in the Chinese Language Using GB18030

Example J.3-1. Example of Person Name Value Representation in the Chinese Language Using GB18030

Person names in the Chinese language may be written in Hanzi (ideographic characters), and/or Latin (alphabetic characters). The Latin representation may be derived using pinyin or another Romanization method, or may be a chosen "westernized" name. The two component groups should be written in the order of alphabetic, then ideographic; the phonetic component group is typically not used (see Table 6.2-1). In this example the simplified script is used.

Note

See notes to Section J.1.

In the example below, the Specific Character Set attribute (0008,0005) would contain:

(0008,0005) GB18030

Text string:

Wang^XiaoDong=王^小东=

Character encoded representation is:

0x57 0x61 0x6e 0x67 0x5e 0x58 0x69 0x61 0x6f 0x44 0x6f 0x6e 0x67 0x3d 0xcd 0xf5 0x5e 0xd0 0xa1 0xb6 0xab 0x3d

Note

The GB18030 encodings for the chinese characters used here are:

王 (CDF5 in GB18030)

小 (D0A1 in GB18030)

东 (B6AB in GB18030)

J.4 Example of Long Text Value Representation in the Chinese Language Using GB18030

Example J.4-1. Example of Long Text Value Representation in the Chinese Language Using GB18030

The following is an example of a Long Text value representation that includes ASCII and GB18030 character set.

Specific Character Set:

(0008,0005) GB18030

Text string:

The first line includes 中文.

The second line includes 中文, too.

The third line.

Character encoded representation is:

0x54 0x68 0x65 0x20 0x66 0x69 0x72 0x73 0x74 0x20 0x6c 0x69 0x6e 0x65 0x20 0x69 0x6e 0x63 0x6c 0x75 0x64 0x65 0x73 0xd6 0xd0 0xce 0xc4 0x2e 0x0d 0x0a 0x54 0x68 0x65 0x20 0x73 0x65 0x63 0x6f 0x6e 0x64 0x20 0x6c 0x69 0x6e 0x65 0x20 0x69 0x6e 0x63 0x6c 0x75 0x64 0x65 0x73 0xd6 0xd0 0xce 0xc4 0x2c 0x20 0x74 0x6f 0x6f 0x2e 0x0d 0x0a 0x54 0x68 0x65 0x20 0x74 0x68 0x69 0x72 0x64 0x20 0x6c 0x69 0x6e 0x65 0x2e 0x0d 0x0a

Note

The underlined byte codes correspond to the GB18030 encodings for the Chinese characters used:

中 (D6D0 in GB18030)

文 (CEC4 in GB18030)

J.5 Person Name Value Representation in Other Languages Using Unicode

Person names in many languages may be written in a local (non-Latin) script, as well as in a transliteration to a Latin script (Romanization). Healthcare information systems in those environments may support one or both name formats. Local scripts may be encoded using Unicode in UTF-8.

For the purpose of exchange in DICOM, there are three typical uses of name component groups using Unicode in UTF-8:

1. Names in a Latin script may be encoded in the first (alphabetic) component group, and names in a local script (alphabet, abugida, or syllabary) in the third (phonetic) component group (see Table 6.2-1). The second (ideographic) component group is null. This is the preferred use for cross-enterprise or international communication.

2. Where the local script historically has a single byte character set defined for Specific Character Set (0008,0005), i.e., Cyrillic, Arabic, Greek, Hebrew, Thai, and the various versions of Latin, only the first name component group might be used. Encoding may be in Unicode in UTF-8, as described in this Annex, as an equivalent for use of that defined single byte character set in the first name component group (see note 1).

3. Names in the local script may be encoded in the first component group, and names in a Latin script in the third component group, both encoded in Unicode in UTF-8.

   Note

   1. A previous edition of DICOM required the first name component group to use a single byte character set (see PS3.5-2008). Unicode in UTF-8 may now be used in that component group simply as a matter of a different character set encoding, but with the same application use of that component group.

   2. Healthcare information systems will use specific scripts in one, two, or three of the Person Name component groups in accordance with local policy. Conformant DICOM Application Entities that receive name attributes must accept multiple name component groups. An Application Entity that is configurable to allow the use of local script for names in either the first or the third component group, and a transliteration script in the other, would support all these typical representations.

   3. The transliteration (from a local script) may be a non-Latin script, e.g., Cyrillic. The same principles apply, and the Cyrillized name might be encoded in the first component group and the local script (which may in fact be a Latin-derived script) in the third component group.

K Character Sets and Person Name Value Representation in the Chinese Language with Code Extensions (Informative)

K.1 Character Sets for the Chinese Language in DICOM

GB 2312 (registered as ISO-IR 58) is used as a Chinese character set in DICOM. This character set is the one most broadly used for the representation of Chinese characters. It can be encoded by ISO 2022 code extension techniques.

The Escape Sequence is shown for reference in Table K.1-1 (see PS3.3)

Table K.1-1. ISO/IEC 2022 Escape Sequence for ISO-IR 58

	ISO-IR 58
G0 set(ASCII)	ESC 02/08 04/02
G1 set	ESC 02/04 02/09 04/01

K.2 Example of Person Name Value Representation in the Chinese Language

Example K.2-1. Example of Person Name Value Representation in the Chinese Language

Person names in the Chinese language may be written in Pinyin (phonetic characters), Hanzi (ideographic characters), or English Name (alphabetic characters). The three component groups should be written in the order of ~~phonetic~~alphabetic (English name), ideographic, and ~~alphabetic (English name)~~phonetic.

