DICOM Correction Proposal

|  |  |
| --- | --- |
| STATUS | Draft Final Text |
| Date of Last Update | 2016/05/26 |
| Person Assigned | Jim Philbin (james.philbin@jhmi.edu) |
| Submitter Name | Jim Philbin (james.philbin@jhmi.edu) |
| Submission Date | 2015/09/13 |

|  |
| --- |
| Correction Number CP1509 |
| Log Summary: Clarify DICOM media types |
| Name of Standard  PS3.2, PS3.17, PS3.18 2016b |
| Rationale for Correction:  The definitions of DICOM media types need to be clarified.  PS3.18 currently refers to DICOM media types having transfer syntax parameters, but only RS services discuss them.  An important design principle of HTTP is that media types are independent of the messages that carry them. This, in turn, means that the transfer syntax parameter should be defined with the DICOM media type specification and should apply to all media types returned from DICOM services. However, there is no requirement added to existing services such as WADO-URI to support a transfer syntax parameter in Accept headers in the request, since there is already a query parameter defined for this purpose, and doing so would reduce interoperability with the installed base.  This CP defines the syntax of DICOM media types and their parameters. It consolidates the information related to DICOM media types into one section.  It also removes the incorrectly used media types introduced by Sup 161 that are of the form 'image/dicom+xxx', which do not exist, have not been registered with IANA, and which are redundant with non-DICOM media types that apply to the same content that is returned and hence are not only unnecessary but are inconsistent with non-DICOM practice (e.g., "image/jpeg" is appropriate instead of "image/dicom+jpeg", which was incorrectly used before, even if the data is not encoded using the Baseline JPEG process).  Since the same media types may be used for both the bulk data of pixel data for DICOM instances and for rendered DICOM instances, and the same resources are used for both types of service, a "rendered" resource is needed to distinguish the two different types of request.  It also corrects the usage of the media type to be surrounded by double-quotes when it occurs as a parameter in multipart media types.  The handling of bulk data retrieval of multi-frame images is clarified to match the other retrieval actions, since only a single bulk data item is referenced from the XML and JSON meta data for pixel data.  The incorrect references to selected frames in the response to Study, Series and Instance retrieval actions are removed and the text clarified to be "all frames".  The default media type for the QIDO-RS response is changed from XML to JSON.  *[Comment: Fujifilm: should also modify the definition of BulkDataURI in section 4, re. not including the Basic Offset Table and Data Stream Fragment Item tags and lengths.]*  *[Comment: Fujifilm: Do rendered and non-rendered responses require multipart/related to be used when returning only a single part, why is there a difference between rendered and non-rendered response, and what re the implications for browsers that cannot handle multipart/related (contrast this with WADO-URI).]*  *[Comment: IANA Expert reviewer: +rle is not a registered media type suffix. As such, the media type name needs to be changed to something like dicom-rle].*  *[Comment: Hologic: 6.4.5: ws-media-type is defined in Section 6.1.1.8.5 <not 6.1.1.8.2>. FIXED].*  *[Comment: Hologic: 6.5.1.1, 6.5.2.1, 6.5.3.1: For "application/dicom", if not specified isn't the transfer syntax the defined default (per section 6.1.1.8) rather than 'chosen freely'? Same comment by McKesson: 6.5.1.1 “If transfer-syntax is not specified the server can freely choose which Transfer Syntax to use”: I thought if no TS was specified, then the default TS was used, and the server is only free to choose if the client specifies “\*”.) AGREED. FIXED.]*  *[Comment: Hologic: (octet-stream comment also applies to 6.5.4.1, 6.5.5.1) Shouldn't "application/octet-stream" also reference section 6.1.1.8? AGREED. Added table reference to 6.1.1.8-3a and -3b as apparopriate.]*  *[Comment: Hologic: 6.5.6.2.2: 'Where {TransferSyntaxUID}' should be 'Where [transfer-syntax-uid]'. FIXED.]*  *[Comment: Hologic: 6.5.8, 2nd paragraph: 'The primarily' should be 'The primary'. FIXED.]*  *[Comment: Hologic: section number 6.8.1.2.2.4 is mislabeled as 1.8.1.2.2.4. FIXED]*  *[Comment: Hologic: the editor's note for Annex HHH should state PS 3.17, not PS 3.18. FIXED]*  *[Comment: Toshiba: various wording changes for 6.1.1.3. FIXED.]*  *[Comment: Toshiba: In 6.1.1.8, do you want a note that this is different than media types into which DICOM instances can be rendered, which are covered in 6.1.1.3? AGREED. Text added.]*  *[Comment: Toshiba: There is a reference to "see 6.1.1.8.2.2" which doesn't exist. FIXED. And several similar incorrect references that should be to 6.1.1.8.1.x not 2.x]*  *[Comment: Toshiba: At the bottom of 6.5 it says "Support for the "transfer-syntax" and "charset" parameters is required." Did you want to make that a "shall" and indicate who the shall applies to? AGREED. Reworded.]*  *[Comment: Toshiba: Clean up "all"/"a" "provider[s] shall" in 6.5.6.1. FIXED.]*  *[Comment: McKesson: Clean up wording of default Transfer Syntax in 6.1.1.8. FIXED.]*  *[Comment: McKesson: Table 6.1.1.8-2, 6.1.1.8-3b: Why list these optional Transfer Syntaxes? ANSWER. Because for Table 6.1.1.8-2 defines the resource category for each of those Transfer Syntaxes, and Table , 6.1.1.8-3b defines the media types to be used for those Transfer Syntaxes.]*  *[Comment: McKesson: Table 6.1.1-2 defines default, required, and optional in the context of Media Type. Do we need an additional definition for the context of Transfer Syntaxes in section 6.1.1.8? FIXED. Added cross-references from 6.1.1.8 to Table 6.1.1-2.]*  *[Comment: McKesson: 6.1.1.8 duplicate item number in note. FIXED.]*  *[Comment: McKesson: 6.1.1.8 “Multi-frame image bulk data is encoded as one frame per part.” Does this differ between compressed and uncompressed? ANSWER. Yes. per the existing text of 6.5, all frames for uncompressed pixel data are returned in a single part: "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". This CP does not propose to change that. Updated the note with clarifying text.]*  *[Comment: McKesson: 6.1.1.8.1: Does it make sense to allow Transfer Syntax to be specified for dicom+xml and dicom+json? AGREED. It does not. Transfer Syntax only applies to application/dicom; amended text in 6.1.1.8.1 and 6.1.1.8.1.2 accordingly.]*  *[Comment: McKesson: Word has converted some double quotes to “smart” quotes. UNDERSTOOD THAT WORD SUCKS. Will not propagate into DocBook generated standard. Fixed font of ';'.]*  *[Comment: McKesson: 6.1.1.8.1.1 On l. 174, where is octet-stream defined? Definition added to 6.1.1.8.1.1.]*  *[Comment: McKesson: 6.1.8.1.2 p.11, l.183 “It is required for RS services.”: What is “it”? Support for transfer syntax parameters, or the presence of those parameters? What needs to be in the conformance statement: that transfer syntax parameters are supported, or the transfer syntaxes that may be specified via parameters? FIXED.]*  *[Comment: McKesson: There is inconsistent use of “mode” and “service” throughout when referring to RS, URI or WS. (And, confusingly, “service” is used to refer to capabilities within RS. This is an artifact of the re-documentation. The current standard uses "mode", but recent CPs and the redoc use "service". I've changed all uses of "mode" to "service".]*  *[Comment: McKesson: 6.1.1.8.1.2 p.11, l.192 “prioritized left to right”: Suggest adding “(most preferred)” after “left”. Also for section 6.1.1.8.1.3, p.12 l.206. FIXED.]*  *[Comment: McKesson: 6.1.1.8.1.2 p.11, l.197 “to respond without needing to transcode”: Is this the only reason? Could an origin server prefer to respond with explicit little-endian regardless of the TS of the original file? I suggest “…accept any transfer syntax. This would allow, for example, the origin server to respond…”. FIXED.]*  *[Comment: McKesson: 6.1.1.8.1.2.1 p.11 l.202: why is this syntax written using a different style? The organization of 6.1.1.8 is getting confusing. Why is the content type header field definition split in two parts (here and under ch6.1.1.8.1.3.1, which refers back to here)? Sections 6.1.1.8.1.2.1 and 6.1.1.8.3.1 have been consolidated into 6.1.3.]*  *[Comment: McKesson: 6.1.1.8.2: referenced section 6.1.1.8.2.2 does not exist. FIXED. .]*  *[Comment: McKesson: 6.1.1.8.4 The resolution process on lines 239-247 makes the second bullet point explicit; remove the second bullet. The third bullet point is stated elsewhere; remove the third bullet. Since there is now only one bullet, combine into previous paragraph, and remove “The Selected Transfer Syntax has the following characteristics:”. AGREED. Changes made as requested.]*  *[Comment: McKesson: 6.1.1.8.4 The resolution process on lines 239-247 do not explicitly account for the “\*” transfer syntax. TBD. Does it need to? That is handled by first determining the Selected Media Type, is it not? FIXED.]*  *[Comment: McKesson: 6.1.1.8.5: I read this to say that application/dicom is not permitted for WS or RS services. Is that correct? There is no multipart-dicom or multipart-dicom-xml defined in 6.1.1.8.1. FIXED. ]*  *[Comment: McKesson: 6.2.2.1.1: Unless it is specified in another section, I read this to say that WADO-URI doesn’t support rendered media types. FIXED. That was not the intent.]*  *[Comment: McKesson: 6.2.2.1.1 “are forbidden for the WADO-URI service”: this seems to contradict 6.1.1.8.5 “Support… is optional for WS and URI services.” Which is correct? FIXED. The itent is to explicitly forbid them so as to not break the URI installed base. Updated 6.1.1.8.5]*  *[Comment: McKesson: 6.5 “DICOM objects returned shall be PS3.10 binary objects.” This doesn’t match the language proposed for WADO-URI (6.3.1.2); is there a reason? By this usage, XML-encoded instances are not “DICOM objects”; is that correct? FIXED. Made 6.5 consistent with 6.3.1.2 and moved separate sentence into encoding bulleted list and reworded for consistency and clarity.]*  *[Comment: McKesson: 6.5 “HTTP Request field Accept is used…”: This paragraph seems out of place here. Except for the use of “WADO-RS” in one spot it is not RS specific. Suggest moving to a more widely applicable section and changing the WADO-RS reference to “DICOM.” (A similar paragraph in STOW-RS, l.731, has mostly been removed.) FIXED. Cleaned up as per STOW.]*  *[Comment: McKesson: 6.5 “The server is required…”: Is it actually required? Or only if the server supports bulk URLs? FIXED. This does raise an interesting question - there is no requirement to return things like PixelData as bulk data, i.e., it could be very inefficiently encoded inline, right? Yes.]*  *[Comment: McKesson: 6.5 This repeats content ("RS Services support the following media types") from 6.1.1.8.5. Suggest removing one or the other. Also should application/octet-stream be included here? FIXED. Made reference to section 6.1.1.8.5 and added sentence for bulk data media types. Also added definitions for Bulk Data Media Type and DICOM Media Type]*  *[Comment: McKesson: 6.5.1.1 If you change to “[dcm-parameters]” there is no longer a “transfer-syntax” to refer to. (Also at 6.5.2.1 and at 6.5.3.1). FIXED. By adding "* *in the dcm-parameters".]*  *[Comment: McKesson: 6.5.1.1: The rendered media types in table 6.1.1.8-3b are not permitted according to 6.5. (Also at 6.5.2.1, at 6.5.4.1, at 6.5.5.1, at 6.6.). NO CHANGE. The text in 6.5 has been amended to recognize these for bulk data in response to another comment.]*  *[Comment: McKesson: Several references to table 6.1.1.8-2 throughout indicate that is contains media types and transfer syntaxes. The current table only contains transfer syntaxes. Either change the references to indicate that the table only contains TS, or add references to the appropriate tables containing media types. FIXED. In all references clarified that this is for the application/dicom media type and that the table specifies transfer syntaxes.]*  *[Comment: McKesson: 6.5.5.1: Definition of server behavior doesn’t really belong in request definition. Suggest moving to response. AGREED. Moved.]*  *[Comment: McKesson: 6.5.6.1 “All WADO-RS providers shall support this media type.”: This statement is redundant with table 6.1.1.8-1b, and inconsistent with the presentation of other WADO-RS services. Suggest removing. Also the statements for dicom+xml and dicom+json are slightly different; if kept, they should be aligned. REJECTED. Important enough not to bury in the table, but a cross reference to the table is added. Made text consistent between xml and json. Changed provider to origin server for both.]*  *[Comment: McKesson: 6.5.6.1 Is it intentional that multipart is used for xml, but not for json? NO CHANGE. Yes, apparently WG 27 in its wisdom decided this; not changed by this CP. See F.2.1 Multiple Results Structure "Multiple results returned in JSON are organized as a single top-level array of JSON objects. This differs from the Native DICOM Model, which returns multiple results as a multi-part collection of singular XML documents."]*  *[Comment: McKesson: On all responses (6.5.1.2, 6.5.2.2, …) it is indicated that dcm-parameters may be included. That would allow multiple transfer syntaxes and multiple character sets to be specified, but it only makes sense for each response (or each part of a response) to specify at most one character set and one transfer syntax. A token that doesn’t allow repeats should be used instead. FIXED. Added section 6.1.3]*  *[Comment: McKesson: 6.5.6.2.2 “[transfer-syntax-uid][charset]”: this doesn’t generate the correct header. Use [ts-uid-parameter] instead; there is no appropriate token defined for a single repetition of a charset parameter. (Getting more pedantic, do we intend to require TS to come before charset?). FIXED. Replaced with [dcm-parameters]]*  *[Comment: McKesson: 6.5.6.2.2 “Where {TransferSyntaxUID} …inline binary data in the XML metadata.”: {TransferSyntaxUID} is being removed from the previous line, so we can’t reference it here. Also the JSON response doesn’t contain XML. FIXED.]*  *[Comment: McKesson: 6.6: Align JSON and XML text. Also, is it an “XML (JSON) request message” or a “request message part containing XML” (is it the whole message or a part)? FIXED. Allowing for difference between multiple XML message parts in on XML request message versus one JSON request message that has a single part that is an array.]*  *[Comment: McKesson: 6.6 “disambiguate the request”: Confirm that this is actually still true. I think the specification of default TS for each media type means that this no longer applies (see l.122). Remove this sentence. REJECTED. It is still true since there are 1:n Media Type: Transfer Syntax permutations, e.g., image/jp2 may be .90 or .91. Reworded for clarity.]*  *[Comment: McKesson: 6.6.1.1: Change “must accept” to “shall accept” throughout. FIXED.]*  *[Comment: McKesson: 6.7.1.1: Add appendix F reference. FIXED.]*  *[Comment: McKesson: The style of referencing PS 3.10 files, PS 3.19 XML, and Appendix F JSON throughout this document varies. The name of the format and the style of the reference to the definition should be standardized throughout. REJECTED. Defer to re-documention effort.]*  *[Comment. Gunter Zeilinger: 6.1.1.8.1.2 Transfer Syntax Parameters: violates RFC 6838 Media Type Specifications and Registration Procedures, which states "* *4.3. Parameter Requirements ... It is an error for a specific parameter to be specified more than once". Some implementations require a unique map indexed by parameter name (e.g., Java EE JAX-RS MediaType.getParameters()). Suggest using multiple media types in the Accept field, abd using the q parameter to specify the user agent's preference. FIXED.]*  *[Comment JFP: The capitalization of ‘transfer syntax’ seems inconsistent – sometimes capitalized and sometimes not.]*  *[Comment JFP: I think we have agreed that a media type should specify only one encoding of a representation. While I resisted this at first, I think it simplifies the exposition of (and differences between) the Accept and Content-Type header field. However, don't some code extension techniques (i.e. 2022) require or allow the specification of multiple character sets, which together create a single encoding? If this is true the "charset" media type parameter must should allow multiple values for these types of encodings.]*  *[Comment JFP: I think that dcm-parameters should be optional for URI – not forbidden. I agree that old implementations of URI should not have to support them, but new implementation should, because they will want to use the same code base to support both protocols.]* |
| Correction Wording: |

