PS3.18

DICOM PS3.18 ~~2017d~~2017e - Web Services

PS3.18: DICOM PS3.18 ~~2017d~~2017e - Web Services

Table of Contents

List of Figures

List of Tables

Notice and Disclaimer

The information in this publication was considered technically sound by the consensus of persons engaged in the development and approval of the document at the time it was developed. Consensus does not necessarily mean that there is unanimous agreement among every person participating in the development of this document.

NEMA standards and guideline publications, of which the document contained herein is one, are developed through a voluntary consensus standards development process. This process brings together volunteers and/or seeks out the views of persons who have an interest in the topic covered by this publication. While NEMA administers the process and establishes rules to promote fairness in the development of consensus, it does not write the document and it does not independently test, evaluate, or verify the accuracy or completeness of any information or the soundness of any judgments contained in its standards and guideline publications.

NEMA disclaims liability for any personal injury, property, or other damages of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, application, or reliance on this document. NEMA disclaims and makes no guaranty or warranty, expressed or implied, as to the accuracy or completeness of any information published herein, and disclaims and makes no warranty that the information in this document will fulfill any of your particular purposes or needs. NEMA does not undertake to guarantee the performance of any individual manufacturer or seller's products or services by virtue of this standard or guide.

In publishing and making this document available, NEMA is not undertaking to render professional or other services for or on behalf of any person or entity, nor is NEMA undertaking to perform any duty owed by any person or entity to someone else. Anyone using this document should rely on his or her own independent judgment or, as appropriate, seek the advice of a competent professional in determining the exercise of reasonable care in any given circumstances. Information and other standards on the topic covered by this publication may be available from other sources, which the user may wish to consult for additional views or information not covered by this publication.

NEMA has no power, nor does it undertake to police or enforce compliance with the contents of this document. NEMA does not certify, test, or inspect products, designs, or installations for safety or health purposes. Any certification or other statement of compliance with any health or safety-related information in this document shall not be attributable to NEMA and is solely the responsibility of the certifier or maker of the statement.

Foreword

This DICOM Standard was developed according to the procedures of the DICOM Standards Committee.

The DICOM Standard is structured as a multi-part document using the guidelines established in [ISO/IEC Directives, Part 2].

PS3.1 should be used as the base reference for the current parts of this standard.

1 Scope

This standard specifies a web-based service for accessing and presenting DICOM (Digital Imaging and Communications in Medicine) objects (e.g., images, medical imaging reports). This is intended for distribution of results and images to healthcare professionals. It provides a simple mechanism for accessing a DICOM object, through HTTP/HTTPS protocol, using DICOM UIDs (Unique Identifiers). Data may be retrieved either in a presentation-ready form as specified by the requester (e.g., JPEG or GIF) or in a native DICOM format. It does not support facilities for web searching of DICOM images. This standard relates only to DICOM Objects (not to non-DICOM objects). Access control beyond the security mechanisms generally available to web applications is outside the scope of this standard.

2 Conformance

Systems claiming conformance to this standard shall function in accordance with all its mandatory sections.

3 Normative References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this part of DICOM. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of DICOM are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

3.1 International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC)

[ISO/IEC Directives, Part 2] ISO/IEC. 2016/05. 7.0. *Rules for the structure and drafting of International Standards*. http://www.iec.ch/members_experts/refdocs/iec/isoiecdir-2%7Bed7.0%7Den.pdf .

[ISO/IEC 2022] ISO/IEC. 1994. *Information technology - Character code structure and extension techniques*.

[ISO/IEC 10918-1] ISO/IEC. 1994. *JPEG Standard for digital compression and encoding of continuous-tone still images. Part 1 - Requirements and implementation guidelines*.

[ISO/IEC 10646] ISO/IEC. 2003. *Information Technology - Universal Multiple-Octet Coded Character Set (UCS)*. *ISO/IEC 10646-2003 is the same as Unicode Version 4.0, available at http://unicode.org* .

[IEC 61966-2.1] IEC. 1999. Ed 1.0 as amended by Amendment A1:2003. *Multimedia systems and equipment - colour measurement and management - Part 2.1: colour management - Default RGB colour space - sRGB*.

3.2 Internet Engineering Task Force (IETF)

[RFC822] IETF. August 1982. *Standard for ARPA Internet Text Messages*. http://tools.ietf.org/html/rfc822 .

[RFC1945] IETF. May 1996. *Hypertext Transfer Protocol Version 1.0 (HTTP/1.0)* . http://tools.ietf.org/html/rfc1945 .

[RFC2045] IETF. November 1996. *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*. http://tools.ietf.org/html/rfc2045 .

[RFC2046] IETF. November 1996. *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*. http://tools.ietf.org/html/rfc2046 .

[RFC2387] IETF. August 1998. *The MIME Multipart/Related Content-type*. http://tools.ietf.org/html/rfc2387 .

[RFC2818] IETF. May 2000. *HTTP Over TLS*. http://tools.ietf.org/html/rfc2818 .

[RFC2978] IETF. October 2000. *IANA Charset Registration Procedures*. http://tools.ietf.org/html/rfc2978 .

[RFC3240] IETF. February 2002. *Digital Imaging and Communications in Medicine (DICOM) - Application/dicom MIME Sub-type Registration*. http://tools.ietf.org/html/rfc3240 .

[RFC3986] IETF. *Uniform Resource Identifiers (URI): Generic Syntax*. http://tools.ietf.org/html/rfc3986 .

[RFC4627] IETF. July 2006. *The application/json Media Type for JavaScript Object Notation (JSON)*. http://tools.ietf.org/html/rfc4627 .

[RFC5234] IETF. January 2008. *Augmented BNF for Syntax Specifications: ABNF*. http://tools.ietf.org/html/rfc5234 .

[RFC6365] IETF. September 2011. *Terminology Used in Internationalization in the IETF*. http://tools.ietf.org/html/rfc6365 .

[RFC6455] IETF. December 2011. *The WebSocket Protocol*. http://tools.ietf.org/html/rfc6455 .

[RFC6570] IETF. March 2012. *URI Template*. http://tools.ietf.org/html/rfc6570 .

[RFC6838] IETF. January 2013. *Media Type Specifications and Registration Procedures*. http://tools.ietf.org/html/rfc6838 .

[RFC7230] IETF. June 2014. *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*. http://tools.ietf.org/html/rfc7230 .

[RFC7231] IETF. June 2014. *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*. http://tools.ietf.org/html/rfc7231 .

[RFC7232] IETF. June 2014. *Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests*. http://tools.ietf.org/html/rfc7232 .

[RFC7233] IETF. June 2014. *Hypertext Transfer Protocol (HTTP/1.1): Range Requests*. http://tools.ietf.org/html/rfc7233 .

[RFC7234] IETF. June 2014. *Hypertext Transfer Protocol (HTTP/1.1): Caching*. http://tools.ietf.org/html/rfc7234 .

[RFC7235] IETF. June 2014. *Hypertext Transfer Protocol (HTTP/1.1): Authentication*. http://tools.ietf.org/html/rfc7235 .

[RFC7236] IETF. June 2014. *Initial Hypertext Transfer Protocol (HTTP) Authentication Scheme Registrations*. http://tools.ietf.org/html/rfc7236 .

[RFC7237] IETF. June 2014. *Initial Hypertext Transfer Protocol (HTTP) Method Registrations*. http://tools.ietf.org/html/rfc7237 .

[RFC7405] IETF. December 2014. *Case-Sensitive String Support in ABNF*. http://tools.ietf.org/html/rfc7405 .

[RFC7525] IETF. May 2015. *TLS Recommendations*. http://tools.ietf.org/html/rfc7525 .

[RFC7540] IETF. May 2015. *Hypertext Transfer Protocol Version 2 (HTTP/2)*. http://tools.ietf.org/html/rfc7540 .

3.3 Health Level Seven (HL7)

[HL7 CDA R2] ANSI/HL7. 2005. *HL7 Version 3 Standard: Clinical Document Architecture Framework, Release 2*. http://www.hl7.org/documentcenter/private/standards/cda/r2/cda_r2_normativewebedition2010.zip .

3.4 Other References

[IHE ITI TF-2x Appendix V] IHE. . . *IT Infrastructure Technical Framework - Web Services for IHE Transactions*. http://www.ihe.net/uploadedFiles/Documents/ITI/IHE_ITI_TF_Vol2x.pdf .

[WADL] W3C. 31 August 2009. . *Member Submission - Web Application Description Language*. http://www.w3.org/Submission/wadl/ .

4 Terms and Definitions

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

Accept Query Parameter	A query parameter that specifies one or more media types acceptable for the representation(s) contained in the response. See Section 6.1.1.5.
Acceptable Character Sets	One or more character sets acceptable to the user agent in the response. See Section 6.1.2.1.
Acceptable Media Types	One or more media type acceptable to the user agent in the response. See Section 6.1.1.4.
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. For the value of a frame of a Pixel Data attribute 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 are encoded. See Section 6.1.1.8.
Charset Query Parameter	A query parameter that specifies one or more character sets for the representation(s) contained in the response. See Section 6.1.2.2.
DICOM Object	An instance of a data object as defined by PS3.3 that has been allocated an unique identifier in the format specified for SOP Instance UID in PS3.3 and has been chosen as an object to be saved securely for some period of time. Within the DICOM Standard, a DICOM Object is referred to as a Composite Service Object Pair (SOP) Instance.
DICOM Resource Categories	A set of categories for the content of DICOM SOP Instances. Examples include images, video, and text. See Section 6.1.1.2.
HTTP	The term HTTP as used in this Standard means the Hypertext Transport Protocol versions 1.0, 1.1 ~~or 2~~, 2 or later. See [RFC1945], [RFC7230] and [RFC7540].
HTTPS	The term HTTPS as used in this Standard means the Hypertext Transport Protocol Secure versions 1.0, 1.1 ~~or 2~~, 2 or later. See [RFC2818] and [RFC7230].
Origin-Server	See [RFC7230] Section 2.1 Client/Server Messaging.
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.
Selected Character Set	The character sets selected by the origin server for the response payload. See Section 6.1.2.4.
Selected Media Type	The media type selected by the origin server for the response payload. See Section 6.1.1.7.
User-Agent	See [RFC7230] Section 2.1 Client/Server Messaging.
Web Access to DICOM Objects	A service enabling the user agent to retrieve DICOM Objects managed by an origin serven, through HTTP/HTTPS protocol.

5 Symbols and Abbreviated Terms

DICOM Digital Imaging and Communications in Medicine

HL7 Health Level Seven

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

HTTPS HyperText Transfer Protocol Secure

IHE Integrating the Healthcare Enterprise

MIME Multipurpose Internet Mail Extensions

QIDO-RS Query based on ID for DICOM Objects by RESTful Services

REST Representational State Transfer

RESTful A RESTful Web service is a Web service implemented using REST architecture and HTTP (see http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf)

SOP Service Object Pair

sRGB A standard RGB color space defined in [IEC 61966-2.1].

STOW-RS STore Over the Web by RESTful Services

UID Unique (DICOM) Identifier

UPS-RS Unified Procedure Step by RESTful Services

URL/URI Uniform Resource Locator / Identifier

UTF-8 Unicode UTF-8 character set defined in [ISO/IEC 10646].

WADL Web Application Description Language

WADO-RS Web Access to DICOM Objects by RESTful Services

WADO-URI Web Access to DICOM Objects by URI

XML eXtensible Markup Language

6 Data Communication Requirements

DICOM Web Services use the HTTP and HTTPS protocols as its transport medium. Web Services supports versions 1.0, 1.1 ~~and 2~~, 2 or later of the protocol. ~~If an origin server supports version 2, it shall also support version 1.1. If an origin server supports version 1.1, it shall also support version 1.0.~~

It is recommended that user agents that want to use HTTP/2 first initiate an HTTP/1.1 connection to the origin server and then upgrade to HTTP/2. If the upgrade fails then the user agent can still use the HTTP/1.1 connection. [RFC7540] Section 3 explains how to initiate HTTP/2 connections.

Note

HTTPS may mean any SSL or TLS version and options. [RFC7525] covers selection of TLS versions and options. There may also be national or local regulations that apply, and site-specific risk analysis may affect the selection. TLS version 1.2 or later is often required and always recommended. Earlier versions of TLS and all versions of SSL are known to be vulnerable to well publicized attacks.

6.1 Interaction

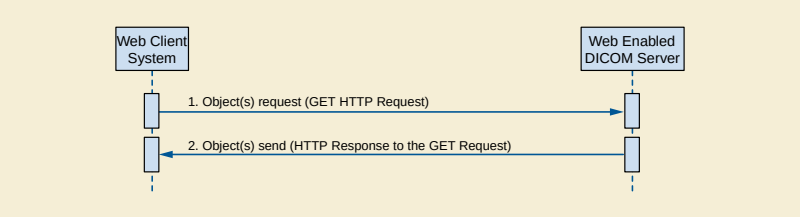

Figure 6-1. Interaction Diagram

The interaction shall be as shown in Figure 6-1.

Multiple communications modes are possible:

- URI based using HTTP Get: WADO-URI request
- RESTful Services (RS) using HTTP Get: WADO-RS, either:
  a. DICOM Requester (Retrieve Study, Series, or Instance DICOM Objects)
  b. Frame Pixel Data Requester (Retrieve Instance Frame Pixel Data)
  c. Bulk Data Requester (Retrieve Study, Series, Instance Bulk Data)
  d. Metadata Requester (Retrieve Study, Series, Instance Metadata)
- RESTful Services (RS) using HTTP Get: QIDO-RS:
  a. Query Requester (Search for Study, Series or Instance DICOM Objects)
- RESTful Services (RS) using HTTP POST: STOW-RS, either:
  a. DICOM Creator (Store Instances)
  b. Metadata and Bulk Data Creator (Store Instances)
- RESTful Services (RS) using HTTP Options: RS Capabilities:
  a. Provided information about the capabilities of a DICOM RESTful web service provider)

6.1.1 Media Types

Media types are identifiers used to define the data format of a representation. HTTP uses media types in the Content-Type and Accept header fields in order to provide open and extensible data typing and type negotiation. The syntax of media types is:

```
media-type = type "/" subtype *(OWS ";" OWS parameter)
```

Where

```
type = token
subtype = token
parameter = token "=" (token / quoted-string) )
```

The <type>/<subtype> may be followed by parameters in the form of name=value pairs.

The type, subtype, and parameter name tokens are case-insensitive, but the case sensitivity of parameter values depends on the semantics of the parameter name. The presence or absence of a parameter might be significant to the processing of a media-type, depending on its definition within the media type registry.

A parameter value can be transmitted either as a token or quoted-string. The quoted and unquoted values are equivalent.

Media types are defined in [RFC7231] Section 3.1.1.1.

IANA maintains a registry of media types at http://www.iana.org/assignments/media-types/media-types.xhtml.

6.1.1.1 Multipart Media Types

Some of the services defined in this Standard support the multipart media types [RFC2387]. The syntax is:

```
multipart-media-type = "multipart" "/" subtype *( OWS ";" OWS parameter )
```

The "application/multipart-related" media type is used by the RS services. Its syntax is:

```
multipart-related = "multipart/related"
                    OWS ";" OWS "type" "=" DQUOTE media-type DQUOTE
                    OWS ";" OWS "boundary" "=" boundary
                    [related-parameters]
```

Where

```
boundary = 0*69bchar bchar-nospace
bchar = bchar-nospace / SP
bchar-nospace = DIGIT / ALPHA / "'" / "(" / ") " / "+" / "_" / "," / "-"
              / "." / "/" / ":" / "=" / "?" "/" / ":" / "=" / "?"
related-parameters = [";" "start" "=" cid]
                     [";" "start-info" "=" cid-list]
cid-list = cid cid-list
cid = token / quoted-string
```

The "type" parameter is required. It contains the media type of the "root" body part. It always contains the special character "/" and thus requires quote marks.

The <cid> is a content identifier. It should be unique for each part of the multipart message.

Typically, the "start" and "start-info" parameters are not specified, and the "root" is the first body part.

6.1.1.2 DICOM Resource Categories

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), 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).
Other	This category includes all resources that are not included above.

6.1.1.3 Rendered Media Types

DICOM instances may be converted by a rendering process into non-DICOM media types in order to 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 specifies the meaning of media type requirement terms used in 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 origin server shall return this media type when none of the Acceptable Media Types (see Section 6.1.1.4) are supported. The origin server shall support this media type.
required	The origin server shall support this media type.
optional	The origin server may support this media type.

Origin servers that support Web 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. Rendered Media Types by Resource Category

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

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).

The user agent may specify the relative degree of preference for media types, whether in the <accept> query parameter or the Accept header field, using the <weight> parameter. See [RFC7231] Section 5.3.1.

```
weight = OWS ";" OWS "q=" qvalue
qvalue = ("0" ["." 0*3DIGIT]) / ("1" ["." 0*3("0") ])
```

If no "q" parameter is present, the default qvalue is 1.

6.1.1.5 Accept Query Parameter

The <accept> query parameter is primarily designed for use in hyperlinks (URLs) embedded in documents, where the Accept header field is not accessible. It is similar to the Accept header field, except that it shall not have wildcards (<type>/* or */*).

The <accept> query parameter has the following syntax:

```
accept = accept-name "=" 1#(media-type [weight])
accept-name = "%s" quoted-string
```

Note

The "%s" that prefixes the <accept-name> specifies that it is a case sensitive token. See [RFC7405].

Its value is a comma-separated list of one or more <media-type>s, possibly including parameters. It shall be supported by the origin server. It is optional for the user agent.

The <accept-name> of the <accept> query parameter is defined by the Service. It is case-sensitive. Table 6.1.1-4 contains the <accept-name> of the <accept> query parameter for some services.

Table 6.1.1-4. <accept> Query Parameter Name by Service

Service	Name
URI	accept-name = "contentType"
RS	accept-name = "accept"

The <accept> query parameter should not be used when the user agent can specify the values in the Accept header field.

All media types present in an <accept> query parameter shall be compatible with a media range in the Accept header field, either explicitly or implicitly through wildcards.

Note

For example, the presence of image/jpeg in the <accept> query parameter will require the Accept header field to include one of the following values: image/jpeg, image/*, or */*.

See Section 6.1.4.

6.1.1.6 Accept Header field

The Accept header field is used to specify media ranges acceptable to the user agent. It has the following syntax:

```
Accept = 1#(media-range [weight])
```

Where,

```
media-range = media-type
            / type "/" "*" parameters
```

/ "*/*" parameters
parameters ; See Section 6.1.1

The Accept header field shall be present. Its value shall be a comma-separated list of one or more media ranges acceptable in the response. See [RFC7231] Section 5.3.2.

A media range is either a media-type or a wildcard. Wildcards use the asterisk ("*") to group media types into ranges, with <type>/* indicating all subtypes of that type, and */* indicating all media types from the target's Resource's Category.

For example, the media range "image/*" matches "image/jpeg", which is the default media type for the Single Frame Image Resource Category, and "text/*" matches "text/html", which is the default media type for the Text Resource Category. The "*/*" media range matches the default media type for the target's Resource Category.

If the origin server receives a request without an Accept header field, but that might have a response payload, it shall return a 406 (Not Acceptable).

Any Accept header field values that are not valid or not supported shall be ignored.

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).

For a set of media types in the <accept> query parameter (step 2 above), or for a set of media ranges in the Accept header field (step 3 above), the highest priority supported media type is determined as follows:

1. Assign a <qvalue> of 1 to any member of the set that does not have a one.
2. Assign each representation supported by the origin server the <qvalue> of the most specific media type that it matches.
3. Select the representation with the highest <qvalue>. If there is a tie, the origin server shall determine which is returned.

For example, consider an origin server that receives a request with the following Accept header field:

Accept: text/*; q=0.5, text/html; q=0.4, text/html; level=1, text/html; level=2; q=0.7, image/png, */*; q=0.4

Suppose that for the resource indicated in the request, the origin server supports representations for the following media types:

text/html(regular, level 1 and level 2)
text/rtf
text/plain
text/x-latex

These media types are assigned the following <qvalue>s, based on the media ranges above:

Media Type	qvalue	Determining Media Range
text/html; level=1	1.0	text/html; level=1
text/html; level=2	0.7	text/html; level=2
text/plain	0.5	text/*
text/rtf	0.5	text/*
text/html	0.4	text/html
text/x-latex	0.4	*/*

Although "image/png" has been assigned a default <qvalue> of 1.0, it is not among the supported media types for this resource, and thus is not listed.

The selected media type is "text/html; level=1" since it is the supported media type in the Text Category with the highest qvalue.

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 Web 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.

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

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

Table 6.1.1.8-1b. Media Types for DICOM Metadata

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

Table 6.1.1.8-1c. Media Types for DICOM Uncompressed Bulk Data

Media Type	Descriptions	URI	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	See Table 6.1.1.8-3a

Table 6.1.1.8-1d. Media Types for DICOM Compressed Bulk Data

Media Type	Descriptions	URI	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-Type header. 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	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 Web 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 Section 10.1 “DICOM Default Transfer Syntax” in PS3.5, 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 SyntaxUID	Transfer Syntax Name	URI	RS
Single Frame Image	1.2.840.10008.1.2.1	Explicit VR Little Endian	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

Category	Transfer SyntaxUID	Transfer Syntax Name	URI	RS
	1.2.840.10008.1.2.4.50	JPEG Baseline (Process 1) :Default Transfer Syntax for Lossy JPEG 8 Bit Image Compression	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
	1.2.840.10008.1.2.4.57	JPEG Lossless, Non-Hierarchical (Process 14)	optional	optional
	1.2.840.10008.1.2.5	RLE Lossless	optional	optional
	1.2.840.10008.1.2.4.80	JPEG-LS Lossless Image Compression	optional	optional
	1.2.840.10008.1.2.4.81	JPEG-LS Lossy (Near-Lossless) Image Compression	optional	optional
	1.2.840.10008.1.2.4.90	JPEG 2000 Image Compression (Lossless Only)	optional	optional
	1.2.840.10008.1.2.4.91	JPEG 2000 Image Compression	optional	optional
	1.2.840.10008.1.2.4.92	JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only)	optional	optional
	1.2.840.10008.1.2.4.93	JPEG 2000 Part 2 Multi-component Image Compression	optional	optional
Multi-frame Image	1.2.840.10008.1.2.1	Explicit VR Little Endian	default	default
	1.2.840.10008.1.2.4.90	JPEG 2000 Image Compression (Lossless Only)	optional	optional
	1.2.840.10008.1.2.4.91	JPEG 2000 Image Compression	optional	optional
	1.2.840.10008.1.2.4.92	JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only)	optional	optional
	1.2.840.10008.1.2.4.93	JPEG 2000 Part 2 Multi-component Image Compression	optional	optional
Video	1.2.840.10008.1.2.1	Explicit VR Little Endian	default	default
	1.2.840.10008.1.2.4.100	MPEG2 Main Profile / Main Level	optional	optional
	1.2.840.10008.1.2.4.101	MPEG2 Main Profile / High Level	optional	optional
	1.2.840.10008.1.2.4.102	MPEG-4 AVC/H.264 High Profile / Level 4.1	optional	optional
	1.2.840.10008.1.2.4.103	MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1	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
	1.2.840.10008.1.2.4.105	MPEG-4 AVC/H.264 High Profile / Level 4.2 For 3D Video	optional	optional
	1.2.840.10008.1.2.4.106	MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2	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 RS service.

Note

No entries are specified for the WADO-URI service, because it does 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	MediaType	Transfer SyntaxUID	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 UIDsfor Compressed Pixel Data in Bulk Data Values

Category	MediaType	Transfer SyntaxUID	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

Category	MediaType	Transfer SyntaxUID	Transfer Syntax Name	RS
		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 be 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 Section 8.2.1 “JPEG Image Compression” in PS3.5.

3. For the media type image/dicom+jp2 and image/dicom+jpx Transfer Syntaxes, the image does not include the jp2 marker segment. See Section 8.2.4 “JPEG 2000 Image Compression” in PS3.5 and Section A.4.4 “JPEG 2000 Image Compression” in PS3.5.

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-mtp     ; see Section 6.1.1.8.1.2.
        / charset-mtp            ; 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 a transfer syntax parameter, but its usage may be constrained by the service for which they are used.

Note

Note. The application/dicom+xml and application/dicom+json Media Types may have a transfer syntax parameter in order to specify the encoding of inline binary data.

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

RS origin servers shall support the transfer syntax parameter.

Transfer syntax parameters are forbidden in URI requests and responses.

