Digital Imaging and Communications in Medicine (DICOM)

Supplement 183: PS3.18 Web Services Re-Documentation

DICOM Standards Committee, Working Group 27: Web Technologies

1300 N. 17th Street Suite 1900

Rosslyn, Virginia 22209 USA

VERSION: Draft Letter Ballot

Developed in accordance with work item 2014-04-A.

Editorial content – to be removed before Letter Ballot Text

Closed Issues

#	Closed Issues
1	Can any transaction or transactions with Query Parameters define new Query Parameters or new values for existing Query Parameters? **Decision**: **Yes**. Any transaction may define new or extend existing Query Parameters. In addition, all origin servers shall ignore any parameters or values it does not support. See Section 8.3. This statement is made only once. It has been removed from transactions that may have stated it.
2	Should we define our own "reason phrases"? **Decision**: **No,** the Standard will not define new Reason Phrases, but an implementation is free to supply its own reason phrases in responses.
3	Should Section 7.7.6 on Caching Header Fields be included? **Decision**: **Yes**, it is informative for people not aware of them.
4	For Query Parameters Names that the origin server doesn't recognize, should it just ignore them and their values? **Decision**: **Yes**, it should ignore both the Query Parameter name and its values and process the request as if they were not there.
5	For Query Parameters Values that the origin server does not recognize, should it just ignore them, or should it return an error response? But what does invalid mean? Does it need to be defined for all Query Parameters. **Decision**: An origin server should ignore valid Query Parameter values it does not recognize and should process the request as if the parameter were not present.
6	Should the origin server be able to define and support additional Query Parameters or additional Query Parameter values for an existing Query Parameter? **Decision: Yes**. 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.
7	Should Conditional Requests and their Header Fields be included? See [RFC7232]. **Decision**: **No**. If needed this can be done in a CP.
8	Should all the applicable normative requirements in PS3.7 for the DIMSE services be replicated in this supplement as normative requirements for the RESTful Services. Our goal is to avoid asking people to read PS3.7. **Decision**: The requirements should be the same, unless the HTTP/S Standard requires a difference
9	Should we define a JSON canonical form to ensure interoperability. **Decision**: **No**, this should be done in a CP if needed.
10	Defines terms: for the current terms service, transaction Do we define Web Services? No. Do we define 'DICOM Web Service' or 'DICOMweb Service'? 'DICOM Web Service. Do we capitalize all http terms? No.
11	What resources should the user agent have to support for the RS Retrieve transaction? **Decision**: User agent does not have support anything. We are not covering conformance statement in this supplement.

Potential CPs after Supplement 183 is Final Text

CP Name	Sections	Potential CP Description
Status Codes	X.Y.3.1	200 with partial 201 should be used whenever a new resource is created 204 should be used for success without a payload. 206 should only be used with Range Requests. Remove and replace with appropriate status code or define a Range request/response for search.
	11.4.3.1	409 is incorrect (which should be a conflict with the *state* of the resource). Not sure of the correct code. 412 or 422? Precondition failed (the precondition of it not already existing)? Elliot Silver
	11.5.1	Add Cache-Control "no-cache"
	11.6.2	400 should be 412 or 422.

	11.7.3.3	If the target Workitem was already in the requested state, the response shall have a 304 status and no payload.
	11.9.3.1	Remove status code 206 (206 should only be used with Range Requests) from Table 11.9.3-1 and replace with 200?
	11.10.3.1	Fix
	all	Remove 409 throughout except when real state conflicts might exist.
Header Fields	8.4	Define origin server and user agent permissions and requirements with respect to header fields.
	11.10.3-2	Fix
Media Type	8.7.8	Improve specification of Selected Media Type and Selected Transfer Syntax
		Add Conditional Requests and their header fields
		Add Range Requests and their header fields
Notification	8.10.4.1	Explain how to establish a WebSocket. There are various methods.
		Allow Open WebSocket Connection to have a media type using the Accept/Content-Type headers in the request/response.
		Add Close Notification Connection
Security	8.11	Consider doing a CP to improve Security and Privacy section.
Worklist	11.2	Why should a dashboard client be required to do creates and updates etc. And the various other cases described in UPS – KOD
		Do we need a requirement to support the three transactions listed in 8.9 Common RESTful transactions?
	11.11	Add 'For a Worklist or a Filtered Worklist resource, the origin server will no longer create new subscriptions to Workitems for the Subscriber and will no longer send Event Reports to the Subscriber related to any Workitems. This transaction corresponds to the UPS DIMSE N-ACTION operation "Suspend Global Subscription".'
Status Report	11.6.3.2	Warning 299 - Isn't that entirely included in a 409 response? What is the added information here?
	11.7.3.2	Everywhere that you've indicated a specific warning message, etc. how should this be handled internationally? Are the messages always in 'en-us', or should they be translated?
		Everywhere that you've indicated a specific warning message, etc. how should this be handled internationally? Are the messages always in 'en-us', or should they be translated?
	11.8.2	Everywhere a specific warning message is indicated, how should this be handled internationally? Are the messages always in 'en-us', or should they be translated?
	11.10.3-2	Fix warnings
		Describe how Request UPS Cancel Action Information codes, etc. are encoded in the payload.
Other	11.8.2	Look at code meaning in PS.3.3 or 3.4. Is that one atomic action, or two? Can another client find the work item in InProgress?
	Annex	Update Table H-1 to include Non-Patient Objects
	11.12	Could write this as a profiling of the Send Event Report Transaction. "Shall implement the Send Event Report Transaction with the following additional requirements." Or some such KOD