*Amend PS3.18, Section 4 Terms and Definitions:*

For the purposes of this part of DICOM, the following terms and definitions apply.

BulkDataURI A Uniform Resource Identifier in accordance with RFC3986 that identifies an octet-stream representing the value of a DICOM attribute.

Note

The octet-stream does not include the Attribute Tag, Value Representation, or Attribute Length. **~~In For~~ The value of a frame o**~~f~~ a Pixel Data attribute **~~under~~ encoded in** a compressed Transfer Syntax**~~, it~~** does **not** include the Basic Offset Table and Data Stream Fragment Item tags and lengths.

**Bulk Data Media Type A media type in which bulk data (such as Pixel Data) extracted from DICOM instances is encoded. See Section 6.1.1.8.**

**DICOM Media Type A media type in which DICOM Instances and Metadata are encoded. See Section 6.1.1.8.**

Rendered Media Type A non-DICOM media type into which DICOM instances may be transformed in order to display them using commonly available non-DICOM software, for example browsers. See [Section 6.1.1.3](#sect_6_1_1_3).

*Amend PS3.18, Section 6.1.1:*

**6.1.1 Media Types**

...

**6.1.1.2 DICOM Resource Categories**

[Table 6.1.1-1](#table_6_1_1_1) defines Resource Categories that correspond to different SOP Classes. The following sections map each Resource Category to appropriate DICOM and Rendered media types.

**Table 6.1.1-1. Resource Categories**

| **Resource Category** | **Definition** |
| --- | --- |
| Single Frame Image | This category includes all resources that:   1. are instances of a single frame SOP Class, or 2. are instances of a multi-frame SOP Class that contain only one frame, or 3. are a single frame selected from an instance of a multi-frame SOP Class. |
| Multi-Frame Image | This category includes all resources that are instances of a multi-frame SOP Class, that are not video and that contain more than one frame. |
| Video | This category includes all resources that contain more than one frame and:   1. are instances encoded in the MPEG family of transfer syntaxes (which includes MP4 and H265), or 2. are time based (motion) multi-frame images that the origin server is capable of encoding in the MPEG family. |
| Text | This category includes all resources that:   1. contain the SR Document Content Module (see [Section C.17.3 “SR Document Content Module” in PS3.3](file:///C:\\Users\\admin\\Downloads\\part03.pdf" \l "sect_C.17.3)), such as narrative text, structured reports, CAD, measurement reports, and key object selection documents, or 2. contain the Encapsulated Document Module (see [Section C.24.2 “Encapsulated Document Module” in PS3.3](file:///C:\\Users\\admin\\Downloads\\part03.pdf" \l "sect_C.24.2)). |
| Other | This category includes all resources that are not included above. |

**6.1.1.3 Rendered Media Types**

DICOM **~~resources~~** **instances** may be converted **by a rendering process** into non-DICOM media types in order to **~~render~~** **display** them using commonly available non-DICOM software, such as browsers.

For example:

1. A DICOM SOP Instance containing an image could be rendered into the image/jpeg or image/png Rendered Media Types.
2. A DICOM SOP Instance containing a multi-frame image in a lossless transfer syntax could be rendered into a video/mpeg or video/mp4 Rendered Media Type.
3. A DICOM SOP Instance containing a Structured Report could be rendered into a text/html, text/plain, or application/pdf Rendered Media Type.

Note

Rendered Media Types are usually consumer format media types. **Some of the same non-DICOM media types are also used as Bulk Data Media Types, that is, for encoding bulk data extracted from Encapsulated Pixel Data (used with compressed Transfer Syntaxes), without applying a rendering process; see Section 6.1.1.8**.

[Table 6.1.1-2](#table_6_1_1_2) specifies the meaning of media type requirement**~~s~~** **terms used** in [Table 6.1.1-3](#table_6_1_1_3) **and the tables in Section 6.1.1.8**.

**Table 6.1.1-2. Definition of Media Type Requirement Terms**

| **Requirement** | **Definition** |
| --- | --- |
| default | **The default media type shall be returned when none of the Acceptable Media Types are supported by the origin server.** The origin server shall support **~~all~~** **~~default~~** **this** media type**~~s~~**. **The origin server shall return this media type when none of the Acceptable Media Types (see 6.1.1.4) are supported.** |
| required | The origin server shall support **~~these~~** **this** media type**~~s~~**. |
| optional | The origin server may support **~~these~~** **this** media type**~~s~~**. |

**Origin servers that support URI, WS or RS services shall support rendering instances of different Resource Categories into Rendered Media Types as specified in** [Table 6.1.1-3](#table_6_1_1_3) **~~defines the Rendered Media Types by their Resource Category for the URI, WS, and RS modes~~**.

**Table 6.1.1-3. Rendered Media Types by Resource Category**

| **Category** | **Media Type** | **URI** | **WS** | **RS** |
| --- | --- | --- | --- | --- |
| Single Frame Image | image/jpeg | default | default | default |
| image/gif | optional | optional | required |
| image/png | optional | optional | required |
| image/jp2 | optional | optional | optional |
| Multi-Frame Image | image/gif | optional | optional | optional |
| Video | video/mpeg | optional | optional | optional |
| video/mp4 | optional | optional | optional |
| video/H265 | optional | optional | optional |
| Text | text/html | default | default | default |
| text/plain | required | required | required |
| text/xml | optional | optional | required |
| text/rtf | optional | optional | optional |
| application/pdf | optional | optional | optional |

When an image/jpeg media type is returned, the image shall be encoded using the JPEG baseline lossy 8 bit Huffman encoded non-hierarchical non-sequential process defined in ISO/IEC 10918-1.

Note

A DICOM encapsulated CDA resource may be returned as a text/xml media type.

The origin server may support additional rendered media types.

**A transfer syntax media type parameter is not permitted for Rendered Media Types.**

**6.1.1.4 Acceptable Media Types**

The term Acceptable Media Types denotes the media types that are acceptable to the user agent in the response. The Acceptable Media Types are those specified in:

* The <accept> query parameter, which may or may not be present.
* The Accept header field, which shall be present.
* **~~The default media type for the target resource, if any.~~**

All requests that expect a response with a payload, shall include the Accept header field. The response to a request without an Accept header field shall be 406 (Not Acceptable). Even if specific media types are provided in the <accept> query parameter, an Accept header field with one or more values shall be present, at a minimum \*/\*.

The Acceptable Media Types shall be either DICOM media-types or Rendered media types, but not both. If the Acceptable Media Types contains both DICOM and Rendered Media Types, the origin server shall return 409 (Conflict).

...

*Amend PS3.18, Section 6.1.1.7 to describe multipart payloads*

#### 6.1.1.7 Selected Media Type

The Selected Media Type is the media type selected by the origin server for the response payload. The media types in the <accept> query parameter and the media ranges in the Accept header field shall each be separately prioritized according to the rules defined in [RFC7231, Section 5.3.1].

**For multipart payloads the Selected Media Type is determined independently for each message part in the response.**

**Note:**

**The Selected Media Type of each message part depends on the Resource Category of the Instance and the Acceptable Media Types for that Resource Category.**

The Selected Media Type is chosen as follows:

1. Select the target's Resource Category
2. Select the representation with the highest priority supported media type for that category in the <accept> query parameter, which is compatible with the Accept header field.
3. If no media type in the <accept> query parameter is supported, select the highest priority supported media type for that category in the Accept header field, if any.
4. Otherwise, select the default media type for the category if the Accept header field contains a wildcard media range matching the category, if any.
5. Otherwise, return a 406 (Not Acceptable).

...

*Insert the following new section in PS3.18, after Section 6.1.1.7*

#### 6.1.1.8 DICOM Media Types and Media Types for Bulk Data

This section defines the media types used to represent DICOM Instances and bulk data. It describes:

* The media type and transfer syntax parameter for DICOM PS3.10 Files
* The media types that can be used for the bulk data of single and multi-frame images and video extracted from Instances.
* The syntax of DICOM Media Types including their transfer syntax and character set parameters.
* The query parameter for transfer syntax.
* The meaning of Acceptable Transfer Syntaxes and Selected Transfer Syntax.
* The media types supported by each service.

The media types defined in this section are distinct from those into which DICOM Instances may be rendered (which are defined in Section 6.1.1.3); some of the same media types are used for both rendered content and bulk data.

Depending on the service, the media types may be single part or multipart, and may have required or optional transfer syntax and/or character set parameters.

Table 6.1.1.8-1a, Table 6.1.1.8-1b, Table 6.1.1.8-1c and Table 6.1.1.8-1d specify the media types used to encode different representations of DICOM Instances for the URI, WS, and RS services. These media types apply to all Resource Categories and have default encodings for images and video data elements contained in the Instances.

The definitions of media type requirements are provided in Table 6.1.1-2.

6.1.1.8-1a: Media Types for DICOM PS3.10 Files

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Media Type | Descriptions | URI | WS | RS |
| application/dicom | Encodes Composite SOP Instances in the DICOM File Format defined in PS3.10, Section 7. | See Table 6.1.1.8-2 | See Table 6.1.1.8-2 | See Table 6.1.1.8-2 |

6.1.1.8-1b: Media Types for DICOM Metadata

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Media Type | Descriptions | URI | WS | RS |
| application/dicom+xml | Encodes Composite SOP Instances as XML Infosets defined in the Native Dicom Model defined in PS3.19. | not applicable | required | required |
| application/dicom+json | Encodes Composite SOP Instances in the JSON format defined in Annex F. | not applicable | not applicable | required |

6.1.1.8-1c: Media Types for DICOM Uncompressed Bulk Data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Media Type | Descriptions | URI | WS | RS |
| application/octet-stream | Encodes a Bulkdata object as a stream of uncompressed bytes, in little endian byte order.  Note: This is the same encoding defined in PS3.19 for the returned value of the getData() call for uncompressed Bulk Data. | not applicable | not applicable | See Table 6.1.1.8-3a |

6.1.1.8-1d: Media Types for DICOM Compressed Bulk Data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Media Type | Descriptions | URI | WS | RS |
| image/\*  video/\* | Encodes Bulkdata values, which in the case of compressed Pixel Data for WADO-RS services, will have each frame encoded as a separate part of a multipart response and identified by an appropriate Content-Typeheader.  Note: This is not the same encoding defined in PS3.19 for the returned value of the getData() call for compressed Pixel Data, which will contain the entire payload of the Pixel Data element encoded in Encapsulated Format as defined in PS3.5 (i.e., as a Sequence of Fragments). | not applicable | not applicable | See Table 6.1.1.8-3b |

Table 6.1.1.8-2 specifies, by Resource Category (see Table 6.1.1-1), the application/dicom media type for PS3.10 Files, along with the default and allowed Transfer Syntax UID combinations for each resource category for the URI, WS and RS services. The default media type for the Resource Category shall be returned when the origin server supports none of the Acceptable Media Types.