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
```

A DICOM Media Type may only have one transfer syntax parameter and it shall have only one 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. E.g., to specify that 1.2.840.10008.1.2.4.50 and to specify that 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 wildcard value "*" 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.

If an Origin server supports the transfer syntax parameter, it shall support the wildcard value.

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

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

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, which shall have only a single value.

The syntax is:

```
charset-mtp = 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 Section 6.1.1.5) and the transfer syntax query parameter 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 media type parameter contained in the accept query parameter (see Section 6.1.1.5)
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 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 and RS APIs support the following DICOM Media Types:

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

Support for the transfer syntax and charset media type parameters is required for RS services.

Support for the "transfer-syntax" and "charset" parameters is forbidden for URI Services (i.e., they may not present in the request or the response).

6.1.2 Character Sets

HTTP uses charset names to indicate or negotiate the character encoding of textual content in representations [RFC6365] Section 3.3.

Character sets may be identified using the value in the IANA Preferred MIME Name column in the IANA Character Set Registry http://www.iana.org/assignments/character-sets/character-sets.xhtml.

Character sets may be identified by using the DICOM Defined Terms for the character set. See Section C.12.1.1.2 “Specific Character Set” in PS3.3, and Section 6.1.2.3 “Encoding of Character Repertoires” in PS3.5.

The origin server shall support the "UTF-8" charset name for RS Retrieve Rendered, but is not required to support the DICOM Defined Term "ISO_IR 192".

The syntax is:

charset = token / defined-term / DQUOTE defined-term DQUOTE

Where

token	A case-insensitive charset name from the Preferred MIME Name in the IANA Character Set Registry.
defined-term	See Section C.12.1.1.2 “Specific Character Set” in PS3.3.

Some DICOM Defined Terms for character sets contain space characters; and shall be enclosed in double quotes in HTTP header fields and percent encoded in URLs.

The Conformance Statement shall document all supported character sets. The Retrieve Capabilities response for all RS Services shall also document all supported character sets.

A request without any <charset> query parameter or Accept-Charset header field implies that the user agent will accept any charset in the response.

Annex D contains a mapping of some Specific Character Set (0008,0005) Defined Terms to IANA charset tokens.

6.1.2.1 Acceptable Character Sets

The term Acceptable Character Sets denotes the character sets that are acceptable to the user agent in the response. The Acceptable Character Sets are those specified in:

- the "charset" media type parameter
- the <character-set> query parameter
- the Accept-Charset header field
- the default character set for the media type, if any

When the Acceptable Character Sets contains a list of one or more Defined Terms they should be ordered as specified in Section C.12.1.1.2 “Specific Character Set” in PS3.3 and Section 6.1.2.3 “Encoding of Character Repertoires” in PS3.5. This is especially important for ISO 2022 character sets.

Any Accept-Charset header field values that are not defined in Annex D or not supported shall be ignored.

6.1.2.2 Character Set Query Parameter

The character set query parameter is primarily designed for use in hyperlinks (URLs) embedded in documents, where the Accept-Charset header field is not accessible.

The character set query parameter has the following syntax:

charset-qp = name "=" 1#(charset [weight])

The character set query parameter value is a comma-separated list of one or more charsets. It is similar to the Accept-Charset header field, except that it shall not have wildcards. It shall be supported by the origin server. It is optional for the user agent.

All charsets present in the character set query parameter may have a corresponding character set in the Accept-Charset header field, either explicitly or implicitly through wildcards.

The name of the character set query parameter is defined by the Service. Table 6.1.2-1 contains the names of the character set query parameter for some services.

Table 6.1.2-1. Character Set Query Parameter Name by Service

Service	Name
URI	name = "charset"
RS Studies	name = "charset"

If this parameter has a value that is not supported it shall be ignored.

See Section 6.1.4.

6.1.2.3 Accept-Charset Header Field

The Accept-Charset header field has the following syntax:

Accept-Charset = 1#(charset [weight]) / ("*" [weight])

The user agent may provide a list of Acceptable Character Sets in the Accept-Charset header field of the request. Its value is a comma-separated list of one or more <charset>s and/or the wildcard value ("*"). It shall be supported by the origin server. It is optional for the user agent.

The values of the Accept-Charset header field values are prioritized by their <weight> parameter.

If no wildcard ("*") is present, then any character sets not explicitly mentioned in the header field are considered "not acceptable" to the client.

A request without an Accept-Charset header field implies that the user agent will accept any charset in response.

If the media type defines a "charset" parameter, it should be included with the media type in the Accept header field, rather than in the Accept-Charset header field.

Any Accept-Charset header field values that are not supported shall be ignored.

6.1.2.4 Selected Character Set

The origin server shall determine the Selected Character Set(s) as follows:

1. Select the first supported character set in the "charset" parameter(s) of the Selected Media Type.
2. Otherwise, select the highest priority supported <charset> in the <character-set> query parameter.
3. Otherwise, select the highest priority supported <charset> in the Accept-Charset header field.
4. Otherwise, if the Selected Media Type has a default character set that is supported, select it.
5. Otherwise, select UTF-8.

Rendered representations returned in the response shall have all contained strings returned in the Selected Character Sets.

If the character set in which the target resource is encoded is not the Selected CharacterSet:

- If the origin server supports transcoding all glyphs used in the target resource into the Selected Character Set,it shall transcode the response payload into the Selected Character Set
- Otherwise, the origin server shall return 406 (Not Acceptable).

Note

This means that some SOP Instances may be convertible and others will not be, even though they have the same Specific Character Set (0008,0005).

All origin servers shall support conversion to the UTF-8 character set for RS Rendered Retrieve.

If the user agent chooses to perform its own conversion rather than have it done by the origin server:

1. The user agent may omit the Accept-Charset header field or send the "*" wildcard
2. The user agent may transcode the character set replacing all unknown characters with a suitable replacement. For example:
   - A question mark ("?"), or other similar character indicating an unknown character.
   - The corresponding Unicode Code Point for the character, represented as "U+xxxx".
   - The four characters "\nnn", where "nnn" is the 3 digit octal representation of each byte (see Section 6.1.2.3 “Encoding of Character Repertoires” in PS3.5).

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 +transfer-syntax-mtp +charset-mtp

If there is a conflict between the Transfer Syntax specified in the media type and the one specified in the File Meta Information Transfer Syntax UID (0002,0010) attribute, the latter has precedence.

6.1.4 Content-type Header Field

[RFC3986] does not permit an empty query component, i.e., if the "?" appears in the Target URI, then there shall be at least one Query Parameter in the URI.

The origin server may define and support additional Query Parameters, or additional Query Parameter values for an existing Query Parameter. If an origin server defines new or extends existing Query Parameters, they shall be included in the Conformance Statement and, if the service supports it, the Retrieve Capabilities response.

The origin server shall ignore any unsupported Query Parameters. The origin server shall process the request as if the unsupported parameters were not present, and should include a payload containing an appropriate warning or error message.

If a supported Query Parameter has an invalid value, the origin server shall return a 400 (Bad Request) error response, and should include should include a payload containing an appropriate warning or error message.

6.2 WADO-URI Request

The HTTP Request used shall use the GET method as defined in [RFC7230].

6.2.1 Parameters of the HTTP Request

The parameters of the <query> component of the Request-URI to be sent to the web Server through the HTTP GET method request shall be represented as defined in [RFC3986].

Note

1. Other components of the Request-URI depend on the configuration, e.g., location and script language of the origin server.
2. The means by which the user agent obtains the value of the necessary parameters for web accessing of DICOM Objects is out of the scope of the standard.

6.2.2 Media Types Acceptable in the Response

6.2.2.1 Query Parameters

See Section 6.1.4.

6.2.2.1.1 Accept Query Parameter

Specifies the Acceptable Media Types for the response payload. See Section 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).

The transfer-syntax and charset media type parameters are forbidden in the request.

Note

1. WADO-URI origin servers support transfer syntax and charset query parameters, which have been used instead of transfer-syntax and charset media type parameters since the inception of the service, even though this is different from the approach used by the later introduced WADO-RS service, which uses transfer-syntax and charset media type parameters instead of query parameters.
2. Only one transferSyntax query parameter is permitted and it may have only one UID value. See Section 8.2.11.

6.2.2.1.2 Character Set Query Parameter

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

charset-qp = %s"charset" "=" 1#(charset [weight])

6.2.2.2 Header Fields

6.2.2.2.1 Accept

The Accept header field specifies the media type(s) acceptable to the user agent in the response. It shall be present. See Section 6.1.1.6 for details.

6.2.2.2.2 Accept-Charset

The Accept-Charset header field specifies the character set(s) acceptable to the user agent in the response. It is optional. See Section 6.1.2.3 for details.

6.2.3 Reserved

6.3 WADO-URI Response

The response shall be an HTTP Response Message as specified in [RFC7230].

Note

The content of the message-body varies according to the Media type as defined below.

6.3.1 Body of Single DICOM MIME Subtype Part Response

6.3.1.1 Media Type

The media type shall be 'application/dicom', as specified in [RFC3240].

6.3.1.2 Payload

The body content shall be a DICOM File that includes File Meta Information as defined in PS3.10.

6.3.1.3 Transfer Syntax

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 6.1.1.8.4.

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).

The transfer-syntax and charset media type parameters are forbidden in the request.

Note

WADO-URI user agents may not depend on the presence of transfer syntax and charset media type parameters, since these have been absent since the inception of the service and are forbidden, even though this is different from the approach used by the later introduced WADO-RS service, which returns transfer-syntax and charset media type parameters in the response. The Transfer Syntax used can be determined from the PS3.10 File Meta Information.

6.3.2 Body of Non-DICOM Media Type Response

6.3.2.1 Media Type

The media type shall be one on the media types defined in the contentType parameter, preferably the most desired by the user agent, and shall be in any case compatible with the Accept header field of the GET method.

Note

The HTTP behavior is that an error (406 - Not Acceptable) is returned if the required media type cannot be served.

6.3.2.2 Content

The content shall be a single MIME part containing the object to be retrieved.

Note

Multiple objects in a response are not supported by this standard. The parameters select only a single object to retrieve. Most current user agents are able to retrieve single objects, within a "non multipart" MIME body, and are not able to support multipart/related or multipart/mixed responses.

6.4 Retired

See PS3.18-2017b.

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.

2. 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.

3. 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.

4. 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.

5. RetrieveBulkdata

   This action retrieves the bulk data for a given BulkDataURI.

6. 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. The syntax is:

  accept = "accept=" 1#media-type

- "charset" The <character-set> query parameter is specified in Section 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) specifying the Acceptable Media Types.

WADO-RS requests may optionally support the "Accept-Charset" header field. See Section 6.1.2.3.

See Section 6.1.4.

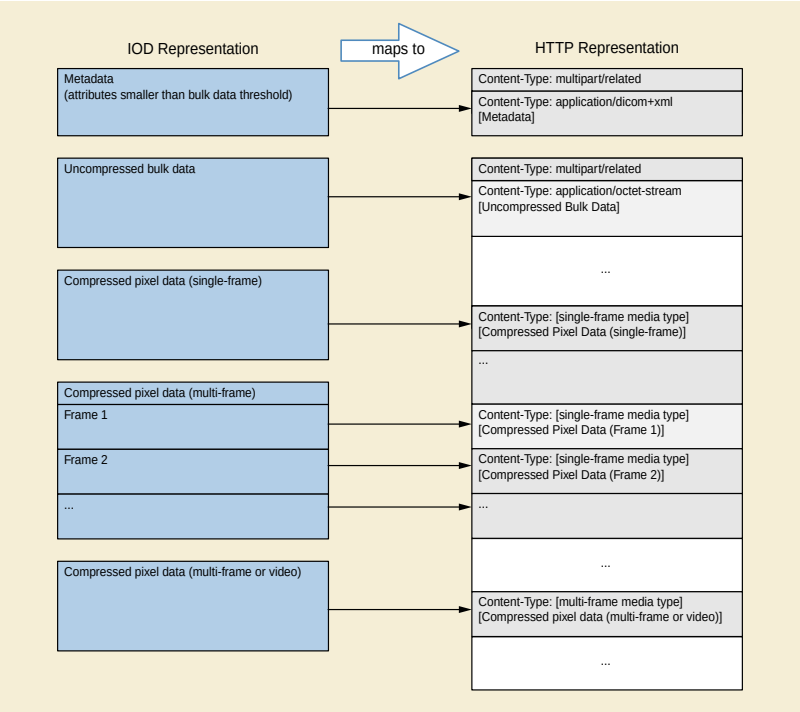

Figure 6.5-1. Mapping between IOD and HTTP message parts

Responses shall be encoded in the following manner: (see 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.
- XML as described in the Native DICOM Model defined in PS3.19 with one message part per XML object.
- JSON as a DICOM JSON Model Object as defined in Annex F.
- Uncompressed bulk and pixel data in a Little Endian format using the application/octet-stream media type with one message part per bulk data item.
- Compressed pixel data encoded as:
  - Single-frame pixel data using a single-frame media type (one message part)
  - Multi-frame pixel data using a single-frame media type (one frame per message part)
  - Multi-frame or video pixel data 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 application/dicom media type and transfer syntaxes specified in Table 6.1.1.8-2.

The request header field Content-Type is used to indicate the media type of the payload.

If the origin server returns XML or JSON responses that contain bulk data references, the origin server is required to support uncompressed bulk 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 media type parameters.

6.5.1 WADO-RS - 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.

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" [dcm-parameters]

      Specifies that the response can be DICOM Instances encoded in PS3.10 format. If transfer-syntax is not specified in the dcm-parameters the origin 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).
    - 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="{media-type}" [dcm-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. 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/jpx"; transfer-syntax=1.2.840.10008.1.2.4.92,, multipart/related; type="image/jpx"; transfer-syntax=1.2.840.10008.1.2.4.93, multipart/related; type="image/jpeg"

    or alternatively, multiple Accept headers:

    Accept: multipart/related; type="image/jpx"; transfer-syntax=1.2.840.10008.1.2.4.92,

    Accept: multipart/related; type="image/jpx"; transfer-syntax=1.2.840.10008.1.2.4.93

    Accept: multipart/related; type="image/jpeg"

6.5.1.2 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 metadata for the Study.

The Server shall return the document(s), or an error code when the document(s) cannot be returned. If the server cannot convert all of the data to any of the requested media types/Transfer Syntaxes, then an error code shall be returned, either a "Not Acceptable" response if no data is returned or a "Partial Content" response if only some data is returned.

The client can compare the SOP Instance UIDs or BulkDataURIs in the metadata and the message response to determine which bulk data elements have been returned.

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

6.5.1.2.1 DICOM Response

- Content-Type:
  - multipart/related; type=application/dicom; boundary={MessageBoundary}
- The entire multipart response contains every instance for the specified Study that can be converted to one of the requested Transfer Syntaxes.
- Each 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="{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-stream
    - Content-Location: {BulkDataURI}
  - an Encapsulated Document (0042,0011) bulk data element from a SOP Instance in the Study encoded in the media type specified in MIME Type of Encapsulated Document (0042,0012) in the Instance with the following header fields:
    - Content-Type: {media-type}
    - Content-Location: {BulkDataURI}
  - a compressed bulk data element from a SOP Instance in the Study encoded in a single-frame compression {MediaType} with the following headers:
    - Content-Type: {media-type} [dcm-parameters]

- Content-Location: {BulkDataURI}

- 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} [dcm-parameters]
  - Content-Location: {BulkDataURI}/frames/{FrameNumber}

    Note

    Each frame will come in a separate part.

- all of the compressed frames from a SOP Instance in the Study encoded in a video media type with the following headers:
  - Content-Type: {media-type} [dcm-parameters]
  - Content-Location: {BulkDataURI}

6.5.2 WADO-RS - 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.

6.5.2.1 Request

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

- Resource
  - {SERVICE}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}, 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.
    - {SeriesInstanceUID} is the series instance UID for a single series.
- 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" [dcm-parameters]

      Specifies that the response can be DICOM Instances encoded in PS3.10 format. If transfer-syntax is not specified in the dcm-parameters the origin 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).
    - 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="{media-type}" [dcm-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. See Section 6.1.3.

6.5.2.2 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 return the document(s), or an error code when the document(s) cannot be returned. If the server cannot convert all of the data to any of the requested media types/Transfer Syntaxes, then an error code shall be returned, either a "Not Acceptable" response if no data is returned or a "Partial Content" response if only some data is returned.

The client can compare the SOP Instance UIDs or BulkDataURIs in the metadata and the message response to determine which bulk data elements have been returned.

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

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 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}
  - multipart/related; type="{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: {BulkDataURI}
  - an Encapsulated Document (0042,0011) bulk data element from a SOP Instance in the Series encoded in the media type specified in MIME Type of Encapsulated Document (0042,0012) in the Instance with the following header fields:
    - Content-Type: {media-type}
    - Content-Location: {BulkDataURI}
  - 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: {media-type} [dcm-parameters]

    - Content-Location: {BulkDataURI}
- 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: {media-type} [dcm-parameters]
    - Content-Location: {BulkDataURI}/frames/{FrameNumber}
- all of the compressed frames from a multi-frame SOP Instance in the Series encoded in a video media type with the following headers:
    - Content-Type: {media-type} [dcm-parameters]
    - Content-Location: {BulkDataURI}

6.5.3 WADO-RS - 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.

6.5.3.1 Request

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

- Resource
    - {SERVICE}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}/instances/{SOPInstanceUID}, 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.
        - {SeriesInstanceUID} is the series instance UID for a single series.
        - {SOPInstanceUID} is the SOP Instance UID for a single SOP Instance.
- 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" [dcm-parameters]

          Specifies that the response can be DICOM Instances encoded in PS3.10 format. If transfer-syntax is not specified in the dcm-parameters the origin 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).
        - 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="{media-type}" [dcm-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. See Section 6.1.3.

6.5.3.2 Response

The Server shall provide either a single DICOM PS3.10 object for the SOP Instance or one or more bulk data items. 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 return the document(s), or an error code when the document(s) cannot be returned. If the server cannot convert all of the bulk data to any of the requested media types, then an error code shall be returned, either a "Not Acceptable" response if no data is returned or a "Partial Content" response if only some data is returned.

The client can compare the BulkDataURIs in the metadata and the message response to determine which bulk data elements have been returned.

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

6.5.3.2.1 DICOM Response

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

See Section 6.1.3.

6.5.3.2.2 Bulk Data Response

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

  See Section 6.1.3.
- 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: {BulkDataURI}
  - an Encapsulated Document (0042,0011) bulk data element encoded in the media type specified in MIME Type of Encapsulated Document (0042,0012) in the Instance with the following header fields:
    - Content-Type: {media-type}
    - Content-Location: {BulkDataURI}
  - a compressed bulk data element from a SOP Instance encoded in a single-frame media type with the following headers:
    - Content-Type: {media-type} [dcm-parameters]
    - Content-Location: {BulkDataURI}
  - a compressed frame from a multi-frame SOP Instance encoded in a single-frame media type with the following headers:

- Content-Type: {media-type} [dcm-parameters]
- Content-Location: {BulkDataURI}/frames/{FrameNumber}

- all of the compressed frames from a multi-frame SOP Instance encoded in a video media type with the following headers:
  - Content-Type: {media-type} [dcm-parameters]
  - Content-Location: {BulkDataURI}

6.5.4 WADO-RS - RetrieveFrames

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

6.5.4.1 Request

The specific Services resources to be used for the RetrieveFrames action shall be as follows:

- Resource
  - {SERVICE}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}/instances/{SOPInstanceUID}/frames/{FrameList}, 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.
    - {SeriesInstanceUID} is the series instance UID for a single series.
    - {SOPInstanceUID} is the SOP Instance UID for a single SOP Instance.
    - {FrameList} is a comma or %2C separated list of one or more non duplicate frame numbers. These may be in any order (e.g., ../frames/1,2,4,3).
- Method
  - GET
- Headers
  - Accept
    - multipart/related; type="application/octet-stream" [dcm-parameters]

      Specifies that the response can be Little Endian uncompressed pixel data
    - multipart/related; type="{media-type}" [dcm-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. See Section 6.1.3.

6.5.4.2 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 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 has a media type of multipart/related with a message boundary separator.

6.5.4.2.1 Pixel Data Response

- Content-Type:
  - multipart/related; type="application/octet-stream"; boundary={MessageBoundary} [dcm-parameters]
  - multipart/related; type="{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: {BulkDataURI}[/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} [dcm-parameters]
    - Content-Location: {BulkDataURI}/frames/{FrameNumber}
  - all of the compressed frames encoded in a video media type (as specified in Table 6.1.1.8-3b) with the following headers:
    - Content-Type: {media-type} [dcm-parameters]
    - Content-Location: {BulkDataURI}[/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.

6.5.5 WADO-RS - RetrieveBulkdata

This action retrieves the bulk data for a given BulkDataURI.

6.5.5.1 Request

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

- Resource
  - {BulkDataURI}, where
    - {BulkDataURI} is the URL of a bulk data element. This may be the URI attribute of a BulkData element received in response to a WADO-RS RetrieveMetadataRequest.
- Method
  - GET
- Headers
  - Accept
    - 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="{media-type}" [dcm-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. See Section 6.1.3.

- Range
  - See [RFC7233] Section 3.1. 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.

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 media 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.

  See Section 6.1.3.
- The entire multipart response contains all bulk data that can be converted to one of the requested media types.
- 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: {BulkDataURI}
  - an Encapsulated Document (0042,0011) bulk data element from a SOP Instance in the Study encoded in the media type specified in MIME Type of Encapsulated Document (0042,0012) in the Instance with the following header fields:
    - Content-Type: {media-type}
    - Content-Location: {BulkDataURI}
  - a compressed bulk data element from a SOP Instance encoded in a single-frame media type with the following headers:
    - Content-Type: {media-type} [dcm-parameters]

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

- a compressed frame from a multi-frame SOP Instance encoded in a single-frame media type with the following headers:
  - Content-Type: {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}/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} [dcm-parameters]

    where {media-type} is of compressed pixel data encoded as specified in Table 6.1.1.8-3b.
  - 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 [RFC7233] Section 4.

6.5.6 WADO-RS - RetrieveMetadata

This action retrieves the DICOM instances presented as the study, series, or instance metadata with the bulk data removed. The response is metadata for the DICOM attributes.

The study, series, or instance metadata includes all attributes; however, a RESTful Service is permitted to replace the Value Field of an attribute with a BulkDataURI for attributes with Value Representations (VR) of DS, FL, FD, IS, LT, OB, OD, OF, OL, OW, SL, SS, ST, UC, UL, UN, US, and UT. The client can use the BulkDataURI with the RetrieveBulkData action to retrieve the original Value Field of that attribute.

Note

1. The server is not required to replace any attribute with a BulkDataURI; this is intended to allow the server to provide clients with metadata of a reasonably small size by leaving out large data Value Fields.
2. OB, OD, OF, OL, OW and UN Attributes not replaced with a BulkDataURL are encoded as XML Base64 binary values.
3. Some DICOM instances, such as SR documents, may be entirely described in the metadata.

6.5.6.1 Request

The specific Services resources to be used for the RetrieveMetadata action shall be as follows:

- Resources
  - {SERVICE}/studies/{StudyInstanceUID}/metadata
  - {SERVICE}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}/metadata
  - {SERVICE}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}/instances/{SOPInstanceUID}/metadata

  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.
  - {SeriesInstanceUID} is the series instance UID for a single series.
  - {SOPInstanceUID} is the SOP Instance UID for a single SOP Instance.

- Method
  - GET
- Headers
  - Accept
    - multipart/related; type="application/dicom+xml"

      Specifies that the response should be PS3.19 XML. WADO-RS origin servers shall support this Media Type. See Table 6.1.1.8-1b.

    - application/dicom+json

      Specifies that the response should be DICOM JSON (see Annex F). WADO-RS origin servers shall support this Media Type. See Table 6.1.1.8-1b.

6.5.6.2 Response

The Server shall provide the document(s) indicated in the request. The Server shall return the document(s) or an error code when the document(s) could not be returned.

The response has a media type of either:

- multipart/related; type="application/dicom+xml", as described in the Native DICOM Model defined in PS3.19, or
- application/dicom+json, as described in Annex F.

The response must include the URL attribute for each BulkData element.

Note

The metadata is consistent with the characteristics of the bulk data on the server. If bulk data is requested using specified Transfer Syntaxes or media types, it is possible that the bulk data retrieved may be inconsistent with the metadata. For example, for a Study whose DICOM Tag (0028,2110) "LossyImageCompression" is set to "00", indicating no lossy compression, calling RetrieveStudy and requesting a lossy compression media type will provide pixel data that is inconsistent with the metadata. It is the responsibility of the client to deal with these inconsistencies appropriately.

6.5.6.2.1 XML Metadata Response

- Content-Type:
  - multipart/related; type="application/dicom+xml" [dcm-parameters]
- The entire multipart response contains all XML metadata for the specified Study, Series, or Instance.
- Each item in the response is the XML encoded metadata for an Instance with the following http headers:
  - Content-Type: application/dicom+xml; [dcm-parameters]

    Where the transfer-syntax in the dcm-parameters is the UID of the DICOM Transfer Syntax used to encode the inline binary data in the XML metadata.

