DICOM Correction Proposal

|  |  |
| --- | --- |
| STATUS | New |
| Date of Last Update | 2016/01/20 |
| Person Assigned | Jim Philbin <james.philbin@jhmi.edu> |
| Submitter Name | Kinson Ho (kinson.ho@agfa.com) |
| Submission Date | 2015/05/25 |

|  |
| --- |
| Correction Number CP1582 |
| Log Summary: Add STOW-RS support for consumer media types |
| Name of Standard  PS 3.18 2015b Section 6.6 |
| Rationale for Correction:  Currently mobile or browser applications want to use STOW-RS to upload DICOM instances, but these user agents may not know how to extract the Pixel Data Macro values from an image in a consumer oriented media type such as JPEG or PNG. This functionality is needed for the IHE WIC Profile.  In addition, in WADO-RS and STOW-RS when an Encapsulated Document (0042,0011) attribute is contain in bulk data they are currently specified to have a Content-Type value of 'application/octet-stream'. It would be better to have a more specific media type for encapsulated documents such as 'application/pdf' or text/xml'. This would be true whether the Content-Type header field is in the message header or a part header in a multipart payload.  Goals:   * Allow mobile and browser applications to upload images and video in non-DICOM media types. * Allow the Content-Type header field of HTTP messages or their parts contain a media type value that is the same as that in the MIME Type of Encapsulated Document (0042,0012) attribute.   Issue:  Should we provide a mechanism that allows any values in the List of MIME Types (0042,0014) attribute to be included in the Content-Type header field, perhaps using a media type parameter? |
| Correction Wording: |

*Update PS3.3, Annex C.7.6.3 as follows:*

**C.7.6.3 Image Pixel Module**

[Table C.7-11a](#table_C_7_11a) describes the Image Pixel Module.

**Table C.7-11a. Image Pixel Module Attributes**

| **Attribute Name** | **Tag** | **Type** | **Attribute Description** |
| --- | --- | --- | --- |
| *Include* [*Table C.7-11b “Image Pixel Macro Attributes”*](#table_C_7_11b) | | | **~~Required unless the image is being sent by a STOW-RS user agent. STOW-RS origin servers supporting consumer media type are required to either:~~**   * **~~Convert the Pixel Data Value Field to an appropriate DICOM Transfer Syntax (see PS3.6), if necessary.~~** * **~~Insert the appropriate Image Pixel Macro Attributes from Table C.7-11b.~~**   **~~Or, if the origin server does not support the requested media type, return a status of 415 (Unsupported Media Type).~~**  **~~The resulting Instance(s) shall fully conform to the requirements of the Standard before being stored by the origin server.~~**  **~~Note:~~**  **~~A STOW-RS user agent may specify some or all of the Pixel Data Macro attributes.~~**  **Required except for STOW-RS user agents. See PS3.18, Section 6.6.** |
| Pixel Data Provider URL | (0028,7FE0) | 1C | A URL of a provider service that supplies the pixel data of the Image.  Required if the image is to be transferred in one of the following presentation contexts identified by Transfer Syntax UID:   * 1.2.840.10008.1.2.4.94 (DICOM JPIP Referenced Transfer Syntax) * 1.2.840.10008.1.2.4.95 (DICOM JPIP Referenced Deflate Transfer Syntax)   Note  The VR of this attribute has changed from UT to UR. |
| Pixel Padding Range Limit | (0028,0121) | 1C | Pixel value that represents one limit (inclusive) of a range of padding values used together with Pixel Padding Value (0028,0120) as defined in the [General Equipment Module](#sect_C_7_5_1). See [Section C.7.5.1.1.2](#sect_C_7_5_1_1_2) for further explanation.  Required if pixel padding is to be defined as a range rather than a single value.  Note   * The Value Representation of this Attribute is determined by the value of Pixel Representation (0028,0103). * Pixel Padding Value (0028,0120) is also required when this Attribute is present. |

[Table C.7-11b](#table_C_7_11b) specifies the common attributes that describe the pixel data of the image.

*Update PS3.18, Section 6.6 as follows:*

**6.5.1.2.2 Bulk Data Response**

* Content-Type:
* multipart/related; type=application/octet-stream; boundary={MessageBoundary}
* multipart/related; type={MediaType}; boundary={MessageBoundary}
* The entire multipart response contains all bulk data for the specified Study that can be converted to one of the requested media types.
* Each item in the response is one of:
* an uncompressed bulk data element encoded in Little Endian binary format with the following headers:
* Content-Type: application/octet-stream
* Content-Location: {BulkDataURL}
* **An encapsulated document element from a SOP Instance in the Study encoded in the media type specified in the MIME Type of Encapsulated Document (0042,0012) attribute of the Instance with the following header fields:**
* **Content-Type: {media-type}**

**Where the media-type is from the MIME Type of Encapsulated Document (0042,0012) attribute.**

* **Content-Location: {BulkDataURL}**
* a compressed bulk data element from a SOP Instance in the Study encoded in a single-frame compression {MediaType} with the following headers:
* Content-Type: {MediaType}
* Content-Location: {BulkDataURL}
* a compressed frame from a multi-frame SOP Instance in the Study encoded in a single-frame media type with the following headers:
* Content-Type: {MediaType}
* Content-Location: {BulkDataURL}/frames/{FrameNumber}

Note

Each frame will come in a separate part.