If no <transfer-syntax> parameter is specified for the media type for PS3.10 Files (application/dicom) then the Explicit VR Little Endian Transfer Syntax "1.2.840.10008.1.2.1" shall be used.

Note:

This is different from the Default Transfer Syntax defined in PS3.5 Section 10.1, which is Implicit VR Little Endian.

**Table 6.1.1.8-2: Transfer Syntax UIDs for 'application/dicom' Media Type  
Instances in the Image or Video Resource Categories**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Category | Transfer Syntax UID | Transfer Syntax Name | URI | WS | RS |
| Single Frame Image | 1.2.840.10008.1.2.1 | Explicit VR Little Endian | default | default | default |
| 1.2.840.10008.1.2.4.70 | JPEG Lossless, Non-Hierarchical, First-Order Prediction  (Process 14 [Selection Value 1]): Default Transfer Syntax for Lossless JPEG Image Compression | optional | optional | optional |
| 1.2.840.10008.1.2.4.50 | JPEG Baseline (Process 1):  Default Transfer Syntax for Lossy JPEG 8 Bit Image Compression | optional | optional | optional |
| 1.2.840.10008.1.2.4.51 | JPEG Extended (Process 2 & 4): Default Transfer Syntax for Lossy JPEG 12 Bit Image Compression (Process 4 only) | optional | optional | optional |
| 1.2.840.10008.1.2.4.57 | JPEG Lossless, Non-Hierarchical (Process 14) | optional | optional | optional |
| 1.2.840.10008.1.2.5 | RLE Lossless | optional | optional | optional |
| 1.2.840.10008.1.2.4.80 | JPEG-LS Lossless Image Compression | optional | optional | optional |
| 1.2.840.10008.1.2.4.81 | JPEG-LS Lossy (Near-Lossless) Image Compression | optional | optional | optional |
| 1.2.840.10008.1.2.4.90 | JPEG 2000 Image Compression (Lossless Only) | optional | optional | optional |
| 1.2.840.10008.1.2.4.91 | JPEG 2000 Image Compression | optional | optional | optional |
| 1.2.840.10008.1.2.4.92 | JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only) | optional | optional | optional |
| 1.2.840.10008.1.2.4.93 | JPEG 2000 Part 2 Multi-component Image Compression | optional | optional | optional |
| Multi-  Frame Image | 1.2.840.10008.1.2.1 | Explicit VR Little Endian | default | default | default |
| 1.2.840.10008.1.2.4.90 | JPEG 2000 Image Compression (Lossless Only) | optional | optional | optional |
| 1.2.840.10008.1.2.4.91 | JPEG 2000 Image Compression | optional | optional | optional |
| 1.2.840.10008.1.2.4.92 | JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only) | optional | optional | optional |
| 1.2.840.10008.1.2.4.93 | JPEG 2000 Part 2 Multi-component Image Compression | optional | optional | optional |
| Video | 1.2.840.10008.1.2.1 | Explicit VR Little Endian | default | default | default |
| 1.2.840.10008.1.2.4.100 | MPEG2 Main Profile @ Main Level | optional | optional | optional |
| 1.2.840.10008.1.2.4.101 | MPEG2 Main Profile @ High Level | optional | optional | optional |
| 1.2.840.10008.1.2.4.102 | MPEG-4 AVC/H.264 High Profile / Level 4.1 | optional | optional | optional |
| 1.2.840.10008.1.2.4.103 | MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1 | optional | optional | optional |
| 1.2.840.10008.1.2.4.104 | MPEG-4 AVC/H.264 High Profile / Level 4.2 For 2D Video | optional | optional | optional |
| 1.2.840.10008.1.2.4.105 | MPEG-4 AVC/H.264 High Profile / Level 4.2 For 3D Video | optional | optional | optional |
| 1.2.840.10008.1.2.4.106 | MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2 | optional | optional | optional |

Table 6.1.1.8-3a and Table 6.1.1.8-3b specify, by Resource Category (see Table 6.1.1-1), the various media types for bulk data, along with the default and allowed media types and Transfer Syntax UID combinations for each resource category for the WS and RS services.

Note:

No entries are specified for the URI or WS services, since they do not support separate retrieval of bulk data.

These media types can be used to retrieve image or video bulk data encoded in a specific Transfer Syntax.

**Table 6.1.1.8-3a: Media Types and Transfer Syntax UIDs**

**for Uncompressed Pixel Data in Bulk Data Values**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Category | Media Type | Transfer Syntax UID | Transfer Syntax Name | RS |
| Single Frame Image | application/octet-stream | 1.2.840.10008.1.2.1 | Explicit VR Little Endian | default |
| Multi-Frame Image | application/octet-stream | 1.2.840.10008.1.2.1 | Explicit VR Little Endian | default |
| Video | application/octet-stream | 1.2.840.10008.1.2.1 | Explicit VR Little Endian | default |

**Table 6.1.1.8-3b: Media Types and Transfer Syntax UIDs  
for Compressed Pixel Data in Bulk Data Values**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Category | Media Type | Transfer Syntax UID | Transfer Syntax Name | RS |
| Single Frame Image | image/jpeg | 1.2.840.10008.1.2.4.70 | JPEG Lossless, Non-Hierarchical, First-Order Prediction  (Process 14 [Selection Value 1]): Default Transfer Syntax for Lossless JPEG Image Compression | default |
| 1.2.840.10008.1.2.4.50 | JPEG Baseline (Process 1):  Default Transfer Syntax for Lossy JPEG 8 Bit Image Compression | optional |
| 1.2.840.10008.1.2.4.51 | JPEG Extended (Process 2 & 4): Default Transfer Syntax for Lossy JPEG 12 Bit Image Compression (Process 4 only) | optional |
| 1.2.840.10008.1.2.4.57 | JPEG Lossless, Non-Hierarchical (Process 14) | optional |
| image/x-dicom-rle | 1.2.840.10008.1.2.5 | RLE Lossless | default |
| image/x-jls | 1.2.840.10008.1.2.4.80 | JPEG-LS Lossless Image Compression | default |
| 1.2.840.10008.1.2.4.81 | JPEG-LS Lossy (Near-Lossless) Image Compression | optional |
| image/jp2 | 1.2.840.10008.1.2.4.90 | JPEG 2000 Image Compression (Lossless Only) | default |
| 1.2.840.10008.1.2.4.91 | JPEG 2000 Image Compression | optional |
| image/jpx | 1.2.840.10008.1.2.4.92 | JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only) | default |
| 1.2.840.10008.1.2.4.93 | JPEG 2000 Part 2 Multi-component Image Compression | optional |
| Multi-  Frame Image | image/jpeg | 1.2.840.10008.1.2.4.70 | JPEG Lossless, Non-Hierarchical, First-Order Prediction  (Process 14 [Selection Value 1]): Default Transfer Syntax for Lossless JPEG Image Compression | default |
| 1.2.840.10008.1.2.4.50 | JPEG Baseline (Process 1):  Default Transfer Syntax for Lossy JPEG 8 Bit Image Compression | optional |
| 1.2.840.10008.1.2.4.51 | JPEG Extended (Process 2 & 4): Default Transfer Syntax for Lossy JPEG 12 Bit Image Compression (Process 4 only) | optional |
| 1.2.840.10008.1.2.4.57 | JPEG Lossless, Non-Hierarchical (Process 14) | optional |
| image/x-dicom+rle | 1.2.840.10008.1.2.5 | RLE Lossless | default |
| image/x-jls | 1.2.840.10008.1.2.4.80 | JPEG-LS Lossless Image Compression | default |
| 1.2.840.10008.1.2.4.81 | JPEG-LS Lossy (Near-Lossless) Image Compression | optional |
| image/jp2 | 1.2.840.10008.1.2.4.90 | JPEG 2000 Image Compression (Lossless Only) | default |
| 1.2.840.10008.1.2.4.91 | JPEG 2000 Image Compression | optional |
| image/jpx | 1.2.840.10008.1.2.4.92 | JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only) | default |
| 1.2.840.10008.1.2.4.93 | JPEG 2000 Part 2 Multi-component Image Compression | optional |
| Video | video/mpeg2 | 1.2.840.10008.1.2.4.100 | MPEG2 Main Profile @ Main Level | optional |
| 1.2.840.10008.1.2.4.101 | MPEG2 Main Profile @ High Level | default |
| video/mp4 | 1.2.840.10008.1.2.4.102 | MPEG-4 AVC/H.264 High Profile / Level 4.1 | default |
| 1.2.840.10008.1.2.4.103 | MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1 | optional |
| 1.2.840.10008.1.2.4.104 | MPEG-4 AVC/H.264 High Profile / Level 4.2 For 2D Video | optional |
| 1.2.840.10008.1.2.4.105 | MPEG-4 AVC/H.264 High Profile / Level 4.2 For 3D Video | optional |
| 1.2.840.10008.1.2.4.106 | MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2 | optional |

The Implicit VR Little Endian (1.2.840.10008.1.2), and Explicit VR Big Endian (1.2.840.10008.1.2.2) transfer syntaxes shall not be used with Web Services.

If a transfer syntax parameter for a DICOM Media Type is not specified in a request or response, the Transfer Syntax in the response shall the Transfer Syntax specified as the default for the Resource Category and media type combination in Table 6.1.1.8-3a or Table 6.1.1.8-3b.

The origin server may support additional Transfer Syntaxes.

Note

1. The compressed bulk data of each part of a multipart payload contains only the compressed bit stream and not the DICOM PS3.5 Encapsulated Sequence or Delimiter Items.
2. For the media type image/dicom+jpeg Transfer Syntaxes, the image may or may not include the JFIF marker segment. See PS3.5 Section 8.2.1.
3. For the media type image/dicom+jp2 and image/dicom+jpx Transfer Syntaxes, the image does not include the jp2 marker segment. See PS3.5 Section 8.2.4 and A.4.4.

4. The resource on the origin server may have been encoded in the Deflated Explicit VR Little Endian (1.2.840.10008.1.2.1.99) transfer syntax. If so, the origin server may inflate it, and then convert it into an Acceptable Transfer Syntax. Alternatively, if the user-agent allowed a Content-Encoding header field of 'deflate', then the deflated bytes may be transferred unaltered, but the transfer syntax parameter in the response should be the Explicit VR Little Endian transfer syntax.

5. Compressed multi-frame image bulk data is encoded as one frame per part. E.g., each frame of a JPEG 2000 multi-frame image will be encoded as a separate part with an image/jp2 media type, rather than as a single part with a video/mj2 (RFC 3745) or uncompressed application/octet-stream media type.

6. Video bulk data is encoded as a single part containing all frames. E.g., all frames of an MPEG-4 video will be encoded as a single part with a video/mp4 (RFC 4337) media type.

7. Many of the media types used for compressed Pixel Data transferred as bulk data values are also used for consumer format media types. The browser may not be able to display the encoded data directly, even though some of the same media types are also used for encoding rendered Pixel Data; see Section 6.1.1.3.

E.g., the media type for bulk data values of lossless 16-bit JPEG 10918-1 encoded Pixel Data is "image/jpeg", the same as might be used for 8-bit JPEG 10918-1 encoded Pixel Data, whether extracted as bulk data, or rendered. The transfer syntax parameter of the Content-Type header field is useful to signal the difference.

##### 6.1.1.8.1 DICOM Media Type Syntax

The syntax of DICOM Media Types is:

dicom-media-type = (dcm-singlepart / dcm-multipart) [dcm-parameters]

Where

dcm-singlepart = dcm-mt-name

dcm-multipart ; see Section 6.1.1.8.1.1.

dcm-parameters = transfer-syntax-parameter ; see Section 6.1.1.8.1.2.

/ charset-parameter ; see Section 6.1.1.8.1.3.

dcm-mt-name = dicom / dicom-xml / dicom-json ; DICOM Media Type name

dicom = "application/dicom"

dicom-xml = "application/dicom**~~-~~+**xml"

dicom-json = "application/dicom+json"

octet-stream = "application/octet-stream"

All DICOM Media Types may have transfer syntax parameter, but its use may be constrained by the defining.

All DICOM Media Types may have a character set parameter, but its use may be constrained by the service for which they are used.

###### 6.1.1.8.1.1 DICOM Multipart Media Types

The syntax of multipart media types is:

dcm-multipart = “multipart/related”

OWS “;” OWS “type" "=” dcm-mp-mt-name

OWS “;” OWS “boundary=” boundary

[dcm-parameters]

[related-parameters]

Where

dcm-mp-mt-name = dicom / dicom-xml / dicom-json / octet-stream

See Section 6.1.1.1 for the definition of <boundary> and <multipart-related>, and see Section 6.1.1.8.2 for the definition of dcm-parameters.

Each multipart media type shall include a “type” parameter that defines the media type of the parts, and shall also include a “boundary” parameter that specifies the <boundary> string that is used to separate the parts.

###### 6.1.1.8.1.2 Transfer Syntax Parameter

All DICOM Media Types may have a single transfer syntax parameter with a single value, but their use may be constrained by the defining service.

Support for transfer syntax parameter is optional for URI and WS Services responses and forbidden in requests.