6.5.6.2.2 JSON Metadata Response

- Content-Type:
  - application/dicom+json [dcm-parameters]

    Where the transfer-syntax in the dcm-parameters is the UID of the DICOM Transfer Syntax used to encode the inline binary data in the JSON metadata.

- 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).

6.5.7 Error Codes

The following error codes are defined and shall be used to report any of the associated error and warning situations. Other error codes may be present for other error and warning situations.

Table 6.5-2. Error Codes

Client Error Code	Client Error Name	Error Situation
206	Partial Content	Accept type, Transfer Syntax or decompression method supported for some but not all requested content.
400	Bad Request	Malformed resource
404	Not Found	Specified resource does not exist
406	Not Acceptable	Accept type, Transfer Syntax or decompression method not supported
410	Gone	Specified resource was deleted
503	Busy	Service is unavailable

6.5.8 WADO-RS - 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 primary differences are the resource component and the query parameters.

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.

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 resource.
{?parameter*}	Zero or more query parameters as defined in Section 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.

6.5.8.1.1 Target Resources

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 Rendered Media Type.

6.5.8.1.2 Query Parameters

The query parameters defined in this section specify various rendering transformations to be applied to the images and video contained in the target resource.

The origin server shall support all of the query parameters defined in this section. An origin server may define additional parameters. If additional parameters are defined, they shall be documented in the Conformance Statement and in the Retrieve Capabilities response. The origin server shall ignore any unknown parameters.

The following rules pertain to all parameters defined in this section:

1. All parameters are optional for the user agent.
2. All parameters are required to be supported by the origin server.
3. These parameters only apply to resources that are images and video.
4. Instances that are not images will be rendered in an Acceptable Media Type, if one exists; otherwise, they will not be rendered.
5. The set of transformations specified by the parameters in this section shall be applied to the images as if they were a Presentation State, that is, in the order specified by the applicable image rendering pipeline specified in PS 3.4.

See Section 6.1.4.

Table 6.5.8-2. Retrieve Rendered Query Parameters

Key	Values	Target Resource	Section
annotation	"patient" and/or "technique"	All	6.5.8.1.2.1
charset	token	All	6.1.2.2
quality	integer	All	6.5.8.1.2.2
viewport	vw, vh, [sx, sy, sw, sh]	Non-Presentation States	6.5.8.1.2.3
viewport	vw, vh,	Presentation States	6.5.8.1.2.3
window	center, width, shape	Non-Presentation States	6.5.8.1.2.4

6.5.8.1.2.1 Image Annotation

The annotation parameter specifies that the rendered images shall be annotated with patient and/or procedure information. Its value is a comma-separated list of one or more keywords. It has the following syntax:

%s"annotation=" 1#(%s"patient" / %s"technique")

Where

"patient"	indicates that the rendered images shall be annotated with patient information (e.g., patient name, birth date, etc.).
"technique"	indicates that the rendered images shall be annotated with information about the procedure that was performed (e.g., image number, study date, image position, etc.).

When this parameter is not present, no annotations shall be applied.

The origin server shall apply the annotations after all other parameters have been applied.

The origin server may support additional keywords, which should be included in the Conformance Statement and the Retrieve Capabilities response.

The origin server shall ignore any unsupported parameter values.

Note

1. The exact nature and presentation of the annotation is determined by the origin server. The annotation is burned into the rendered image pixels.
2. A user agent wanting more control over annotations may retrieve an image, omitting the "annotation" parameter; and separately retrieve the metadata; and create customized annotations on the image.

6.5.8.1.2.2 Image Quality

The "quality" parameter specifies the requested quality of the rendered images. It has the following syntax:

%s"quality=" integer

Where

integer	is an unsigned integer between 1 and 100 inclusive, with 100 being the best quality.

If the value of this parameter is less than 1 or greater than 100, the response shall be a 400 (Bad Request), and should include a payload containing an appropriate error message.

The "quality" parameter is only supported for media types that allow lossy compression.

Note

1. Decompression and re-compression may degrade the image quality if the original image was already irreversibly compressed. If the image has been already lossy compressed using the same format as required (e.g., jpeg), it may be sent as it is without decompressing and re-compressing it.
2. The specific interpretation of the meaning of this parameter is determined by the origin server.

6.5.8.1.2.3 Scaling Regions of Source Images to a Viewport

The "viewport" parameter specifies a rectangular region of the source image(s) to be cropped, and a rectangular region corresponding to the size of the user agent's viewport to which the cropped image should be scaled.

If the target resource is a Presentation State Instance, the syntax for this parameter is:

%s"viewport=" vw "," vh

Otherwise it is:

%s"viewport=" vw "," vh ["," [sx] "," [sy] "," [sw] "," [sh]]

Where

vw and vh	are positive integers specifing the width and height, in pixels, of the rendered image.
sx and sy	are decimal numbers whose absolute values specify, in pixels, the top-left corner of the region of the source image(s) to be rendered; if either <sx> or <sy> is not specified it defaults to 0.
sw and sh	are decimal numbers whose absolute values specify, in pixels, the width and height of the region of the source image(s) to be rendered; if <sw> is not specified, the origin server shall render to the right edge of the source image; if <sh> is not specified, the origin server shall render to the bottom edge of the source image; if <sw> is a negative value, the image is flipped horizontally; if <sh> is a negative value, the image is flipped vertically.

The source image region parameters (sx, sy, sw, and sh) shall not be present when rendering a Presentation State Instance. If they are present the origin server shall return a 409 (Conflict).

The origin server shall first crop, if specified, then scale the source images, maintaining their original aspect ratio, until either the rendered image width is the same as the viewport width or the image height is the same as the viewport height, whichever avoids truncation. In other words, viewport scaling makes the image(s) as large as possible, within the viewport, without overflowing the viewport area and without distorting the image.

If any of the optional parameter values are not present the default value shall be used. Individual values may be elided, but the commas between the values shall be present. For example:

viewport=512,512,,,512,512

The missing <sx> and <sy> parameter values shall default to 0.

If trailing values are elided, then the trailing commas shall be omitted. For example:

viewport=1024,1024

The missing <sx>, <sy>, <sw>, <sh> will have their default values, which means the image(s) will not be cropped, and the full image will be rendered.

If the viewport parameter is not present, the rendered image(s) shall not be scaled, i.e., the rendered image(s) shall contain the same sized pixel matrix as the source DICOM image.

If this parameter specifies an ill-defined source region or viewport, the origin server shall return a 400 (Bad Request) response, and should include a payload containing an appropriate error message.

Note

The default values for <sx> and <sy> differ from the defaults in the Specified Displayed Area in Presentation States, which uses integer values with the top left corner being (1,1). See Section C.10.4 “Displayed Area Module” in PS3.3.

6.5.8.1.2.4 Windowing

The "window" parameter controls the windowing of the images as defined in Section C.8.11.3.1.5 “VOI Attributes” in PS3.3. It has the following syntax:

%s"window=" center "," width "," function

center	is a decimal number containing the window-center value
width	is a decimal numbercontaining the window-width value
function	is one of the following keywords: "linear", "linear-exact", or "sigmoid".

Note

These correspond to the differently capitalized and punctuated values of VOI LUT Function (0028,1056). See Section C.11.2.1.2 “Window Center and Window Width” in PS3.3.

All three parameter shall be present with valid values.

If any of the parameter values are missing or invalid, the origin server shall return a 400 (Bad Request) response, and should include a payload containing an appropriate error message.

If the target resource is a Presentation State, this parameter shall not be used. If this parameter is present when the target resource is a Presentation state, the origin server shall return a 400 (Bad Request).

6.5.8.1.3 Header Fields

Required: Accept

The values of the Accept header field shall be one or more Rendered Media Types.

6.5.8.1.4 Payload

This request has no payload.

6.5.8.2 Behavior

The target resource(s) are rendered according to the query parameters, by applying the transformations according to the appropriate rendering pipeline specified in Section N.2 “Pixel Transformation Sequence” in PS3.4.

It the target resource is not a single instance, Presentation State Instances contained in the target resource shall not be rendered.

Rendered images shall contain no more than 8 bits per channel.

6.5.8.2.1 Presentation State Instance

If the target resource is a Presentation State Instance, that instance may contain references to one or more series, each of which may contain one or more instances, each of which may contain one or more frames. The response shall return rendered versions of all supported Instances and frames referenced by the Presentation State Instance.

For example, if the Presentation State instance references a multi-frame image, then the response will contain all frames specified by the target resource, or if the Presentation State instance references a series, then the response will contain all instances contained in that series.

If the target resource is a Presentation State Instance, then only the Charset, Annotation, Quality, and Viewport parameters may also be present. If any other Retrieve Rendered Transaction Query Parameters are present the response shall be 400 (Bad Request), and should include a payload containing an appropriate error message.

If the Presentation State Instance contains a Blending Sequence, then the rendered images in the response shall correspond to the frames of the input that have a Blending Sequence Item with a Blending Position (0070,0405) value of UNDERLYING. See Section C.11.14.1.1 “Blending Sequence” in PS3.3.

The origin server shall render all of the images referenced by the Presentation State in an Acceptable Media Type using the rendering pipeline specified in PS3.4.

If there is more than one image in the response they shall be ordered according to the:

1. Dimension Index Values (0020,9157) attribute, if present
2. Image Position (Patient) (0020,0032) attribute, if present
3. Image Position Volume (0020,9301), if present
4. Order of the instance references in the presentation state

If the above does not fully specify the ordering of the frames, then the origin server shall resolve any remaining ambiguity in the ordering.

If the Presentation Size Mode is TRUE SIZE it shall be treated as SCALE TO FIT.

If the Presentation Size Mode is SCALE TO FIT, the origin server shall scale the Specified Displayed Area in the Presentation State, maintaining its original aspect ratio, until either the rendered image width is the same as the viewport width or the rendered image height is the same as the viewport height, whichever comes first. In other words, viewport scaling makes the displayed area selection as large as possible, within the viewport, without overflowing the viewport area and without distorting the image. If the viewport parameter is not present, the returned images shall have the dimensions of the Specified Displayed Area.

If the Presentation Size Mode is MAGNIFY, then the referenced images shall be scaled to the Specified Displayed Area in the Presentation State, and then they shall be cropped to the size specified by the "viewport" parameter. If the request does not contain a "viewport" parameter, then the referenced images shall not be cropped.

Any Specified Displayed Area relative annotations in the Presentation State shall be rendered relative to the Specified Displayed Area within the Presentation State, not the size of the viewport.

Though the output of the Presentation State is defined in DICOM to be in P-Values (grayscale values intended for display on a device calibrated to the DICOM Grayscale Standard Display Function PS3.14), the grayscale or color space for the rendered images is not defined by this standard.

6.5.8.3 Response

The Retrieve Rendered service has the following response message syntax:

```
version SP status-code SP reason-phrase CRLF
Content-Type: rendered-media-type CRLF
*(header-field CRLF)
CRLF
payload
```

Where

version	is the HTTP version, for example "HTTP/1.1"
rendered-media-type	is a Rendered Media Type; see Section 6.1.1.3.
payload	is one or more representations in a Rendered Media Type.

6.5.8.3.1 Status Codes

The response shall include a status code from Table 6.5.8-3, if applicable; otherwise, an appropriate status code shall be used.

Table 6.5.8-3. Common Status Codes

Status Code	Meaning
200 Success	The origin server successfully rendered and is returning representations for the resource.
206 Partial Content	The origin server successfully rendered and is returning representations for part, but not all, of the resource.
406 Not Acceptable	The origin server does not support any of the Acceptable Media Types.
413 Payload Too Large	The target resource is too large to be rendered by the origin server.

6.5.8.3.2 Header Fields

Required: Content-Type

The value of the Content-Type header field shall be a Rendered Media Type.

6.5.8.3.3 Payload

The origin server shall include all successfully rendered representations in the payload.

Rendered images that do not contain a color management profile (e.g., an ICC profile), shall be assumed to be in sRGB space.

6.5.8.4 Media Types

The origin server shall be capable of returning representations in Rendered Media Types identified as default and required in Section 6.1.1.3.

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 multipart messages. The organization of SOP Instances into message parts depends on whether the SOP Instances are structured as PS3.10 binary instances, or metadata and bulk data.

PS3.10 binary instances shall be encoded with one message part per DICOM Instance.

When the request message contains compressed bulk data with a Content Type that is one of the media types specified in Table 6.6-1, the request may omit the Image Pixel Description Macro attributes and the origin server will derive them from the compressed bit stream. Some media types do not directly correspond to a DICOM Transfer Syntax and the origin server will transform the received bit stream into an uncompressed or lossless (reversibly) compressed Transfer Syntax.

Note

1. This allows a user agent to use consumer media types to encode the pixel data even though it may not have:
   - the pixel data in a form that directly corresponds to a lossless (reversible) DICOM Transfer Syntax, or
   - an API to access the information required to populate the Image Pixel Description Macro.
2. If the supplied compressed bit stream is in a lossless (reversible) format, the intent is to allow full fidelity retrieval of the decompressed pixels, not the format in which it happened to be submitted.
3. If the supplied compressed bit stream is in a lossy (irreversible) format, there will be a corresponding DICOM Transfer Syntax, and the origin server is not expected to recompress it causing further loss.

Metadata and bulk data requests will be encoded in the following manner:(see Figure 6.5-1 Mapping between IOD and HTTP message parts):

- All XML request messages shall be encoded as described in the Native DICOM Model defined in PS3.19 with one message part per XML object; the attributes of the Image Pixel Description Macro may be omitted for the media types specified in Table 6.6-1.
- All JSON request messages shall be encoded as an array of DICOM JSON Model Objects defined in Annex F in a single message part; the attributes of the Image Pixel Description Macro may be omitted for the media types specified in Table 6.6-1.
- Bulk data (with the exception of encapsulated document element) and uncompressed 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)

- An Encapsulated Document (0042,0011) bulk data element shall be encoded using the media-type from the MIME Type of Encapsulated Document (0042,0012) attribute with one message part per bulk data item.

Compressed pixel data shall be encoded using the media types and transfer syntaxes specified in Table 6.1.1.8-3b. Media types corresponding to several DICOM Transfer Syntax UIDs may require a transfer-syntax parameter to convey the Transfer Syntax the compressed pixel data is encoded in.

The request header field Content-Type is used to indicate the media type of the payload.

The Service shall support uncompressed bulk data (multipart/related; type="application/octet-stream").

Table 6.6-1 contains a list of media types containing compressed pixel data from which an origin server shall be able to derive the Image Pixel Data Description Macro Attribute values.

Requirements are specified in Table 6.6-1 as follows:

- Transform - No DICOM Transfer Syntax exists; shall be transformed by the origin server into an uncompressed or lossless compressed Transfer Syntax (the choice of which is at the discretion of the origin server).
- Unchanged - Shall be encapsulated in the corresponding DICOM Transfer Syntax without further lossy compression

Table 6.6-1. Media Type Transformation to Transfer Syntaxes

Media Type	Requirement
image/gif	Transform
Image/jp2	Unchanged
image/jpeg	Unchanged
image/jpx	Unchanged
image/png	Transform
video/mp4	Unchanged
video/mpeg2	Unchanged

Note

1. In the case of pixel data supplied as image/gif or image/png, the origin server may transform the color representation from indexed color to true color (RGB) as necessary to conform to any Photometric Interpretation constraints specified by the IOD (i.e., if PALETTE COLOR is not permitted); such a transformation is considered lossless.
2. If the number of bits per channel of an image/png file is not supported by the IOD, a lossless transformation cannot be performed.
3. An animated image/gif will be converted into a multi-frame image; image/png does not support animation, and MNG is not included in Table 6.6-1.
4. Any transparency information present in an image/gif or image/png file will be discarded, since DICOM does not support the concept of transparency.
5. If an alpha channel is supplied in an image/png file, and the IOD does not support the RGBA Photometric Interpretation, the alpha channel will be discarded (i.e., considered to consist of all opaque values, consistent with the policy of discarding any transparency information).

6.6.1 STOW-RS - Store Instances

This action stores one or more DICOM instances associated with one or more study instance unique identifiers (SUID). The request message can be DICOM or metadata and bulk data depending on the "Content-Type", and is encapsulated in a multipart request body.

6.6.1.1 Request

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

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

      Note

      It is not necessary that the study referenced by the StudyInstanceUID in the resource (and in the provided instances) exists on the server, however it is necessary that it be a valid UID. The client may have obtained an appropriate UID from elsewhere or generated it as described in Chapter 9 “Unique Identifiers (UIDs)” in PS3.5 and Annex B “Creating a Privately Defined Unique Identifier (Informative)” in PS3.5.
- Method
  - POST
- Headers
  - Content-Type - The representation scheme being posted to the RESTful service. The types allowed for this request header are as follows:
    - multipart/related; type="application/dicom"; boundary={messageBoundary}

      Specifies that the post is PS3.10 binary instances. All STOW-RS providers shall accept this Content-Type.
    - multipart/related; type="application/dicom+xml"; boundary={messageBoundary}

      Specifies that the post is PS3.19 XML metadata and bulk data. All STOW-RS providers shall accept this Content-Type.
    - multipart/related; type="application/dicom+json"; boundary={messageBoundary}

      Specifies that the post is DICOM JSON metadata and bulk data. All STOW-RS providers 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}

- The multipart request body contains all the metadata and bulk data to be stored. If the number of bulk data parts does not correspond to the number of unique BulkDataURIs in the metadata then the entire message is invalid and will generate an error status line.
- Each body part is either DICOM PS3.19 XML metadata or a bulk data item from a SOP Instance sent as part of the Store operation. The first part of the multipart message shall be XML metadata.
- Each bulk data item shall be preceded by all metadata items that contain a reference to it.

  Note

  This requires that all bulk data items for an instance shall be preceded by the XML metadata for that instance and if a bulk data item is included in multiple instances it shall be preceded by the XML metadata for each instance in which it is included.
- The first part in the multipart request will contain the following HTTP headers:
  - Content-Type: application/dicom+xml; transfer-syntax={TransferSyntaxUID}
- Subsequent items will contain the following HTTP headers (order is not guaranteed):
  - additional metadata with the following headers:
    - Content-Type: application/dicom+xml; transfer-syntax={TransferSyntaxUID}

    Where {TransferSyntaxUID} is the UID of the DICOM Transfer Syntax used to encode the inline binary data in the XML metadata.
  - an encapsulated document with the following headers:
    - Content-Type: {media-type}
    - Content-Location: {BulkDataURI}
  - an uncompressed bulk data element encoded in Little Endian binary format with the following headers:
    - Content-Type: application/octet-stream
    - Content-Location: {BulkDataURI}
  - a compressed pixel data object from a SOP Instance in the Study with the following headers:
    - Content-Type: {media-type} [dcm-parameters]
    - Content-Location: {BulkDataURI}
- Metadata and its associated bulk data shall always be sent in the same POST request.

  Note

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

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). 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}

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

- Subsequent items will be one of the following:
  - an encapsulated document with the following headers:
    - Content-Type: {media-type}
    - Content-Location: {BulkDataURI}
  - an uncompressed bulk data element encoded in Little Endian binary format with the following headers:
    - Content-Type: application/octet-stream
    - Content-Location: {BulkDataURI}
  - a compressed pixel data object from a SOP Instance in the Study with the following headers:
    - Content-Type: {media-type} [dcm-parameters]
    - Content-Location: {BulkDataURI}
- JSON Metadata and its associated bulk data shall always be sent in the same POST request.

  Note

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

6.6.1.2 Action

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

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

Note

For more information on populating the Original Attribute Sequence, see Section C.12.1 “SOP Common Module” in PS3.3.

The origin server shall encapsulate or convert any compressed pixel data received as bulk data into an appropriate DICOM Transfer Syntax, as defined in Table 6.6-1.

The origin server shall populate the attributes of the Image Pixel Description Macro, if absent from the Metadata, by deriving them from the compressed pixel data received as bulk data.

The stored Instance(s) shall fully conform to the IOD and encoding requirements of PS3.3 and PS3.5, respectively.

The origin server shall return a status of 415 (Unsupported Media Type) if it cannot convert the bulk data or populate the Image Pixel Description Macro Attribute values.

6.6.1.3 Response

The RESTful Service shall return an HTTP status line, including a status code and associated textual phrase for the entire set of stored SOP Instances, followed by a message body containing the Store Instances Response Module as defined in Table 6.6.1-2. The message body shall be encoded as either:

- an XML object as described in the Native DICOM Model defined in PS3.19, or

- a DICOM JSON Model Object defined as defined in Annex F.

6.6.1.3.1 Response Status Line

If the status for all instances included in the POST request is Success, the RESTful Service shall return an "HTTP 200 - Success" response code.

If the status for all instances included in the POST request is Failure, the RESTful Service shall return an appropriate failure status line with a response code from Table 6.6.1-1. If there are instance specific errors, the response code shall be a 409 and the response payload shall contain the Store Instances Response Module, which contains additional information regarding instance errors.

In all other conditions, the RESTful Service shall return an "HTTP 202 - Accepted" response code. The response payload may contain a Store Instances Response Module, which specifies additional information regarding instance warnings or failures.

Table 6.6.1-1. HTTP Standard Response Code

Service Status	HTTP Status Codes	STOW-RS Description
Failure	400 - Bad Request	This indicates that the STOW-RS Service was unable to store any instances due to bad syntax.
	401 - Unauthorized	This indicates that the STOW-RS Service refused to create or append any instances because the client is not authorized.
	403 - Forbidden	This indicates that the STOW-RS Service understood the request, but is refusing to fulfill it (e.g., an authorized user with insufficient privileges).
	409 - Conflict	This indicates that the STOW-RS Service request was formed correctly but the service was unable to store any instances due to a conflict in the request (e.g., unsupported SOP Class or StudyInstanceUID mismatch). This may also be used to indicate that a STOW-RS Service was unable to store any instances for a mixture of reasons. Additional information regarding the instance errors can be found in the XML response message body.
	415 - Unsupported Media Type	This indicates that the STOW-RS Service does not support the Content-Type specified in the storage request (e.g., the service does not support JSON metadata).
	503 - Busy	This indicates that the STOW-RS Service was unable to store any instances because it was out of resources.
Warning	202 - Accepted	This indicates that the STOW-RS Service stored some of the instances but warnings or failures exist for others. Additional information regarding this error can be found in the XML response message body.
Success	200 - OK	This indicates that the STOW-RS Service successfully stored all the instances.

Note

HTTP Status Codes for Failures and Warnings are returned in HTTP response headers. It is recommended that the text returned in the HTTP Response Warning contain a DICOM Status Code and descriptive reason as defined in Section 6.6.1.3.2.1. For example,

Warning: "A700: Out of memory"

6.6.1.3.2 Response Message Body

The message body shall provide appropriate status codes for individual SOP Instances indicating success, warning, or failure as defined below.

The message body may also include details about the processing of attributes by the service.

The message body shall also include details of failures that are not associated with a specific SOP Instance.

Table 6.6.1-2 defines the Attributes for referencing SOP Instances that are contained in a Store Instances Response Module in the response message body.

Table 6.6.1-2. Store Instances Response Module Attributes

Attribute Name	Tag	Type	Attribute Description
Retrieve URL	(0008,1190)	2	The URL where the Study is available for retrieval via a WADO-RS Retrieve Study service. Note The VR of this attribute has changed from UT to UR.
Failed SOP Sequence	(0008,1198)	1C	A sequence of Items where each Item references a single SOP Instance for which storage could not be provided. Required if one or more SOP Instances failed to store.
>Table 10-11 "SOP Instance Reference Macro Attributes" in PS3.3			
>Failure Reason	(0008,1197)	1	The reason that storage could not be provided for this SOP Instance. See Section 6.6.1.3.2.1.2.
Referenced SOP Sequence	(0008,1199)	1C	A sequence of Items where each Item references a single SOP Instance that was successfully stored. Required if one or more SOP Instances were successfully stored.
>Table 10-11 "SOP Instance Reference Macro Attributes" in PS3.3			
>Retrieve URL	(0008,1190)	2	The URL where the SOP Instance is available for retrieval via a WADO-RS service. Note The VR of this attribute has changed from UT to UR.
>Warning Reason	(0008,1196)	1C	The reason that this SOP Instance was accepted with warnings. Required if there was a warning for this SOP Instance. See Section 6.6.1.3.2.1.1.
>Original Attributes Sequence	(0400,0561)	3	Sequence of Items containing all attributes that were removed or replaced by other values. One or more Items are permitted in this sequence.
>>Attribute Modification DateTime	(0400,0562)	1	Date and time the attributes were removed and/or replaced.
>>Modifying System	(0400,0563)	1	Identification of the system that removed and/or replaced the attributes.
>>Reason for the Attribute Modification	(0400,0565)	1	Reason for the attribute modification. Defined terms are: COERCE = Replace values of attributes such as Patient Name, ID, Accession Number, for example, during import of media from an external institution, or reconciliation against a master patient index. CORRECT = Replace incorrect values, such as Patient Name or ID, for example, when incorrect worklist item was chosen or operator input error.