* a set of compressed frames from a SOP Instance in the Study encoded in a multi-frame media type with the following headers:
* Content-Type: {MediaType}
* Content-Location: {BulkDataURL}[/frames/{FrameList}]
* {FrameList} is a list of frames separated by %2C (comma). It may be omitted if the message part includes all frames for the specified bulk pixel data object.

*Update PS3.18, Section 6.6 as follows:*

**6.6 STOW-RS Request/Response**

The STOW-RS Service defines one action type. An implementation shall support the following action type:

1. Store Instances

This action creates new resources for the given SOP Instances on the Server or appends to existing resources on the Server.

All request messages are HTTP/1.1 multipart messages. The organization of SOP Instances into message parts depends on whether the SOP Instances are structured as [PS3.10](file:///D:\MyFiles\Healthcare\dicom\CPs\part10.pdf#PS3.10) binary instances, or metadata and bulk data.

[PS3.10](file:///D:\\MyFiles\\Healthcare\\dicom\\CPs\\part10.pdf" \l "PS3.10) binary instances shall be encoded with one message part per DICOM Instance.

Metadata and bulk data requests will be encoded in the following manner:(see [Figure 6.5-1 Mapping between IOD and HTTP message parts](file:///D:\MyFiles\Healthcare\dicom\CPs\part18.docx#figure_6_5_1)):

* All XML request messages shall be encoded as described in the Native DICOM Model defined in [PS3.19](file:///D:\MyFiles\Healthcare\dicom\CPs\part19.pdf#PS3.19) with one message part per XML object.
* All JSON requests shall be encoded as an array of DICOM JSON Model Objects defined in [Annex F](file:///D:\MyFiles\Healthcare\dicom\CPs\part18.docx#chapter_F).
* Uncompressed bulk and pixel data shall be encoded in a Little Endian format using the application/octet-stream media type with one message part per bulk data item.
* Compressed pixel data shall be encoded in one of two ways:
* Single-frame pixel data encoded using a single-frame media type (one message part)
* Multi-frame or video pixel data encoded using a multi-frame media type (multiple frames in one message part)

**The origin server shall encode DICOM Instances with compressed** **~~Compressed~~** pixel data **~~shall be encoded~~** using the Media Types as described in [Table 6.5-1](#table_6_5_1) **~~WADO-RS Media Type Mapping to Transfer Syntax UID~~**. **DICOM** Media Types **~~corresponding to several DICOM Transfer Syntax UIDs~~** may require a transfer-syntax parameter **~~to disambiguate the request~~**. **See Section 6.1.1.8.**

**~~HTTP Request field Content-Type is used in the header lines by the client in an HTTP/1.1 transaction to indicate the type of data being sent to the Service. All lines are~~** [**~~[RFC 822]~~**](#biblio_RFC_822) **~~or~~** [**~~[RFC 7230]~~**](#biblio_RFC_7230) **~~format headers. All HTTP header fields whose use is not defined by STOW-RS shall have the meaning defined by the HTTP standard.~~**

* **Consumer Media Types:**

**Some user agents may not be able to support DICOM image or video media types, such user agents are permitted to include consumer media types in the Pixel Data (7FE0,0010) Attribute. If a DICOM Instance contains a consumer media type the Content-Type header field of the message or message part shall contain that media type.**

**Table X.Y contains a list of recommended consumer media types that origin servers should support.**

**Table X.Y: Consumer Media Types Supported by Store Studies Service**

|  |  |  |
| --- | --- | --- |
| **Resource Category** | **Media Type** | **Notes** |
| **Image** | **image/gif** |  |
| **~~Image/jp2~~** |  |
| **image/jpeg** |  |
| **~~image/jpx~~** |  |
| **image/png** |  |
| **~~image/tiff~~** |  |
| **~~image/webp~~** |  |
| **Video** | **~~video/jpeg~~** |  |
| **~~video/H264~~** |  |
| **video/H265** |  |
| **video/mp4** |  |
| **video/mpeg** |  |
| **~~video/mj2~~** |  |
| **~~video/jpeg2000~~** |  |
| **Text** | **~~application/xml~~** | **??** |
| **application/hl7-cda+xml** | **Encapsulated CDA** |
| **~~application/hl7-v2~~** |  |
| **application/hl7-v2+xml** |  |
| **~~application/hl7-v3+xml~~** |  |
| **application/pdf** | **Encapsulated PDF** |
| **text/html** | **??** |
| **text/xml** |  |

**User agents and origin server may support other consumer media types. Any consumer media types supported by user agents and/or origin servers should be included in their Conformance Statement. Origin servers should also include them in their Capabilities document.**

The Service is required to support uncompressed bulk and pixel data (multipart/related; type= application/octet-stream).