RS origin servers shall support transfer syntax parameter.

Origin servers that support transfer syntax parameters shall specify in their conformance statement those values of the transfer syntax parameter that are supported in the response.

User agents that support transfer syntax parameters shall specify in their conformance statement those transfer syntax parameter values that may be supplied in the request.

The syntax is:

transfer-syntax-mtp = OWS ";" OWS $s"transfer-syntax=" ts-value

ts-value = transfer-syntax-uid / "\*"

transfer-syntax-uid ; a UID from PS3.6 Table A-1 with a UID Type of Transfer Syntax

The value of the transfer syntax parameter may be either a Transfer Syntax UID or the token “\*”.

For example, to specify that 1.2.840.10008.1.2.4.50 is the acceptable Transfer Syntaxes, an Accept header field could be:

Accept: application/dicom; transfer-syntax=1.2.840.10008.1.2.4.50

There may only be one transfer syntax parameter it shall only have a single value.

Note: Per RFC 6838 Media Type Specifications and Registration Procedures, it is an error for a specific parameter to be specified more than once. If a choice of Transfer Syntaxes is acceptable. more than one media type may be provided in the Accept header with different q parameter values to indicate preference. For example, to specify that both 1.2.840.10008.1.2.4.50 and 1.2.840.10008.1.2.4.57 are acceptable, but 1.2.840.10008.1.2.4.50 is preferred, an Accept header field could be:

Accept: multipart/related; application/dicom; transfer-syntax=1.2.840.10008.1.2.4.50, application/dicom; transfer-syntax=1.2.840.10008.1.2.4.57;q=0.5

The token “\*” indicates that the user agent will accept any Transfer Syntax. This allows, for example, the origin server to respond without needing to transcode an existing representation to a new Transfer Syntax, or to respond with the Explicit VR Little Endian Transfer Syntax regardless of the Transfer Syntax stored.

###### 6.1.1.8.1.3 Character Set Parameter

The DICOM Media Type character set parameter is used to specify Acceptable Character Sets for the response. A DICOM Media Type may have a single character set parameter.

The syntax is:

charset-parameter = OWS ";" OWS %s"charset" "=" charset

All DICOM Media Types shall have a Default Character Set of UTF-8.

See Section 6.1.2 for character set details.

##### 6.1.1.8.2 Transfer Syntax Query Parameter

The transfer syntax query parameter specifies a comma-separated list of one or more Transfer Syntax UIDs, as defined in PS3.6. It is optional.

The syntax is:

transfer-syntax-qp = ts-parameter-name "=" (1#transfer-syntax-uid / "\*")

ts-parameter-name = %s quoted-string

The URI service defines the ts-parameter-name to be “transferSyntax”, which is case-sensitive.

The RS service uses the transfer syntax parameter in the "accept" query parameter (see 6.1.1.5) and ts-parameter-name is not supported.

##### 6.1.1.8.3 Acceptable Transfer Syntaxes

Each media type in the Acceptable Media Types has an associated set of Acceptable Transfer Syntaxes.

The Acceptable Transfer Syntaxes for a media type can be specified in any of the following ways, depending on the service:

1. The "transfer-syntax" parameters to a media type contained in the <accept> query parameter.
2. The value(s) contained in the transfer-syntax query parameter (see Section 6.1.1.8.4)
3. The "transfer-syntax" media type parameter contained in a media type in the Accept header field.

##### 6.1.1.8.4 Selected Transfer Syntax

The Selected Transfer Syntax is the transfer syntax selected by the origin server to encode a single message part in the response.

The origin server shall first determine the Selected Media Type as defined in Section 6.1.1.7 and then determine the Selected Transfer Syntax.

If the Selected Media Type was contained in the <accept> query parameter, then the Selected Transfer Syntax is determined as follows:

1. Select the value of the transfer syntax parameter of the Selected Media Type, if any;
2. Otherwise, select the value of the transfer syntax in the transfer-syntax query parameter value for the Selected Media Type, if any;
3. Otherwise select the default transfer syntax for the Selected Media Type

If the Selected Media Type was contained in the Accept header field, then the Selected Transfer Syntax is determined as follows:

1. Select the transfer syntax parameter for the Selected Media Type, if any;
2. Otherwise, select the default transfer syntax for the Selected Media Type.

Note

1. The Selected Transfer Syntax may be different for each message part contained in a response.
2. Implementers may use a different selection algorithm as long as the result is the same.

##### 6.1.1.8.5 Support for DICOM Media Types by Service

The URI, WS, and RS APIs support the following DICOM Media Types:

uri-media-type = dicom [dcm-parameters]

ws-media-type = dicom-xml [dcm-parameters]

rs-media-types = (dcm-multipart / dicom-json) [dcm-parameters]

Support for the "transfer-syntax" and "charset" media type parameters is required for RS services, but optional for the URI and WS Services.

*Insert PS3.18, Section 6.1.3 to describe multipart payloads*

#### 6.1.3 Content-Type Header Field

The Content-Type header field specifies the media type of the payload. It should only be present when a payload is present, and any media type parameters shall specify the encoding of the corresponding message part.

In particular, a DICOM Media Type used as the value of a Content-Type header field shall have zero or one transfer syntax parameter (see Section 6.1.1.8.1.2), and zero or one charset parameter (see Section 6.1.1.8.1.3), which corresponds to the character encoding of the corresponding message part.

Content-Type: dicom-media-type +ts-uid-parameter +charset

*Modify PS3.18 Section 6.2.2.1 as follows:*

**6.2 WADO-URI Request**

...

**6.2.2 Media Types Acceptable in the Response**

**6.2.2.1 Query Parameters**

**6.2.2.1.1 Accept Query Parameter**

Specifies the Acceptable Media Types for the response payload. See [Section 6.1.1.4](#sect_6_1_1_4). The name of the parameter is "contentType", which is case-sensitive. Its syntax is:

accept = %s"contentType" "=" 1#rendered-media-type / 1#**uri-**media-type

**The WADO-URI service supports Rendered Media Types (see Section 6.1.1.3) or the uri-media-type (see Section 6.1.1.8.5).**

**Support for the "transfer-syntax" and "charset" parameters is optional for the uri-media-type.**

Note:

Since the “transfer-syntax” and “charset” media type parameters were added to Web Services long after the WADO-URI standard was created, most origin servers are likely to ignore them. WADO-URI origin servers support the “transferSyntax” and “charset” query parameters, which should be used instead.

**6.2.2.1.2 Character Set Query Parameter**

Specifies the Acceptable Character Sets for the response payload. See [Section 6.1.2.1](#sect_6_1_2_1). The name of the parameter is "charset", which is case-sensitive. Its syntax is:

character-set = %s"charset" "=" 1#(token / [DQUOTE defined-term DQUOTE])

*Update PS3.18 Section 6.3.1.3 as follows:*

**6.3 WADO-URI Response**

...

**6.3.1 Body of Single DICOM Media Subtype Part Response**

**6.3.1.1 Media Type**

The media type shall be 'application/dicom', as specified in [[RFC 3240]](#biblio_RFC_3240).

**6.3.1.2 ~~Content~~ Payload**

The body content shall be a **~~"Part 10~~** **DICOM** File**~~"~~** that includes **~~a meta-header~~** **File Meta Information** as defined in [PS3.10](file:///C:\Users\admin\Downloads\part10.pdf#PS3.10).

**6.3.1.3 Transfer Syntax**

**~~The returned DICOM object shall be encoded using one of the transfer syntaxes specified in the transfer syntax query parameter as defined in~~** [**~~Section 8.2.11~~**](#sect_8_2_11) **~~below. By default, the transfer syntax shall be "Explicit VR Little Endian".~~**

**~~Note~~**

**~~This implies that retrieved images are sent uncompressed by default.~~**

**Since the Selected Media Type is a DICOM Media Type, the representations in the response shall be encoded using the Selected Transfer Syntax. See Section 6.1.1.8.6.**

**The UID of the Selected Transfer Syntax may optionally be supplied as a media type parameter in the Content-Type header field to convey the Transfer Syntax used to encode the data set in the PS3.10 File Format in the response.**

*Insert PS3.18 Section 6.4.5 as follows:*

**6.4 WADO-WS Request/Response**

...

**6.4.5 DICOM Media Type**

The WADO-WS service supports the ws-media-type. See Section **~~6.1.1.8.2~~** **6.1.1.8.5**

Support for the "transfer-syntax" and "charset" parameters is optional for the WADO-WS Service.

*Update PS3.18 Section 6.5 as follows:*

**6.5 WADO-RS Request/Response**

The DICOM RESTful Service defines several action types. An implementation shall support all the following six action types:

1. RetrieveStudy

This action retrieves the set of DICOM instances associated with a given study unique identifier (UID). The response can be DICOM or bulk data depending on the "Accept" type, and is encapsulated in a multipart MIME response.

1. RetrieveSeries

This action retrieves the set of DICOM instances associated with a given study and series UID. The response can be DICOM or bulk data depending on the "Accept" type, and is encapsulated in a multipart MIME response.

1. RetrieveInstance

This action retrieves the DICOM instance associated with the given study, series, and SOP Instance UID. The response can be DICOM or bulk data depending on the "Accept" type, and is encapsulated in a multipart MIME response.

1. RetrieveFrames

This action retrieves the DICOM frames for a given study, series, SOP Instance UID, and frame numbers. The response is pixel data, and encapsulated in a multipart MIME response.

1. RetrieveBulkdata

This action retrieves the bulk data for a given bulk data URL. **~~The response is a single bulk data item.~~**

1. RetrieveMetadata

This action retrieves the DICOM instances presented as the study, series, or instance metadata with the bulk data removed.

WADO-RS requests may contain the following query parameters:

* "accept" The <accept> query parameter is specified in [Section 6.1.1.5](#sect_6_1_1_5). The syntax is:

accept = "accept=" 1#media-type

* "charset" The <character-set> query parameter is specified in [Section 6.1.2.2](#sect_6_1_2_2). The syntax is:

character-set = "charset" = 1#charset

WADO-RS requests shall include an "Accept" header field (see [Section 6.1.1.6](#sect_6_1_1_6)) specifying the Acceptable Media Types.

WADO-RS requests may optionally support the "Accept-Charset" header field. See [Section 6.1.2.3](#sect_6_1_2_3).

**~~DICOM objects returned shall be~~** [**~~PS3.10~~**](file:///C:\Users\admin\Downloads\part10.pdf#PS3.10) **~~binary objects encoded in a requested Transfer Syntax (Explicit VR Little Endian by default) with one message part per DICOM Instance.~~**

**~~Other types of r~~R**esponses **~~will~~** **shall** be encoded in the following manner: (see [Figure 6.5-1](#figure_6_5_1)).

* **DICOM Files as defined in PS3.10, encoded in a requested Transfer Syntax (Explicit VR Little Endian by default) with one message part per DICOM Instance**
* **~~All~~** XML responses **~~shall be encoded~~** as described in the Native DICOM Model defined in [PS3.19](file:///C:\Users\admin\Downloads\part19.pdf#PS3.19) with one message part per XML object.
* **~~All~~** JSON responses **~~shall be encoded~~** as a DICOM JSON Model Object as defined in [Annex F](#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 **~~may be~~** encoded **~~in one of three ways~~as**:
* Single-frame pixel data **~~encoded~~** using a single-frame media type (one message part)
* Multi-frame pixel data **~~encoded~~** using a single-frame media type (one frame per message part)
* Multi-frame or video pixel data **~~encoded~~** using a multi-frame media type (multiple frames in one message part)

**The compressed pixel data consists of the compressed bit stream only, and shall not include any Sequence Items and Delimiters from the PS3.5 Encapsulated Pixel Data format.**

Compressed pixel data shall be encoded using the **~~following Media Type~~** **application/dicom media type and transfer syntaxes specified in Table 6.1.1.8-2**. **~~Media Types corresponding to several DICOM Transfer Syntax UIDs require a transfer-syntax parameter, as shown in~~** [**~~Table 6.5-1~~**](#table_6_5_1)**~~, to disambiguate the request.~~**