Attribute Name	Tag	Type	Attribute Description
>>Modified Attributes Sequence	(0400,0550)	1	Sequence that contains all the Attributes, with their previous values, that were modified or removed from the main data set. Only a single Item shall be included in this sequence.
>>Any Attribute from the main data set that was modified or removed; may include Sequence Attributes and their Items.			
Other Failures Sequence	(0008,119A)	1C	Reasons not associated with a specific SOP Instance that storage could not be provided. Each Item references a single storage failure. Required if there are one or more failures not associated with a specific SOP Instance.
>Failure Reason	(0008,1197)	1	The reason that storage could not be provided for this message item. See Section 6.6.1.3.2.1.2.

6.6.1.3.2.1 Store Instances Response Attribute Description

6.6.1.3.2.1.1 Warning Reason

Table 6.6.1-3 defines the semantics for which the associated value shall be used for the Warning Reason (0008,1196):

Table 6.6.1-3. Store Instances Response Warning Reason Values

Status Code (hexadecimal)	Status Code (decimal)	Meaning	Explanation
B000	45056	Coercion of Data Elements	The STOW-RS Service modified one or more data elements during storage of the instance. See Section 6.6.1.3.
B006	45062	Elements Discarded	The STOW-RS Service discarded some data elements during storage of the instance. See Section 6.6.1.3.
B007	45063	Data Set does not match SOP Class	The STOW-RS Service observed that the Data Set did not match the constraints of the SOP Class during storage of the instance.

Additional codes may be used for the Warning Reason (0008,1196) to address the semantics of other issues.

In the event that multiple codes may apply, the single most appropriate code shall be used.

6.6.1.3.2.1.2 Failure Reason

Table 6.6.1-4 defines the semantics for which the associated value shall be used for the Failure Reason (0008,1197). Implementation specific warning and error codes shall be defined in the conformance statement:

Table 6.6.1-4. Store Instances Response Failure Reason Values

Status Code (hexadecimal)	Status Code (decimal)	Meaning	Explanation
A7xx	42752 - 43007	Refused out of Resources	The STOW-RS Service did not store the instance because it was out of resources.
A9xx	43264 - 43519	Error: Data Set does not match SOP Class	The STOW-RS Service did not store the instance because the instance does not conform to its specified SOP Class.
Cxxx	49152 - 53247	Error: Cannot understand	The STOW-RS Service did not store the instance because it cannot understand certain Data Elements.

Status Code (hexadecimal)	Status Code (decimal)	Meaning	Explanation
C122	49442	Referenced Transfer Syntax not supported	The STOW-RS Service did not store the instance because it does not support the requested Transfer Syntax for the instance.
0110	272	Processing failure	The STOW-RS Service did not store the instance because of a general failure in processing the operation.
0122	290	Referenced SOP Class not supported	The STOW-RS Service did not store the instance because it does not support the requested SOP Class.

Additional codes may be used for the Failure Reason (0008,1197) to address the semantics of other issues.

In the event that multiple codes may apply, the single most appropriate code shall be used.

6.6.1.3.2.2 Response Message Body Example

The following is an example of a PS3.18 XML Store Instances Response Module in the response message body containing 2 failed SOP Instances, 1 successful SOP Instance, and 1 accepted SOP Instance with a warning:

```
<?xml version="1.0" encoding="utf-8" xml:space="preserve"?>
<NativeDicomModel xmlns="http://dicom.nema.org/PS3.19/models/NativeDICOM"
xsi:schemaLocation="http://dicom.nema.org/PS3.19/models/NativeDICOM"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <DicomAttribute tag="00081198" vr="SQ" keyword="FailedSOPSequence">
  <Item number="1">
   <DicomAttribute tag="00081150" vr="UI" keyword="ReferencedSOPClassUID">
    <Value number="1">1.2.840.10008.3.1.2.3.1</Value>
   </DicomAttribute>
   <DicomAttribute tag="00081155" vr="UI"
   keyword="ReferencedSOPInstanceUID">
    <Value number="1">
    2.16.124.113543.6003.1011758472.49886.19426.2085542308</Value>
   </DicomAttribute>
   <DicomAttribute tag="00081197" vr="US" keyword="FailureReason">
    <Value number="1">290</Value>
   </DicomAttribute>
  </Item>
  <Item number="2">
   <DicomAttribute tag="00081150" vr="UI" keyword="ReferencedSOPClassUID">
    <Value number="1">1.2.840.10008.3.1.2.3.1</Value>
   </DicomAttribute>
   <DicomAttribute tag="00081155" vr="UI"
   keyword="ReferencedSOPInstanceUID">
    <Value number="1">
    2.16.124.113543.6003.1011758472.49886.19426.2085542309</Value>
   </DicomAttribute>
   <DicomAttribute tag="00081197" vr="US" keyword="FailureReason">
    <Value number="1">290</Value>
   </DicomAttribute>
  </Item>
 </DicomAttribute>
 <DicomAttribute tag="00081199" vr="SQ" keyword="ReferencedSOPSequence">
  <Item number="1">
   <DicomAttribute tag="00081150" vr="UI" keyword="ReferencedSOPClassUID">
    <Value number="1">1.2.840.10008.5.1.4.1.1.2</Value>
   </DicomAttribute>
   <DicomAttribute tag="00081155" vr="UI"
   keyword="ReferencedSOPInstanceUID">
    <Value number="1">
```

```
      2.16.124.113543.6003.189642796.63084.16748.2599092903</Value>
    </DicomAttribute>
    <DicomAttribute tag="00081190" vr="UR" keyword="RetrieveURL">
      <Value number="1">
      https://wadors.hospital.com/studies/2.16.124.113543.6003.1154777499.30246.19789.3503430045/
      series/2.16.124.113543.6003.2588828330.45298.17418.2723805630/
      instances/2.16.124.113543.6003.189642796.63084.16748.2599092903</Value>
    </DicomAttribute>
  </Item>
  <Item number="2">
    <DicomAttribute tag="00081150" vr="UI" keyword="ReferencedSOPClassUID">
      <Value number="1">1.2.840.10008.5.1.4.1.1.2</Value>
    </DicomAttribute>
    <DicomAttribute tag="00081155" vr="UI"
    keyword="ReferencedSOPInstanceUID">
      <Value number="1">
      2.16.124.113543.6003.189642796.63084.16748.2599092905</Value>
    </DicomAttribute>
    <DicomAttribute tag="00081196" vr="US" keyword="WarningReason">
      <Value number="1">45056</Value>
    </DicomAttribute>
    <DicomAttribute tag="00081190" vr="UR" keyword="RetrieveURL">
      <Value number="1">
      https://wadors.hospital.com/studies/2.16.124.113543.6003.1154777499.30246.19789.3503430045/
      series/2.16.124.113543.6003.2588828330.45298.17418.2723805630/
      instances/2.16.124.113543.6003.189642796.63084.16748.2599092905</Value>
    </DicomAttribute>
  </Item>
 </DicomAttribute>
 <DicomAttribute tag="00081190" vr="UR" keyword="RetrieveURL">
  <Value number="1">
  https://wadors.hospital.com/studies/2.16.124.113543.6003.1154777499.30246.19789.3503430045</Value>
 </DicomAttribute>
</NativeDicomModel>
```

6.7 QIDO-RS Request/Response

DICOM QIDO-RS defines several action types. An implementation shall support the following action types:

a. SearchForStudies

   This action searches for DICOM Studies that match specified search parameters and returns a list of matching studies and the requested attributes for each study.

b. SearchForSeries

   This action searches for DICOM Series that match specified search parameters and returns a list of matching series and the requested attributes for each series.

c. SearchForInstances

   This action searches for DICOM Instances that match specified search parameters and returns a list of matching instances and the requested attributes for each instance.

6.7.1 QIDO-RS - Search

6.7.1.1 Request

The specific resources to be used for the search actions shall be as follows:

- Resource
  - SearchForStudies
    - {+SERVICE}/studies{?query*,fuzzymatching,limit,offset}
  - SearchForSeries
    - {+SERVICE}/studies/{StudyInstanceUID}/series{?query*,fuzzymatching,limit,offset}
    - {+SERVICE}/series{?query*,fuzzymatching,limit,offset}
  - SearchForInstances
    - {+SERVICE}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}/instances{?query*,fuzzymatching,limit,offset}
    - {+SERVICE}/studies/{StudyInstanceUID}/instances{?query*,fuzzymatching,limit,offset}
    - {+SERVICE}/instances{?query*,fuzzymatching,limit,offset}

  where

  - {+SERVICE} is the base URL for the QIDO RESTful service. This may be a combination of protocol (http or https), authority, and path.
  - {StudyInstanceUID} is the unique Study Instance UID for a single study.
  - {SeriesInstanceUID} is the unique Series Instance UID for a single series.
- Method
  - GET
- Headers
  - Accept - The Media Type of the query results. The types allowed for this request header are:
    - multipart/related; type="application/dicom+xml"

      Specifies that the results should be DICOM PS3.19 XML (one part per result)
    - application/dicom+json (default)

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

    A QIDO-RS provider shall support both Accept header values
  - Cache-control: no-cache (recommended)

    If included, specifies that search results returned should be current and not cached.
- {query}
  - {attributeID}={value}

    0-n / {attributeID}={value} pairs allowed
  - includefield={attributeID} | all

    0-n includefield / {attributeID} pairs allowed, where "all" indicates that all available attributes should be included for each response.

  Each {attributeID} must refer to one of:

  - Patient IE attributes

- Study IE attributes
- Series IE attributes (SearchForSeries or SearchForInstances requests only)
- Composite Instance IE attributes (SearchForInstances requests only)
- Additional Query/Retrieve Attributes (Section C.3.4 in PS3.4)
- Timezone Offset From UTC (0008,0201)

See Section 6.7.1.1.1 for {attributeID} and {value} encoding rules

- fuzzymatching=true | false
- limit={limit}

  The “limit” parameter value is an unsigned integer, which specifies the maximum number of results the origin server shall return. If the “limit” parameter is not present the origin server shall return the maximum number of results in a single response that it supports.

- offset={offset}

  The “offset” parameter value is an unsigned integer, which specifies the number of results the origin server shall skip before the first returned result. If the “offset” query parameter is not present, its value is 0.

See Section 6.1.4.

6.7.1.1.1 {attributeID} encoding rules

Each {attributeID} query key shall be unique unless the associated DICOM Attribute allows UID List matching (see Section C.2.2.2.2 in PS3.4), in which case each {value} will be interpreted to be an element of the UID List.

The acceptable values for {value} are determined by the types of matching allowed by C-FIND for its associated {attributeID} (see Section C.2.2.2 in PS3.4). All characters in {value} that are disallowed for URIs shall be percent-encoded. See [RFC3986] for details.

If an {attributeID} is passed as the value of an "includefield" query key this is equivalent to C-FIND Universal matching for the specified attribute (see Section C.2.2.2.3 in PS3.4).

{attributeID} can be one of the following:

- {dicomTag}
- {dicomKeyword}
- {dicomTag}.{attributeID}, where {attributeID} is an element of the sequence specified by {dicomTag}
- {dicomKeyword}.{attributeID}, where {attributeID} is an element of the sequence specified by {dicomKeyword}

{dicomTag} is the eight character hexadecimal string corresponding to the Tag of a DICOM Attribute (see Chapter 6 in PS3.6).

{dicomKeyword} is the Keyword of a DICOM Attribute (see Chapter 6 in PS3.6).

Note

Examples of valid values for {attributeID}:

- 0020000D
- StudyInstanceUID
- 00101002.00100020
- OtherPatientIDsSequence.PatientID
- 00101002.00100024.00400032

- OtherPatientIDsSequence.IssuerOfPatientIDQualifiersSequence.UniversalEntityID

Note

Examples of valid QIDO-RS URLs:

- http://dicomrs/studies?PatientID=11235813
- http://dicomrs/studies?PatientID=11235813&StudyDate=20130509
- http://dicomrs/studies?00100010=SMITH*&00101002.00100020=11235813&limit=25
- http://dicomrs/studies?00100010=SMITH*&OtherPatientIDsSequence.00100020=11235813
- http://dicomrs/studies?PatientID=11235813&includefield=00081048&includefield=00081049&includefield=00081060
- http://dicomrs/studies?PatientID=11235813&StudyDate=20130509-20130510
- http://dicomrs/studies?StudyInstanceUID=1.2.392.200036.9116.2.2.2.2162893313.1029997326.94587%2c1.2.392.200036.9116.2.2.2.2162893313.1029997326.94583

6.7.1.2 Response

The origin server shall perform the query indicated in the request.

The search requests shall be idempotent, that is, two separate search requests with the same target resource, query parameters, and header fields shall return the same ordered list of results, if the set of matching results on the origin server has not changed.

The results returned in the response are determined as follows:

~~M~~matches is the number of matches resulting from the search.

~~smax~~maxResults is the maximum number of results the origin server allows in a single response.

offset is the value of the "offset" query parameter. It is the index of the first element in results.

limit the value of the "limit" query parameter.

~~R~~results is the ~~length~~number of ~~the~~ returned results. It is equal to the minimum of:

- ~~**M - offset** the length of available matches starting at offset,~~
- ~~**smax - offset** the length before reaching smax, and~~
- ~~**limit** the value of the "limit" query parameter.~~
- matches – offset, where if the result is less than zero, the result is zero
- maxResults
- limit

remaining is the number of remaining matches. It is equal to: ~~M~~matches – (offset +~~R~~ results).

The response is determined as follows:

If (~~R~~<results=0) then there were no matches, and a 204 (No Content) response shall be returned with an empty payload.

Otherwise, a 200 (OK) response shall be returned with a payload containing results

If (remaining > 0) the response shall include a Warning header field (see [RFC7234] Section 5.5) containing the following:

Warning: 299 {+service}: There are <remaining> additional results that can be requested

If the set of matching results has changed due to changes in the origin server contents, then the ordered list of results may be different for subsequent transactions with identical requests, and the results of using the "limit" and "offset" parameters may be inconsistent

The response will be in an Acceptable Media Type.

6.7.1.2.1 Matching

The matching semantics for each attribute are determined by the types of matching allowed by C-FIND (see Section C.2.2.2 in PS3.4).

Matching results shall be generated according to the Hierarchical Search Method described in Section C.4.1.3.1.1 in PS3.4.

Combined Datetime matching shall be performed (see Section C.2.2.2.5 in PS3.4).

Note

If a QIDO-RS provider is acting as a proxy for a C-FIND SCP that does not support combined Datetime matching the QIDO-RS provider will need to perform a C-FIND request using Date only and filter results outside the time range before returning a QIDO-RS response

If the TimezoneOffsetFromUTC / 00080201 query key is included in the request, dates and times in the request are to be interpreted in the specified time zone.

If the "fuzzymatching=true" query key/value is included in the request and it is supported then additional fuzzy semantic matching of person names shall be performed in the manner specified in the DICOM Conformance Statement for the service provider.

If the "fuzzymatching=true" query key/value is included in the request and it is not supported, the response shall include the following HTTP Warning header (see [RFC7234] Section 5.5):

Warning: 299 {SERVICE}: "The fuzzymatching parameter is not supported. Only literal matching has been performed."

where {SERVICE} is the base URL for the QIDO-RS provider. This may be a combination of scheme (http or https), host, port, and application.

Note

The Warning header is separate from the Status Line and does not affect the returned Status Code.

6.7.1.2.1.1 Study Matching

Providers of the SearchForStudies service shall support the search query keys described in Table 6.7.1-1:

Table 6.7.1-1. QIDO-RS STUDY Search Query Keys

Key Word	Tag
StudyDate	00080020
StudyTime	00080030
AccessionNumber	00080050
ModalitiesInStudy	00080061
ReferringPhysicianName	00080090
PatientName	00100010
PatientID	00100020
StudyInstanceUID	0020000D
StudyID	00200010

6.7.1.2.1.2 Series Matching

Providers of the SearchForSeries service shall support the search query keys described in Table 6.7.1-1a:

Table 6.7.1-1a. QIDO-RS SERIES Search Query Keys

Key Word	Tag
Modality	00080060
SeriesInstanceUID	0020000E
SeriesNumber	00200011
PerformedProcedureStepStartDate	00400244
PerformedProcedureStepStartTime	00400245
RequestAttributeSequence	00400275
>ScheduledProcedureStepID	00400009
>RequestedProcedureID	00401001

If {StudyInstanceUID} is not specified in the URL and this form of Relational Query is supported, all Study-level attributes specified in Table 6.7.1-1 shall also be supported.

6.7.1.2.1.3 Instance Matching

Providers of the SearchForInstances service shall support the search query keys described in Table 6.7.1-1b:

Table 6.7.1-1b. QIDO-RS INSTANCE Search Query Keys

Key Word	Tag
SOPClassUID	00080016
SOPInstanceUID	00080018
InstanceNumber	00200013

If {StudyInstanceUID} is not specified in the URL and this form of Relational Query is supported, all Study-level attributes specified in Table 6.7.1-1 shall also be supported.

If {SeriesInstanceUID} is not specified in the URL and this form of Relational Query is supported, all Series-level attributes specified in Table 6.7.1-1a shall also be supported.

6.7.1.2.2 Query Result Attributes

6.7.1.2.2.1 Study Result Attributes

For each matching Study, the QIDO-RS provider shall return all attributes in accordance with Table 6.7.1-2:

Table 6.7.1-2. QIDO-RS STUDY Returned Attributes

Attribute Name	Tag	Notes
Specific Character Set	(0008,0005)	If necessary for encoding any returned attributes
Study Date	(0008,0020)	
Study Time	(0008,0030)	
Accession Number	(0008,0050)	
Instance Availability	(0008,0056)	
Modalities in Study	(0008,0061)	
Referring Physician's Name	(0008,0090)	
Timezone Offset From UTC	(0008,0201)	May be absent if no value is available

Attribute Name	Tag	Notes
Retrieve URL	(0008,1190)	Shall be empty if the resource cannot be retrieved via WADO-RS Note The VR of this attribute has changed from UT to UR.
Patient's Name	(0010,0010)	
Patient ID	(0010,0020)	
Patient's Birth Date	(0010,0030)	
Patient's Sex	(0010,0040)	
Study Instance UID	(0020,000D)	
Study ID	(0020,0010)	
Number of Study Related Series	(0020,1206)	
Number of Study Related Instances	(0020,1208)	
All other Study Level DICOM Attributes passed as {attributeID} query keys that are supported by the service provider as matching or return attributes		
All other Study Level DICOM Attributes passed as "includefield" query values that are supported by the service provider as return attributes		
All available Study Level DICOM Attributes if the "includefield" query key is included with a value of "all"		

Series Level and Instance Level attributes passed as "includefield" query values shall not be returned.

Note

The above list is consistent with those required for IHE RAD-14 (see http://www.ihe.net/Technical_Framework/upload/IHE_RAD_TF_Vol2.pdf Table 4.14-1).

6.7.1.2.2.2 Series Result Attributes

For each matching Series, the QIDO-RS provider shall return all attributes listed in Table 6.7.1-2a:

Table 6.7.1-2a. QIDO-RS SERIES Returned Attributes

Attribute Name	Tag	Notes
Specific Character Set	(0008,0005)	If necessary for encoding any returned attributes
Modality	(0008,0060)	
Timezone Offset From UTC	(0008,0201)	May be absent if no value is available
Series Description	(0008,103E)	May be absent if no value is available
Retrieve URL	(0008,1190)	Shall be empty if the resource cannot be retrieved via WADO-RS Note The VR of this attribute has changed from UT to UR.
Series Instance UID	(0020,000E)	
Series Number	(0020,0011)	
Number of Series Related Instances	(0020,1209)	
Performed Procedure Step Start Date	(0040,0244)	May be absent if no value is available

Attribute Name	Tag	Notes
Performed Procedure Step Start Time	(0040,0245)	May be absent if no value is available
Request Attribute Sequence	(0040,0275)	May be absent if no value is available
>Scheduled Procedure Step ID	(0040,0009)	
>Requested Procedure ID	(0040,1001)	
All other Series Level DICOM Attributes passed as {attributeID} query keys that are supported by the service provider as matching or return attributes		
All other Study or Series Level DICOM Attributes passed as "includefield" query values that are supported by the service provider as return attributes		
All available Instance Level DICOM Attributes if the "includefield" query key is included with a value of "all"		
If {StudyInstanceUID} is not specified, all Study-level attributes specified in Table 6.7.1-2		

Instance Level attributes passed as "includefield" query values shall not be returned.

Note

The above list is consistent with the attributes required for IHE RAD-14 (see http://www.ihe.net/Technical_Framework/upload/IHE_RAD_TF_Vol2.pdf Table 4.14-1).

6.7.1.2.2.3 Instance Result Attributes

For each matching instance, the QIDO-RS provider shall return all attributes listed in Table 6.7.1-2b:

Table 6.7.1-2b. QIDO-RS Instance Returned Attributes

Attribute Name	Tag	Notes
Specific Character Set	(0008,0005)	If necessary for encoding any returned attributes
SOP Class UID	(0008,0016)	
SOP Instance UID	(0008,0018)	
Instance Availability	(0008,0056)	
Timezone Offset From UTC	(0008,0201)	May be absent if no value is available
Retrieve URL	(0008,1190)	Shall be empty if the resource cannot be retrieved via WADO-RS Note The VR of this attribute has changed from UT to UR.
Instance Number	(0020,0013)	
Rows	(0028,0010)	Only present for Image Instances
Columns	(0028,0011)	Only present for Image Instances
Bits Allocated	(0028,0100)	Only present for Image Instances
Number of Frames	(0028,0008)	Only present for Multi-frame image instances
All other Instance Level DICOM Attributes passed as {attributeID} query keys that are supported by the service provider as matching or return attributes		
All other Study, Series or Instance Level DICOM Attributes passed as "includefield" query values that are supported by the service provider as return attributes		
All available Instance Level DICOM Attributes if the "includefield" query key is included with a value of "all"		
If {StudyInstanceUID} is not specified, all Study-level attributes specified in Table 6.7.1-2		
If {SeriesInstanceUID} is not specified, all Series-level attributes specified in Table 6.7.1-2a		

Note

The above list is consistent with the attributes required for IHE RAD-14 (see http://www.ihe.net/Technical_Framework/upload/IHE_RAD_TF_Vol2.pdf Table 4.14-1 and Table 4.14-2).

6.7.1.2.3 Query Result Messages

The server shall support returning query results as:

- XML Results
- JSON Results

The result format used shall depend on the Accept header of the request.

6.7.1.2.3.1 XML Results

- Content-Type: multipart/related; type="application/dicom+xml"
  - The response is a multipart message body where each part is a DICOM PS3.19 XML NativeDicomModel element containing the attributes for one matching Study, Series or Instance (see Section A.1 in PS3.19).
  - The provider of the QIDO service may use a BulkData reference at its discretion (see Table A.1.5-2 in PS3.19 and Section 6.5.6). For example, this might be done to avoid encoding a large DICOM Value Field, such as an image thumbnail.
  - If there are no matching results, the message body will be empty.
- Each part in the multipart response will contain the following HTTP headers:
  - Content-Type: application/dicom+xml

6.7.1.2.3.2 JSON Results

- Content-Type: application/dicom+json
  - The response is a DICOM JSON message containing a DICOM JSON property for each matching study, series or instance containing sub-properties describing the matching attributes for each study, series or instance (see Section F.2).
  - The provider of the QIDO service may use a BulkDataURI reference at its discretion (see Section F.2.6). For example, this might be done to avoid encoding a large DICOM Value Field, such as an image thumbnail.
  - If there are no matching results, the JSON message is empty.

6.7.1.3 Status Codes

Table 6.7-1 lists the HTTP status codes that shall be used to report any of the associated error and warning situations. Other error codes may be present for other error and warning situations.

Table 6.7-1. QIDO-RS HTTP Status Codes

Code	Name	Description
Success		
200	OK	The query completed and any matching results are returned in the message body.
Failure		
400	Bad Request	The QIDO-RS Provider was unable to perform the query because the Service Provider cannot understand the query component.
401	Unauthorized	The QIDO-RS Provider refused to perform the query because the client is not authenticated.

Code	Name	Description
403	Forbidden	The QIDO-RS Provider understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
413	Request entity too large	The query was too broad and a narrower query or paging should be requested. The use of this status code should be documented in the conformance statement.
503	Busy	Service is unavailable.

