**Log Summary: Type 1 SQ Empty Items in Functional Groups**

**Name of Standard**
PS 3.3 2011  
PS 3.5 2011

**Rationale for Correction**

CP-994 clarified that if a functional group sequence was not required for every frame, the sequence could be omitted from those per-frame functional group sequence items. It proposed that when the information was not required, the sequence itself should not be sent, rather than sending the sequence with a single empty item.

While a step forward, CP 994 did not quite resolve all the issues around inclusion of Functional Groups. This CP makes clear the difference between, and the conditions for, an absent Sequence, an empty Sequence, and a non-empty Sequence with empty Items.

Some Functional Groups are invoked as Mandatory in the IOD specification, consist of a Type 1 SQ attribute, but have content (Attributes in Items) that is Type 1C. [Example: in Enhanced CT Image IOD, Plane Position (Patient) Functional Group is Mandatory; that Functional Group consists of a Type 1 attribute Plane Position Sequence, and its Item has a single Type 1C attribute Image Position (Patient).]

In such a situation, the question arises as to the expected encoding for the frames where the Type 1C condition is not met. CP-994 stated that the Functional Group Sequence should then not be present – but this conflicts with the Mandatory invocation in the IOD.

Alternatively, the Functional Group could be present with an empty Item. The content of a Sequence Item is dictated by whatever the PS 3.3 IOD defines as being the content of the Item, and that this concept is distinct from whether or not the Sequence Attribute or the Sequence Item itself is required to be sent. I.e., an IOD that permits empty items for a required sequence is valid. This is distinct from the concept of Type 1 versus Type 2 sequences; for Type 1 sequences, Items must be present (even if the Items themselves are empty), whereas for Type 2 sequences, no Item is required (and indeed sending an empty Item when the IOD dataset for that Item requires content is not permitted). This is not explicit in PS 3.5, but should be made so.

A slightly different issue arises with the Type 1 Per-frame Functional Groups Sequence itself, within the Multi-frame Functional Groups Module, which requires one Item for each frame. In some cases, this attribute could possibly have no functional groups to encode, and so would require empty Items corresponding to the number of frames. [Example: Multi-frame Grayscale Byte SC Image has no mandatory functional groups, but the Mandatory Multi-frame Module has the Type 1 Per-frame Functional Groups Sequence.]. This can be resolved by the same solution, i.e., recognizing that each such Item may be empty.

This CP clarifies the validity of a Type 1 SQ attribute with one or more empty Items. It also clarifies the requirement for empty Functional Groups in PS 3.3.

**Correction Wording:**

None specified in the document.
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 PS 3.6. 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.

Notes: 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 delimiting (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 PS 3.6. These Data Elements shall be included in the Data Set and their absence is a protocol violation.

Notes: 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 PS 3.3.

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.

**Notes:**
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.

**Notes:**
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.

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, 0004 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.

No Item Data Element (FFFE,E000) and Item Delimitation Data Element (FFFE,E00D) shall be present if the sequence of Items contains zero Items.

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.

For reference PS3.3 Sections 5.2 and 5.4

5.2 SEQUENCES

Certain Tables in this Standard describe Sequences of Items by using the symbol: ‘>’. The symbol ‘>’ precedes the Attribute (or Module) Name of the members of an Item. All marked Attributes (or Modules) belong to the generic description of an Item that may be repeated to form a Sequence of Items. This Sequence of Items is nested in the Attribute (or Module) that precedes in the table the first member marked with a ‘>’.

Note: The following table describes the "Referenced Series Sequences" Attribute as a Sequence of one or more Items where each Item contains the three Attributes marked by a ‘>’. The Sequence of Items is nested inside the value of the Referenced Series Sequence Attribute. The following Attribute (not marked) is not part of the Items of the Sequence.

<table>
<thead>
<tr>
<th>Referenced Series Sequence</th>
<th>.....</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;Series Date</td>
<td>.....</td>
</tr>
<tr>
<td>&gt;Series Time</td>
<td>.....</td>
</tr>
<tr>
<td>&gt;Series Instance UID</td>
<td>.....</td>
</tr>
<tr>
<td>Modality</td>
<td>.....</td>
</tr>
</tbody>
</table>

This notation may be used to create nested hierarchical structures by using ‘>>’ at the second level of nesting and so on.

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. See PS 3.5.

For describing the number of items in the attribute description the following sentences are preferred:

<table>
<thead>
<tr>
<th>Sequence Attribute Type</th>
<th>Number of Items</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 or 1C</td>
<td>1</td>
<td>Only a single Item shall be included in this sequence.</td>
</tr>
<tr>
<td>1 or 1C</td>
<td>1-n</td>
<td>One or more Items shall be included in this sequence.</td>
</tr>
<tr>
<td>2 or 2C</td>
<td>0-1</td>
<td>Zero or one Item shall be included in this sequence.</td>
</tr>
<tr>
<td>2 or 2C</td>
<td>0-n</td>
<td>Zero or more Items shall be included in this sequence.</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>Only a single Item is permitted in this sequence.</td>
</tr>
<tr>
<td>3</td>
<td>1-n</td>
<td>One or more Items are permitted in this sequence.</td>
</tr>
</tbody>
</table>

Note: The encoding of empty sequence attributes is described in PS 3.5.

In a number of cases for Normalized IODs, the Data Element Type and Conditions are defined in the appropriate Service definition in PS 3.4, in other cases in the attribute description in PS 3.3. It is not
necessary to specify for any Attribute within a Sequence the condition that it is “required if a Sequence item is present”, since this is always implicit, whether or not there are additional requirements.

...  

5.4 ATTRIBUTE MACROS

Some tables contain references to Attribute Macros. This convention is used in cases where the same Attributes are used in multiple tables or multiple places in one Module. The reference means that the Attributes of the Attribute Macro shall be included in the Module in place of the row that contains the reference to the Attribute Macro.