**~~Note~~**

**~~If the Transfer Syntax is not specified, then a reversible (lossless) encoding is used.~~**

**~~Table 6.5-1. Media Type Mapping to Transfer Syntax~~**

| **~~DICOM Transfer Syntax UID~~** | **~~Media Type and Parameters~~** |
| --- | --- |
| **~~Single-frame media types~~** | |
| **~~1.2.840.10008.1.2.4.50~~** | **~~image/dicom+jpeg; transfer-syntax=1.2.840.10008.1.2.4.50~~** |
| **~~1.2.840.10008.1.2.4.51~~** | **~~image/dicom+jpeg; transfer-syntax=1.2.840.10008.1.2.4.51~~** |
| **~~1.2.840.10008.1.2.4.57~~** | **~~image/dicom+jpeg; transfer-syntax=1.2.840.10008.1.2.4.57~~** |
| **~~1.2.840.10008.1.2.4.70~~** | **~~image/dicom+jpeg~~** |
| **~~1.2.840.10008.1.2.4.70~~** | **~~image/dicom+jpeg; transfer-syntax=1.2.840.10008.1.2.4.70~~** |
| **~~1.2.840.10008.1.2.5~~** | **~~image/dicom+rle~~** |
| **~~1.2.840.10008.1.2.5~~** | **~~image/dicom+rle; transfer-syntax=1.2.840.10008.1.2.5~~** |
| **~~1.2.840.10008.1.2.4.80~~** | **~~image/dicom+jpeg-ls~~** |
| **~~1.2.840.10008.1.2.4.80~~** | **~~image/dicom+jpeg-ls; transfer-syntax=1.2.840.10008.1.2.4.80~~** |
| **~~1.2.840.10008.1.2.4.81~~** | **~~image/dicom+jpeg-ls; transfer-syntax=1.2.840.10008.1.2.4.81~~** |
| **~~1.2.840.10008.1.2.4.90~~** | **~~image/dicom+jp2~~** |
| **~~1.2.840.10008.1.2.4.90~~** | **~~image/dicom+jp2; transfer-syntax=1.2.840.10008.1.2.4.90~~** |
| **~~1.2.840.10008.1.2.4.91~~** | **~~image/dicom+jp2; transfer-syntax=1.2.840.10008.1.2.4.91~~** |
| **~~1.2.840.10008.1.2.4.92~~** | **~~image/dicom+jpx~~** |
| **~~1.2.840.10008.1.2.4.92~~** | **~~image/dicom+jpx; transfer-syntax=1.2.840.10008.1.2.4.92~~** |
| **~~1.2.840.10008.1.2.4.93~~** | **~~image/dicom+jpx; transfer-syntax=1.2.840.10008.1.2.4.93~~** |
| **~~Multi-frame media types~~** | |
| **~~1.2.840.10008.1.2.4.92~~** | **~~image/dicom+jpx~~** |
| **~~1.2.840.10008.1.2.4.92~~** | **~~image/dicom+jpx; transfer-syntax=1.2.840.10008.1.2.4.92~~** |
| **~~1.2.840.10008.1.2.4.93~~** | **~~image/dicom+jpx; transfer-syntax=1.2.840.10008.1.2.4.93~~** |
| **~~1.2.840.10008.1.2.4.100~~** | **~~video/mpeg; transfer-syntax=1.2.840.10008.1.2.4.100~~** |
| **~~1.2.840.10008.1.2.4.101~~** | **~~video/mpeg; transfer-syntax=1.2.840.10008.1.2.4.101~~** |
| **~~1.2.840.10008.1.2.4.102~~** | **~~video/mp4; transfer-syntax=1.2.840.10008.1.2.4.102~~** |
| **~~1.2.840.10008.1.2.4.103~~** | **~~video/mp4; transfer-syntax=1.2.840.10008.1.2.4.103~~** |

**~~Note~~**

**~~For the media type image/dicom+jp2 Transfer Syntaxes, 1.2.840.10008.1.2.4.90 and 1.2.840.10008.1.2.4.91, the image does not include the jp2 wrapper.~~**

**~~HTTP Request field Accept is used in the header lines by the client in a HTTP protocol transaction to indicate the data responses that are acceptable from the server. HTTP Response fields~~** **The request header field** Content-Type **~~and parameters are~~ is** used **~~in the header lines by the server in a HTTP protocol transaction~~** to indicate the **media** type **~~and encoding~~** of **~~data returning to the client~~ the payload**. **~~All lines are~~** [**~~[RFC 822]~~**](#biblio_RFC_822) **~~format headers. All HTTP header fields whose use is not defined by WADO-RS are presumed to have the meaning defined by the HTTP standard.~~**

**If the origin server returns XML or JSON responses that contain bulk data references, tT**he **origin** server is required to support uncompressed bulk **~~and pixel~~** data (application/octet-stream) and must be able to deliver all bulk data in that form **(i.e., decompress it from its original form if necessary)** unless it is available only in a lossy-compressed format.

**The DICOM Media Types supported are defined in Section 6.1.1.8.5.**

**The Bulk Data Media Types supported are defined in Table 6.1.1.8-1c and Table 6.1.1.8-1d.**

**The origin server shall support the "transfer-syntax" and "charset" parameters.**

*Update PS3.18 Section 6.5.1.1 as follows:*

**6.5.1 WADO-RS - RetrieveStudy**

*…*

**6.5.1.1 Request**

The specific Services resource to be used for the RetrieveStudy action shall be as follows:

* Resource
* {SERVICE}/studies/{StudyInstanceUID}, where
* {SERVICE} is the base URL for the service. This may be a combination of protocol (either http or https), host, port, and application.
* {StudyInstanceUID} is the study instance UID for a single study.
* Method
* GET
* Headers
* Accept - A comma-separated list of representation schemes, in preference order, which will be accepted by the service in the response to this request. The types allowed for this request header are as follows:
* multipart/related; type=**"**application/dicom**";** [**~~transfer-syntax={TransferSyntaxUID}~~dcm-parameters**]

Specifies that the response can be DICOM Instances encoded in [PS3.10](file:///Z:\dicom\DICOM\WG-27\Supplements\Re-Doc%20Part%2018\CPs\In%20Development\part10.pdf#PS3.10) format. If transfer-syntax is not specified **in the dcm-parameters,** the server **~~can freely choose which Transfer Syntax to use~~** **shall use the Explicit VR Little Endian Transfer Syntax "1.2.840.10008.1.2.1"** for each Instance **(see Section 6.1.1.8)**.

* multipart/related; type=**"**application/octet-stream**" [dcm-parameters]**

Specifies that the response can be Little Endian uncompressed bulk data. **See Section 6.1.3.**

* multipart/related; type=**"**{**~~Image-media-type~~media-type**}**" [dcm-parameters]**

Specifies that the response can be pixel data encoded using **~~a {MediaType} listed in~~** [**~~Table 6.5-1~~**](#table_6_5_1) **~~(including parameters).~~ the media types and transfer syntaxes specified in Table 6.1.1.8-3b**. **See Section 6.1.3.**

Note

An example of a more complicated accept header with multiple transfer syntaxes:

User is interested in receiving JPEG2000 pixel data in lossless or compressed format but is willing to accept JPEG as well.

The Accept request would contain the following comma-separated parameters:

Accept: multipart/related**~~=~~**; **type="**image/**~~dicom+~~**jpx**"**; transfer-syntax=1.2.840.10008.1.2.4.92, multipart/related**~~=~~**; **type="**image/**~~dicom+~~**jpx**"**; transfer-syntax=1.2.840.10008.1.2.4.93, multipart/related**~~=~~**; **type="**image/**~~dicom+~~**jpeg**"**

or alternatively, multiple Accept headers:

Accept: multipart/related**~~=~~**; **type="**image/**~~dicom+~~**jpx**"**; transfer-syntax=1.2.840.10008.1.2.4.92

Accept: multipart/related**~~=~~**; **type="**image/**~~dicom+~~**jpx**"**; transfer-syntax=1.2.840.10008.1.2.4.93

Accept: multipart/related**~~=~~**; **type="**application/**~~dicom+~~**jpeg**"**

*Update PS3.18 Section 6.5.1.2 as follows:*

**6.5.1.2 Response**

...

**6.5.1.2.1 DICOM Response**

* Content-Type:
* multipart/related; type=**"**application/dicom**"**; boundary={MessageBoundary} [dcm-parameters]
* The entire multipart response contains every instance for the specified Study that can be converted to one of the requested Transfer Syntaxes.
* Each **~~item~~part** in the multipart response represents a DICOM SOP Instance with the following http headers:
* Content-Type: application/dicom**; [dcm-parameters]**

**See Section 6.1.3.**

**6.5.1.2.2 Bulk Data Response**

* Content-Type:
* multipart/related; **type="**application/octet-stream**"**; boundary={MessageBoundary}**; [dcm-parameters]**
* multipart/related; **type="**{**~~MediaType~~media-type**}**"**; boundary={MessageBoundary}**; [dcm-parameters]**

**See Section 6.1.3.**

* 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-stream6
* Content-Location: {BulkDataURL}
* a compressed bulk data element from a SOP Instance in the Study encoded in a single-frame compression **media type ~~{MediaType}~~** with the following headers:
* Content-Type: **{media-type}**
* 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: **{media-type}**
* Content-Location: {BulkDataURL}/frames/{FrameNumber}

Note

Each frame will come in a separate part.

* **~~a set~~** **all** of **the** compressed frames from a SOP Instance in the Study encoded in a **~~multi-frame~~ video** media type with the following headers:
* Content-Type: **{media-type}**
* 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 6.5.2.1 as follows:*

**6.5.2 WADO-RS - RetrieveSeries**

*…*

**6.5.2.1 Request**

*…*

* multipart/related; type=**"**application/dicom**"**; [**~~transfer-syntax={TransferSyntaxUID}~~dcm-parameters**]

Specifies that the response can be DICOM Instances encoded in [PS3.10](file:///C:\Users\admin\Downloads\part10.pdf#PS3.10) format. If transfer-syntax is not specified **in the dcm-parameters** the server **~~can freely choose which Transfer Syntax to use~~** **shall use the Explicit VR Little Endian Transfer Syntax "1.2.840.10008.1.2.1"** for each Instance **(see Section 6.1.1.8)**.

* multipart/related; type=**"**application/octet-stream**"**; **[dcm-parameters]**

Specifies that the response can be Little Endian uncompressed bulk data.

* multipart/related; type=**"~~{MediaType}~~ {media-type}"; [dcm-parameters]**

**~~Specifies that the response can be pixel data encoded using a {MediaType} listed in~~** [**~~Table 6.5-1~~**](#table_6_5_1) **~~(including parameters).~~**

**Specifies that the response can be pixel data encoded using the media types and transfer syntaxes specified in Table 6.1.1.8-3b.**

*Update PS3.18 Section 6.5.2.2 as follows:*

**6.5.2.2 Response**

...

**6.5.2.2.1 DICOM Response**

* Content-Type:
* multipart/related; type=**"**application/dicom**"**; boundary={MessageBoundary}
* The entire multipart response contains every instance for the specified Series that can be converted to one of the requested Transfer Syntaxes.
* Each **~~item~~part** in the multipart response represents a DICOM SOP Instance with the following http headers:
* Content-Type: application/dicom**; [dcm-parameters]**

**See Section 6.1.3.**

**6.5.2.2.2 Bulk Data Response**

* Content-Type:
* multipart/related; type=**"**application/octet-stream**"**; boundary={MessageBoundary}**; [dcm-parameters]**
* multipart/related; type=**"**{**~~MediaType~~media-type**}**"**; boundary={MessageBoundary}**; [dcm-parameters]**

**See Section 6.1.3.**

* The entire multipart response contains all bulk data for the specified Series 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}
* a compressed bulk data element from a SOP Instance in the Series encoded in a single-frame media type with the following headers:
* Content-Type: {**~~MediaType~~ media-type}**
* Content-Location: {BulkDataURL}
* a compressed frame from a multi-frame SOP Instance in the Series encoded in a single-frame media type with the following headers:
* Content-Type: {**~~MediaType~~** **media-type**}
* Content-Location: {BulkDataURL}/frames/{FrameNumber}
* **~~a set~~** **all** of **the** compressed frames from a multi-frame SOP Instance in the Series encoded in a **~~multi-frame~~ video** media type with the following headers:
* Content-Type:**{media-type}**
* 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 6.5.3.1 as follows:*

**6.5.3 WADO-RS - RetrieveInstance**

*…*

**6.5.3.1 Request**

*…*

* multipart/related; type=**"**application/dicom**"**; [**~~transfer-syntax={TransferSyntaxUID}~~dcm-parameters**]

Specifies that the response can be DICOM Instances encoded in [PS3.10](file:///C:\Users\admin\Downloads\part10.pdf#PS3.10) format. If transfer-syntax is not specified **in the dcm-parameters** the server **~~can freely choose which Transfer Syntax to use~~** **shall use the Explicit VR Little Endian Transfer Syntax "1.2.840.10008.1.2.1"** for each Instance **(see Section 6.1.1.8)**.

* multipart/related; type=**"**application/octet-stream**"; [dcm-parameters]**

Specifies that the response can be Little Endian uncompressed bulk data. **See Section 6.1.3.**

* multipart/related; type=**"**{**~~MediaType~~media-type**}**"; [dcm-parameters]**

**~~Specifies that the response can be pixel data encoded using a {MediaType} listed in~~** [**~~Table 6.5-1~~**](#table_6_5_1) **~~(including parameters).~~**

**Specifies that the response can be pixel data encoded using the application/dicom media type and transfer syntaxes specified in Table 6.1.1.8-2**. **See Section 6.1.3.**

*Update PS3.18 Section 6.5.3.2 as follows:*

**6.5.3.2 Response**

...

**6.5.3.2.1 DICOM Response**

* Content-Type:
* multipart/related; type=**"**application/dicom**"**; boundary={MessageBoundary}
* The multipart response contains a single **~~item~~part** representing the specified DICOM SOP Instance with the following http headers:
* Content-Type: application/dicom**; [dcm-parameters]**

**6.5.3.2.2 Bulk Data Response**

* Content-Type:
* multipart/related; type=**"**application/octet-stream**"**; boundary={MessageBoundary} **[dcm-parameters]**
* multipart/related; type=**"**{**~~MediaType~~media-type**}**"**; boundary={MessageBoundary} **[dcm-parameters]**
* The entire multipart response contains all bulk data for the specified Instance 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}
* a compressed bulk data element from a SOP Instance encoded in a single-frame media type with the following headers:
* Content-Type: **{media-type}**
* Content-Location: {BulkDataURL}
* a compressed frame from a multi-frame SOP Instance encoded in a single-frame media type with the following headers:
* Content-Type: **{media-type}**
* Content-Location: {BulkDataURL}/frames/{FrameNumber}
* **~~a set~~** **all** of **the** compressed frames from a multi-frame SOP Instance encoded in a **~~multi-frame~~ video** media type with the following headers:
* Content-Type: **{media-type}**
* 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 6.5.4.1 as follows:*

**6.5.4 WADO-RS - RetrieveFrames**

*…*

**6.5.4.1 Request**

*…*

* multipart/related; type=**"**application/octet-stream**"; [dcm-parameters]**

Specifies that the response can be Little Endian uncompressed pixel data **as specified in Table 6.1.1.8-3a.**

* multipart/related; type=**"**{**~~MediaType~~media-type**}**" [dcm-parameters]**

**~~Specifies that the response can be pixel data encoded using a {MediaType} listed in~~** [**~~Table 6.5-1~~**](#table_6_5_1) **~~(including parameters).~~**

**Specifies that the response can be pixel data encoded using the media types and transfer syntaxes specified in Table 6.1.1.8-3b**.