---- End of Editorial content – to be removed before Final Text ----

Scope and Field of Application of this Supplement

This supplement re-documents PS3.18 Web Services.

35 The goals of this re-documentation are:

- Factor out text that is common to multiple services and in doing so 1) ensure uniformity and 2) make clearer and concise for readers.
- Use a uniform format and style for documenting DICOM web services, making it easier to navigate and more efficient for readers implementing multiple services
- Bring the Standard into conformance with current Web Standards, especially [RFC7230 – 7234], and [RFC3986 – 3987].
- Use the Augmented Backus-Naur Form (ABNF) defined in [RFC5234] and [RFC7405] to specify the syntax of request and response messages.
- Use consistent terminology throughout the Standard.
- Use a consistent format for documenting services and transactions.

The most important aspect of the re-documentation is that technical requirements of PS3.18 should not be changed. Errors, ambiguities, and underspecified aspects of the current PS3.18 have been corrected through the CP process prior to the finalization of this supplement.

PS3.18

Web Services

PS3.18: DICOM PS3.18 2015c - Web Services

Table of Contents

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 purposes or needs. NEMA does not undertake to guarantee the performance of any individual manufacturer or seller's products or services by 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 3].

PS3.1 should be used as the base reference for the current Parts of this Standard.

1 Scope

PS3.18 specifies web services (using the HTTP family of protocols) for managing and distributing DICOM (Digital Imaging and Communications in Medicine) Information Objects, such as medical images, annotations, reports, etc. to healthcare organizations, providers, and patients. The term DICOMweb is used to designate the RESTful Web Services described here.

Security considerations, including access control, authorization, and auditing are beyond the scope of PS3.18. Refer to PS3.15 Security and System Management Profiles.

2 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, IEC, and IETF maintain registries of currently valid International Standards.

HL7 CDA	Health Level Seven, Clinical Document Architecture (CDA)
IEC 61966-2.1	IEC. 1999. *Multimedia systems and equipment - colour measurement and management - Part 2.1: colour management - Default RGB colour space – sRGB. ISBN: 2-8318-4989-6 - ICS codes: 33.160.60, 37.080 - TC 100 - 51 pp. as amended by Amendment A1:2003*. https://en.wikipedia.org/wiki/RGB_color_space
IETF RFC822	Standard for ARPA Internet Text Messages http://tools.ietf.org/html/rfc822
IETF RFC2045	and followings MIME Multipurpose Internet Mail Extension http://tools.ietf.org/html/rfc2045
IETF RFC2046	Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types https://tools.ietf.org/html/rfc2046
IETF RFC2387	The MIME Multipart/Related Content-type https://tools.ietf.org/html/rfc2387
IETF RFC2978	IANA Charset Registration Procedures https://tools.ietf.org/html/rfc2978
IETF RFC3240	Application/dicom MIME Subtype Registration http://tools.ietf.org/html/rfc3240
IETF RFC3536	Terminology Used in Internationalization in the IETF https://tools.ietf.org/html/rfc6365#section-3.3
IETF RFC3986	Uniform Resource Identifiers (URI): Generic Syntax http://tools.ietf.org/html/rfc3986
IETF RFC4648	The Base16, Base32, and Base64 Data Encodings. https://tools.ietf.org/html/rfc4648
IETF RFC5234	Augmented BNF for Syntax Specifications: ABNF http://tools.ietf.org/html/rfc5234
IETF RFC6338	Media Type Specifications and Registration https://tools.ietf.org/html/rfc6838
IETF RFC6365	Terminology Used in Internationalization in the IETF https://tools.ietf.org/html/rfc6365
IETF RFC6455	The WebSocket Protocol http://tools.ietf.org/html/rfc6455
IETF RFC6365	Terminology Used in Internationalization in the IETF https://tools.ietf.org/html/rfc6365
IETF RFC6570	URI Template http://tools.ietf.org/html/rfc6570
IETF RFC6838	Media Type Specifications and Registration Procedures <https://tools.ietf.org/html/rfc6838>
IETF RFC7159	The JavaScript Object Notation (JSON) Data Interchange Format. https://tools.ietf.org/html/rfc7159
IETF RFC7230	Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing http://tools.ietf.org/html/rfc7230
IETF RFC7231	Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content http://tools.ietf.org/html/rfc7231
IETF RFC7232	Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests

	http://tools.ietf.org/html/rfc7232
IETF RFC7233	Hypertext Transfer Protocol (HTTP/1.1): Range Requests http://tools.ietf.org/html/rfc7233
IETF RFC7234	Hypertext Transfer Protocol (HTTP/1.1): Caching http://tools.ietf.org/html/rfc7234
IETF RFC7235	Hypertext Transfer Protocol (HTTP/1.1): Authentication http://tools.ietf.org/html/rfc7235
IETF RFC7236	Initial Hypertext Transfer Protocol (HTTP) Authentication Scheme Registrations http://tools.ietf.org/html/rfc7236
IETF RFC7237	Initial Hypertext Transfer Protocol (HTTP) Method Registrations http://tools.ietf.org/html/rfc7237
IETF RFC7303	XML Media Type
IETF-RFC7405	Case-Sensitive String Support in ABNF https://tools.ietf.org/html/rfc7405
IHE ITI TF-2x: Appendix V	IHE IT Infrastructure Technical Framework, Volume 2x, Appendix V (Web Services for IHE Transactions)

ISO/IEC Directives, Part 3 ISO/IEC. 1989. *Drafting and presentation of International Standards*.

ISO/IEC 10918	JPEG Standard for digital compression and encoding of continuous-tone still images
ISO/IEC 2022:1994	Information technology -- Character code structure and extension http://www.ecma-international.org/publications/standards/Ecma-035.htm
SUBM-wadl-20090831	Web Application Description Language (WADL), W3C Member Submission 31 August 2009 http://www.w3.org/Submission/wadl/)

3 Definitions

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

3.1 DICOM Conformance

PS3.18 makes use of the following terms defined in PS3.2:

Conformance Statement

3.2 DICOM Information Object

This PS3.18 makes use of the following terms defined in PS3.3:

Multi-frame image

3.3 DICOM Service Class Specifications

This PS3.18 makes use of the following terms defined in PS3.4:

Real-World Activity

Service-Object Pair (SOP) Class

Service-Object Pair (SOP) Instance

3.4 DICOM Data Structures and Encoding

PS3.18 makes use of the following terms defined in PS3.5:

Data Element

Data Element Tag

Unique Identifier (UID)

3.5 DICOM Message Exchange

PS3.18 makes use of the following terms defined in PS3.7:

DICOM Message Service Element (DIMSE)

3.6 HyperText Transfer Protocol (HTTP/HTTPS)

PS3.18 makes use of the following terms defined in IETF RFC 7230 Section 2.1 Client/Server Messaging:

HTTP

HTTPS

origin server

user agent

3.7 DICOM Web Services

PS3.18 defines the following terms.

Bulkdata

An object that contains an octet-stream containing one or more Value Fields (typically containing large data, such as Pixel Data) extracted from a DICOM Dataset. See Metadata.

Notes

1. The octet-stream does not include the Attribute Tag, Value Representation, or Attribute Length.

2. 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.

Bulkdata URI

A Uniform Resource Identifier that references Bulkdata.

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 typically a Composite Service Object Pair (SOP) Instance.

DICOM Resource

One or more DICOM Objects that are referenced by a URI.

Event Report

A Dataset containing elements describing an event that occurred on the origin server. See Section 11.12.

Metadata

A DICOM Dataset where zero or more elements (typically containing large data, such as Pixel Data) have been replaced with Bulkdata URIs.

DIMSE Proxy

An origin server that responds to DICOM Web Service requests by executing DIMSE transactions to a backend server.

RESTful Web Service

A web service is RESTful if it is implemented using the REST architecture and principles. See https://en.wikipedia.org/wiki/Representational_state_transfer.