*Update PS3.18, Section 6.6.1.1 as follows:*

**6.6.1.1 Request**

The specific Service resource to be used for the Store Instances action shall be as follows:

* Resource
* {SERVICE}/studies[/{StudyInstanceUID}], where
* {SERVICE} is the base URL for the service. This may be a combination of scheme (either HTTP or HTTPS), host, port, and application.
* {StudyInstanceUID} (optional) is the study instance UID for a single study. If not specified, instances can be from multiple studies. If specified, all instances shall be from that study; instances not matching the StudyInstanceUID shall be rejected.

**Note**

**It is not necessary that the study referenced by the Study Instance UID in the resource (and in the provided instances) exists on the server, however it is necessary that it be a valid UID. The client may have obtained an appropriate UID from elsewhere or generated it as described in** [**Chapter 9 “Unique Identifiers (UIDs)” in PS3.5**](part05.pdf#chapter_9) **and** [**Annex B “Creating a Privately Defined Unique Identifier (Informative)” in PS3.5**](part05.pdf#chapter_B)**.**

* Method
* POST
* Headers
* Content-Type - The representation scheme being posted to the RESTful service. The types allowed for this request header are as follows:
* multipart/related; type=application/dicom; boundary={messageBoundary}

Specifies that the post is [PS3.10](part10.pdf#PS3.10) binary instances. All STOW-RS providers must accept this Content-Type.

* multipart/related; type=application/dicom+xml; boundary={messageBoundary}

Specifies that the post is [PS3.19](part19.pdf#PS3.19) XML metadata and bulk data. All STOW-RS providers must accept this Content-Type.

* multipart/related; type=application/json; boundary={messageBoundary}

Specifies that the post is DICOM JSON metadata and bulk data. A STOW-RS provider may optionally accept this Content-Type.

* **Payload**

**The STOW request payload will typically contain Instances in DICOM Media Types; however, since some user agents, such as mobile or browser applications, may not be able to create DICOM Media Types, the user agent is allowed to send image or video instances in consumer media types. See Table X.Y.**

**Any image or video Instances encoded in consumer media types are only required to have a Pixel Data (7FE0, 0010) Attribute, which contains the image(s) or video encoded in a consumer media type. All other Image Pixel Macro Attributes (see Table C.7-11b) are optional, but if supplied should conform to the requirements of the Pixel Data Macro.**

**~~Note~~**

**~~It is not necessary that the study referenced by the StudyInstanceUID in the resource (and in the provided instances) exists on the server, however it is necessary that it be a valid UID. The client may have obtained an appropriate UID from elsewhere or generated it as described in~~** [**~~Chapter 9 “Unique Identifiers (UIDs)” in PS3.5~~**](part05.pdf#chapter_9) **~~and~~** [**~~Annex B “Creating a Privately Defined Unique Identifier (Informative)” in PS3.5~~**](part05.pdf#chapter_B)**~~.~~**

*Update PS3.18, Section 6.6.1.2 as follows:*

**6.6.1.2 Action**

The **~~Service~~** **origin server** may coerce or replace values of attributes such as Patient Name, ID, Accession Number, for example, during import of media from an external institution, reconciliation against a master patient index, or reconciliation against an imaging procedure order. The Service may correct, or replace incorrect values, such as Patient Name or ID, for example, when incorrect worklist item was chosen or operator input error occurs.

If any element is coerced or corrected, the Original Attribute Sequence (0400,0561) shall be included in the DICOM Object that is stored and may be included in the PS3.18 XML Store Instances Response Module in the response.

Note

For more information on populating the Original Attribute Sequence, see [Section C.12.1 “SOP Common Module” in PS3.3](part03.pdf#sect_C.12.1).