6.8 RS Capabilities Service

DICOM RS Capabilities Service defines a single transaction type which shall be supported by all implementations

a. RetrieveCapabilities

This transaction retrieves the parameters for services supported by the server.

6.8.1 Retrieve Capabilities

6.8.1.1 Request Message

The Retrieve Server Options transaction can be requested for the following resources:

- {+SERVICE}/{InformationEntity*}
  - where {+SERVICE} is the base URL for the service. This may be a combination of protocol (either http or https), host, port, and application.
  - where {InformationEntity} is the path to a defined DICOM RESTful service resource, such as:
    - WADO-RS (see 6.5.1, 6.5.2, 6.5.3, 6.5.4, 6.5.5, 6.5.6)
    - STOW-RS (see 6.6.1)
    - QIDO-RS (see 6.7.1)
    - UPS-RS (see 6.9.1, 6.9.2, 6.9.3, 6.9.4, 6.9.5, 6.9.6, 6.9.7, 6.9.8, 6.9.9)

6.8.1.1.1 Method = OPTIONS

The Retrieve Server Options Service request messages use the OPTIONS method.

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/json

6.8.1.2 Response message

All responses are http single part messages. A successful response will return a Web Application Description Language (WADL) document encoded in a Media Type consistent with the Accept header of the request.

The WADL document shall contain one top-level "application" element.

The "application" element shall contain one "resources" element whose "base" attribute value is {SERVICE}, where {SERVICE} is the base URL for the service. This may be a combination of protocol (either http or https), host, port, and application.

Additionally, the WADL content shall include a "resource" element for the resource specified in the request (see 6.8.1.1) describing all methods (see 6.8.1.2.2.2 for description and examples) and child resources (see 6.8.1.2.2.1 for description and examples) for the specified resource and each of its children.

6.8.1.2.1 Resources

The full WADL resource tree follows directly and unambiguously from the RESTful resource endpoints defined in 6.5, 6.6, 6.7 and 6.9.

For informative purposes, the full resource tree and the methods defined for each resource are described in Table 6.8-1.

Table 6.8-1. Resources and Methods

Resource	Methods supported (excluding RetrieveCapabilities)	Reference
{+SERVICE}	N/A	N/A
studies	SearchForStudies	6.8.1.2.2.3
	StoreInstances	6.8.1.2.2.2
{StudyInstanceUID}	RetrieveStudy	6.8.1.2.2.1
	StoreStudyInstances	6.8.1.2.2.3
metadata	RetrieveStudyMetadata	6.8.1.2.2.1
series	SearchForStudySeries	6.8.1.2.2.3
{SeriesInstanceUID}	RetrieveSeries	6.8.1.2.2.1
metadata	RetrieveSeriesMetadata	6.8.1.2.2.1
instances	SearchForStudySeriesInstances	6.8.1.2.2.3
{SOPInstanceUID}	RetrieveInstance	6.8.1.2.2.1
metadata	RetrieveInstanceMetadata	6.8.1.2.2.1
frames	N/A	N/A
{framelist}	RetrieveFrames	6.8.1.2.2.1
instances	SearchForStudyInstances	6.8.1.2.2.3
series	SearchForSeries	6.8.1.2.2.3
{SeriesInstanceUID}	N/A	N/A
{instances}	SearchForInstances	6.8.1.2.2.3
instances	SearchForInstances	6.8.1.2.2.3
{BulkDataURI}	RetrieveBulkData	6.8.1.2.2.1
workitems	SearchForUPS	6.8.1.2.2.3
	CreateUPS	6.8.1.2.2.2
{UPSInstanceUID}	RetrieveUPS	6.8.1.2.2.1
	UpdateUPS	6.8.1.2.2.4
state	ChangeUPSState	6.8.1.2.2.4
cancelrequest	RequestUPSCancel	6.8.1.2.2.4

Resource	Methods supported (excluding RetrieveCapabilities)	Reference
subscribers	N/A	N/A
{AETitle}	CreateSubscription	6.8.1.2.2.5
	DeleteSubscription	6.8.1.2.2.5
1.2.840.10008.5.1.4.34.5	N/A	N/A
subscribers	N/A	N/A
{AETitle}	CreateSubscription	6.8.1.2.2.5
	DeleteSubscription	6.8.1.2.2.5
suspend	SuspendGlobalSubscription	6.8.1.2.2.5
1.2.840.10008.5.1.4.34.5.1	N/A	N/A
subscribers	N/A	N/A
{AETitle}	CreateSubscription	6.8.1.2.2.5
	DeleteSubscription	6.8.1.2.2.5
suspend	SuspendGlobalSubscription	6.8.1.2.2.5

6.8.1.2.2 Methods

6.8.1.2.2.1 Retrieve Methods

The Retrieve methods define the capabilities of a WADO-RS resource (see 6.5) or a RetrieveUPS resource (see 6.9.4).

The Retrieve methods shall contain the following attributes:

- A "name" attribute with a value of "GET"
- An "id" attribute with a value of "RetrieveStudy", "RetrieveSeries", "RetrieveInstance", "RetrieveBulkData", "RetrieveFrames", "RetrieveStudyMetadata", "RetrieveSeriesMetadata", "RetrieveInstanceMetadata" or "RetrieveUPS"

The Retrieve methods shall contain a "request" element with "param" elements documenting the following:

- supported Accept header values
  - if the same Media Type is supported with multiple Transfer Syntaxes there should be one entry for each combination of Media Type and Transfer Syntax

The Retrieve methods shall contain one or more "response" elements documenting the following:

- supported Status Codes
- Media Types returned for each Status Code (if applicable)
  - if the same Media Type is supported with multiple Transfer Syntaxes there should be one entry for each combination of Media Type and Transfer Syntax

    Note

    More than one Status Code can be described by a single "response" element.

Example:

```
<method name="GET" id="RetrieveStudies">
 <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 />
   <option value="multipart/related; type=application/dicom";
         transfer-syntax=1.2.840.10008.1.2.1 />
   <option value="multipart/related; type=application/octet-stream" />
   <option value="multipart/related; type=image/dicom+jpx" />
   <option value="multipart/related; type=image/dicom+jpx;
         transfer-syntax=1.2.840.10008.1.2.4.92" />
   <option value="multipart/related; type= video/mpeg;
         transfer-syntax=1.2.840.10008.1.2.4.100" />
  </param>
 </request>
 <response status="200,206">
  <representation mediaType="multipart/related; type=application/dicom";
               transfer-syntax=1.2.840.10008.1.2 />
  <representation mediaType="multipart/related; type=application/dicom";
               transfer-syntax=1.2.840.10008.1.2.1 />
  <representation mediaType="multipart/related; type=application/octet-stream" />
  <representation mediaType="multipart/related; type= image/dicom+jpx" />
  <representation mediaType="multipart/related; type= image/dicom+jpx;
               transfer-syntax=1.2.840.10008.1.2.4.92" />
  <representation mediaType="multipart/related; type= video/mpeg;
               transfer-syntax=1.2.840.10008.1.2.4.100" />
 </response>
 <response status="400 404 406 410 503" />
</method>
```

6.8.1.2.2.2 Store Methods

The Store methods define the capabilities of a STOW-RS resource (see 6.6) or a CreateUPS resource (see 6.9.1).

The Store methods shall contain the following attributes:

- A "name" attribute with a value of "POST"
- An "id" attribute with a value of "StoreInstances", "StoreStudyInstances" or "CreateUPS"

The Store methods shall contain a "request" element with "param" elements documenting the following:

- supported Accept header values
- supported Representations
  - if the same Media Type is supported with multiple Transfer Syntaxes there should be one entry for each combination of Media Type and Transfer Syntax

The Store methods shall contain one or more "response" elements documenting the following:

- supported Status Codes
- Media Types returned for each Status Code (if applicable)
- Headers returned for each Status Code

Note

More than one Status Code can be described by a single "response" element.

Example:

```
<method name="GET" 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" />
  <representation mediaType="multipart/related; type=application/dicom;
              transfer-syntax=1.2.840.10008.1.2.1" />
  <representation mediaType="multipart/related; type=application/dicom+xml" />
  <representation mediaType="multipart/related; type=application/dicom+xml;
              transfer-syntax=1.2.840.10008.1.2" />
  <representation mediaType="multipart/related; type=application/dicom+xml;
     transfer-syntax=1.2.840.10008.1.2.1" />
  <representation mediaType="multipart/related; type=application/dicom+xml;
              transfer-syntax=1.2.840.10008.1.2.4.92" />
  <representation mediaType="multipart/related; type=application/dicom+xml;
              transfer-syntax=1.2.840.10008.1.2.4.100" />
 </request>
 <response status="200" />
 <response status="202,409">
  <representation mediaType="application/dicom+xml" />
 </response>
 <response status="400,401,403,503" />
</method>
```

6.8.1.2.2.3 Search Methods

The Search methods define the capabilities of a QIDO-RS resource (see 6.7) or a SearchForUPS resource (see 6.9.3).

The Search methods shall contain the following attributes:

- A "name" attribute with a value of "GET"
- An "id" attribute with a value of "SearchForStudies", "SearchForStudySeries", "SearchForSeries", "SearchForStudySeriesInstances", "SearchForStudyInstances", "SearchForSeriesInstances", "SearchForInstances" or "SearchForUPS"

The Search methods shall contain a "request" element with "param" elements documenting the following:

- supported Accept header values
- support for the Cache-control header
- support of "limit", "offset" and "fuzzymatching" query parameters
- supported search parameters (both tag and keyword variants shall be listed)
- supported options for the "includefield" parameter (both tag and keyword variants shall be listed)

The Search methods shall contain one or more "response" elements documenting the following:

- supported Status Codes
- returned "header" parameters, including use of "warning headers"
- Media Types returned for each Status Code (if applicable)

  Note

  More than one Status Code can be described by a single "response" element.

Example:

```
<method name="GET" id="SearchForStudies">
 <request>
  <param name="Accept" style="header" default="application/dicom+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="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>
```

6.8.1.2.2.4 Update Methods

The Update methods define the capabilities of an UpdateUPS, a ChangeUPSState or a RequestUPSCancellation resource (see 6.9.2).

The Update methods shall contain the following attributes:

- A "name" attribute with a value of "POST" for UpdateUPS and RequestUPSCancel
- A "name" attribute with a value of "PUT" for ChangeUPSState
- An "id" attribute with a value of "UpdateUPS", "ChangeUPSState" or "RequestUPSCancellation"

The Update methods shall contain a "request" element with "param" elements documenting the following:

- supported Representations

The Update methods shall contain one or more "response" elements documenting the following:

- supported Status Codes
- Headers returned for each Status Code

  Note

  More than one Status Code can be described by a single "response" element.

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>
```

6.8.1.2.2.5 Subscribe Methods

The Subscribe methods define the capabilities of a CreateSubscription, a SuspendGlobalSubscription or a DeleteSubscription resource (see 6.9.7, 6.9.8 and 6.9.9).

The Subscribe methods shall contain the following attributes:

- A "name" attribute with a value of "POST" for CreateSubscription and SuspendGlobalSubscription
- A "name" attribute with a value of "DELETE" for DeleteSubscription
- An "id" attribute with a value of "CreateSubscription", "SuspendGlobalSubscription" or "DeleteSubscription"

The Subscribe methods shall contain a "request" element with "param" elements documenting the following:

- supported Representations

The Subscribe methods shall contain one or more "response" elements documenting the following:

- supported Status Codes
- Headers returned for each Status Code

  Note

  More than one Status Code can be described by a single "response" element.

Example:

```
<method name="POST" id="CreateSubscription">
 <request>
  <param name="deletionlock" style="query" default="false">
   <option value="true" />
   <option value="false" />
  </param>
 </request>
 <response status="201">
  <param name="Warning" style="header" fixed="299 {+SERVICE}: Deletion Lock not granted." />
 </response>
 <response status="403">
  <param name="Warning" style="header" fixed="299 {+SERVICE}: The Origin-Server does not support
                                        Global Subscription Filtering." />
```