Service

When used in PS3.18 the term Service means a set of transactions and resources to which those transactions apply.

Status Report

A Status Report is information contained in a response payload describing warnings or errors related to a request.

Subscriber

The creator or owner of a Subscription, typically a user agent.

sRGB

A standard RGB color space <https://en.wikipedia.org/wiki/RGB_color_space> defined in [IEC 61966-2.1].

Target URI

The URI contained in a request message. It designates the resource that is the target of the request.

Transaction

When used in PS3.18 the term Transaction means an HTTP/HTTPS request/response message pair.

UTF-8

The Unicode Standard, Version 11.0, Section D95 (2018 June 5)

4 Symbols and Abbreviated Terms

ABNF	Augmented Backus-Naur Form. See [RFC5234] and [RFC7405].
DICOM	Digital Imaging and Communications in Medicine
HL7	Health Level Seven
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTP/1.1	Version 1.1 of the HyperText Transfer Protocol
HTTP/2	Version 2 of the HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
HTTPS/1.1	Version 1.1 of the HyperText Transfer Protocol
HTTPS/2	Version 2 of the HyperText Transfer Protocol
IHE	Integrating the Healthcare Enterprise
REST, RESTful	REpresentational State Transfer, a web services architecture. A service implemented in this architecture is described as RESTful.
RS	The RESTful web services API defined in PS3.18
SOP	Service Object Pair
UID	Unique (DICOM) Identifier
URI	A Uniform Resource Identifier (see RFC3986).
WADL	Web Application Description Language
XML	eXtensible Markup Language

5 Conventions

This section defines conventions used throughout the rest of PS3.18.

5.1 URIs versus URLs

Throughout this Part the term URI is used exclusively as is recommended in [RFC3986] Section 3.1.1 <https://tools.ietf.org/html/rfc3986#section-1.1.3>. The term URL is not used.

5.2 Message Syntax

The syntax of the request and response messages for transactions are defined using the ABNF Grammar used in [RFC7230], which is based on the ABNF defined in [RFC5234]. PS3.18 also uses the ABNF extensions in [RFC7405], which defines '%s' prefix for denoting case sensitive strings.

The syntax rules defined herein are valid for the US-ASCII character set or character sets that are supersets of US-ASCII, e.g., Unicode UTF-8.

In the ABNF used to define the syntax of messages, the following conventions are used:

1. Syntactic variables are lowercase.
2. Terminal rules are uppercase. For example, 'SP' stands for the US-ASCII space (0x20) literal character, and 'CRLF' stands for the ASCII carriage return (0xD) and line feed (0xA) literal characters.
3. Header Field names are capitalized and quotation marks that denote literal strings for header field names are omitted. The Header Field names are the only capitalized names used in the grammar. See [RFC7231] Section 1.2 <https://tools.ietf.org/html/rfc7231#section-1.2 >. For example:

```
Accept: media-type CRLF
```

is equivalent to

```
"Accept:" media-type CRLF
```

In PS3.18, as with HTTP in general, resources are identified by URIs [RFC3986]. Each service defines the resources it manages, and the URI Templates used to define the structure of the URIs that reference them.

In HTTP RFCs, ABNF rules for `obs-text` and `obs-fold` denote "obsolete" grammar rules that appear for historical reasons. These rules are not used in DICOM Web Services syntax definitions.

See Annex A for the Combined ABNF for DICOM Web Services.

5.2.1 Common Syntactic Rules for Data Types

Table 5.2-1 defines the syntax of some common rules used in defining data values in PS3.18.

Table 5.2-1. ABNF for Common Syntactic Values

Name	Rule
int	= [+ / -] 1*DIGIT ; An integer
uint	= 1*DIGIT ; An unsigned integer
pos-int	= NON-ZERO-DIGIT *DIGIT ; An integer greater than zero
decimal	= int ["." uint] [("E" / "e") int] ; a fixed or floating point number with at most 16 characters
string	= %s 1*QCHAR ; A case sensitive string
base64	; Use base64URI defined in [RFC4648] <https://tools.ietf.org/html/rfc4648#section-5>
uid	= uid-root 1*("." uid-part)
uid-root	= "0" / "1" / "2"
uid-part	= "0" / pos-int

5.2.2 URI Templates

The URI Template [RFC6570] syntax has been extended to allow case sensitive variable names. This has been done by modifying the varchar production (see [RFC6570] Section 2.3) as follows:

```
varchar   = %x20-21 / %x23-7E / pct-encoded
```

5.2.3 List Rule ('#')

The ABNF has been extended with the List Rule, which is used to define comma-separated lists. It does not allow empty lists, empty list elements, or the legacy list rules defined in [RFC7230] Section 7<https://tools.ietf.org/html/rfc7230#section7>.

```
1#element      = element *(OWS "," OWS element)
#element       = 1#element
<n>#<m>element = element <n-1>*<m-1> (OWS "," OWS element)
```

Where

```
n >= 1 and m > n
```

5.3 Web Service Section Structure

TODO: Review this section after public comment and Update

PS3.18 is organized so that new Services may be appended as new numbered sections at the end of the document.