*Update PS3.18 Section 6.5.4.2 as follows:*

**6.5.4.2 Response**

...

**6.5.4.2.1 Pixel Data Response**

* Content-Type:
* multipart/related; type=**"**application/octet-stream**"**; boundary={MessageBoundary} **[dcm-parameters]**
* multipart/related; type=**"**{**~~MediaType~~media-type**}**"**; boundary={MessageBoundary} **[dcm-parameters]**
* The entire multipart response contains all requested Frames for the specified Instance.
* Each item in the response is one of:
* an uncompressed frame encoded in Little Endian binary format **(as specified in Table 6.1.1.8-3a)** with the following headers:
* Content-Type: application/octet-stream
* Content-Location: {BulkDataURL}[/frames/{FrameNumber}]
* a compressed frame encoded in a single-frame media type **(as specified in Table 6.1.1.8-3b)** with the following headers:
* Content-Type: **{media-type}**
* Content-Location: {BulkDataURL}/frames/{FrameNumber}
* a set of compressed frames encoded in a **~~multi-frame~~ video** media type **(as specified in Table 6.1.1.8-3b)** with the following headers:
* Content-Type: **{media-type}**
* 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.
* The frames will be returned in the order specified by the Frame List.

*Update PS3.18 Section 6.5.5.1 as follows:*

**6.5.5 WADO-RS - RetrieveBulkdata**

This action retrieves the bulk data for a given bulk data URL. **~~The response is a single bulk data item.~~**

**6.5.5.1 Request**

The specific Services resource to be used for the RetrieveBulkdata action shall be as follows:

* Resource
* {BulkDataURL}, where
* {BulkDataURL} is the URL of a bulk data element. This may be the URL attribute of a BulkData element received in response to a WADO-RS RetrieveMetadataRequest.
* **~~The server shall always return the same bulk data for a specified BulkData URL if the data is available.~~**
* **~~If the resource specified by the BulkData URL is not available, the server shall return:~~**
* **~~404 - Not Found, if the server expects to be able to return the resource again in the future~~**
* **~~410 - Gone, if the server does not expect the resource to be valid in the future~~**
* **~~The server determines the period of time a BulkData URL resource is available.~~**
* Method
* GET
* Headers
* Accept
* multipart/related; type=**"**application/octet-stream**" [dcm-parameters]**

Specifies that the response can be Little Endian uncompressed bulk data.

* multipart/related; type=**"**{**~~MediaType~~media-type**}**" [dcm-parameters]**

**~~Specifies that the response can be pixel data encoded using a {Image-media-type} listed in~~** [**~~Table 6.5-1~~**](#table_6_5_1) **~~(including parameters).~~**

**Specifies that the response can be compressed pixel data encoded using the media types and transfer syntaxes specified in Table 6.1.1.8-3b**.

* Range
* See [[RFC 7233]](#biblio_RFC_7233) [Section 3.1](http://tools.ietf.org/html/rfc7233#section-5.5). If omitted in the request the server shall return the entire bulk data object.

**6.5.5.2 Response**

The Server shall provide the document(s) indicated in the request. **~~In order to parse the bulk data items it is necessary to also retrieve the corresponding metadata for the specified Study, Series, or Instance.~~**

**The server shall always return the same bulk data for a specified BulkData URL if the data is available.**

**If the resource specified by the BulkData URL is not available, the server shall return:**

**• 404 - Not Found, if the server expects to be able to return the resource again in the future**

**• 410 - Gone, if the server does not expect the resource to be valid in the future**

**The server determines the period of time a BulkData URL resource is available.**

The Server shall return the document(s) or an error code when the document(s) cannot be returned. If the server cannot encode the pixel data using any of the requested media types, then an error status shall be returned.

All response formats have a content type of multipart/related with a message boundary separator. The response format depends on the Accept header specified in the request.

**6.5.5.2.1 Bulk Data Response**

* Content-Type:
* multipart/related; type=**"**application/octet-stream**"**; boundary={MessageBoundary} **[dcm-parameters]**
* **multipart/related; type="{media-type}"; boundary={MessageBoundary} [dcm-parameters]**

**where {media-type} is of compressed pixel data encoded as specified in Table 6.1.1.8-3b.**

* **The entire multipart response contains all bulk data that can be converted to one of the requested media types.**
* **~~The single item~~ Each part 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 **[dcm-parameters]**
* Content-Location: {BulkDataURL}
* a compressed bulk data elementfrom a SOP Instance encoded in a single-frame media type with the following headers:
* Content-Type:{**~~MediaType~~media-type**} **[dcm-parameters]**

**where {media-type} is of compressed pixel data encoded as specified in Table 6.1.1.8-3b.**

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

**Note**

**Each frame will come in a separate part.**

* **all of the compressed frames from a SOP Instance encoded in a video media type with the following headers:**
* **Content-Type: {media-type}**
* **Content-Location: {BulkDataURL}**
* If the Range header is specified in the request, the server shall return only the specified bytes of the bulk data object. See [[RFC 7233]](#biblio_RFC_7233) [Section 4](http://tools.ietf.org/html/rfc7233#section-4).

*Update PS3.18 Section 6.5.6.1 as follows:*

**6.5.6 WADO-RS - RetrieveMetadata**

...

**6.5.6.1 Request**

...

* Headers
* Accept
* multipart/related; type=**"**application/dicom+xml**"**

Specifies that the response should be [PS3.19](file:///C:\Users\admin\Downloads\part19.pdf#PS3.19) XML. **~~All~~** WADO-RS **~~providers~~** **~~must~~** **origin servers** **shall** support this media type **(see Table 6.1.1.8-1b)**.

* application/**dicom+**json

Specifies that the **~~results~~** **response** should be DICOM JSON (see [Annex F](#chapter_F)). **~~A~~** WADO-RS **~~provider~~** **~~optionally~~** **origin servers** **shall** support this media type **(see Table 6.1.1.8-1b).**

*Update PS3.18 Section 6.5.6.2 as follows:*

**6.5.6.2 Response**

....

The response has a content type of either:

* multipart/related; type=**"**application/dicom+xml**"**, as described in the Native DICOM Model defined in [PS3.19](file:///C:\Users\admin\Downloads\part19.pdf#PS3.19), or
* application/**dicom+**json, as described in [Annex F](#chapter_F).

*Update PS3.18 Section 6.5.6.2.2 as follows:*

**6.5.6.2.2 JSON Metadata Response**

* Content-Type:
* application/**dicom+**json **~~transfer-syntax={TransferSyntaxUID}~~ [dcm-parameters]**

**~~Where {TransferSyntaxUID} is the UID of the DICOM Transfer Syntax used to encode the inline binary data in the XMLmetadata.~~**

**Specifies that the response can be DICOM Metadata encoded in [the](file:///C:\\Users\\admin\\Downloads\\part10.pdf" \l "PS3.10) ‘application/dicom+json’ media type. If transfer-syntax is not specified in the dcm-parameters, the server shall use the Explicit VR Little Endian Transfer Syntax "1.2.840.10008.1.2.1" for each Instance (see Section 6.1.1.8).**

* The response is a JSON array that contains all metadata for the specified Study.
* Each element in the array is the DICOM JSON encoded metadata for an Instance (see [Annex F](#chapter_F)).

*Update PS3.18 Section 6.5.8 as follows:*

**6.5.8 WADO-RS - Retrieve Rendered Transaction**

**~~The Retrieve Rendered transaction~~** **This action** retrieves DICOM instances rendered as: images, text-based documents, or other appropriate representations depending on the target resource.

Its primary use case is to provide user agents with a simple interface for displaying medical images and related documents, without requiring deep knowledge of DICOM data structures and encodings. It is similar to the Retrieve DICOM service in that it uses the same method, resources, header fields and status codes. The **~~primarily~~** **primary** difference**s** **~~is~~are** the additional **resource component and the** query parameters **~~and media types supported~~**.

The origin server shall document the Composite SOP classes that it supports for this transaction in the Conformance Statement and in the response to the Retrieve Capabilities request, and shall be able to render all valid instances for which conformance is claimed, e.g., all photometric interpretations that are defined in the IOD for the SOP class.

**~~If the origin server supports this transaction, it shall also support the Retrieve DICOM transaction (WADO-RS).~~**

**6.5.8.1 Request**

The Retrieve Rendered service has the following request message syntax:

GET SP /{+resource}{?parameter\*} SP version CRLF  
 Accept: 1#rendered-media-type CRLF  
 \*(header-field CRLF)  
 CRLF

Where

|  |  |
| --- | --- |
| {+resource} | References a **~~non-Presentation State~~** resource. |
| {?parameter\*} | Zero or more query parameters as defined in [Section 6.5.8.1.2](#sect_6_5_8_1_2). |
| version | HTTP version = "HTTP/1.1" |
| 1#rendered-media-type | One or more Rendered Media Types See [Section 6.1.1.3](#sect_6_1_1_3). |

**6.5.8.1.1 Target Resources**

[Table 6.5.8-1](#table_6_5_8_1) shows the resources supported by the Retrieve Rendered transaction along with their associated URI templates.

**Table 6.5.8-1. Resources, Templates and Description**

| **Target Resource** | **Resource URI Template** |
| --- | --- |
| Study | /studies/{study\_uid}**/rendered**  Retrieves a study in acceptable Rendered Media Types. |
| Series | /studies/{study\_uid}/series/{series\_uid}**/rendered**  Retrieves a series in an acceptable Rendered Media Type. |
| Instance | /studies/{study\_uid}/series/{series\_uid}/instances/{instance\_uid}**/rendered**  Retrieves an instance in an acceptable Rendered Media Type. |
| Frames | /studies/{study\_uid}/series/{series\_uid}/instances/{instance\_uid}/frames/{frame\_list}**/rendered**  Retrieves one or more frames in an acceptable **R~~r~~**endered **M~~m~~**edia **T~~t~~**ype. |

...

*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:///Z:\dicom\DICOM\WG-27\Supplements\Re-Doc%20Part%2018\CPs\In%20Development\part10.pdf#PS3.10) binary instances, or metadata and bulk data.

[PS3.10](file:///Z:\dicom\DICOM\WG-27\Supplements\Re-Doc%20Part%2018\CPs\In%20Development\part10.pdf#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](#figure_6_5_1)):

* All XML request messages shall be encoded as described in the Native DICOM Model defined in [PS3.19](file:///Z:\dicom\DICOM\WG-27\Supplements\Re-Doc%20Part%2018\CPs\In%20Development\part19.pdf#PS3.19) with one message part per XML object.
* All JSON request **message**s shall be encoded as an array of DICOM JSON Model Objects defined in [Annex F](#chapter_F) **in a single message part**.
* 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)

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~~** **using the media types and transfer syntaxes specified in Table 6.1.1.8-3b.**

**~~.~~** Media **~~T~~t**ypes corresponding to several DICOM Transfer Syntax UIDs may require a transfer-syntax parameter to **~~disambiguate the request~~convey the Transfer Syntax the compressed pixel data is encoded in**.

**~~HTTP Request~~** **The request header** field Content-Type is used **~~in the header lines by the client in an HTTP/1.1 transaction~~** to indicate the **media** type of **~~data being sent to the Service~~ the payload. ~~All lines are RFC822 or RFC7230 format headers. All HTTP header fields whose use is not defined by STOW-RS shall have the meaning defined by the HTTP standard.~~**

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

**6.6.1 STOW-RS - Store Instances**

...

**6.6.1.1 Request**

...

* 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](file:///C:\Users\admin\Downloads\part10.pdf#PS3.10) binary instances. All STOW-RS providers **~~must~~** **shall** accept this Content-Type.

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

Specifies that the post is [PS3.19](file:///C:\Users\admin\Downloads\part19.pdf#PS3.19) XML metadata and bulk data. All STOW-RS providers **~~must~~** **shall** accept this Content-Type.

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

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

...

**6.6.1.1.1 DICOM Request Message Body**

The DICOM Request Message has a multipart body.

* Content-Type:
* multipart/related; type=application/dicom; boundary={MessageBoundary}
* The multipart request body contains every instance to be stored. Each instance is in a separate part of the multipart body.
* Each part in the multipart body represents a DICOM SOP Instance with the following HTTP headers:
* Content-Type: application/dicom

**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}

...

**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/**dicom+**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/**dicom+**json; transfer-syntax={TransferSyntaxUID}

...

*Update PS3.18 Section 6.7.1.1 as follows:*

**6.7.1 QIDO-RS - Search**

**6.7.1.1 Request**

...