**The origin server shall convert any Instances encoded with consumer media types into an appropriate DICOM Media Type as follows:**

* **Convert the Pixel Data Attribute to an appropriate DICOM Media Type and Transfer Syntax.**
* **Insert or update any required Pixel Data Macro Attributes (see Table C.7-11b) as necessary to create a valid DICOM Instance.**

**The resulting Instance(s) shall fully conform to the requirements of the Standard before being stored by the origin server.**

**if the origin server does not support the requested consumer media type, it shall return a status of 415 (Unsupported Media Type).**

*Update PS3.18, Section 6.6.1.1.2 as follows:*

**6.6.1.1.2 XML Metadata and Bulk Data Request Message Body**

The XML Metadata and Bulk Data Request Message has a multipart body.

* Content-Type:
* multipart/related; type=application/dicom+xml; boundary={MessageBoundary}
* The multipart request body contains all the metadata and bulk data to be stored. If the number of bulk data parts does not correspond to the number of unique BulkDataURIs in the metadata then the entire message is invalid and will generate an error status line.
* Each body part is either DICOM [PS3.19](file:///G:\CP_1582\part19.pdf#PS3.19) XML metadata or a bulk data item from a SOP Instance sent as part of the Store operation. The first part of the multipart message must be XML metadata.
* Each bulk data item must be preceded by all metadata items that contain a reference to it.

Note

This requires that all bulk data items for an instance must be preceded by the XML metadata for that instance and if a bulk data item is included in multiple instances it must be preceded by the XML metadata for each instance in which it is included.

* The first part in the multipart request will contain the following HTTP headers:
* Content-Type: application/dicom+xml; transfer-syntax={TransferSyntaxUID}
* Subsequent items will contain the following HTTP headers (order is not guaranteed):
* additional metadata with the following headers:
* Content-Type: application/dicom+xml; transfer-syntax={TransferSyntaxUID}

Where {TransferSyntaxUID} is the UID of the DICOM Transfer Syntax used to encode the inline binary data in the XML metadata.

* **~~an encapsulated PDF document with the following headers~~**
* **~~Content-Type: application/dicom+pdf~~**
* **~~Content-Location: {BulkDataURI}~~**
* **~~an encapsulated CDA document with the following headers~~**
* **~~Content-Type: application/dicom+cda~~**
* **~~Content-Location: {BulkDataURI}~~**
* **an encapsulated document with the following headers**
* **Content-Type: {media-type}**

**Where the media-type is from the MIME Type of Encapsulated Document (0042,0012) attribute.**

* **Content-Location: {BulkDataURI}**
* an uncompressed bulk data element encoded in Little Endian binary format with the following headers:
* Content-Type: application/octet-stream
* Content-Location: {BulkDataURI}
* a compressed pixel data object from a SOP Instance in the Study with the following headers:
* Content-Type: {MediaType}
* Content-Location: {BulkDataURI}
* Metadata and its associated bulk data shall always be sent in the same POST request.

Note

It is not intended that metadata and bulk data be stored separately in multiple POST requests since the service always requires the metadata for context.

*Update PS3.18, Section 6.6.1.1.3 as follows:*

**6.6.1.1.3 JSON Metadata and Bulk Data Request Message Body**

The JSON Metadata and Bulk Data Request Message has a multipart body.

* Content-Type:
* multipart/related; type=application/json; boundary={MessageBoundary}
* The multipart request body contains all the metadata and bulk data to be stored. If the number of bulk data parts does not correspond to the number of unique BulkDataURIs in the metadata then the entire message is invalid and will generate an error status line.
* The first part in the multipart request will contain a JSON array of DICOM JSON Model Objects (defined in [Annex F](#chapter_F)). Each array element is the metadata from a SOP Instance sent as part of the Store operation. This message part will have the following headers:
* Content-Type: application/json; transfer-syntax={TransferSyntaxUID}

Where {TransferSyntaxUID} is the UID of the DICOM Transfer Syntax used to encode the inline binary data in the JSON metadata.

* Subsequent items will be one of the following:
* **~~an encapsulated PDF document with the following headers~~**
* **~~Content-Type: application/dicom+pdf~~**
* **~~Content-Location: {BulkDataURI}~~**
* **~~an encapsulated CDA document with the following headers~~**
* **~~Content-Type: application/dicom+cda~~**
* **~~Content-Location: {BulkDataURI}~~**
* **an encapsulated document with the following headers**
* **Content-Type: {media-type}**

**Where the media-type is from the MIME Type of Encapsulated Document (0042,0012) attribute.**

* **Content-Location: {BulkDataURI}**
* an uncompressed bulk data element encoded in Little Endian binary format with the following headers:
* Content-Type: application/octet-stream
* Content-Location: {BulkDataURI}
* a compressed pixel data object from a SOP Instance in the Study with the following headers:
* Content-Type: {MediaType}
* Content-Location: {BulkDataURI}
* JSON Metadata and its associated bulk data shall always be sent in the same POST request.

Note

It is not intended that metadata and bulk data be stored separately in multiple POST requests since the service always requires the metadata for context.