Each Web Service defined in PS3.18 is described using the structure shown in Annex I. This structure is intended to facilitate consistency/uniformity in the documentation and ensure all the important details are considered. Note that this structure allows new transactions to be added to a Service later.

5.4 Request and Response Header Field Tables

Request header field requirements are described using tables of the following form:

Table 5.4-1. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	

The Name column contains the name of the HTTP header field as defined in [RFC7230, RFC7231].

The Value column defines either the value type or the specific value contained in the header field.

The Usage User Agent column defines requirements for the user agent to supply the header field in the request.

The Usage Origin Server column defines requirements for the origin server to support the header field.

The content of the Usage columns is either:

- M Mandatory
- C Conditional
- O Optional

The Description column of conditional request header fields specifies the condition for the presence of the header field.

- "Shall be present if <condition>" means that if the <condition> is true, then the header field shall be present; otherwise, it shall not be present.
- "May be present otherwise" is added to the description if the header field may be present, even if the condition is not true

Response header field requirements are described using tables of the following form:

Table 5.4-2. Response Header Fields

Name	Value	Origin Server Usage	Description

65 For response header fields the Usage column defines requirements for the origin server to supply the header field.

6 Conformance

An implementation claiming conformance to PS3.18 shall function in accordance with all its mandatory sections.

DICOM Web Services are used to transmit Composite SOP Instances. All Composite SOP Instances transmitted shall conform to the requirements specified in other Parts of the Standard.

An implementation may conform to the DICOM Web Services by supporting the role of origin server or user agent, or both, for any of the Services defined in PS3.18. The structure of Conformance Statements is specified in PS3.2.

An implementation shall describe in its Conformance Statement the Real-World Activity associated with its use of DICOM Web Services, including any proxy functionality between a Web Service and the equivalent DIMSE Service.

An implementation shall describe in its Conformance Statement the security mechanisms utilized by the implementation.

7 Overview of DICOM Web Services (Informative)

7.1 DICOM Web Service Types

PS3.18 defines DICOM Web Services. Each service allows a user agent to interact with an origin server in order to manage a set of DICOM Resources. Each DICOM Web Service operates on a set of resources and defines a set of Transactions that operate on those resources. All Transactions are defined in terms of HTTP request/response message pairs.

When used in PS3.18 the term HTTP refers to the family of HTTP protocols including: HTTP/1.1, HTTPS/1.1, HTTP/2, and HTTPS/2, as defined by the relevant IETF RFCs, but does not include HTTP/1.0 or HTTPS/1.0. The HTTP standards are normative for all aspects of HTTP message format and transmission.

There are two general types of DICOM Web Services: URI and RESTful. This distinction is based on the type of web service protocol used to specify resources and transactions.

7.1.1 URI Web Service

The URI Web Service retrieves representations of its resources, those resources being Composite SOP Instances. The URI service defines two transactions that retrieve SOP Instances in different media types. All URI transactions use the query component of the URI in the request message to specify the transaction.

The functionality of the URI Web Service Transactions is like, but more limited than, the Retrieve Transaction of the Studies Web Service.

7.1.2 RESTful Web Services and Resources

Each RESTful Web Service defines the set of resources, and the transactions that can be applied to those resources.

The defined RESTful Web Services are:

Studies Web Service

Enables a user agent to manage DICOM Studies stored on an origin server.

Worklist Web Service

Enables a user agent to manage a Worklist containing Workitems stored on an origin server.

Non-Patient Instance Web Service

Enables a user agent to manage Non-Patient Instances, e.g., Color Palettes, stored on an origin server.

7.2 Resources, Representations, and Target URIs

In RESTful Web Services (https://en.wikipedia.org/wiki/Representational_state_transfer), a resource is an abstract object with a type, associated data, relationships to other resources, and a set of methods that operate on it. Resources are grouped into collections. Collections are themselves resources as well. Each collection is unordered and contains only one type of resource. Collections can exist globally, at the top level of an API, but can also be contained inside a resource. In the latter case, we refer to these collections as sub-collections. Sub-collections usually express some kind of “contained in” relationship.