* Headers
* Accept - The media type of the query results. The types allowed for this request header are:
* multipart/related; type=**"**application/dicom+xml**"** **~~(default)~~**

Specifies that the results should be DICOM [PS3.19](file:///C:\Users\admin\Downloads\part19.pdf#PS3.19) XML (one part per result)

* application/**dicom+**json **(default)**

Specifies that the results should be DICOM JSON **as defined in** [**Annex F**](#chapter_F) **(the one and only part contains all results)**

A QIDO-RS provider shall support both Accept header values.

*Update PS3.18 Section 6.7.1.2.3 as follows:*

**6.7.1.2.3 Query Result Messages**

....

**6.7.1.2.3.1 XML Results**

* Content-Type: multipart/related; type=**"**application/dicom+xml**"**

...

**6.7.1.2.3.2 JSON Results**

* Content-Type: application/**dicom+**json

*Update PS3.18 Section 6.8.1.1.2 as follows:*

**6.8.1.1.2 Header Fields**

The Retrieve Server Options Service request messages can include the following header fields:

* Accept:
* application/vnd.sun.wadl+xml
* application/**dicom+**json

*Update PS3.18 Section 6.8.1.2.2.3 as follows:*

**6.8.1.2.2.3 Search Methods**

…

Example:

<method name="GET" id="SearchForStudies">  
 <request>  
 <param name="Accept" style="header" default="multipart/related; type=application/dicom+xml">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="fuzzymatching" style="query" />  
 <param name="StudyDate" style="query" />  
 <param name="00080020" style="query" />  
 <param name="StudyTime" style="query" />  
 <param name="00080030" style="query" />  
 …  
 <param name="includefield" style="query" repeating="true" />  
 <option value="all" />  
 <option value="00081049" />  
 <option value="PhysiciansOfRecordIdentificationSequence" />  
 <option value="00081060" />  
 <option value="NameOfPhysiciansReadingStudy" />  
 …  
 </param>  
</request>  
<response status="200">  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
</response>  
<response status="400 401 403 413 503" />  
</method>

*Update PS3.18 Section 6.8.1.2.2.4 as follows:*

**6.8.1.2.2.4 Update Methods**

…

Example:

<method name="POST" id="UpdateUPS">  
 <request>  
 <representation mediaType="application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
 </request>  
 <response status="200">  
 <param name="Warning" style="header" fixed="299 {+SERVICE}: The UPS was created with modifications." />  
 <param name="Warning" style="header" fixed="299 {+SERVICE}: Requested optional Attributes are not supported." />  
 </response>  
 <response status="409">  
 <param name="Warning" style="header" fixed="299 {+SERVICE}: The Transaction UID is missing." />  
 <param name="Warning" style="header" fixed="299 {+SERVICE}: The Transaction UID is incorrect." />  
 <param name="Warning" style="header" fixed="299 {+SERVICE}: The submitted request is inconsistent  
 with the current state of the UPS Instance." />  
 </response>  
 <response status="400 401 403 404 503" />  
</method>

*Update PS3.18 Section 6.9.1.1 as follows:*

**6.9.1.1 Request**

The request message shall be formed as follows:

* Resource
* {+SERVICE}/workitems{?AffectedSOPInstanceUID}

where

* {+SERVICE} is the base URL for the service. This may be a combination of protocol (either HTTP or HTTPS), authority and path.
* {AffectedSOPInstanceUID} specifies the SOP Instance UID of the UPS Instance to be created
* Method
* POST
* Headers
* Content-Type - The representation scheme being posted to the RESTful service. The types allowed for this request header are as follows:
* application/dicom+xml

Specifies that the post is DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML metadata. See [Section 6.9.1.1.1](#sect_6_9_1_1_1).

* application/**dicom+**json

Specifies that the post is DICOM [PS3.18](#PS3_18) JSON metadata. See [Section 6.9.1.1.1](#sect_6_9_1_1_1).

* The request body shall convey a single Unified Procedure Step Instance. The instance shall comply with all requirements in the Req. Type N-CREATE column of [Table CC.2.5-3 in PS3.4](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part04.pdf#table_CC.2.5-3).

**6.9.1.1.1 Request Message**

The Request Message has a single part body.

* Content-Type:
* application/dicom+xml
* application/**dicom+**json
* The request body contains all attributes to be stored in either DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML or DICOM JSON. Any binary data contained in the message shall be inline.

*Update PS3.18 Section 6.9.2.1 as follows:*

**6.9.2.1 Request**

The request message shall be formed as follows:

* Resource
* {+SERVICE}/workitems/{UPSInstanceUID}{?transaction}

where

* {+SERVICE} is the base URL for the service. This may be a combination of protocol (either HTTP or HTTPS), authority and path.
* {UPSInstanceUID} is the UID of the Unified Procedure Step Instance
* {transaction} specifies the Transaction UID / Locking UID for the specified Unified Procedure Step Instance

If the UPS instance is currently in the SCHEDULED state, {transaction} shall not be specified.

If the UPS instance is currently in the IN PROGRESS state, {transaction} shall be specified.

* Method
* POST
* Headers
* Content-Type - The representation scheme being posted to the RESTful service. The types allowed for this request header are as follows:
* application/dicom+xml

Specifies that the post is DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML metadata. See [Section 6.9.2.1.1](#sect_6_9_2_1_1).

* application/**dicom+**json

Specifies that the post is DICOM [PS3.18](#PS3_18) JSON metadata. See [Section 6.9.2.1.1](#sect_6_9_2_1_1).

* The request body describes changes to a single Unified Procedure Step Instance. It shall include all Attributes for which Attribute Values are to be set. The changes shall comply with all requirements described in [Section CC.2.6.2 in PS3.4](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part04.pdf#sect_CC.2.6.2).
* Because the request will be treated as atomic (indivisible) and idempotent (repeat executions have no additional effect), all changes contained in the request shall leave the UPS instance in an internally consistent state.

**6.9.2.1.1 Request Message**

The Request Message has a single part body.

* Content-Type:
* application/dicom+xml
* application/**dicom+**json
* The request body contains all the attributes to be updated in either DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML or DICOM [PS3.18](#PS3_18) JSON. Any binary data contained in the message shall be inline.

...

**6.9.3.1 Request**

The request message shall be formed as follows:

* Resource
* {+SERVICE}/workitems/{?query\*}

where

* {+SERVICE} is the base URL for the service. This may be a combination of protocol (either HTTP or HTTPS), authority and path.
* Method
* GET
* Headers
* Accept - The representation scheme in which the RESTful service is requested to return the results. The types allowed for this request header are as follows:
* multipart/related; type=application/dicom+xml; boundary={messageBoundary}

Specifies that the results should be DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML metadata.

* application/**dicom+**json

Specifies that the results should be DICOM [PS3.18](#PS3_18) JSON metadata.

...

**6.9.3.3.2 Query Result Attribute**

...

**6.9.3.3.3.2 JSON Response Message**

* Content-Type:
* application/**dicom+**json
* The response is a DICOM JSON message containing a DICOM JSON property for each matching UPS Instance containing sub-properties describing the matching attributes for each UPS Instance (see [Section F.2](#sect_F_2)).

...

**6.9.4 RetrieveUPS**

This resource supports the retrieval of a UPS Instance.

**6.9.4.1 Request**

The request message shall be formed as follows:

* Resource
* {+SERVICE}/workitems/{UPSInstanceUID}

where

* {+SERVICE} is the base URL for the service. This may be a combination of protocol (either HTTP or HTTPS), authority and path.
* {UPSInstanceUID} is the UID of the Unified Procedure Step Instance
* Method
* GET
* Headers
* Accept - The representation scheme in which the RESTful service is requested to return the result. The types allowed for this request header are as follows:
* application/dicom+xml

Specifies that the result should be DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML metadata.

* application/**dicom+**json

...

**6.9.4.3.2 Response Message**

...

**6.9.4.3.2.1 XML Response Message**

* Content-Type:
* application/dicom+xml
* The response contains a DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML DicomNativeModel element containing the attributes for the requested UPS Instance (see [Section A.1 in PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#sect_A.1)).

**6.9.4.3.2.2 JSON Response Message**

* Content-Type:
* application/**dicom+**json

...

**6.9.5 ChangeUPSState**

....

**6.9.5.1 Request**

The request message shall be formed as follows:

* Resource
* {+SERVICE}/workitems/{UPSInstanceUID}/state

where:

* {+SERVICE} is the base URL for the service. This may be a combination of protocol (either HTTP or HTTPS), authority and path.
* {UPSInstanceUID} is the UID of the Unified Procedure Step Instance
* Method
* PUT
* Headers
* Content-Type - The representation scheme being posted to the RESTful service. The types allowed for this request header are as follows:
* application/dicom+xml

Specifies that the post is DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML metadata. See [Section 6.9.5.1.1](#sect_6_9_5_1_1).

* application/**dicom+**json

...

**6.9.5.1.1 Request Message**

The Request Message has a single part body.

* Content-Type:
* application/dicom+xml
* application/**dicom+**json

...

**6.9.6 RequestUPSCancellation**

This resource records a request that the specified UPS Instance be canceled.

**6.9.6.1 Request**

* Resource
* {+SERVICE}/workitems/{UPSInstanceUID}/cancelrequest

where:

* {+SERVICE} is the base URL for the service. This may be a combination of protocol (either HTTP or HTTPS), authority and path.
* {UPSInstanceUID} is the UID of the Unified Procedure Step Instance
* Method
* POST
* Headers
* Content-Type - The representation scheme being posted to the RESTful service. The types allowed for this request header are as follows:
* application/dicom+xml

Specifies that the post is DICOM [PS3.19](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part19.pdf#PS3.19) XML metadata. See [Section 6.9.5.1.1](#sect_6_9_5_1_1).

* application/**dicom+**json

Specifies that the post is DICOM [PS3.18](#PS3_18) JSON metadata. See [Section 6.9.5.1.1](#sect_6_9_5_1_1).

* The request body describes a request to cancel a single Unified Procedure Step Instance. The request body shall comply with all attribute requirements described in [Table CC.2.2-1 in PS3.4](file:///C:\Users\jfp\AppData\Local\Microsoft\Windows\INetCache\Content.Outlook\AJ745Z9D\part04.pdf#table_CC.2.2-1).

**6.9.6.1.1 Request Message**

The Request Message has a single part body.

* Content-Type:
* application/dicom+xml
* application/**dicom+**json

...

*Update PS3.18 Annex F.2 as follows:*

**F.2 DICOM JSON Model**

The DICOM JSON Model follows the Native DICOM Model for XML very closely, so that systems can take advantage of both formats without much retooling. The Media Type for DICOM JSON is application/**dicom+**json. The default character repertoire shall be UTF-8 / ISO\_IR 192.

*Update PS3.17 Annex HHH as follows:*

**HHH.7.1 WADL Example (XML)**

The following WADL XML example contains all the required elements for an origin-server that supports WADO-RS, QIDO-RS and STOW-RS with all required services and parameters.

<application xsi:schemaLocation="http://wadl.dev.java.net/2009/02 wadl.xsd"  
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
 xmlns="http://wadl.dev.java.net/2009/02">  
 <resources base="http://medical.examplehospital.org/dicomweb">  
 <resource path="studies">  
 <method name="GET" id="SearchForStudies">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="StudyDate" style="query" />  
 <param name="00080020" style="query" />  
 <param name="StudyTime" style="query" />  
 <param name="00080030" style="query" />  
 <param name="AccessionNumber" style="query" />  
 <param name="00080050" style="query" />  
 <param name="ModalitiesInStudy" style="query" />  
 <param name="00080061" style="query" />  
 <param name="ReferringPhysicianName" style="query" />  
 <param name="00080090" style="query" />  
 <param name="PatientName" style="query" />  
 <param name="00100010" style="query" />  
 <param name="PatientID" style="query" />  
 <param name="00100020" style="query" />  
 <param name="StudyInstanceUID" style="query" repeating="true" />  
 <param name="0020000D" style="query" repeating="true" />  
 <param name="StudyID" style="query" />  
 <param name="00200010" style="query" />  
 <param name="includefield" style="query" repeating="true">  
 <option value="all" />  
 </param>  
 </request>  
 <response status="200">  
 <param name="Warning" style="header"  
 fixed="299 {SERVICE}: The fuzzymatching parameter is not supported.  
 Only literal matching has been performed." />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
 </response>