```
  </response>
  <response status="400 401 404 409 503" />
</method>
```

6.8.1.3 Status Codes

Table 6.8-2 lists the HTTP status codes that shall be used to report any of the associated error and warning situations. Other error codes may be present for other error and warning situations.

Table 6.8-2. Server Options HTTP Status Codes

Code	Name	Description
Success		
200	OK	The query completed and any matching results are returned in the message body.
Failure		
400	Bad Request	The Server Options Provider was unable to perform the query because the Service Provider cannot understand the query component.
401	Unauthorized	The Server Options Provider refused to perform the query because the client is not authenticated.
403	Forbidden	The Server Options Provider understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
503	Busy	Service is unavailable.

6.9 UPS-RS Worklist Service

This DICOM Web Service defines a RESTful interface to the UPS SOP Classes (see PS3.3 and PS3.4). It consists of the following action types:

1. CreateUPS

   This action requests the creation of a UPS Instance on the Origin-Server. It corresponds to the UPS DIMSE N-CREATE operation.

2. UpdateUPS

   This action sets the attributes of a UPS Instance managed by the Origin-Server. It corresponds to the UPS DIMSE N-SET operation.

3. SearchForUPS

   This action searches for UPS Instances known to the Origin-Server. It corresponds to the UPS DIMSE C-FIND operation.

4. RetrieveUPS

   This action retrieves a UPS Instances. It corresponds to the UPS DIMSE N-GET operation.

5. ChangeUPSState

   This action sets the state of a UPS Instance managed by the Origin-Server. It corresponds to the UPS DIMSE N-ACTION operation "Change UPS State".

6. RequestUPSCancellation

   This action requests the cancellation of a UPS Instance managed by the Origin-Server. It corresponds to the UPS DIMSE N-ACTION operation "Request UPS Cancel".

7. CreateSubscription

This action subscribes to a UPS Instance or the Global Worklist managed by the Origin-Server. It corresponds to the UPS DIMSE N-ACTION operation "Subscribe to Receive UPS Event Reports".

8. SuspendGlobalSubscription

   This action suspends an existing subscription to the Global Worklist managed by the Origin-Server. It corresponds to the UPS DIMSE N-ACTION operation "Suspend Global Subscription".

9. DeleteSubscription

   This action cancels an existing subscription to a UPS Instance or the Global Worklist managed by the Origin-Server. It corresponds to the UPS DIMSE N-ACTION operation "Unsubscribe from Receiving UPS Event Reports".

10. OpenEventChannel

    This action initiates a WebSocket connection to allow the User-Agent to start receiving Event Report messages.

11. SendEventReport

    This action sends an Event Report using an open WebSocket connection. It corresponds to the UPS DIMSE N-EVENT-REPORT operation.

An Origin-Server shall support all of the above action types.

The requirements for a UPS-RS Origin-Server that is also a Unified Worklist and Procedure Step SCP are described in Section CC.1 in PS3.4

Table 6.9-1. UPS Interface Mapping

Action Type	Section	Method & Resource
CreateUPS	6.9.1	POST {+SERVICE}/workitems{?AffectedSOPInstanceUID}
UpdateUPS	6.9.2	POST {+SERVICE}/workitems/{UPSInstanceUID}{?transaction}
SearchForUPS	6.9.3	GET {+SERVICE}/workitems{?query*}
RetrieveUPS	6.9.4	GET {+SERVICE}/workitems/{UPSInstanceUID}
ChangeUPSState	6.9.5	PUT {+SERVICE}/workitems/{UPSInstanceUID}/state
RequestUPSCancellation	6.9.6	POST {+SERVICE}/workitems/{UPSInstanceUID}/cancelrequest
CreateSubscription	6.9.7	POST {+SERVICE}/workitems/{UPSInstanceUID}/subscribers/{AETitle}{?deletionlock} POST {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5/ subscribers/{AETitle}{?deletionlock} POST {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5.1/ subscribers/{AETitle}{?deletionlock,query*}
SuspendGlobalSubscription	6.9.8	POST {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5/ subscribers/{AETitle}/suspend POST {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5.1/ subscribers/{AETitle}/suspend
DeleteSubscription	6.9.9	DELETE {+SERVICE}/workitems/{UPSInstanceUID}/ subscribers/{AETitle}
OpenEventChannel	6.9.10	GET {+WSSERVICE}/subscribers/{AETitle}
SendEventReport	6.9.11	N/A

The Origin-Server shall comply with all requirements placed on the SCP for the corresponding services in Annex CC “Unified Procedure Step Service and SOP Classes (Normative)” in PS3.4.

6.9.1 CreateUPS

This resource allows a User-Agent to instruct an Origin-Server to create a UPS instance.

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 XML metadata. See Section 6.9.1.1.1.
    - application/dicom+json

      Specifies that the post is DICOM PS3.18 JSON metadata. See Section 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.

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 XML or DICOM JSON. Any binary data contained in the message shall be inline.

6.9.1.2 Behavior

The Origin-Server shall create and maintain UPS instances as instructed by CreateUPS requests and as specified by the SCP behavior in Section CC.2.5.3 in PS3.4.

The Origin-Server shall return the HTTP Status Line applicable to the associated request.

6.9.1.3 Response

The Origin-Server shall return an HTTP response message.

6.9.1.3.1 Response Status Line

If the Create request is successful, the Origin-Server shall return an HTTP "201 - Created" response code.

If the request fails, the Origin-Server shall return an appropriate failure status line with a response code from Table 6.9.1-1.

Table 6.9.1-1. Status Codes

HTTP Code	Reason Phrase	Description
201	Created	The UPS instance was created and the new resource can be retrieved at the Content-Location specified in the response
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
409	Conflict	The UID of the posted UPS Instance corresponds to an existing UPS Instance.
503	Busy	Service is unavailable.

6.9.1.3.2 Response Headers

If the request is successful, the HTTP response message shall include the following HTTP header:

- Content-Location: {+WorkitemURL}

  Where {+WorkitemURL} is the URL from which the created UPS Instance can be retrieved (see Section 6.9.4)

If the UPS instance was created with modifications, the response message shall include the following HTTP header:

- Warning: 299 {+SERVICE}: The UPS was created with modifications.

6.9.1.3.3 Response Message Body

The response message body shall be empty.

6.9.2 UpdateUPS

This resource supports the modification of attribute values of an existing UPS Instance.

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 XML metadata. See Section 6.9.2.1.1.
    - application/dicom+json

      Specifies that the post is DICOM PS3.18 JSON metadata. See Section 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.
- 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 XML or DICOM PS3.18 JSON. Any binary data contained in the message shall be inline.

6.9.2.2 Behavior

The Origin-Server shall support the Attribute changes to the UPS instance specified by the User-Agent in the UpdateUPS request and as specified by the SCP behavior in Section CC.2.6.3 in PS3.4.

The Origin-Server shall return the HTTP Status applicable to the associated request.

6.9.2.3 Response

The Origin-Server shall return an HTTP response message.

6.9.2.3.1 Response Status Line

If the Set request is successful, the Origin-Server shall return an HTTP "200 - OK" response code.

If the request fails, the Origin-Server shall return an appropriate failure status line with a response code from Table 6.9.2-1.

Table 6.9.2-1. Status Codes

HTTP Code	Reason Phrase	Description
200	OK	The UPS instance was updated

HTTP Code	Reason Phrase	Description
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
404	Not found	The specified UPS Instance does not exist or is not managed by this Origin-Server.
409	Conflict	The request cannot be performed for one of the following reasons: • the submitted request is inconsistent with the current state of the UPS Instance • the Transaction UID is missing • the Transaction UID is incorrect
503	Busy	Service is unavailable.

6.9.2.3.2 Response Headers

If the UPS instance was updated but with modifications made by the Origin-Server, the response message shall include the following HTTP header:

- Warning: 299 {+SERVICE}: The UPS was created with modifications.

If optional attributes were rejected, the response message shall include the following HTTP Warning header field:

- Warning: 299 {+SERVICE}: Requested optional Attributes are not supported.

If the request was rejected with an HTTP 409 status code, the response message shall include one of following messages encoded in an HTTP Warning header field describing the nature of the conflict:

- Warning: 299 {+SERVICE}: The Transaction UID is missing.
- Warning: 299 {+SERVICE}: The Transaction UID is incorrect.
- Warning: 299 {+SERVICE}: The submitted request is inconsistent with the current state of the UPS Instance.

6.9.2.3.3 Response Message Body

The response message body shall be empty.

6.9.3 SearchForUPS

This resource returns a list of UPS Instances that match specified search query parameters along with requested attributes for each Instance.

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 XML metadata.

    - application/dicom+json

      Specifies that the results should be DICOM PS3.18 JSON metadata.

  - Cache-control: no-cache (recommended)

    If included, specifies that search results returned should be current and not cached.

- {query}
  - {attributeID}={value}

    0-n / {attributeID}={value} pairs allowed

  - includefield={attributeID} | all

    0-n includefield / {attributeID} pairs allowed, where "all" indicates that all attributes with values should be included for each response.

    Each {attributeID} shall refer to an attribute of the Unified Procedure Step IOD (see Section B.26.2 in PS3.3).

    See Section 6.7.1.1 for {attributeID} and {value} encoding rules

  - fuzzymatching=true | false
  - limit={maximumResults}
  - offset={skippedResults}

6.9.3.2 Behavior

The Origin-Server shall perform a search according the requirements for the QIDO-RS services (see Section 6.7.1.2).

6.9.3.2.1 Matching

An Origin-Server shall support matching against all Unified Procedure Step Instance Attributes in Table CC.2.5-3 in PS3.4 with a Match Key Type value of U, R or *.

See Section 6.7.1.2.1 for matching behavior.

6.9.3.3 Response

The Origin-Server shall return an HTTP response message.

6.9.3.3.1 Response Status Line

If the SearchForUPS request is successful, the Origin-Server shall return an HTTP "200 - OK" response code.

If the request fails, the Origin-Server shall return an appropriate failure status line with a response code from Table 6.9.3-1.

Table 6.9.3-1. Status Codes

HTTP Code	Reason Phrase	Description
200	OK	The query completed and any matching results are returned in the message body.
206	Partial Content	Only some of the query results were returned and the rest can be requested through the appropriate UPS-RS request.
400	Bad Request	The UPS-RS Origin-Server was unable to perform the query because the Service Provider cannot understand the query component.
401	Unauthorized	The UPS-RS Origin-Server refused to perform the query because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
413	Request entity too large	The query was too broad and a narrower query or paging should be requested.
503	Busy	Service is unavailable.

6.9.3.3.2 Query Result Attribute

For each matching UPS Instance, the Origin-Server shall return:

- All Unified Procedure Step Instance Attributes in Table CC.2.5-3 in PS3.4 with a Return Key value of 1 and 2.
- All Unified Procedure Step Instance Attributes in Table CC.2.5-3 in PS3.4 with a Return Key value of 1C for which the conditional requirements are met.
- All other Unified Procedure Step Instance Attributes passed as {attributeID} query keys that are supported by the Origin-Server as matching or return attributes
- All other Unified Procedure Step Instance Attributes passed as "includefield" query values that are supported by the Origin-Server as return attributes.

6.9.3.3.3 Response Message

The response message body contains the results.

The format of the response message body shall contain one of the Media Types specified by the request Accept header field. An Origin-Server shall support all Media-Types allowed in the request.

6.9.3.3.3.1 XML Response Message

- Content-Type:
  - multipart/related; type="application/dicom+xml"
- The response is a multipart message body where each part is a DICOM PS3.19 XML DicomNativeModel element containing the attributes for one matching UPS Instance (see Section A.1 in PS3.19).
- If there are no matching results, the message body shall be empty.
- Each part in the multipart body includes the following HTTP headers:
  - Content-Type: application/dicom+xml

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).
- If there are no matching results, the JSON message shall be empty.

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 XML metadata.
    - application/dicom+json

      Specifies that the result should be DICOM PS3.18 JSON metadata.
  - Cache-control: no-cache (recommended)

    If included, specifies that results returned should be current and not cached.

6.9.4.2 Behavior

The Origin-Server shall return, via the HTTP response, the indicated Unified Procedure Step Instance to the User-Agent.

Note

The requirement for the Origin-Server to respond to GET requests for UPS Instances that have moved to the COMPLETED or CANCELED state is limited. See Section CC.2.1.3 in PS3.4.

The Origin-Server shall not return the Transaction UID (0008,1195) Attribute. This is necessary to preserve this Attribute's role as an access lock.

The Origin-Server shall return the HTTP Response Status Code applicable to the associated request. A Failure Code shall indicate that the Origin-Server has not returned the SOP Instance.

6.9.4.3 Response

The Origin-Server shall return an HTTP response message.

6.9.4.3.1 Response Status Line

If the Retrieve request is successful, the Origin-Server shall return an HTTP "200 - OK" response code.

If the request fails, the Origin-Server shall return an appropriate failure status line with a response code from Table 6.9.4-1.

Table 6.9.4-1. Status Codes

HTTP Code	Reason Phrase	Description
200	OK	The requested instance is returned.
400	Bad Request	The UPS-RS Origin-Server was unable to perform the query because the Service Provider cannot understand the query component.
401	Unauthorized	The UPS-RS Origin-Server refused to perform the query because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
404	Not found	The specified UPS Instance does not exist or is not managed by this Origin-Server.
503	Busy	Service is unavailable.

6.9.4.3.2 Response Message

The response message body contains the results.

The format of the response message body shall contain one of the Media Types specified by the request Accept header field. An Origin-Server shall support all Media-Types allowed in the request.

6.9.4.3.2.1 XML Response Message

- Content-Type:
  - application/dicom+xml
- The response contains a DICOM PS3.19 XML DicomNativeModel element containing the attributes for the requested UPS Instance (see Section A.1 in PS3.19).

6.9.4.3.2.2 JSON Response Message

- Content-Type:
  - application/dicom+json
- The response is a DICOM JSON array containing a DICOM JSON representation of the requested UPS Instance (see Section F.2).

6.9.5 ChangeUPSState

This resource supports the modification of the state of an existing UPS Instance.

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 XML metadata. See Section 6.9.5.1.1.
    - application/dicom+json

      Specifies that the post is DICOM PS3.18 JSON metadata. See Section 6.9.5.1.1.
- The request body describes a state change to a single Unified Procedure Step Instance. It shall include all Attributes required for an SCU in Table CC.2.1-1 in PS3.4.

6.9.5.1.1 Request Message

The Request Message has a single part body.

- Content-Type:
  - application/dicom+xml
  - application/dicom+json
- The request body contains attributes in either DICOM PS3.19 XML or DICOM PS3.18 JSON format.

6.9.5.2 Behavior

The Origin-Server shall support the state changes to the UPS instance specified in the request as described by the SCP behavior in Section CC.2.1.3 in PS3.4.

After completing the ChangeUPSState request, the Origin-Server shall return the HTTP Response Line applicable to the associated request.

6.9.5.3 Response

The Origin-Server shall return an HTTP response message.

6.9.5.3.1 Response Status Line

If the State Change was successful, the Service shall return an HTTP "200 - OK" response code.

If the State Change fails, the Service shall return an appropriate failure status line with a response code from Table 6.9.5-1.

Table 6.9.5-1. Status Codes

HTTP Code	Reason Phrase	Description
200	OK	The UPS instance was updated
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request

HTTP Code	Reason Phrase	Description
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
404	Not found	The specified UPS Instance does not exist or is not managed by this Origin-Server.
409	Conflict	The request cannot be performed for one of the following reasons: • the submitted request is inconsistent with the current state of the UPS Instance • the Transaction UID is missing • the Transaction UID is incorrect
503	Busy	Service is unavailable.

6.9.5.3.2 Response Headers

If the User-Agent specifies a Procedure Step State (0074,1000) attribute with a value of "CANCELED" and the UPS Instance is already in that state, the response message shall include the following HTTP Warning header field:

- Warning: 299 {+SERVICE}: The UPS is already in the requested state of CANCELED.

If the User-Agent specifies a Procedure Step State (0074,1000) attribute with a value of "COMPLETED" and the UPS Instance is already in that state, the response message shall include the following HTTP Warning header field:

- Warning: 299 {+SERVICE}: The UPS is already in the requested state of COMPLETED.

If the request was rejected with an HTTP 409 status code, the response message shall include one of following messages in the HTTP Warning header field describing the nature of the conflict:

- Warning: 299 {+SERVICE}: The Transaction UID is missing.
- Warning: 299 {+SERVICE}: The Transaction UID is incorrect.
- Warning: 299 {+SERVICE}: The submitted request is inconsistent with the current state of the UPS Instance.

6.9.5.3.3 Response Message Body

The response message body shall be empty.

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 XML metadata. See Section 6.9.5.1.1.
    - application/dicom+json

      Specifies that the post is DICOM PS3.18 JSON metadata. See Section 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.

6.9.6.1.1 Request Message

The Request Message has a single part body.

- Content-Type:
  - application/dicom+xml
  - application/dicom+json
- The request body contains attributes in either DICOM PS3.19 XML or DICOM PS3.18 JSON format.

6.9.6.2 Behavior

RequestUPSCancellation is used to request to the Origin-Server that the state of a UPS Instance be changed to CANCELED as shown in Figure CC.1.1-1 in PS3.4. The Origin-Server shall process the request as described by the SCP behavior in Section CC.2.2.3 in PS3.4.

The request may include a Reason For Cancellation and/or a proposed Procedure Step Discontinuation Reason Code Sequence.

The request may also include a Contact Display Name and/or a Contact URI for the person with whom the cancel request may be discussed.

Note

An HTTP Status Code indicating success means that the Request was accepted, not that the UPS has been canceled. The system performing the UPS is not obliged to honor the request to cancel and in some scenarios, may not even receive notification of the request. See Section CC.2.4 in PS3.4.

To cancel an IN PROGRESS UPS that the User-Agent is itself performing, the User-Agent shall instead use the ChangeUPSState action as described in Section 6.9.5.

6.9.6.3 Response

The Origin-Server shall return an HTTP response message.

6.9.6.2.1 Response Status Line

If the cancel request was accepted, the Service shall return an HTTP "202 - Accepted" response code.

If the cancel request was rejected, the Service shall return an appropriate failure status line with a response code from Table 6.9.6-1.

Table 6.9.6-1. Status Codes

HTTP Code	Reason Phrase	Description
202	Accepted	The cancel request was accepted
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
404	Not found	The specified UPS Instance does not exist or is not managed by this Origin-Server.
409	Conflict	The cancellation request is inconsistent with the current state of the UPS Instance
503	Busy	Service is unavailable.

6.9.2.5.2 Response Headers

If the UPS Instance is already in a canceled state, the response message shall include the following HTTP Warning header field:

- Warning: 299 {+SERVICE}: The UPS is already in the requested state of CANCELED.

6.9.5.2.3 Response Message Body

The response message body shall be empty.

6.9.7 CreateSubscription

This resource records subscribers to whom future events associated with the specified UPS Instances will be reported.

6.9.7.1 Request

The request message shall be formed as follows:

- Resource
  - {+SERVICE}/workitems/{UPSInstanceUID}/subscribers/{AETitle}}{?deletionlock}
  - {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5/subscribers/{AETitle}{?deletionlock}
  - {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5.1/subscribers/{AETitle}{?deletionlock,query*}

    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 or a well-known UID
    - {AETitle} is an Application Entity Title that conforms to the "AE" Value Representation (see Table 6.2-1 in PS3.5) and identifies the Application Entity to be subscribed
    - {deletionlock}, if present, shall have a value of either "true" or "false", indicating whether or not the User-Agent is requesting a Deletion Lock
    - {query} specifies the query key/value pairs describing the filter parameters
- Method
  - POST

- Headers
  - Content-Length: 0
- {query}
  - deletionlock=true | false
  - {attributeID}={value}

    0-n / {attributeID}={value} pairs allowed

    Each {attributeID} shall refer to an attribute of the Unified Procedure Step IOD (see Section B.26.2 in PS3.3).

    See Section 6.7.1.1 for {attributeID} and {value} encoding rules.
- The request body shall be empty.

6.9.7.2 Behavior

The Origin-Server shall support the management of UPS instance subscriptions as specified by the SCP behavior in Section CC.2.3.3 in PS3.4.

Upon receipt of the CreateSubscription, SuspendGlobalSubscription or DeleteSubscription request, the Origin-Server shall attempt to update the Global Subscription State, Filtered Global Subscription and/or UPS Subscription State of the specified Application Entity with respect to the specified SOP Instance UID as described in Table CC.2.3-2 in PS3.4 and then return the appropriate HTTP response.

6.9.7.3 Response

6.9.7.3.1 Response Status Line

The Service shall return an HTTP status line, including a status code and associated reason phrase.

If the CreateSubscription request was successful, the Service shall return an "HTTP 201 - Created" response code. The response shall contain a "Content-Location" header of the following format:

- Content-Location: {WSSERVICE}

  where:

  - {WSSERVICE} is the base URL for the WebSocket service. This shall include the WebSocket protocol (either WS or WSS) and may include a combination of authority and path.

If the subscription fails, the Service shall return an appropriate failure status line with a response code from Table 6.9.7-2.

Table 6.9.7-2. Status Codes

HTTP Code	Reason Phrase	Description
201	Created	The subscription was created.
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., the Origin-Server does not support global subscription filtering or an authenticated user has insufficient privileges).
404	Not found	The specified UPS Instance or well-known UID does not exist or is not managed by this Origin-Server.
409	Conflict	Specified action not appropriate for specified instance.
503	Busy	Service is unavailable.

6.9.7.3.2 Response Headers

If the CreateSubscriptionrequest was accepted but the deletion lock was not, the response message shall include the following HTTP Warning header field:

- Warning: 299 {+SERVICE}: Deletion Lock not granted.

If the request was rejected with an HTTP 403 status code because Filtered Global Subscription is not supported, the response message shall include the following HTTP Warning header field:

- Warning: 299 {+SERVICE}: The Origin-Server does not support Global Subscription Filtering.

6.9.7.3.3 Response Message Body

The response message body shall be empty.

6.9.8 SuspendGlobalSubscription

This resource suspends an existing Global Subscription or Filtered Global Subscription. The Origin-Server will no longer automatically subscribe the User-Agent to newly-created UPS Instances. This does not delete any existing subscriptions to specific UPS Instances.

6.9.8.1 Request

The request message shall be formed as follows:

- Resource
  - {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5/subscribers/{AETitle}/suspend
  - {+SERVICE}/workitems/1.2.840.10008.5.1.4.34.5.1/subscribers/{AETitle}/suspend

    where

    - {+SERVICE} is the base URL for the service. This may be a combination of protocol (either HTTP or HTTPS), authority and path.
    - {AETitle} identifies the subscribed Application Entity.
- Method
  - POST
- The request body shall be empty.

6.9.8.2 Behavior

The SuspendGlobalSubscription Origin-Server shall behave as described in Section 6.9.7.2.

6.9.8.3 Response

6.9.8.3.1 Response Status Line

The Service shall return an HTTP status line, including a status code and associated reason phrase.

If the SuspendGlobalSubscriptionrequest was successful, the Service shall return an HTTP "200 - OK" response code.

If the subscription change fails, the Service shall return an appropriate failure status line with a response code from Table 6.9.8-1.

Table 6.9.8-1. Status Codes

HTTP Code	Reason Phrase	Description
200	OK	The subscription was suspended.
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
404	Not found	The specified UPS Instance or well-known UID does not exist or is not managed by this Origin-Server.
409	Conflict	Specified action not appropriate for specified instance.
503	Busy	Service is unavailable.

6.9.8.2.2 Response Message Body

The response message body shall be empty.

6.9.9 DeleteSubscription

This resource removes existing subscriptions from the specified UPS Instances.

6.9.9.1 Request

The request message shall be formed as follows:

- Resource
  - {+SERVICE}/workitems/{UPSInstanceUID}/subscribers/{AETitle}

    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 or a well-known UID.
    - {AETitle} identifies the subscribed Application Entity.
- Method
  - DELETE
- The request body shall be empty.

6.9.9.2 Behavior

The DeleteSubscription Origin-Server shall behave as described in Section 6.9.7.2.

6.9.9.3 Response

6.9.9.3.1 Response Status Line

The Service shall return an HTTP status line, including a status code and associated reason phrase.

If the DeleteSubscriptionrequest was successful, the Service shall return an HTTP "200 - OK" response code.

If the subscription fails, the Service shall return an appropriate failure status line with a response code from Table 6.9.7-1.

Table 6.9.7-1. Status Codes

HTTP Code	Reason Phrase	Description
200	OK	The subscription was removed.
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
404	Not found	The specified UPS Instance or well-known UID does not exist or is not managed by this Origin-Server.
409	Conflict	Specified action not appropriate for specified instance.
503	Busy	Service is unavailable.

6.9.9.3.2 Response Message Body

The response message body shall be empty.

6.9.10 OpenEventChannel

This resource opens a WebSocket channel that will be used to send Event Reports to the client.

See [RFC6455] for details on the WebSocket protocol.

6.9.10.1 Request

The request message shall be formed as follows:

- Resource
  - {+WSSERVICE}/subscribers/{AETitle}

    where

    - {+WSSERVICE} is the base URL for the WebSocket service. This shall include the WebSocket protocol (either WS or WSS) and may include a combination of authority and path
    - {AETitle} identifies the subscribed Application Entity.
- Method
  - GET

6.9.10.2 Behavior

The Origin-Server maintains the active WebSocket connection and uses it to send Event Report messages for UPS Instances which have subscriptions association with {AETitle} (see Section 6.9.7.2).

If the WebSocket connection is lost at any point the User-Agent can re-establish it by repeating the request.

The state of a WebSocket connection does not affect subscriptions and an Origin-Server is not required to queue messages when the connection is down.

Note

A User-Agent will only receive the initial state of a newly-subscribed UPS Instance if the WebSocket connection was initiated before creating the subscription

6.9.10.3 Response

6.9.10.3.1 Response Status Line

The Service shall return an HTTP status line, including a status code and associated reason phrase.

If the request was successful, the Service shall return an HTTP "101 - Switching Protocols" response code.

If the request fails, the Service shall return an appropriate failure status line with a response code from Table 6.9.10-1.

Table 6.9.10-1. Status Codes

HTTP Code	Reason Phrase	Description
101	Switching Protocols	The WebSocket connection was established.
400	Bad Request	The UPS-RS Origin-Server was unable to understand the request
401	Unauthorized	The UPS-RS Origin-Server refused to accept the request because the client is not authenticated.
403	Forbidden	The UPS-RS Origin-Server understood the request, but is refusing to perform the query (e.g., an authenticated user with insufficient privileges).
503	Busy	Service is unavailable.

6.9.10.3.2 Response Message Body

The response message body shall be empty.

The connection remains open and may be used by the server to send Event messages (see Section 6.9.11).

6.9.11 SendEventReport

This operation sends an Event Report over an established WebSocket connection.

6.9.11.1 Request

The request message shall be formed as follows:

- Resource
  - N/A
- Method
  - WebSocket Data Frame transmission
- The Event Report shall contain all mandatory attributes described indescribed in Table CC.2.4-1 in PS3.4 and Table 10.3-1 in PS3.7 for the event type.

6.9.11.1.1 Request Message Body

WebSocket Events are encoded as WebSocket data frames with an opcode of "%x1" (text).

The frame payload data shall be a DICOM JSON dataset containing the attributes of the Event Report.

Note

1. Example WebSocket payload:

```
{
"00000002": { "vr" : "UI", [ "1.2.840.10008.5.1.4.34.6.4" ] },
"00000100": { "vr" : "US", [ 256 ] },
```

```
"00000110": { "vr" : "US", [ 23 ] },
"00000800": { "vr" : "US", [ 0 ] },
"00001000": { "vr" : "UI", [ "1.2.840.10008.5.1.4.34.6.4.2.3.44.22231" ] },
"00001001": { "vr" : "US", [ 1 ] },
"00741238": { "vr" : "CS", [ "SCHEDULED" ] },
"00744041": { "vr" : "CS", [ "READY" ] }
}
```

2. The WebSocket protocol does not allow content negotiation so it is not possible to support both XML and JSON encoding of Event Report messages without extending the protocol.

6.9.11.2 Behavior

Section CC.2.4.3 in PS3.4 describes the scenarios in which an Origin-Server sends Event Reports to a subscriber and the content of the Event Report messages.

6.9.11.3 Response

None.

6.10 RS Non-patient Instance (NPI) Storage

The RS Non-Patient Instance (NPI) Storage Services define a set of RESTful transactions that enable a user agent to retrieve, store, and search an origin server for instances that are not related to a patient.

An NPI Service manages a collection of resources belonging to the categories specified in Section 6.10.1. All NPI Storage Service origin servers shall support the Retrieve Capabilities, Retrieve, and Search transactions. Support for the Store transaction is optional. All NPI Storage Service user agents one or more of the Retrieve Capabilities, Retrieve, Store, or Search transactions.

6.10.1 Resources

An NPI Service manages resources from the same NPI Category. The target resource URIs have the following templates:

/{npi-name}

/{npi-name}/{uid}

Where

npi-name	= "color-palettes"
	/ "defined-procedure-protocols"
	/ "hanging-protocols"
	/ "implant-templates"
uid	; is the Unique Identifier of an NPI Instance

Table 6.10.1-1 contains the templates for the NPI Resource Categories. It also includes the PS3.3 Section in which the corresponding IOD is defined.

Table 6.10.1-1. Resource Categories, URI Templates and Descriptions

Resource Category	URI Template and Description	IOD	Storage Class	Information Model
Color Palette	/color-palettes{/uid}	A.58 in PS3.3	GG in PS3.4	X.1.3 in PS3.4
Defined Procedure Protocol	/defined-procedure-protocols{/uid}	A.82 in PS3.3	GG in PS3.4	HH.1.3 in PS3.4
Hanging Protocol	/hanging-protocols{/uid}	A.44 in PS3.3	GG in PS3.4	U.1.3 in PS3.4
Implant Template	/implant-templates{/uid}	A.61 in PS3.3	GG in PS3.4	BB.1.3 in PS3.4

The NPI SOP Classes are listed in Table GG.3-1 in PS3.4.

6.10.2 General Query Parameters

The Query Parameters in this section can be used with all NPI transactions.

6.10.2.1 Accept

The origin server shall support the Accept query parameter for all NPI transactions. See Section 6.1.1.5, “Accept Query Parameter”.

6.10.2.2 Character Set

The origin server shall support the Charset query parameter for all NPI transactions. See Section 6.1.2.2, “Character Set Query Parameter”.

6.10.3 Transactions

The NPI Service defines the transactions listed in Table 6.10.3-1

Table 6.10.3-1. NPI Service Transactions

Transaction	Method	Resource	Payload		Description
			Request	Response	
RetrieveCapabilities	OPTIONS	/	N/A	Capabilities Description	Retrieves a description of the capabilities of the NPI Service, including transactions, resources, query parameters, etc.
Retrieve	GET	/{npi-name}/{uid}	N/A	Instance and/or Status Report	Retrieves an Instance, specified by the target resource in an Acceptable DICOM Media Type.
Store	POST	/{npi-name}{/uid}	Instance(s)	Status Report	Stores one or more DICOM Instances in a DICOM media type, contained in the request payload, in the location referenced by the target resource URL.
Search	GET	/{npi-name} ?{params*}	N/A	Result(s) and/or Status Report	Searches the target resource for Instances that match the search parameters and returns a list of matches in an Acceptable DICOM Media Type.

The npi-name specifies the type of resource(s) contained in the payload.

Table 6.10.3-2 shows the target resources permitted for each transaction.

Table 6.10.3-2. Resources by Transaction

Resource	URI	Retrieve	Store	Search	Capabilities
NPI Service	/				X
All Instances	/{npi-name}		X	X	
Instance	/{npi-name}/{uid}	X	X		

6.10.3.1 Retrieve Capabilities Transaction

The Retrieve Capabilities transaction retrieves a machine-readable description of the NPI service implemented by an origin server. The response contains a machine-readable Capabilities Description. The Capabilities Description describes the transactions, resources, representations, etc. that are supported by the service(s).

An origin server implementation of an NPI Service shall support the Retrieve Capabilities transaction.

6.10.3.1.1 Request

The Retrieve Capabilities request uses the OPTIONS method and has the following format:

```
OPTIONS SP / SP version CRLF
Accept: 1#media-type CRLF
*(header-field CRLF)
CRLF
```

6.10.3.1.1.1 Resource

The target URL shall reference the Base URI ("/") of the service.

6.10.3.1.1.2 Query Parameters

There are no additional Query Parameters.

6.10.3.1.1.3 Request Header Fields

Table 6.10.3.1.1.3-1 shows the most common Mandatory, Conditional, and Optional header fields for this transaction.

Table 6.10.3.1.1.3-1. Request Header Fields

Header Fields	Value	Usage	Description
Accept	media-range	M	See 6.1.1.7.
Accept-Charset	1#charset	O	See 6.1.2.3.

6.10.3.1.1.4 Request Payload

The request has no payload.

6.10.3.1.2 Behavior

The origin server shall return a machine-readable description of its capabilities in an Acceptable Media Type.

6.10.3.1.3 Response

The format of the response is as follows:

```
version SP status-code SP reason-phrase CRLF
Content-Type: media-type CRLF
*(header-field CRLF)
CRLF
payload
```

6.10.3.1.3.1 Status Codes

A success response shall have a status code of 200 (OK) or 204 (No Content).

A failure response shall have a 400 or 500 level status code.

6.10.3.1.3.2 Response Header Fields

Table 6.10.3.1.3.2-1. Response Header Fields

Header Field	Value	Usage	Requirements
Content-Type	media-type	M	

Header Field	Value	Usage	Requirements
Content-Length	uint	C	Shall be present if no transfer coding has been applied. Shall be absent otherwise.
Transfer-Encoding	encoding	C	Shall be present if a transfer coding has been applied. Shall be absent otherwise.
~~ETag~~	~~entity-tag~~	~~C~~	~~Shall be present if the response status code is 200.~~

6.10.3.1.3.3 Response Payload

A success response shall have a payload containing a Capabilities Description in the Selected Media Type.

A failure response shall have a payload describing the error.

6.10.3.2 Retrieve Transaction

The Retrieve transaction retrieves the target NPI resource in a DICOM Media Type.

6.10.3.2.1 Request

The Retrieve request has the following syntax:

```
GETSP /{npi-name}/{uid} SP version CRLF
Accept: 1#dicom-media-type CRLF
[If-None-Match: entity-tag CRLF]
*(header-field CRLF)
CRLF
```

6.10.3.2.1.1 Resources

The target URL shall reference one of the resources shown in Table 6.10.3.2.1.1-1.

An origin server shall specify all supported npi-names in its Conformance Statement and in its response to the Retrieve Capabilities transaction.

Table 6.10.3.2.1.1-1. Resources and URI Templates

Resource	URI Template
Instance	/{npi-name}/{uid}

6.10.3.2.1.2 Query Parameters

There are no additional query parameters.

6.10.3.2.1.3 Request Header Fields

Table 6.10.3.2.1.3-1 shows the most common Mandatory, Conditional, and common Optional header fields for this transaction.

Table 6.10.3.2.1.3-1. Request Header Fields

Header Field	Value	Usage
Accept	dicom-media-type	M

6.10.3.2.1.4 Request Payload

The request shall have no payload.

6.10.3.2.2 Behavior

The origin server shall try to locate the target resource and if found, return it in an Acceptable DICOM Media Type.

6.10.3.2.3 Response

The response has the following syntax:

version SP status-code SP reason-phrase CRLF
Content-Type: dicom-media-type CRLF
[~~ETag: entity-tag CRLF~~]
[Last-Modified: HTTP-date CRLF]
*(header-field CRLF)
CRLF
Payload

6.10.3.2.3.1 Status Codes

The response shall have an appropriate status code. Table 6.10.3.2.3.1-1 contains the most common status codes for this transaction.

Table 6.10.3.2.3.1-1. Status Codes

Code	Description
200 (OK)	Indicates that the instance was successfully retrieved.
304 (Not Modified)	Indicates that the user agent's current representation is up to date, so no payload was returned. This status code shall only be returned for a Conditional Retrieve request containing an If-None-Match header field.
400 (Bad Request)	Indicates that the origin server did not store any of the representations contained in the request payload because of errors in the request message. For example, an invalid Query Parameter or an invalid SOP instance.
404 (Not Found)	Indicates that the origin server did not find a current representation for the target resource or is not willing to disclose that one exists. For example, an unsupported IOD, or SOP Instance not on server.
406 (Unsupported Media Type)	Indicates that the origin server does not support any of the Acceptable Media Types.

See [RFC7231] Section 6.

6.10.3.2.3.2 Response Header Fields

Table 6.10.3.2.3.2-1. Request Header Fields

Header Field	Value	Usage	Requirements
Content-Type	dicom-media-type	M	
Content-Length	uint	C	Shall be present if no transfer coding has been applied. Shall be absent otherwise.
Transfer-Encoding	encoding	C	Shall be present if a transfer coding has been applied. Shall be absent otherwise.
~~ETag~~	~~entity-tag~~	~~C~~	~~Shall be present if the response status code is 200 or 204.~~

6.10.3.2.3.3 Response Payload

A success response shall have a payload containing the DICOM instance specified by the target resource.

A failure response shall have a payload describing the error.

6.10.3.3 Store Transaction

This transaction requests that the origin server store the representations of the NPIs contained in the request payload so that they may be retrieved in the future using the Instance UIDs.

6.10.3.3.1 Request

Transactions in this service use the POST method. The request syntax is:

POST SP /{npi-name} {/uid} SP version CRLF
Content-Type: dicom-media-type CRLF
*(header-field CRLF)
CRLF
payload

6.10.3.3.1.1 Resources

The target URL shall reference one of the resources shown in Table 6.10.3.3.1.1-1.

An origin server shall specify all supported npi-names in its Conformance Statement and in its response to the Retrieve Capabilities transaction.

Table 6.10.3.3.1.1-1. Resources and URI Templates

Resource	URI Template	Description
All Instances	/{npi-name}	Stores representations of a set of Instances.
Instance	/{npi-name} {/uid}	Stores a representation of a single Instance with a UID equal to uid.

6.10.3.3.1.2 Query Parameters

There are no additional Query Parameters.

6.10.3.3.1.3 Request Header Fields

Table 6.10.3.3.1.3-1. Store Request Header Fields

Header Field	Value	Usage	Requirements
Content-Type	dicom-media-type	M	
Accept	dicom-media-type	M	
Content-Length	uint	C	Shall be present if no transfer coding has been applied. Shall be absent otherwise.
Transfer-Encoding	encoding	C	Shall be present if a transfer coding has been applied. Shall be absent otherwise.

6.10.3.3.1.4 Request Payload

The request payload shall be present and shall contain one or more representations in the DICOM Media Type specified by the Content-Type header field of the message, or for multipart payloads the Content-Type header field of each part.

6.10.3.3.2 Behavior

The origin server stores the representations contained in the request payload so that they may be retrieved later using the Retrieve transaction.

Before storing the representations, the origin server may coerce data elements.

If any element is coerced, the Original Attribute Sequence (0400,0561) (see C.12.1 in PS3.3) shall be included in the stored DICOM instances. Both the Original Attribute Sequence and the response shall describe the modifications.

6.10.3.3.3 Response

The response shall have the following syntax:

version SP status-code SP reason-phrase CRLF
*(header-field CRLF)
CRLF
[Status Report]

6.10.3.3.3.1 Status Codes

The response shall have an appropriate status code. Table 6.10.3.3.3.1-1 contains the most common status codes for this transaction.

Table 6.10.3.3.3.1-1. Common Status Codes

Status Code	Description
200 (OK)	Indicates that the origin server successfully stored or created at least one of the representations contained in the request payload and is returning a response payload.
201 (Created)	Indicates that the origin server successfully created at least one of the representations contained in the request payload and may be returning a response payload.
202 (Accepted)	Indicates that the origin server successfully validated the request message, but has not yet stored or created the representations in the request payload. The origin server may or may not have validated the payload. The user agent can use a Query or Retrieve transaction later to determine if the request has completed.
204 (No Content)	Indicates that the origin server successfully stored all the representations contained in the request payload without any modifications and is not returning a response payload.
400 (Bad Request)	Indicates that the origin server did not store any of the representations contained in the request payload because of errors in the request message. For example, an invalid Query Parameter or an invalid SOP instance.
404 (Not Found)	Indicates that the origin server did not find a current representation for the target resource or is not willing to disclose that one exists. For example, an unsupported IOD, or SOP Instance not on server.
409 (Conflict)	Indicates that the request could not be completed due to a conflict with the current state of the target resource.
415 (Unsupported Media Type)	Indicates that the origin server does not support the media type specified in the Content-Type header field of the request, and none of the representations contained in the request were processed or stored.

6.10.3.3.3.2 Response Header Fields

Table 6.10.3.3.3.2-1. Store Response Header Fields

Header Field	Value	Usage	Requirements
Content-Type	dicom-media-type	M	
Content-Length	uint	C	Shall be present if no transfer coding has been applied. Shall be absent otherwise.
Transfer-Encoding	encoding	C	Shall be present if a transfer coding has been applied. Shall be absent otherwise.

6.10.3.3.3.3 Response Payload

If the origin server failed to store or modified any representations in the request payload, the response payload shall contain a Status Report describing any additions, modifications, or deletions to the stored representations. The Status Report may also describe any warnings or other useful information.

6.10.3.4 Search Transaction

The Search transaction searches the collection of NPI Instances contained in the target resource. The search criteria are specified in the query parameters. Each match includes the default and requested attributes from the matching Instance. A successful response returns a list describing the matching Instances.

6.10.3.4.1 Request

The Search transaction uses the GET method and has the following syntax:

GET SP /{npi-name} {?parameter*} SP version CRLF
Accept: 1#dicom-media-type CRLF
*(header-field CRLF)
CRLF

6.10.3.4.1.1 Resources

The target URL shall reference one of the resources shown in Table 6.10.3.4.1.1-1.

An origin server shall specify all supported npi-names in its Conformance Statement and in its response to the Retrieve Capabilities transaction.

Table 6.10.3.4.1.1-1. Resources and URI Templates

Resource	URI Template	Description
All Instances	/{npi-name}	Searches a collection of NPI Instances

6.10.3.4.1.2 Query Parameters

The parameters in the query component of the target URL specify the matching criteria, the attribute values to be returned, and the results to be returned. The URI template for the query parameters is:

{?parameter*} = "?" {&match*} {&include*} {&offset} {&limit}

See Section 6.7.1.1, “Request” for a description of the syntax of Search Query Parameters.

6.10.3.4.1.2.1 Attributes and Behaviors

For each Resource Category the origin server supports, it shall support the behaviors and matching key attributes specified in the corresponding sections in Table 6.10.3.4.1.2.1-1.

Table 6.10.3.4.1.2.1-1. NPI Resource Search Attributes

Resource Category	Defined Attributes and Matching Key Types
Color Palette	X.6.1.2 in PS3.4
Defined Procedure Protocol	HH.6.1.2 in PS3.4
Hanging Protocol	U.6.1.2 in PS3.4
Implant Template	BB.6.1.2 in PS3.4

6.10.3.4.1.3 Request Header Fields

Table 6.10.3.4.1.3-1. Search Request Header Fields

Header Field	Value	Usage
Accept	dicom-media-type	M

6.10.3.4.1.4 Request Payload

The request has no payload.

6.10.3.4.2 Behavior

The origin server shall perform the search indicated by the request, using the matching behavior specified in Section 6.7.1.2 and in the corresponding sections in Table 6.10.3.4.1.2.1-1, and return a response containing the search results, or an appropriate Status Report.

The rules for search results are specified in Section 6.7.1.2.

6.10.3.4.3 Response

A success response shall have a status code of 200 (OK) and a payload containing the search results in the Selected Media Type.

A failure response shall contain a Status Report describing the error(s) encountered.

6.10.3.4.3.1 Status Codes

The response shall have an appropriate status code. Table 6.10.3.4.3.1-1 contains the most common status codes for this transaction.

Table 6.10.3.4.3.1-1. Common Status Codes

Status Code	Description
200 (OK)	Indicates that the origin server found and returned at least one resource matching the request.
400 (Bad Request)	Indicates that the origin server did not return any search results because of errors in the request message.
404 (Not Found)	Indicates that the origin server did not find any resources matching the request, or is not willing to disclose that any exist.
406 (Unsupported Media Type)	Indicates that the origin server does not support any of the Acceptable Media Types.
409 (Conflict)	Indicates that the request could not be completed due to a conflict with the current state of the target resource.

6.10.3.4.3.2 Response Header Fields

Table :

Table 6.10.3.15. Search Response Header Fields

Header Field	Value	Usage	Requirement
Content-Type	dicom-media-type	M	
Content-Length	uint	C	Shall be present if no transfer coding has been applied. Shall be absent otherwise.
Transfer-Encoding	encoding	C	Shall be present if a transfer coding has been applied. Shall be absent otherwise.

6.10.3.4.3.3 Response Payload

A success response payload shall contain Search results.

A failure response payload shall contain a Status Report describing any failures, warnings or other useful information.

6.10.4 Media Types

The origin server shall support the media types listed as Default or Required in Table 6.10.4-1 for all NPI transactions.

Table 6.10.4-1. Default, Required, and Optional Media Types

Media Type	Usage
application/dicom	Required
application/dicom+json	Default
multipart/related; type="application/dicom+xml"	Optional

6.10.5 Conformance

The origin server shall support the transactions listed as Required in Table 6.10.5-1.

Table 6.10.5-1. Required and Optional Transactions

Transaction	Support	Section
Retrieve Capabilities	Required	6.10.3.1
Retrieve	Required	6.10.3.2
Store	Optional	6.10.3.3
Search	Required	6.10.3.4

Implementations shall specify in their Conformance Statement (see PS3.2) and the Capabilities Description (see Section 6.8.1.2):

- The implementations role: origin server, user agent, or both
- The supported resources (IODs) for each role

In addition, for each supported transaction they shall specify:

- The supported Query Parameters, including optional attributes, if any
- The supported DICOM Media Types
- The supported character sets (if other than UTF-8)

7 Object Types

Retired. See Section 6.1.1.

8 Parameters of the WADO-URI Request

8.1 Parameters Available for all DICOM Objects

Parameters specified in this section are applicable to all supported DICOM SOP Classes.

Some of the Query Parameters specified in this section have values that are UIDs. Table 8.1-1 lists error status codes related to UIDs.

Table 8.1-1. UID Related Errors

Status Code	Reason
400 (Bad Request)	The UID is not a correctly formatted, or it references a resource that is not a SOP Instance (i.e., references an instance of a different entity, e.g., a Study).
404 (Not Found)	No resource corresponding to the UID exists.
410 (Gone)	The resource corresponding to the UID, once existed, but no longer exists.

Note

An origin server that does not check the format of the UID, or does not maintain a history of removed resources, may return a 404 (Not Found).

An error response may include a payload containing an appropriate error message.

Note

To identify a DICOM Object, only one UID is required, because any UID is globally unique. However, the standard requires that the UID of the higher levels in the DICOM Information Model are specified (i.e., series and study), in order to support the use of DICOM devices that support only the baseline hierarchical (rather than extended relational) Query/Retrieve model, which requires the Study Instance UID and Series Instance UID to be defined when retrieving an SOP Instance, as defined in PS3.4.

8.1.1 Request Type

This parameter specifies that this is a URI service request. The parameter name shall be "requestType", and the value shall be "WADO". It is REQUIRED.

If the value is other than "WADO", and the origin server does not support the value, the response shall be 400 (Bad Request), and may include a payload containing an appropriate error message.

8.1.2 Unique Identifier of the Study

Study Instance UID as defined in PS3.3. This parameter is REQUIRED.

The parameter name shall be "studyUID" for URI mode.

The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length with a NULL character.

Error status codes related to UIDs are specified in Table 8.1-1.

8.1.3 Unique Identifier of the Series

Series Instance UID as defined in PS3.3. This parameter is REQUIRED.

The parameter name shall be "seriesUID" for URI mode.

The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length with a NULL character.

Error status codes related to UIDs are specified in Table 8.1-1.

8.1.4 Unique Identifier of the Object

SOP Instance UID as defined in PS3.3. This parameter is REQUIRED.

The parameter name shall be "objectUID" for URI mode.

The value shall be encoded as a unique identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length with a NULL character.

Error status codes related to UIDs are specified in Table 8.1-1.

8.1.5 Acceptable Media Type of the Response

This parameter contains one or more Acceptable Media Types as defined in Section 6.1.1.4. This parameter is OPTIONAL for URI mode.

In URI mode the parameter name shall be "contentType", and its value shall contain one or more media types.

See Section 6.1.1 for details.

8.1.6 Acceptable Character Sets

Character set with which the returned objects are to be encoded, as defined in the [RFC7230]. This parameter is OPTIONAL for URI mode.

The parameter name shall be "charset" for URI mode.

See Section 6.1.2 for details.

8.1.7 Anonymize Object

Removal of all patient identification information from within the DICOM Objects, if not already done, as defined in PS3.15. This parameter is OPTIONAL. For the URI mode, it shall only be present if contentType is application/dicom.

The parameter name shall be "anonymize" for URI mode.

The value shall be "yes".

The Server may return an error if it either cannot or refuses to anonymize these objects.

The Server shall return a new SOP Instance UID if the content of the object has not already been anonymized.

If this parameter has any other value than "yes", the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

Note

1. This standard does not introduce any security-related requirements. It is likely that the information contained within DICOM Objects identifies the patient. The protocol used (that is HTTP) can be replaced by HTTPs, which is its secure extension, to protect the information in transit. The underlying DICOM implementation decides whether or not to grant access to a particular DICOM object based on whatever security policy or mechanism it has in place. A server is unlikely to fulfill a request from an unknown user (e.g., accessed via the HTTP protocol) unless it is certain that the data requested has no patient identifying information within it and has been approved for public viewing.

2. The Anonymize object enables, for example, teaching files systems or clinical trial applications to offer an access to original images stored in a PACS, without disclosing the patient's identity, and requiring storage of a (de-identified) copy of the original image. Anonymization is the responsibility of the Server. In order to preserve patient confidentiality, the

Server likely will refuse to deliver an anonymized SOP instance to an unknown or unauthorized person unless the Server is certain that the SOP instance holds no patient identifying information. This would include "blanking out" any annotation area(s) containing nominative information burned into the pixels or in the overlays.

8.1.9 Retired

See PS3.18-2017b.

8.2 Parameters for DICOM Images

These parameters shall only be included when a request is made for a Single Frame or Multi-frame Image or Video Objects as defined in Section 6.1.1.2.

If any of these parameters are included in a request for non-Image objects (e.g., for SR objects), then the response shall be a 400 (Bad Request), and may include a payload containing an appropriate error message.

8.2.1 Annotation on the Object

Annotation of objects retrieved and displayed as an image. This parameter is OPTIONAL for the URI mode. It shall not be present if contentType is application/dicom, or is a non-image media type (e.g., text/*). When it is not present for image objects, no additional annotation may be burnt in.

When used in conjunction with a presentation state object, it shall be applied after the presentation on the images. When used in conjunction with the region parameter, it shall be applied after the selection of the region.

The parameter name shall be "annotation" for URI mode. Its value is a non-empty list of one or more of the following items, separated by a "," character:

- "patient", for displaying patient information on the image (e.g., patient name, birth date,…)
- "technique", for displaying technique information of the image (e.g., image number, study date, image position,…).

Note

The exact nature and presentation of the annotation is determined by the Server. The annotation is burned into the returned image pixels.

The origin server may support additional values for this parameter.

The origin server shall ignore any values it does not support. If unsupported values are present, the origin server shall include a following Warning header field:

Warning: 299 {+service}: The following annotation values are not supported: <values>

and may include a payload containing an appropriate warning message.

8.2.2 Viewport Dimensions

The viewport parameters specify the dimensions of the user agent's viewport. The Viewport Rows and Columns parameters specify the height and width, in pixels, of the returned image.

If these parameters specify viewport dimensions that are either ill-defined or not supported, then the response shall be a 400 (Bad Request), and may include a payload containing an appropriate error message.

8.2.2.1 Number of Pixel Rows

The parameter name shall be "rows" for URI mode.

The value shall be expressed as an integer, representing the image height to be returned. It is OPTIONAL for the URI mode. It shall not be present if contentType is application/dicom.

If both "rows" and "columns" are specified, then each shall be interpreted as a maximum, and a size will be chosen for the images within these constraints, maintaining the correct aspect ratio. If the number of rows is absent and the number of columns is present, the number of rows shall be chosen in order to maintain the correct aspect ratio. If both are absent, the images (or selected region) are sent in their original size (or the size of the presentation state applied on the images), resulting as one pixel of screen image for each value in the images data matrix.

The value shall be encoded as an integer string (IS), as specified in PS3.5.

8.2.2.2 Number of Pixel Columns

The parameter name shall be "columns" for URI mode.

The value shall be expressed as an integer, representing the image width to be returned. It is OPTIONAL for the URI mode. It shall not be present if contentType is application/dicom.

If both "rows" and "columns" are specified, then each shall be interpreted as a maximum, and a size will be chosen for the images within these constraints, maintaining the correct aspect ratio. If the number of columns is absent and the number of rows is present, the number of columns shall be chosen in order to maintain the correct aspect ratio. If both are absent, the images (or selected region) is sent in its original size (or the size of the presentation state applied on the images), resulting as one pixel of screen for one pixel of the images.

The value shall be encoded as an integer string (IS), as specified in PS3.5.

8.2.3 Reserved

8.2.4 Region of the Image

This parameter allows selection of a rectangular region of an image matrix to be retrieved. The purpose of this parameter is to allow a user to view a selected area of the image matrix, for example at higher magnification.

The parameter is OPTIONAL for the URI mode.

The parameter name shall be "region" for URI mode.

It shall only be present if the Acceptable Media Types are Rendered Media Types. See Section 6.1.1.3.

It shall not be present if the Unique Identifier of the Presentation Object parameter is present.

The value shall be expressed as a list of four positive decimal strings, separated by the ',' character, representing the region of the source images to be returned. These decimal values shall be values in a normalized coordinate system relative to the size of the original image matrix measured in rows and columns, with values ranging from 0.0 to 1.0, and representing in the following order:

- the x position of the top left hand corner of the region to be retrieved, 0.0 corresponding to the first column of the image matrix.
- the y position of the top left hand corner of the region to be retrieved, 0.0 corresponding to the top row of the image matrix.
- the x position of the bottom right hand extent of the region, 1.0 corresponding to the last column of the image matrix, 0.0 being forbidden.
- the y position of the bottom right hand extent of the region, 1.0 corresponding to the last row of the image matrix, 0.0 being forbidden.

  Note

  The Server may or may not support this parameter.

If this parameter is supported, an image matrix corresponding to the specified region shall be returned with size corresponding to the specified normalized coordinate values otherwise the complete image matrix shall be returned. If the presentationUID parameter is present, the region shall be selected after the corresponding presentation state has been applied on the images.

If this parameter specifies an ill-defined region, the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

8.2.5 Windowing

The Windowing parameters are optional; however, if either is present, both shall be present. If only one is present the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

The Windowing and Presentation State parameters shall not be present in the same request. If both are present the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

The Windowing parameters shall not be present if contentType is application/dicom; if either is present the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

8.2.5.1 Window Center of the Image

The parameter name shall be "windowCenter" for URI mode.

It controls the window center of the images as defined in PS3.3. This parameter is OPTIONAL for the URI mode.

The value shall be encoded as a decimal string (DS), as specified in PS3.5.

8.2.5.2 Window Width of the Image

The parameter name shall be "windowWidth" for URI mode.

It controls the window width of the images as defined in PS3.3. This parameter is OPTIONAL for the URI mode.

The value shall be encoded as a decimal string (DS), as specified in PS3.5.

8.2.6 Reserved

8.2.7 Frame Number

The parameter name shall be "frameNumber" for URI mode.

Specifies that the single frame with that number within a multi-frame image object, as defined in PS3.3 that shall be returned. It is OPTIONAL. It shall not be present if contentType is application/dicom.

If the target resource is:

- a single frame image and the frame number is present and not 1, or
- a multi-frame image and the frame number is not between 1 and the number of frames in the image (inclusive), or
- not an image

then the response shall be a 400 (Bad Request), and may include a payload containing an appropriate error message.

The value shall be encoded as an integer string (IS), as specified in PS3.5.

8.2.8 Image Quality

The parameter name shall be "imageQuality" for URI mode. It is OPTIONAL for the URI mode. It shall not be present if contentType is application/dicom, except if the transferSyntax parameter is present and corresponds to a lossy compression.

If the requested media type is for a lossy compressed image (e.g., image/jpeg), this parameter indicates the required quality of the image to be returned within the range 1 to 100, 100 being the best quality.

The value shall be encoded as an integer string (IS), as specified in PS3.5.

Note

Decompression and re-compression may degrade the image quality if the original image was already irreversibly compressed. In case the image has been already lossy compressed using the same format as required (e.g., jpeg), it may be sent as it is without decompressing and re-compressing it.

If the value of this parameter is less than 1 or greater than 100, then the response shall be a 400 (Bad Request), and may include a payload containing an appropriate error message.

Note

The specific interpretation of the meaning of this parameter is left to the interpretation of the implementers of the standard.

8.2.9 Unique Identifiers of the Presentation State Object

The parameters in this Section specify the Series and SOP Instance UIDs of a Presentation State. They are OPTIONAL; however, if either is present, both shall be present. If only one is present the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

If the Presentation State parameters are present, then only the Annotation, Image Quality, Viewport and Region parameters may also be present. If any of the other image rendering parameters described in Section 8.2 are present the response shall be 400 (Bad Request), and may include a payload containing an appropriate error message.

If the target resource is not a Presentation State then the response will be 400 (Bad Request), and may include a payload containing an appropriate error message.

The Presentation State parameters shall not be present if contentType is application/dicom; if either is present the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

8.2.9.1 Unique Identifier of the Presentation State SOP Instance

The parameter name shall be "presentationUID" for URI mode.

The value is the SOP Instance UID of the Presentation State storage object to be applied to the images.

The value shall be encoded as a unique identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length with a NULL character.

If this parameter is present, then the Region of the Image parameter shall not be present. See Section 8.2.4.

If the Presentation Size Mode in the presentation state is SCALE TO FIT or TRUE SIZE, then the displayed area specified in the presentation shall be scaled to fit the size specified by the rows and columns parameters if present, otherwise the displayed area selected in the presentation state will be returned without scaling.

Note

1. The intent of the TRUE SIZE mode in the presentation state cannot be satisfied, since the physical size of the pixels displayed by the web browser is unlikely to be known. If the Presentation Size Mode in the presentation state is MAGNIFY, then the displayed area specified in the presentation shall be magnified (scaled) as specified in the presentation state. It will then be cropped to fit the size specified by the rows and columns parameters, if present.

2. Any Displayed Area relative annotations specified in the presentation state are rendered relative to the Specified Displayed Area within the presentation state, not the size of the returned image.

Though the output of the presentation state is defined in DICOM to be in P-Values (grayscale values intended for display on a device calibrated to the DICOM Grayscale Standard Display Function PS3.14), the grayscale or color space for the images returned by the request is not defined by this standard.

8.2.9.2 Unique Identifier of the Series Containing the Presentation SOP Instance

The parameter name shall be "presentationSeriesUID" for URI mode.

The value is the Series Instance UID of the Series containing the Presentation State storage object to be applied on the images.

If this parameter is present, then the Region of the Image parameter shall not be present. See Section 8.2.4.

The value shall be encoded as a unique identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length with a NULL character.

Note

As specified in DICOM, the Presentation State will be in the same study as the images it applies to.

8.2.10 Reserved

8.2.11 Transfer Syntax UID

For the URI mode the parameter name shall be "transferSyntax" containing one value.

The Transfer Syntax to be used within the DICOM image objects, as specified in PS3.6. This parameter is OPTIONAL for the URI mode. It shall not be present if contentType is other than application/dicom.

By default the DICOM object(s) returned shall be encoded in Explicit VR Little Endian. Neither Implicit VR, nor Big Endian shall be used. The response shall be the Transfer Syntax requested if possible. If it is not possible for the response to be sent using the requested transfer syntax then the Explicit VR Little Endian Uncompressed Transfer Syntax shall be used, unless the pixel data in its compressed form is of such length that it cannot be encoded in the Explicit VR Little Endian Uncompressed Transfer Syntax.

Note

~~The transfer syntax can be one of the JPIP Transfer Syntaxes, in which case the returned objects will contain the URL of the JPIP provider for retrieving the pixel data.~~

1. If transcoding to the Explicit VR Little Endian Transfer Syntax, a VR of UN may be needed for the encoding of Data Elements with explicit VR whose value length exceeds 65534 (2^{16}-2) (FFFEH, the largest even length unsigned 16 bit number) but which are defined to have a 16 bit explicit VR length field. See Section 6.2.2 in PS3.5.

2. The transfer syntax can be one of the JPIP Transfer Syntaxes, in which case the returned objects will contain the URL of the JPIP provider for retrieving the pixel data.

The value(s) shall be encoded as a unique identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length with a NULL character.

A URI Query Component Syntax (Normative)

This Standard uses the URI syntax as defined in [RFC3986] *Uniform Resource Identifier (URI): Generic Syntax* and extends it by specifying the syntax of the query component of DICOM URIs. The grammar for the query component is defined using [RFC5234] *Augmented BNF for Syntax Specifications: ABNF*.

DICOM URIs may use the query component of the URI to specify request parameters. The following grammar defines the general syntax of parameters contained in the query component of the URI. Specific HTTP transactions defined elsewhere in this standard may further refine the legal <name> and/or <value> rules.

```
query-component = parameter [ *("&" parameter) ]
parameter       = name "="" value
name            = *qchar
value           = *qchar
qchar           = unreserved / pct-encoded / qspecial
qspecial        = "/" / "?" / ":" / "@" / "!" / "$" / "'"
                / "(" / ")" / "*" / "+" / "," / ";"
```

The following rules are defined in [RFC3986] (Normative). They are reproduced here for convenience.