7.2.1 DICOM RESTful Resources

The DICOM Resources defined in PS3.18 are typically either a DICOM Web Services or DICOM Information Objects). Examples include Studies, Series, Instances, Worklists, and Workitems.

DICOM Resources are grouped into collections and hierarchies. The following resources are examples collections:

Resource Path	Contents
/studies	A collection of Studies
/series	A collection of Series
/instance	A collection of Instance
/frames	

The following resources are examples of hierarchies:

/studies/{study}/series	- contains a collection of Series
/studies/{study}/series/{series}/instances	- contains a collection of Instances
/studies/{study}/series/{series}/instances/{instance}/frames	- contains a sequence of frames.

A DICOM Web Service origin server manages a collection of resources. This might not be done directly, for example, an origin server could act as a proxy, converting RESTful requests into DIMSE requests, and DIMSE responses into RESTful responses.

Resources are typically created and/or accessed by user agents.

7.2.2 Representations

A resource is an abstract concept that is reified by a representation. which is a data object encoded in an octet-stream. For example, a DICOM Study (abstract) might be represented by an encoding it in an application/dicom or application/dicom+json media type.

A media type describes the format or encoding of a representation. Examples of media types are application/dicom, application/dicom+json, image/jpeg, and text/html.

7.2.3 Target URIs

Resources are identified by URIs. Each service defines the resources that it manages and the format of the URIs used to reference those resources. The format of URIs is defined using URI Templates. See [RFC6570].

8 Common Aspects of DICOM Web Services

This section describes details and requirements that are common to all Web Services defined in PS3.18.

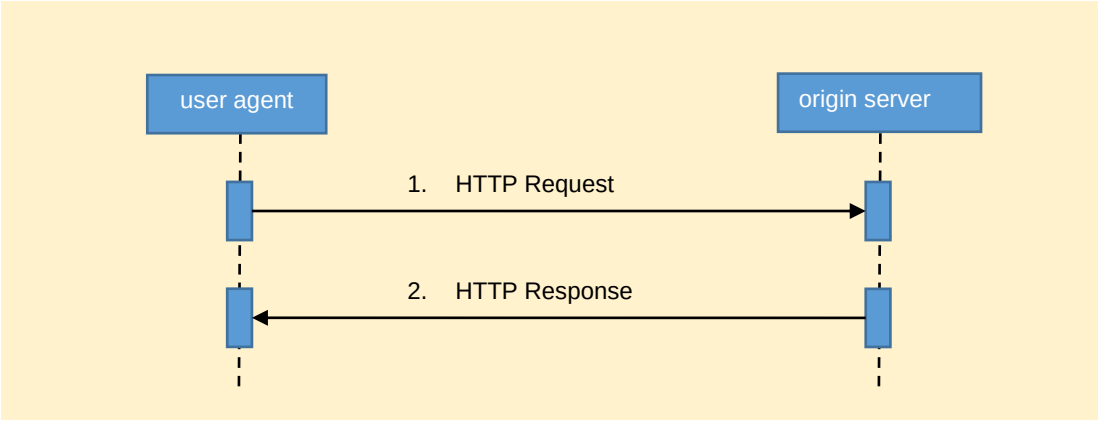
8.1 Transactions

Each transaction is composed of a request message and a response message, sometimes referred to as a request/response pair. When used in PS3.18 the term "request" means "request message", and "response" means "response message", unless clearly stated otherwise. Figure 7.3-1 is an interaction diagram that shows the message flow of a transaction. When it receives the request, the origin server processes it and returns a response.

The request includes a method, the URI of the Target Resource, and header fields. It might also include Query Parameters and a payload.

The response includes a status code, a reason phrase, header fields, and might also include a payload.

Figure 7.3-1. Interaction Diagram for Transactions



8.1.1 Request Message Syntax

PS3.18 uses the ABNF defined in Section 5.2 to define the syntax of transactions.

All Web Services API request messages have the following syntax:

```
method SP target-uri SP version CRLF
*(header-field CRLF)
CRLF
[payload]
```

Where,

`method = "CONNECT" / "DELETE" / "GET" / "HEAD" / "OPTIONS" / "POST" / "PUT"`

Each transaction defines the method it uses.