<response status="400 401 403 413 503" />  
 </method>  
 <method name="POST" id="StoreInstances">  
 <request>  
 <param name="Accept" style="header" default="application/dicom+xml">  
 <option value="application/dicom+xml" />  
 </param>  
 <representation mediaType="multipart/related; type=application/dicom" />  
 <representation mediaType="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 </request>  
 <response status="202 409">  
 <representation mediaType="application/dicom+xml" />  
 </response>  
 <response status="400 401 403 503" />  
 </method>  
 <resource path="{StudyInstanceUID}">  
 <method name="GET" id="RetrieveStudy">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom">  
 <option value="multipart/related; type=application/dicom" />  
 <option value="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <option value="multipart/related; type=application/octet-stream" />  
 </param>  
 </request>  
 <response status="200 206">  
 <representation mediaType="multipart/related; type=application/dicom" />  
 <representation mediaType="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <representation mediaType="multipart/related; type=application/octet-stream" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 <method name="POST" id="StoreStudyInstances">  
 <request>  
 <param name="Accept" style="header" default="application/dicom+xml">  
 <option value="application/dicom+xml" />  
 </param>  
 <representation mediaType="multipart/related; type=application/dicom" />  
 <representation mediaType="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 </request>  
 <response status="202 409">  
 <representation mediaType="application/dicom+xml" />  
 </response>  
 <response status="400 401 403 503" />  
 </method>  
 <resource path="series">  
 <method name="GET" id="SearchForStudySeries">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="Modality" style="query" />  
 <param name="00080060" style="query" />  
 <param name="SeriesInstanceUID" style="query" repeating="true" />  
 <param name="0020000E" style="query" repeating="true" />  
 <param name="SeriesNumber" style="query" />  
 <param name="00200011" style="query" />  
 <param name="PerformedProcedureStepStartDate" style="query" />  
 <param name="00400244" style="query" />  
 <param name="PerformedProcedureStepStartTime" style="query" />  
 <param name="00400245" style="query" />  
 <param name="RequestAttributeSequence" style="query" />  
 <param name="00400275" style="query" />  
 <param name="RequestAttributeSequence.ScheduledProcedureStepID" style="query" />  
 <param name="00400275.00400009" style="query" />  
 <param name="RequestAttributeSequence.RequestedProcedureID" style="query" />  
 <param name="00400275.00401001" style="query" />  
 <param name="includefield" style="query" repeating="true">  
 <option value="all" />  
 </param>  
 </request>  
 <response status="200">  
 <param name="Warning" style="header"  
 fixed="299 {SERVICE}: The fuzzymatching parameter is not supported.  
 Only literal matching has been performed." />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
 </response>  
 <response status="400 401 403 413 503" />  
 </method>  
 <resource path="{SeriesInstanceUID}">  
 <method name="GET" id="RetrieveSeries">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom">  
 <option value="multipart/related; type=application/dicom" />  
 <option value="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <option value="multipart/related; type=application/octet-stream" />  
 </param>  
 </request>  
 <response status="200 206">  
 <representation mediaType="multipart/related; type=application/dicom" />  
 <representation mediaType="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <representation mediaType="multipart/related; type=application/octet-stream" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 <resource path="instances">  
 <method name="GET" id="SearchForStudySeriesInstances">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="SOPClassUID" style="query" repeating="true" />  
 <param name="00080016" style="query" repeating="true" />  
 <param name="SOPInstanceUID" style="query" repeating="true" />  
 <param name="00080018" style="query" repeating="true" />  
 <param name="InstanceNumber" style="query" />  
 <param name="00200013" style="query" />  
 <param name="includefield" style="query" repeating="true">  
 <option value="all" />  
 </param>  
 </request>  
 <response status="200">  
 <param name="Warning" style="header"  
 fixed="299 {SERVICE}: The fuzzymatching parameter is not supported.  
 Only literal matching has been performed." />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
 </response>  
 <response status="400 401 403 413 503" />  
 </method>  
 <resource path="{SOPInstanceUID}">  
 <method name="GET" id="RetrieveInstance">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom">  
 <option value="multipart/related; type=application/dicom" />  
 <option value="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <option value="multipart/related; type=application/octet-stream" />  
 </param>  
 </request>  
 <response status="200 206">  
 <representation mediaType="multipart/related; type=application/dicom" />  
 <representation mediaType="multipart/related; type=application/dicom;  
 transfer-syntax=1.2.840.10008.1.2.1" />  
 <representation mediaType="multipart/related; type=application/octet-stream" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 <resource path="frames">  
 <resource path="{framelist}">  
 <method name="GET" id="RetrieveFrames">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/octet-stream">  
 <option value="multipart/related; type=application/octet-stream" />  
 </param>  
 </request>  
 <response status="200">  
 <representation mediaType="multipart/related; type=application/octet-stream" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 </resource>  
 </resource>  
 <resource path="metadata">  
 <method name="GET" id="RetrieveInstanceMetadata">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 </request>  
 <response status="200">  
 <representation mediaType=" multipart/related; type=application/dicom+xml" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 </resource>  
 </resource>  
 </resource>  
 <resource path="metadata">  
 <method name="GET" id="RetrieveSeriesMetadata">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 </request>  
 <response status="200">  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 </resource>  
 </resource>  
 </resource>  
 <resource path="instances">  
 <method name="GET" id="SearchForStudyInstances">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="SOPClassUID" style="query" />  
 <param name="00080016" style="query" />  
 <param name="SOPInstanceUID" style="query" repeating="true" />  
 <param name="00080018" style="query" repeating="true" />  
 <param name="Modality" style="query" />  
 <param name="00080060" style="query" />  
 <param name="SeriesInstanceUID" style="query" repeating="true" />  
 <param name="0020000E" style="query" repeating="true" />  
 <param name="SeriesNumber" style="query" />  
 <param name="00200011" style="query" />  
 <param name="InstanceNumber" style="query" />  
 <param name="00200013" style="query" />  
 <param name="PerformedProcedureStepStartDate" style="query" />  
 <param name="00400244" style="query" />  
 <param name="PerformedProcedureStepStartTime" style="query" />  
 <param name="00400245" style="query" />  
 <param name="RequestAttributeSequence" style="query" />  
 <param name="00400275" style="query" />  
 <param name="RequestAttributeSequence.ScheduledProcedureStepID" style="query" />  
 <param name="00400275.00400009" style="query" />  
 <param name="RequestAttributeSequence.RequestedProcedureID" style="query" />  
 <param name="00400275.00401001" style="query" />  
 <param name="includefield" style="query" repeating="true">  
 <option value="all" />  
 </param>  
 </request>  
 <response status="200">  
 <param name="Warning" style="header"  
 fixed="299 {SERVICE}: The fuzzymatching parameter is not supported.  
 Only literal matching has been performed." />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
 </response>  
 <response status="400 401 403 413 503" />  
 </method>  
 </resource>  
 <resource path="metadata">  
 <method name="GET" id="RetrieveStudyMetadata">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 </request>  
 <response status="200">  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 </resource>  
 </resource>  
 </resource>  
 <resource path="series">  
 <method name="GET" id="SearchForSeries">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="StudyDate" style="query" />  
 <param name="00080020" style="query" />  
 <param name="StudyTime" style="query" />  
 <param name="00080030" style="query" />  
 <param name="AccessionNumber" style="query" />  
 <param name="00080050" style="query" />  
 <param name="Modality" style="query" />  
 <param name="00080060" style="query" />  
 <param name="ModalitiesInStudy" style="query" />  
 <param name="00080061" style="query" />  
 <param name="ReferringPhysicianName" style="query" />  
 <param name="00080090" style="query" />  
 <param name="PatientName" style="query" />  
 <param name="00100010" style="query" />  
 <param name="PatientID" style="query" />  
 <param name="00100020" style="query" />  
 <param name="StudyInstanceUID" style="query" repeating="true" />  
 <param name="0020000D" style="query" repeating="true" />  
 <param name="SeriesInstanceUID" style="query" />  
 <param name="0020000E" style="query" />  
 <param name="StudyID" style="query" />  
 <param name="00200010" style="query" />  
 <param name="SeriesNumber" style="query" />  
 <param name="00200011" style="query" />  
 <param name="PerformedProcedureStepStartDate" style="query" />  
 <param name="00400244" style="query" />  
 <param name="PerformedProcedureStepStartTime" style="query" />  
 <param name="00400245" style="query" />  
 <param name="RequestAttributeSequence" style="query" />  
 <param name="00400275" style="query" />  
 <param name="RequestAttributeSequence.ScheduledProcedureStepID" style="query" />  
 <param name="00400275.00400009" style="query" />  
 <param name="RequestAttributeSequence.RequestedProcedureID" style="query" />  
 <param name="00400275.00401001" style="query" />  
 <param name="includefield" style="query" repeating="true">  
 <option value="all" />  
 </param>  
 </request>  
 <response status="200">  
 <param name="Warning" style="header"  
 fixed="299 {SERVICE}: The fuzzymatching parameter is not supported.  
 Only literal matching has been performed." />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
 </response>  
 <response status="400 401 403 413 503" />  
 </method>  
 <resource path="{SeriesInstanceUID}">  
 <resource path="instances">  
 <method name="GET" id="SearchForSeriesInstances">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="SOPClassUID" style="query" repeating="true" />  
 <param name="00080016" style="query" repeating="true" />  
 <param name="SOPInstanceUID" style="query" repeating="true" />  
 <param name="00080018" style="query" repeating="true" />  
 <param name="InstanceNumber" style="query" />  
 <param name="00200013" style="query" />  
 <param name="includefield" style="query" repeating="true">  
 <option value="all" />  
 </param>  
 </request>  
 <response status="200">  
 <param name="Warning" style="header"  
 fixed="299 {SERVICE}: The fuzzymatching parameter is not supported.  
 Only literal matching has been performed." />  
 <representation mediaType="application/**dicom+**json" />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 </response>  
 <response status="400 401 403 413 503" />  
 </method>  
 </resource>  
 </resource>  
 </resource>  
 <resource path="instances">  
 <method name="GET" id="SearchForInstances">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/dicom+**~~xml~~json**">  
 <option value="multipart/related; type=application/dicom+xml" />  
 <option value="application/**dicom+**json" />  
 </param>  
 <param name="Cache-control" style="header">  
 <option value="no-cache" />  
 </param>  
 <param name="limit" style="query" />  
 <param name="offset" style="query" />  
 <param name="SOPClassUID" style="query" repeating="true" />  
 <param name="00080016" style="query" repeating="true" />  
 <param name="SOPInstanceUID" style="query" repeating="true" />  
 <param name="00080018" style="query" repeating="true" />  
 <param name="StudyDate" style="query" />  
 <param name="00080020" style="query" />  
 <param name="StudyTime" style="query" />  
 <param name="00080030" style="query" />  
 <param name="AccessionNumber" style="query" />  
 <param name="00080050" style="query" />  
 <param name="Modality" style="query" />  
 <param name="00080060" style="query" />  
 <param name="ModalitiesInStudy" style="query" />  
 <param name="00080061" style="query" />  
 <param name="ReferringPhysicianName" style="query" />  
 <param name="00080090" style="query" />  
 <param name="PatientName" style="query" />  
 <param name="00100010" style="query" />  
 <param name="PatientID" style="query" />  
 <param name="00100020" style="query" />  
 <param name="StudyInstanceUID" style="query" repeating="true" />  
 <param name="0020000D" style="query" repeating="true" />  
 <param name="SeriesInstanceUID" style="query" repeating="true" />  
 <param name="0020000E" style="query" repeating="true" />  
 <param name="SeriesNumber" style="query" />  
 <param name="00200011" style="query" />  
 <param name="InstanceNumber" style="query" />  
 <param name="00200013" style="query" />  
 <param name="PerformedProcedureStepStartDate" style="query" />  
 <param name="00400244" style="query" />  
 <param name="PerformedProcedureStepStartTime" style="query" />  
 <param name="00400245" style="query" />  
 <param name="RequestAttributeSequence" style="query" />  
 <param name="00400275" style="query" />  
 <param name="RequestAttributeSequence.ScheduledProcedureStepID" style="query" />  
 <param name="00400275.00400009" style="query" />  
 <param name="RequestAttributeSequence.RequestedProcedureID" style="query" />  
 <param name="00400275.00401001" style="query" />  
 <param name="includefield" style="query" repeating="true">  
 <option value="all" />  
 </param>  
 </request>  
 <response status="200">  
 <param name="Warning" style="header"  
 fixed="299 {SERVICE}: The fuzzymatching parameter is not supported.  
 Only literal matching has been performed." />  
 <representation mediaType="multipart/related; type=application/dicom+xml" />  
 <representation mediaType="application/**dicom+**json" />  
 </response>  
 <response status="400 401 403 413 503" />  
 </method>  
 </resource>  
 <resource path="{BulkDataURL}">  
 <method name="GET" id="RetrieveBulkData">  
 <request>  
 <param name="Accept" style="header"  
 default="multipart/related; type=application/octet-stream">  
 <option value="multipart/related; type=application/octet-stream" />  
 </param>  
 </request>  
 <response status="200">  
 <representation mediaType="multipart/related; type=application/octet-stream" />  
 </response>  
 <response status="400 404 406 410 503"></response>  
 </method>  
 </resource>  
 </resources>  
</application>

*Update PS3.2 Annex K as follows:*

**K.4.2.1.1 QIDO-RS Search for Studies**

**Table K.4.2-1. QIDO-RS Search for Studies Specification**

| **Parameter** | **Restrictions** |
| --- | --- |
| Media Types | Restricted to "multipart/related; type=application/dicom+xml" or "application/**dicom+**json" |
| Matching Attributes | ... |

...

**K.4.2.1.2 QIDO-RS Search for Series**

**Table K.4.2-2. QIDO-RS Search for Series Specification**

| **Parameter** | **Restrictions** |
| --- | --- |
| Media Types | Restricted to "multipart/related; type=application/dicom+xml" or "application/**dicom+**json" |
| Matching Attributes | ... |

...

**K.4.2.1.3 QIDO-RS Search for Instances**

**Table K.4.2-3. QIDO-RS Search for Instances Specification**

| **Parameter** | **Restrictions** |
| --- | --- |
| Media Types | Restricted to "multipart/related; type=application/dicom+xml" or "application/**dicom+**json" |
| Matching Attributes | ... |