Log Summary: Fixes to Part 5 for recently added VRs OV, SV and UV (and regarding inconsistent and incomplete definitions)

Name of Standard
PS3.5

Rationale for Correction:
The list of Value Representations (VR) that are not affected by the special handling in PS3.5 Section 6.2.2 is missing the recently added VRs OV, SV and UV. Instead of adding them to the existing negative list, it is proposed to rather refer to the 16-bit Length Field in case of using a Transfer Syntax with Explicit VR encoding.

In PS3.5 Section 6.4, the list of VRs having a fixed Value Multiplicity (VM) of one is again (see CP-1799) not consistent with the normative specification in Section 6.2: OV is missing.

Since the terms “Explicit VR” and “Implicit VR” are already used quite consistently throughout PS3.5, it is proposed to introduce official definitions for them (and to use them completely consistently throughout the standard text). Also other definitions seem to be missing, e.g. “Date Element Tag Field” and “Value Length Field”.

Editorial changes:

- The spelling of “Undefined Length” is not consistent throughout PS3.5 (and probably other Parts of the Standard), i.e. sometimes “undefined length” is used. It is proposed to consistently use the upper case version “Undefined Length”. The same is true for “Value” vs. “value” (if really the definition in Section 3.10 is meant), “Value Length” vs. “value length”, “VR Field”, “Value Length Field” and “Value Field”.

- There are also some inconsistencies regarding the defined term “Value Length Field”, which should be fixed: e.g. sometimes only “Length Field” is used (i.e. “Value” is missing), sometimes “Data Element Length Field”, and for Items even “Item Length Field”…

- The new definitions “Implicit VR” and “Explicit VR” should be used consistently throughout PS3.5 and all other Parts of the DICOM standard.

Correction Wording:

Change PS3.5 Section 3.7

3.7 DICOM Information Object Definitions

This Part of the Standard makes use of the following terms defined in PS3.3:

Attribute Tag

Attribute_Tag.

Information Entity

Information_Entity.

Information Object Definition (IOD)

Information_Object_Definition.
3.10 DICOM Data Structures and Encoding Definitions

The following definitions are commonly used in this Standard:

[...]

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 Tag Field
The field within a Data Element structure that contains the Data Element Tag.

[...]

Explicit VR
Encoding of a Data Element structure where the Value Representation of that Data Element is specified explicitly in Value Representation Field.

Implicit VR
Encoding of a Data Element structure where the Value Representation of that Data Element is specified implicitly, i.e. not in Value Representation Field. This encoding is used for the DICOM Default Little Endian Transfer Syntax (see Section 10.1).

[...]

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.

[...]

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.

[...]

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 structure that contains the Value(s) of that Data Element.

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

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

[...]

Value Representation (VR) Field
The field where the Value Representation of a Data Element is stored in the encoding of a Data Element structure with Explicit VR.

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 Little Endian Transfer Syntax (Implicit VR) encoding, 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 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 \((2^{16}-2)\) 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 byte ordering and shall not be 'byte-swapped' (see Section 7.3). In the case of undefined Value 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 DICOM Default Little Endian Transfer Syntax, and not having been translated to the DICOM Default Little Endian Transfer Syntax 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).

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 Value Length Field.

5. The Value of UN may contain the value of 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 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.

Change PS3.5 Section 6.4

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 older versions of this Standard, 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 delimitier 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 multi-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 multi-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 LT, OB, OD, OF, OL, OV, SQ, ST, UN, UR or UT shall always have a Value Multiplicity of one. See Table 6.2-1.

Change PS3.5 Section 7.1.2

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 and a Sequence Delimitation Item marks 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.

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

<table>
<thead>
<tr>
<th>Tag</th>
<th>VR</th>
<th>Value Length</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Number (16-bit unsigned integer)</td>
<td>Element Number (16-bit unsigned integer)</td>
<td>VR (2 single byte characters)</td>
<td>Reserved (2 bytes) set to a value of 0000H</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tag</th>
<th>VR</th>
<th>Value Length</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Number (16-bit unsigned integer)</td>
<td>Element Number (16-bit unsigned integer)</td>
<td>VR (2 single byte characters) (16-bit unsigned integer)</td>
<td>Even number of bytes containing the Data Element Value(s) encoded according to the VR and negotiated Transfer Syntax.</td>
</tr>
<tr>
<td>2 bytes</td>
<td>2 bytes</td>
<td>2 bytes</td>
<td>2 bytes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>'Value Length' bytes</td>
</tr>
</tbody>
</table>

**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, Value 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.