`SP= %x20`

The US-ASCII Space character

`target-uri = "/" {/resource} {?parameters*}`

Each transaction defines a URI Template for the Target Resource. The template specifies the format of URIs that reference the Target Resource(s) of a request. See Section 8.1.1.2.

`version = ("HTTP" / "HTTPS") "/" ("1.1" / "2")`

The version of the HTTP protocol; one of "HTTP/1.1", "HTTP/2", "HTTPS/1.1", or "HTTPS/2".

`CRLF = %x0D.0A`

A US-ASCII carriage return (%x0D) followed by a linefeed (%x0A).

`*(header-field CRLF)`

Zero or more header fields each followed by a CRLF delimiter.

`[payload] = *OCTET / multipart-payload`

An optional payload containing zero or more 8-bit OCTETs.

Note:

The method, SP, version, CRLF, Accept, header-field, and payload are all HTTP productions from [RFC7230], and [RFC7231]. The definitions are reproduced here for convenience.

8.1.1.1 Method

The request method is one of the HTTP methods, such as CONNECT, DELETE, GET, HEAD, OPTIONS, POST, PUT. See [RFC7230] Section 4<https://tools.ietf.org/html/rfc7231#section-4>.

8.1.1.2 Target Resource

The Target Resource of a request is specified within a Target URI contained in the request message. URI Templates are used to specify the format of the Target URI.

The most general template for a target URI is:

```
target-uri = "/" {/resource} {?parameter}
```

The Target Resources of the RESTful Services typically correspond to DICOM Information Entities. The general format of such resource URI Templates is:

/resource-type/{resource-id} (/resource-subtype/{sub-resource-id})* {?parameter*}

Where,

resource-type	is a literal string dependent on the service, for example "studies", and
{resource-id}	is a variable, typically a UID, that identifies a specific resource
resource-subtype	is a literal string dependent on the parent resource, for example "series"
{sub-resource-id}	is a variable, typically a UID, that identifies a specific sub-resource
{?parameters*}	is a URI Template for zero or more Query Parameters

8.1.1.3 Query Parameters

Query parameters are contained in the query component (see RFC3986) of the URI. The user agent may use Query Parameters to supply parameters to the request. See Section 8.3.

8.1.1.4 Request Header Fields

Request header fields are used to specify metadata for the request. Most requests have one or more Content Negotiation (see Section 8.4.2) header fields. If a request has a payload, the request will have the corresponding Content Representation (see Section 8.4.3) and Payload (see Section 8.4.4) header fields.

8.1.1.5 Request Payload

The payload of the request is an octet-stream containing the content of the message. See Section 8.6. The presence of a payload in a request is signaled by a Content-Length or Content-Encoding header field.

8.1.2 Response Message Syntax

The syntax of a response message is:

```
version SP status-code SP reason-phrase CRLF
*(header-field CRLF)
CRLF
[payload]
```

Where

`status-code = 3DIGIT`

A three-digit code specifying the status of the response.

`reason-phrase = *(HTAB / SP / VCHAR)`

A human readable phrase that corresponds to the status. An implementation may define its own reason phrases. The reason-phrase syntax is slightly modified from that in [RFC7230]; PS3.18 does not allow obsolete text (obs-text) in the reason-phrase.

Note:

The status-code production is from [RFC7230].

The origin server shall always return a response message.

8.1.2.1 Status Codes

The response message shall always include a valid 3-digit status code. Section 8.5 defines the status codes used by transactions. IANA maintains a registry of HTTP Status codes. See http://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml.

8.1.2.2 Response Header Fields

Response header fields are used to specify metadata for the response. If the response will have the corresponding Content Representation (see Section 8.4.3) and Payload (see Section 8.4.4) header fields.

8.1.2.3 Response Payload

The payload of the response is an octet-stream containing one or more representations. See Section 8.6.

A transaction typically defines two types of payloads for a response message: a success payload, and a failure payload.

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

8.2 Target Resources

Transaction specifications define what resource types are valid Target Resources for the transaction and define the format of the URI for the Target Resource (and Query Parameters) using URI Templates. The URI of a Target Resource is referred to as the Target URI. Transaction specifications also define what resource types are valid resources for the response.

A Target URI is composed of three components: The Base URI, the Target Resource Path, and Query Parameters (which are often optional).

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.

