DICOM Change Proposal

STATUS	Assigned
Date of Last Update	2025/04/12
Person Assigned	David Clunie dclunie@dclunie.com
Submitter Name	Jeroen Medema, jeroen.medema@philips.com
Submission Date	2025/03/11

Change Number CP-2223
Log Summary: Clarify VRs that may have Undefined Length and usage
Name of Standard PS3.5
Rationale for Change: There are two situations in which a Data Element may have an Undefined Length rather than fixed length: 1. The Data Element is Sequence and is encoded with an SQ or UN VR 2. The Data Element is Encapsulated Pixel Data and is encoded in (7FE0,0010) with an OB VR. The recipient may know the VR either because it is explicitly encoded as SQ, OB or UN, or is implicit and known from the data dictionary (particularly in the case of (7FE0,0010) and Encapsulated Transfer Syntaxes). In both situations, the Value Field of the Data Element will consist of Items. In the case of a Sequence there may also be Undefined Length Items that are delimited by an Item Delimiter. In the case of Encapsulated Pixel Data the Items are always of fixed length. There are no other situations in which Undefined Length may be used, otherwise the length of the Value Field would be indeterminate. In particular, there are no mechanisms described for using an Undefined Length for any non-Sequence Data Elements other than (7FE0,0010), and for any VR other than OB, which is always used for all Encapsulated Pixel Data Transfer Syntaxes. It has come to pass that as new VRs have been added and the means of describing various VRs has been amended, they have inadvertently been included amongst those that can theoretically be of Undefined Length. This despite no means for actually encoding them that way in any of the standard Transfer Syntaxes having ever been described. There was no intent to extend the use of Undefined Length beyond the two situations already described. Also, even in the original 1993 description, it was suggested that OW could be used, when in fact only OB was defined for Encapsulated Pixel Data, so this was inconsistent. It is proposed to amend the description of those VRs that may be of Undefined Length, to remove all those that cannot currently be encoded that way. *[Ed.Note: Original text from PS3.3 1993 is highlighted in yellow for historical perspective; not all sections are highlighted, only relevant ones. Text from Sup 14 that introduced the UN VR is highlighted in purple.]*
Change Wording: See below. It is assumed section 7.1.2 is correct.

3.10 DICOM Data Structures and Encoding Definitions

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.

…

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.

…

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 Delimitation Data Elements, respectively.

…

6.2 Value Representation (VR)

…

All new VRs defined in future versions of DICOM shall be of the same Data Element Structure as defined in Section 7.1.2 with reserved bytes after the VR and a 32-bit unsigned integer VL (i.e., following the format for VRs such as OB or UT), and may or may not permit Undefined Length.

…

6.2.2 Unknown (UN) Value Representation

The Unknown (UN) VR shall only be used for Private Data Elements and Standard Data Elements previously encoded as some DICOM VR other than UN using Implicit VR encoding, and whose Value Representation is currently unknown, or whose known Value Representation is one of those that have a 16-bit Value Length Field (see Section 7.1.2) when using Explicit VR encoding and whose Value Length exceeds 65534 (2^{16}-2). As long as the VR is unknown the Value Field is insensitive to 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. .
2. ....
3. …
4. ...
5. The Value Length Field of VR UN may contain Undefined Length (FFFFFFFFH), in which case the contents **of the Value Field** can be assumed to be encoded with Implicit VR. See Section 7.5.1 to determine how to parse Data Elements with an Undefined Length.
6. ...

…

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:

...

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 Value 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

  1. The decoder of a Data Set should support both Explicit and Undefined Lengths for VRs of SQ and UN and, when applicable **depending on the Transfer Syntax**, for VRs of ~~**OW and**~~ OB **for the Pixel Data (7FE0,0010) Data Element**.

  2. ...

Value Field ...

Value Fields with Undefined Length are ~~**delimited**~~ **encoded** through the use of Sequence Delimitation Items**, Items of Fixed Length, or Items of Undefined Length** and Item Delimitation Data Elements, which are described further in Section 7.5 **and Section A.4**.

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 AE, AS, AT, CS, DA, DS, DT, FL, FD, IS, LO, LT, PN, SH, SL, SS, ST, TM, UI, UL and US the Value Length Field is the 16-bit unsigned integer following the two byte VR Field (Table 7.1-2). The value of the Value Length Field shall equal the length of the Value Field.

- for all other VRs the 16 bits following the two byte 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.

  - for VRs of OB, ~~**OD, OF, OL, OV, OW,**~~ SQ and UN, 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**, the Value Field will consist of Item Elements,** and a Sequence Delimitation Item ~~**marks**~~ **will be present at** the end of the Value Field.

  - for all other VRs with a 32-bit Value Length Field, the Value Length Field shall contain a value equal to the length (in bytes) of the Value Field.

    Note

    ~~**VRs of SV, UC, UR, UV and UT may not have an Undefined Length, i.e., a Value Length of FFFFFFFFH.**~~ **The decoder of a Data Set will need to parse the Value Field processing the Item Elements to determine the end of the Value Field. It may not scan for a Sequence Delimitation Item since the series of bytes of which it is composed may be a valid value within the Value Field and there is no escape mechanism.**

…

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, **for VRs of OB and SQ,** the Value Field has an Undefined Length**, the Value Field will consist of Item Elements,** and a Sequence Delimitation Item ~~marks~~ **will be present at** the end of the Value Field.

Note

The decoder of a Data Set will need to parse the Value Field processing the Item Elements to determine the end of the Value Field. It may not scan for a Sequence Delimitation Item since the series of bytes of which it is composed may be a valid value within the Value Field and there is no escape mechanism.

…

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.

…

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 Value (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 Value (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 Value (Item) Length Field shall be equal to 00000000H if the Item contains no Data Set.

b. Undefined Length: The Value (Item) Length Field shall contain the value FFFFFFFFH to indicate an Undefined 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 Value (Item) 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.

…

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 Value (Sequence) 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 Value (Sequence) Length Field shall contain a Value FFFFFFFFH to indicate a Sequence of Undefined 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 a Value (Item) Length Field 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.

…

8.2 Native or Encapsulated Format Encoding

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

…

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 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).

Each Fragment conveys its own explicit even length (see Section A.4).

The Sequence of Fragments of the encapsulated stream is terminated by a Sequence Delimiter Item, thus allowing the support of encoding processes where the resulting length of the entire stream is not known until it is entirely encoded. Encapsulated Formats support both Single-frame and Multi-frame images (as defined in PS3.3). At least one Frame shall be present, and hence at least one Fragment will be present.

…

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. ...

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

- ...
- For the Value Representations OB, OL, OV and OW, the encoding shall meet the following specification depending on the Data Element Tag:
  - Pixel Data (7FE0,0010) 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 defined 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:

    - …

    If encapsulated, it has the Value Representation OB and is an octet-stream 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-frame 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 Value (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.

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

    Note

    The decoder of a Data Set will need to parse the Value Field processing the Item Elements to determine the end of the Value Field. It may not scan for a Sequence Delimitation Item since the series of bytes of which it is composed may be a valid value within the Value Field and there is no escape mechanism.

- …