Specific Character Set:

(0008,0005) \ISO 2022 IR 58

Character String:

Zhang^XiaoDong=张^小东=

Encoded String:

Zhang^XiaoDong= ESC 02/04 02/09 04/01 张~~小东~~ ^ESC 02/~~08 04/02~~04 02/09 04/01小东 =

Character encoded representation (GB2312) is:

0x5A 0x68 0x61 0x6E 0x67 0x5E 0x58 0x69 0x61 0x6F 0x44 0x6F 0x6E 0x67 0x3D **0x1B 0x24 0x29 0x41** 0xD5 0xC5-[[0xD0 0xA1 0xB6 0xAB]]- -[[ ]]-+[[0x5E]]+ **0x1B -[[0x28 0x42]]-+[[0x24 0x29 0x41]]+** 0xD0 0xA1 0xB6 0xAB 0x3D 0x20

Note

~~The underlined byte codes correspond to double byte characters, the bold byte codes to escape sequences.~~

1. The underlined byte codes correspond to double byte characters, the bold byte codes to escape sequences.
2. The multi-byte character set (ISO-IR 58) and single-byte character set (ISO 646) can be used intermixed without any explicit escape sequence after the initial escape sequence, up to the next delimiter (^ or =) or the end of the value field. Once ISO 646 has been designated to G0 and ISO-IR 58 to G1, each character set has a different code area, thus can be used intermixed. The decoder will check the most significant bit of a character to know whether it is a two byte

character in the G1 area (high bit one) or a one byte character in the G0 area (high bit zero). There does not need to be an explicit escape to invoke ISO 646 into G0 at the end of the string prior to a delimiter (^ or =) or the end of the value field. However, there does need to be a new invocation of ISO-IR 58 in each name component in which it is used.

K.3 Example of Long Text Value Representation in the Chinese Language with ~~Explicit Escape Sequences Between GB2312 G0 and GB2312~~GB2312 G1

Example K.3-1. Example of Long Text Value Representation in the Chinese Language with ~~Explicit Escape Sequences Between GB2312 G0 and GB2312~~GB2312 G1

Chinese (ISO 2022 IR 58) and ASCII (ISO 646) character sets can be used intermingled ~~with~~without explicit escape sequences between them. The Chinese character set ISO IR 58 is invoked to the G1 area, and the ASCII character set is invoked to the G0 area. The following is an example of a Long Text value representation that includes ASCII and Chinese character set. Every line ~~must start in ASCII, end in ASCII~~is presumed to start in the default character set and requires an explicit invocation of GB 2312 into G1, but does not require re-invocation of the default ASCII character set into G0.

Specific Character Set:

(0008,0005) \ISO 2022 IR 58

Character String (with CR LF after each line):

1) 第一行文字。

2) 第二行文字。

3) 第三行文字。

Encoded String:

1) ESC 02/04 02/09 04/01 第一行文字。~~ESC 02/08 04/02~~

2) ESC 02/04 02/09 04/01 第二行文字。~~ESC 02/08 04/02~~

3) ESC 02/04 02/09 04/01 第三行文字。~~ESC 02/08 04/02~~

Character encoded representation (GB2312):

0x31 0x2e **0x1B 0x24 0x29 0x41** 0xB5 0xDA 0xD2 0xBB 0xD0 0xD0 0xCE 0xC4 0xD7 0xD6 0xA1 0xA3 ~~**0x1B 0x28 0x42**~~ 0x0D 0x0A

0x32 0x2e **0x1B 0x24 0x29 0x41** 0xB5 0xDA 0xB6 0xFE 0xD0 0xD0 0xCE 0xC4 0xD7 0xD6 0xA1 0xA3 ~~**0x1B 0x28 0x42**~~ 0x0D 0x0A

0x33 0x2e **0x1B 0x24 0x29 0x41** 0xB5 0xDA 0xC8 0xFD 0xD0 0xD0 0xCE 0xC4 0xD7 0xD6 0xA1 0xA3 ~~**0x1B 0x28 0x42**~~ 0x0D 0x0A[[0x20]]

Note

The underlined byte codes correspond to double byte characters, the bold byte codes to escape sequences.

Table K.3-1. Character Sets and Escape Sequences used in the Examples of Person Name

Character Set Description	Component Group	Value of (0008,0005) Defined Term	ISO registration number	Standard for Code Extension	ESC Sequence	Code Element	Character Set: Purpose of Use
Chinese	First: ~~Phonetic~~Alphabetic (English name)	Value 1: none	ISO-IR 6			G0	ISO 646:
	Second: Ideographic	Value 1: ISO 2022 IR 58	ISO-IR58	ISO 2022	ESC 02/04 02/09 04/01	G1	ISO 2022 CN: Chinese
	Third: ~~Alphabetic (English name)~~Phonetic	Value 1: none	ISO-IR 6	ISO 2022	ESC 02/08 04/02	G0	ISO 646: For delimiters