In some cases, the Attribute Macro is used in a Sequence (the VR of the Data Element in which the Attribute is encoded is SQ, see PS 3.5). When this is done, the reference is preceded by one or more “>” characters. The number of “>” characters indicates the level in the sequence that all of the Attributes in the Attribute Macro occupy.

There may be specialization of the description of the Attributes in the Attribute Macro. In these cases, this specialization is described in the Description column of the Module.

Following is an example of this convention.

Table 5.4-1 is an example of a Module table using the Attribute Macro convention.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Type</th>
<th>Attribute Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute A</td>
<td>(aaaa,aaaa)</td>
<td>1</td>
<td>This is an example.</td>
</tr>
<tr>
<td>Attribute B Sequence</td>
<td>(bbbb,bbbb)</td>
<td>1</td>
<td>This is an example of a Sequence Attribute</td>
</tr>
</tbody>
</table>

>Include ‘Example Macro’ Table 5.4-2

In this Module, Attribute D (dddd,dddd) is Type 1

Table 5.4-2 is an example of the Attribute Macro referenced in Table 5.4-1.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Type</th>
<th>Attribute Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute C</td>
<td>(cccc,cccc)</td>
<td>1</td>
<td>This is an example.</td>
</tr>
<tr>
<td>Attribute D</td>
<td>(dddd,dddd)</td>
<td>3</td>
<td>This Attribute is generally a Type 3.</td>
</tr>
</tbody>
</table>

The contents of the Example Module Table, if it had not been described with the Example Macro would have been as shown in Table 5.4-3

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Type</th>
<th>Attribute Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute A</td>
<td>(aaaa,aaaa)</td>
<td>1</td>
<td>This is an example.</td>
</tr>
<tr>
<td>Attribute B Sequence</td>
<td>(bbbb,bbbb)</td>
<td>1</td>
<td>This is an example of a Sequence Attribute.</td>
</tr>
<tr>
<td>&gt;Attribute C</td>
<td>(cccc,cccc)</td>
<td>1</td>
<td>This is an example.</td>
</tr>
<tr>
<td>&gt;Attribute D</td>
<td>(dddd,dddd)</td>
<td>1</td>
<td>In this Module, this Attribute has been specialized to Type 1 as indicated in Table 5.4-1.</td>
</tr>
</tbody>
</table>
C.7.6.16 Multi-frame Functional Groups Module

Table C.7.6.16-1 specifies the attributes of the Multi-frame Functional Groups Module. This module is included in SOP instances even if there is only one frame in the instance.

**Table C.7.6.16-1**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Type</th>
<th>Attribute Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Functional Groups Sequence</td>
<td>(5200,9229)</td>
<td>2</td>
<td>Sequence that contains the Functional Group Macros that are shared for all frames in this SOP Instance and Concatenation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: The contents of this sequence are the same in all SOP Instances that comprise a Concatenation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Zero or one Item shall be included in this sequence. See section C.7.6.16.1.1 for further explanation.</td>
</tr>
<tr>
<td>&gt;Include one or more Functional Group Macros that are shared by all frames. The selected Functional Group Macros shall not be present in the Per-frame Functional Groups Sequence (5200,9230).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Per-frame Functional Groups Sequence</td>
<td>(5200,9230)</td>
<td>1</td>
<td>Sequence that contains the Functional Group Sequence Attributes corresponding to each frame of the Multi-frame Image. The first Item corresponds with the first frame, and so on. One or more Items shall be included in this sequence. The number of Items shall be the same as the number of frames in the Multi-frame image. See Section C.7.6.16.1.2 for further explanation.</td>
</tr>
<tr>
<td>&gt;Include one or more Functional Group Macros.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

C.7.6.16.1 Multi-frame Functional Groups Module Attribute Description

C.7.6.16.1.1 Functional Group

A Functional Group is a set of Attributes that are logically related and may vary together. Functional Groups are defined by editorial convention in Macros. Those Functional Groups Macros that apply to all frames are included in the Shared Functional Groups Sequence (5200,9229). Functional Groups...
**Macros** whose attribute values may vary from frame to frame are included in the Per-frame Functional Groups Sequence (5200,9230).

A single Functional Group **Macro** shall not be included in both the Shared Functional Groups Sequence (5200,9229) and the Per-frame Functional Groups Sequence (5200,9230).

Notes: 1. In the case of a SOP Instance containing a single frame, some Functional Groups **Macros** may be contained in the Shared Functional Groups Sequence (5200,9229) and others in the one Item of the Per-frame Functional Groups Sequence (5200,9230).

2. Even if there are no Functional Groups **Macros** in the Per-frame Functional Groups Sequence (5200,9230) an empty Item is encoded for every frame, **which is permitted for a Type 1 Sequence, as described in PS 3.5**.

It may happen that **the Data Set for the Item of** a Functional Group Sequence **Attribute** does not contain any **value Attributes** (e.g., a condition for a single Type 1C attribute in the sequence is not met). **In this case, the Item is included but is empty**.

**Notes:** PS 3.5 describes that an empty Item is permitted for a Type 1 or Type 2 Sequence depending on what the IOD in PS 3.3 defines for the Data Set that is defined for that Sequence Item.

It may happen that **a Functional Group Sequence Attribute** or is not required for a particular frame (e.g., an optional Functional Group). In this case the Functional Group Sequence **Attribute** is not included in the Shared Functional Groups Sequence (5200,9229) or the Per-frame Functional Groups Sequence (5200,9230) for a particular frame.

**Note:** The absence of the sequence attribute corresponding to a particular functional group **macro** indicates that the functional group is not used for a particular frame.