```
unreserved     = ALPHA / DIGIT / "-" / "." / "_" / "~"
pct-encoded    = "%" HEXDIG HEXDIG
```

Note

1. This grammar allows the query component to contain any of the legal characters as defined by [RFC3986].
2. No whitespace is permitted in URIs. Whitespace around line breaks and the line breaks themselves should be stripped before parsing the URI (See [RFC3986] Appendix C).
3. [RFC3986] does not permit an empty query component, i.e., if the "?" appears in the URI then there must be some legal query parameters in the URI.
4. The <qchar> rule defined above is the <pchar> rule of [RFC3986], which defines the legal character for the query component, minus the characters "=" and "&".

B Examples (Informative)

B.1 Retrieving a Simple DICOM Image in JPEG

http://www.hospital-stmarco/radiology/wado.php?requestType=WADO
&studyUID=1.2.250.1.59.40211.12345678.678910
&seriesUID=1.2.250.1.59.40211.789001276.14556172.67789
&objectUID=1.2.250.1.59.40211.2678810.87991027.899772.2

B.2 Retrieving a DICOM SR in HTML

http://server234/script678.asp?requestType=WADO
&studyUID=1.2.250.1.59.40211.12345678.678910
&seriesUID=1.2.250.1.59.40211.789001276.14556172.67789
&objectUID=1.2.250.1.59.40211.2678810.87991027.899772.2
&charset=UTF-8

B.3 Retrieving a Region of A DICOM Image

Retrieving a region of a DICOM image, converted if possible in JPEG2000, with annotations burned into the image containing the patient name and technical information, and mapped into a defined image size:

https://aspradio/imageaccess.js?requestType=WADO
&studyUID=1.2.250.1.59.40211.12345678.678910
&seriesUID=1.2.250.1.59.40211.789001276.14556172.67789
&objectUID=1.2.250.1.59.40211.2678810.87991027.899772.2
&contentType=image%2Fjp2;level=1,image%2Fjpeg;q=0.5
&annotation=patient,technique
&columns=400
&rows=300
&region=0.3,0.4,0.5,0.5
&windowCenter=-1000
&windowWidth=2500

B.4 Retrieving As A DICOM Media Type

Retrieving a DICOM image object using the baseline 8-bit lossy JPEG transfer syntax, and de-identified:

http://www.medical-webservice.st/RetrieveDocument?requestType=WADO
&studyUID=1.2.250.1.59.40211.12345678.678910
&seriesUID=1.2.250.1.59.40211.789001276.14556172.67789
&objectUID=1.2.250.1.59.40211.2678810.87991027.899772.2
&contentType=application%2Fdicom
&anonymize=yes
&transferSyntax=1.2.840.10008.1.2.4.50

C Applications (Informative)

There are multiple applications, in which DICOM and "web-based" environments are interacting. "Web-based" means information and communication systems that are using Internet related technologies (Web, e-mail…). The basic feature supported by this standard is a mechanism for the "Web-based" system to retrieve a DICOM object from the "DICOM-based" system.

Typical applications are:

i. Referencing an image or a report from an electronic patient record (EPR)

ii. Including references to images in an e-mail

iii. Providing access by outside referring doctors to a hospital web server that contains references to reports, images and waveforms

iv. Providing access to anonymized DICOM reports, images and waveforms via a web server, for teaching purposes and for clinical trials.

To retrieve DICOM Objects using "WADO", the "web-based" system must "know" the UIDs (Study, Series, SOP Instance) of the objects it needs to retrieve. These may be obtained through different methods (reception of a standardized message containing a document containing the reference to the DICOM Objects, query of other systems…) that are beyond the scope of this standard.

D IANA Character Set Mapping

Table D-1 provides a mapping of some IANA Character Set Registry Preferred MIME Names to DICOM Specific Character Set Defined Terms.

Table D-1. IANA Character Set Mapping

IANA Preferred MIME Name	DICOM Defined Terms for Specific Character Set (0008,0005)	Language
ISO-8859-1	ISO_IR 100	Latin-1 Latin alphabet
ISO-8859-2	ISO_IR 101	Latin-2 Eastern European
ISO-8859-3	ISO_IR 109	Latin alphabet #3
ISO-8859-4	ISO_IR 110	Latin alphabet #4
ISO-8859-5	ISO_IR 144	Cyrillic
ISO-8859-6	ISO_IR 127	Arabic
ISO-8859-7	ISO_IR 126	Greek
ISO-8859-8	ISO_IR 138	Hebrew
ISO-8859-9	ISO_IR 148	Latin alphabet #5
TIS-620	ISO_IR 166	Thai
ISO-2022-JP	ISO 2022 IR 13\ISO 2022 IR 87	Japanese
ISO-2022-KR	ISO 2022 IR 6\ISO 2022 IR 149	Korean
ISO-2022-CN	ISO 2022 IR 6\ISO 2022 IR 58	Chinese
GB18030	GB18030	Chinese
GBK	GBK	Chinese
UTF-8	ISO_IR 192	Unicode

E Retired

F DICOM JSON Model

F.1 Introduction to JavaScript Object Notation (JSON)

JSON is a text-based open standard, derived from JavaScript, for representing data structures and associated arrays. It is language-independent, and primarily used for serializing and transmitting lightweight structured data over a network connection. It is described in detail by the Internet Engineering Task Force (IETF) in [RFC4627], available at http://www.ietf.org/rfc/rfc4627.txt.

The DICOM JSON Model complements the XML-based Native DICOM Model, by providing a lightweight representation of data returned by DICOM web services. While this representation can be used to encode any type of DICOM Data Set it is expected to be used by client applications, especially mobile clients, such as described in the QIDO-RS use cases (see Annex HHH “Transition from WADO to RESTful Services (Informative)” in PS3.17).

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.

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.

F.2.1.1 Examples

F.2.1.1.1 Native DICOM Model

```
<?xml version="1.0" encoding="UTF-8" xml:space="preserve" ?>
<NativeDicomModel>
  <DicomAttribute tag="0020000D" vr="UI" keyword="StudyInstanceUID">
    <Value number="1">1.2.392.200036.9116.2.2.2.1762893313.1029997326.945873</Value>
  </DicomAttribute>
</NativeDicomModel>
…
<?xml version="1.0" encoding="UTF-8" xml:space="preserve" ?>
<NativeDicomModel>
  <DicomAttribute tag="0020000D" vr="UI" keyword="StudyInstanceUID">
    <Value number="1">1.2.444.200036.9116.2.2.2.1762893313.1029997326.945876</Value>
  </DicomAttribute>
</NativeDicomModel>
```

F.2.1.1.2 DICOM JSON Model

```
[
  {
     "0020000D": {
      "vr": "UI",
      "Value": [ "1.2.392.200036.9116.2.2.2.1762893313.1029997326.945873" ]
    }
  },
  {
    "0020000D" : {
      "vr": "UI",
      "Value": [ "1.2.392.200036.9116.2.2.2.2162893313.1029997326.945876" ]
    }
```

```
  }
]
```

F.2.2 DICOM JSON Model Object Structure

The DICOM JSON Model object is a representation of a DICOM Data Set.

The internal structure of the DICOM JSON Model object is a sequence of objects representing attributes within the DICOM Data Set.

Attribute objects within a DICOM JSON Model object must be ordered by their property name in ascending order.

Group Length (gggg,0000) attributes shall not be included in a DICOM JSON Model object.

The name of each attribute object is:

- The eight character uppercase hexadecimal representation of a DICOM Tag

Each attribute object contains the following named child objects:

- vr: A string encoding the DICOM Value Representation. The mapping between DICOM Value Representations and JSON Value Representations is described in Section F.2.3.
- At most one of:
  - Value: An array containing one of:
    - The Value Field elements of a DICOM attribute with a VR other than PN, SQ, OB, OD, OF, OL, OW, or UN (described in Section F.2.4)

      The encoding of empty Value Field elements is described in Section F.2.5
    - The Value Field elements of a DICOM attribute with a VR of PN. The non-empty name components of each element are encoded as a JSON strings with the following names:
      - Alphabetic
      - Ideographic
      - Phonetic
    - JSON DICOM Model objects corresponding to the sequence items of an attribute with a VR of SQ

      Empty sequence items are represented by empty objects
- BulkDataURI: A string encoding the WADO-RS URL of a bulk data item describing the Value Field of an enclosing Attribute with a VR of DS, FL, FD, IS, LT, OB, OD, OF, OL, OW, SL, SS, ST, UC, UL, UN, US, or UT (described in Section F.2.6)
- InlineBinary: A base64 string encoding the Value Field of an enclosing Attribute with a VR of OB, OD, OF, OL, OW, or UN (described in Section F.2.7)

  Note

  1. For Private Data Elements, the group and element numbers will follow the rules specified in Section 7.8.1 in PS3.5
  2. The person name representation is more closely aligned with the DICOM Data Element representation than the DICOM PS3.19 XML representation.

F.2.3 DICOM JSON Value Representation

The value representation (VR) is included in each DICOM JSON Model attribute object and named "vr". For example:

"vr": "CS"

All DICOM Value Representations are mapped to specified JSON Data Types (see Table F.2.3-1). The JSON encodings shall conform to the Definition, Character Repertoire (if applicable) and Length of Value specified for that Value Representation (see Section 6.2 “Value Representation (VR)” in PS3.5) with the following exceptions:

- Attributes with a Value Representation of AT shall be restricted to eight character uppercase hexadecimal representation of a DICOM Tag

Table F.2.3-1. DICOM VR to JSON Data Type Mapping

VR Name	Type	JSON Data Type
AE	Application Entity	String
AS	Age String	String
AT	Attribute Tag	String
CS	Code String	String
DA	Date	String
DS	Decimal String	Number
DT	Date Time	String
FL	Floating Point Single	Number
FD	Floating Point Double	Number
IS	Integer String	Number
LO	Long String	String
LT	Long Text	String
OB	Other Byte	Base64 encoded octet-stream
OD	Other Double	Base64 encoded octet-stream
OF	Other Float	Base64 encoded octet-stream
OL	Other Long	Base64 encoded octet-stream
OW	Other Word	Base64 encoded octet-stream
PN	Person Name	Object containing Person Name component groups as strings (see Section F.2.2)
SH	Short String	String
SL	Signed Long	Number
SQ	Sequence of Items	Array containing DICOM JSON Objects
SS	Signed Short	Number
ST	Short Text	String
TM	Time	String
UC	Unlimited Characters	String
UI	Unique Identifier (UID)	String
UL	Unsigned Long	Number
UN	Unknown	Base64 encoded octet-stream
UR	Universal Resource Identifier or Universal Resource Locator (URI/URL)	String
US	Unsigned Short	Number
UT	Unlimited Text	String

Although data, such as dates, are represented in the DICOM JSON model as strings, it is expected that they will be treated in the same manner as the original attribute as defined by Chapter 6 in PS3.6.

F.2.4 DICOM JSON Value Multiplicity

The value or values of a given DICOM attribute are given in the "Value" array. The value multiplicity (VM) is not contained in the DICOM JSON object.

For example:

"Value": ["bar", "foo"]

or:

"Value": ["bar"]

F.2.5 DICOM JSON Model Null Values

If an attribute is present in DICOM but empty (i.e., Value Length is 0), it shall be preserved in the DICOM JSON attribute object containing no "Value", "BulkDataURI" or "InlineBinary".

If a multi-valued attribute has one or more empty values these are represented as "null" array elements. For example:

"Value": ["bar", null, "foo"]

If a sequence contains empty items these are represented as empty JSON object in the array.

"Value": [{ … }, { }, { … }]

F.2.6 BulkDataURI

If an attribute contains a "BulkDataURI" , this contains the URI of a bulk data element as defined in Table A.1.5-2 in PS3.19.

F.2.7 InlineBinary

If an attribute contains an "InlineBinary", this contains the base64 encoding of the enclosing attribute's Value Field.

There is a single InlineBinary value representing the entire Value Field, and not one per Value in the case where the Value Multiplicity is greater than one. E.g., a LUT with 4096 16 bit entries that may be encoded in DICOM with a Value Representation of OW, with a VL of 8192 and a VM of 1, or a US VR with a VL of 8192 and a VM of 4096 would both be represented as a single InlineBinary string.

All rules (e.g., byte ordering and swapping) in DICOM PS3.5 apply.

Note

Implementers should in particular pay attention to the PS3.5 rules regarding the value representations of OD, OF, OL and OW.

F.3 Transformation with other DICOM Formats

F.3.1 Native DICOM Model XML

The transformation between the Native DICOM Model XML and the DICOM JSON model cannot be done through the use of generic XML - JSON converters.

The mapping between the two formats is as follows (see also Table F.3.1-1):

- The XML "NativeDicomModel" element maps to the DICOM JSON Model Object
- Each "DicomAttribute" element maps to an attribute object within the DICOM JSON model object
  - The "tag" attribute maps to the JSON object name

    - The Native DICOM Model XML allows for duplicate Tag values and the DICOM JSON model does not. To resolve this, private attribute Tag values must be remapped according to the conflict avoidance rules specified in Section 7.8.1 “Private Data Element Tags” in PS3.5.
  - The "vr" attribute maps to the "vr" child string
- "Value" elements map to members of the "Value" child array
  - A "Value" element with the attribute "number=n" maps to "Value[n-1]"
  - Empty "Value" elements are represented by "null" entries in the "Value" array
- "PersonName" elements map to objects within the "Value" array. For a "PersonName" element with the attribute "number=n":
  - The "Alphabetic" element maps to "Value[**n-1**].Alphabetic"
  - The "Ideographic" element maps to "PersonName[**n**].Ideographic"
  - The "Phonetic" element maps to "PersonName[**n**].Phonetic"
- "Item" elements map to members of the "Value" child array
  - An "Item" element with the attribute "number=n" maps to "Value[n-1]"
  - Empty "Item" elements are represented by empty JSON property entries in the "Value" array
- The "uri" attribute of the "BulkData" element maps to the "BulkDataURI" string
- The "InlineBinary" element maps to the "InlineBinary" string

Table F.3.1-1. XML to JSON Mapping

DICOM PS3.19 XML	DICOM JSON Model

<NativeDicomModel>
<DicomAttribute tag= **ggggee01** … />
<DicomAttribute tag= **ggggee02** … />
…
</NativeDicomModel>
	{
	ggggee01 : { … },
	ggggee02 : { … },
	…
	}

<DicomAttribute
tag= **ggggeeee**
vr= **VR** >
<Value number="1"> **Value** </Value>
</DicomAttribute>
	ggggeeee : {
	"vr": **VR** ,
	"Value": [**Value**]
	}

<DicomAttribute tag= **ggggeeee** … >
<Value number="1"> **Value1** </Value>
<Value number="2"> **Value2** </Value>
…
</DicomAttribute>
	ggggeeee : {
	…
	"Value": [**Value1** ,
	Value2 , …
	]
	}

DICOM PS3.19 XML	DICOM JSON Model

<DicomAttribute tag= **ggggeeee** … >
</DicomAttribute>
	ggggeeee : {
	…
	}

DICOM PS3.19 XML	DICOM JSON Model
<DicomAttribute tag= **ggggeeee** vr="PN" … >	**ggggeeee** : {
<PersonName number="1">	…
<Alphabetic>	"vr": "PN",
<FamilyName> **SB1**	"Value": [
</FamilyName>	{
<GivenName> **SB2**	**"** Alphabetic **"** : **"SB1^SB2^SB3^SB4^SB5",**
</GivenName>	"Ideographic": **"ID1^ID2^ID3^ID4^ID5"** ,
<MiddleName> **SB3**	"Phonetic": **"PH1^PH2^PH3^PH4^PH5"**
</MiddleName>	},
<NamePrefix> **SB4**	{
</NamePrefix>	"Alphabetic":
<NameSuffix> **SB5**	**"** **SB6** **"**
</NameSuffix>	}
</Alphabetic>	]
<Ideographic>	}
<FamilyName> **ID1**	
</FamilyName>	
…	
</Ideographic>	
<Phonetic>	
<FamilyName> **PH1**	
</FamilyName>	
…	
</Phonetic>	
</PersonName>	
<PersonName number="2">	
<Alphabetic>	
<FamilyName> **SB6**	
</FamilyName>	
</Alphabetic>	
</PersonName>	
</DicomAttribute>	

DICOM PS3.19 XML	DICOM JSON Model

<DicomAttribute tag= **ggggeeee** vr="SQ" … >
<Item number="1">
<DicomAttribute tag= **ggggee01** … />
<DicomAttribute tag= **ggggee02** … />
…
</Item>
<Item number="2">
<DicomAttribute tag= **ggggee01** … />
<DicomAttribute tag= **ggggee02** … />
…
</Item>
<Item number="3">
</Item>
…
</DicomAttribute>
	ggggeeee : {
	…
	"vr": "SQ",
	"Value":
	[
	{
	ggggee01 : { … },
	ggggee02 : { … },
	…
	}
	{
	ggggee01 : { … },
	ggggee02 : { … },
	…
	}
	{ }
	…
	]
	}

<DicomAttribute tag= **ggggeeee** … >
<BulkData URI= **BulkDataURI** >
</DicomAttribute>
	ggggeeee : {
	…
	"BulkDataURI": **BulkDataURI**
	}

<DicomAttribute tag= **ggggeeee** … >
<InlineBinary> **Base64String** </InlineBinary>
</DicomAttribute>
	ggggeeee : {
	…
	"InlineBinary": " **Base64String"**
	}

<DicomAttribute tag= **gggg00ee** PrivateCreator= **PrivateCreator** … >
…
</DicomAttribute>
	ggggXXee : {
	…
	}

F.4 DICOM JSON Model Example

// The following example is a QIDO-RS SearchForStudies response consisting
// of two matching studies, corresponding to the example QIDO-RS request:

```
// GET http://qido.nema.org/studies?PatientID=12345&includefield=all&limit=2
[
  {  // Result 1
    "00080005": {
      "vr": "CS",
      "Value": [ "ISO_IR 192" ]
    },
    "00080020": {
      "vr": "DT",
      "Value": [ "20130409" ]
    },
    "00080030": {
      "vr": "TM",
      "Value": [ "131600.0000" ]
    },
    "00080050": {
      "vr": "SH",
      "Value": [ "11235813" ]
    },
    "00080056": {
      "vr": "CS",
      "Value": [ "ONLINE" ]
    },
    "00080061": {
      "vr": "CS",
      "Value": [
        "CT",
        "PET"
      ]
    },
    "00080090": {
      "vr": "PN",
      "Value": [
       {
         "Alphabetic": "^Bob^^Dr."
       }
      ]
    },
    "00081190": {
      "vr": "UR",
      "Value": [ "http://wado.nema.org/studies/
      1.2.392.200036.9116.2.2.2.1762893313.1029997326.945873" ]
    },
    "00090010": {
      "vr": "LO",
      "Value": [ "Vendor A" ]
    },
    "00091002": {
      "vr": "UN",
      "InlineBinary": [ "z0x9c8v7" ]
    },
    "00100010": {
      "vr": "PN",
      "Value": [
       {
         "Alphabetic": "Wang^XiaoDong",
         "Ideographic": "王^小東"
       }
      ]
    },
```

```
    "00100020": {
       "vr": "LO",
       "Value": [ "12345" ]
    },
    "00100021": {
       "vr": "LO",
       "Value": [ "Hospital A" ]
    },
    "00100030": {
       "vr": "DT",
       "Value": [ "19670701" ]
    },
    "00100040": {
       "vr": "CS",
       "Value": [ "M" ]
    },
    "00101002": {
       "vr": "SQ",
       "Value": [
          {
             "00100020": {
                "vr": "LO",
                "Value": [ "54321" ]
             },
             "00100021": {
                "vr": "LO",
                "Value": [ "Hospital B" ]
             }
          },
          {
             "00100020": {
                "vr": "LO",
                "Value": [ "24680" ]
             },
             "00100021": {
                "vr": "LO",
                "Value": [ "Hospital C" ]
             }
          }
       ]
    },
    "0020000D": {
       "vr": "UI",
       "Value": [ "1.2.392.200036.9116.2.2.2.1762893313.1029997326.945873" ]
    },
    "00200010": {
       "vr": "SH",
       "Value": [ "11235813" ]
    },
    "00201206": {
       "vr": "IS",
       "Value": [ 4 ]
    },
    "00201208": {
       "vr": "IS",
       "Value": [ 942 ]
    }
 },
 {  // Result 2
    "00080005": {
```

```
    "vr": "CS",
    "Value": [ "ISO_IR 192" ]
},
"00080020": {
    "vr": "DT",
    "Value": [ "20130309" ]
},
"00080030": {
    "vr": "TM",
    "Value": [ "111900.0000" ]
},
"00080050": {
    "vr": "SH",
    "Value": [ "11235821" ]
},
"00080056": {
    "vr": "CS",
    "Value": [ "ONLINE" ]
},
"00080061": {
    "vr": "CS",
    "Value": [
        "CT",
        "PET"
    ]
},
"00080090": {
    "vr": "PN",
    "Value": [
      {
        "Alphabetic": "^Bob^^Dr."
      }
    ]
},
"00081190": {
    "vr": "UR",
    "Value": [ "http://wado.nema.org/studies/
    1.2.392.200036.9116.2.2.2.2162893313.1029997326.945876" ]
},
"00090010": {
    "vr": "LO",
    "Value": [ "Vendor A" ]
},
"00091002": {
    "vr": "UN",
    "InlineBinary": [ "z0x9c8v7" ]
},
"00100010": {
    "vr": "PN",
    "Value": [
      {
        "Alphabetic": "Wang^XiaoDong",
        "Ideographic": "王^小東"
      }
    ]
},
"00100020": {
    "vr": "LO",
    "Value": [ "12345" ]
},
```

```
    "00100021": {
      "vr": "LO",
      "Value": [ "Hospital A" ]
    },
    "00100030": {
      "vr": "DT",
      "Value": [ "19670701" ]
    },
    "00100040": {
      "vr": "CS",
      "Value": [ "M" ]
    },
    "00101002": {
      "vr": "SQ",
      "Value": [
        {
          "00100020": {
            "vr": "LO",
            "Value": [ "54321" ]
          },
          "00100021": {
            "vr": "LO",
            "Value": [ "Hospital B" ]
          }
        },
        {
          "00100020": {
            "vr": "LO",
            "Value": [ "24680" ]
          },
          "00100021": {
            "vr": "LO",
            "Value": [ "Hospital C" ]
          }
        }
      ]
    },
    "0020000D": {
      "vr": "UI",
      "Value": [ "1.2.392.200036.9116.2.2.2.2162893313.1029997326.945876" ]
    },
    "00200010": {
      "vr": "SH",
      "Value": [ "11235821" ]
    },
    "00201206": {
      "vr": "IS",
      "Value": [ 5 ]
    },
    "00201208": {
      "vr": "IS",
      "Value": [ 1123 ]
    }
  }
]
```


F.5 References

[RFC4627] (Normative JSON definition)

JSON. http://www.json.org/ (Informative)

Wikipedia, definition of JSON. http://en.wikipedia.org/wiki/JSON (Informative)

JSON in FHIR. http://www.hl7.org/implement/standards/fhir/formats.htm#json (Informative)

G WADL JSON Representation

G.1 Introduction

While the WADL specification only specifies an XML encoding for the WADL payload, the data structure can easily be represented using JSON. Additionally, conversion from XML to JSON and vice-versa can be done in a lossless manner.

G.2 XML Elements

The JSON encoding of WADL XML elements depends on whether the element is:

- a "doc" element
- an element that is unique within a particular parent element (e.g., "request")
- an element that can be repeated within a particular parent element (e.g., "param")

G.2.1 Doc Elements

A "doc" element is represented as an array of objects, where each object may contain:

- a "@xml:lang" string
- a "@title" string
- a "value" string

Example:

```
"doc": [
 {
  "@xml:lang": "en",
  "value": "Granular cell tumor"
 },
 {
  "@xml:lang": "ja",
  "value": "顆粒細胞腫"
 },
 {
  "@xml:lang": "fr",
  "value": "Tumeur à cellules granuleuses"
 }
]
```

G.2.2 Unique Elements

All unique WADL XML elements are represented as an object whose name is the name of the XML element and where each member may contain:

- a "@{attribute}" string for each XML attribute of the name {attribute}
- a child object for each child element that must be unique
- a child array for each child element that may not be unique

Example:

```
"request": {
  "param": [ ... ],
```

```
  "representation": [ ... ]
}
```

G.2.3 Repeatable Elements

All repeatable WADL XML elements are represented as an array of objects whose name is the name of the XML element and where each may contain:

- a "@{attribute}" string for each XML attribute of the name {attribute}
- a child object for each child element that must be unique
- a child array for each child element that may not be unique

Example:

```
"param": [
 {
  "@name": "Accept",
  "@style": "header"
 },
 {
  "@name": "Cache-control",
  "@style": "header"
 }
]
```