The most general template for a Target URI is:

```
target-uri = "/" {/resource} {?optional*}
```

or if any of the Query Parameters are required

```
target-uri = "/" {/resource} ?{required*}{&optional*}
```

Where

`"/"`	The slash character ('/') is used to designate the Base URI.
`{/resource}`	A URI template for the Target Resource Path, a relative path component that references the Target Resource. The '/' in the template indicates that reserved characters, such as '/', can be used in the template expansion. See [RFC6570]. "/{/resource}" indicates the absolute URI to the Target Resource on the origin server.
`{required*}`	A URI Template for one or more required query parameters. See 8.1.1 for an example.
`{&optional*)`	A URI Template for zero or more optional query parameters. See 8.3.1.4 for an example.

The Base URI of a Service is an absolute URI that specifies the location of the origin server implementing the Service. Each target-uri defined by PS3.18 starts with a "/", which is a shorthand that designates the Base URI of the Service. The Base URI may support more than one Service.

The Service Root Path is the Base URI without the Scheme and Authority components.

The Target Resource Path is a relative URI that specifies the path to the resource from the Base URI of the Service. It is specified by a URI Template that uses Path Expansion {/var} as defined in [RFC6570].

For example, given the URI:

```
http://dicom.nema.org/service/studies/2.25.123456789/series/2.25.987654321
```

The Base URI is:

```
http://dicom.nema.org/service
```

The Service Root Path is:

```
/service
```

The Target Resource Path is:

```
/studies/2.25.123456789/series/2.25.987654321
```

The URI Template for this resource is:

```
/studies/{study}/series/{series}
```

Where

`{study}` is the Study Instance UID of a Study
`{series}` is the Series Instance UID of a Series

8.3 Query Parameters

Query Parameters are specified in the query component of the URI (see [RFC3986] Section 3.4 <https://tools.ietf.org/html/rfc3986#section-3.4>).

The query component of a request URI may be used to specify one or more Query Parameters. These parameters are referred to as Query Parameters to distinguish them from header field parameters or other types of parameters that may be contained in the payload.

The Query Parameters are specified using a URI Template that uses Form {?var} and Query Continuation Style {&var} Query Expansion as defined in [RFC6570].

If a Target URI includes a "query component" (see [RFC3986] Section 3.4 <https://tools.ietf.org/html/rfc3986#section-3.4>), it shall contain Query Parameters that conform to the syntax defined here.

The Services and Transactions defined elsewhere in PS3.18 may further refine the qp-name and qp-value rules defined below.

[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 may return a response containing appropriate warning and/or error messages.

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

8.3.1 Query Parameter Syntax

Query parameters have the following syntax:

```
query-parameters = "?" parameter [*("&" parameter)]
```

Each parameter after the first, is separated from the following parameter by the "&" character. Each parameter has the following syntax:

```
parameter  = qp-name
           / qp-name "=" 1#qp-value
           / qp-name "=" 1#attribute
           / attribute
           / attribute "=" 1#qp-value
```

The qp-name is case sensitive, and starts with an alphabetic or underscore character, followed by zero or more alphanumeric or underscore "_" characters:

```
name = %s DQ 1*(ALPHA / "_") *(ALPHA / DIGIT / "_") DQ
```

A qp-name by itself (with no values) is a legal Query Parameter. A parameter <name> may also be followed by a comma-separated list of one or more qp-values, or one or more attributes.

Qp-values are case-sensitive. A qp-value is composed of qp-chars, where qp-char is the set of legal query component characters as defined by [RFC3986]<https://tools.ietf.org/html/rfc3986>, minus the equal ("="), ampersand ("&"), and comma (",") characters.

```
qp-value   = %s DQ 1*qp-char DQ
qp-char    = unreserved / pct-encoded / qp-special
qp-special = "!" / "$" / "'" / "(" / ")" / "*" / "+" / ";" /":" / "@" / "/" / "?"
```

The only visible US-ASCII characters disallowed in the query component by [RFC3986] are "#", "[", "]". PS3.18 further disallows "&", "=", and ",". However, the characters ("#", "[", "]" "&", "=", and ",".) may be included in qp-values if they are percent encoded.

There are two kinds of attribute simple and sequence:

```
attribute = simple-attribute / sequence-attribute
```

A simple-attribute is a single Data Element Tag or Keyword (see PS3.6 Table 6-1) that does not have a VR of SQ:

```
simple-attribute = keyword / tag
keyword          = %s DQ 1*ALPHA *(ALPHA / DIGIT) DQ
tag              = 8HEXDIG
```

DICOM keywords are case sensitive; they shall start with an alphabetic character followed by zero or more alphanumeric characters. See PS3.6.

A sequence-attribute is two or more attributes separated by the dot character ("."), all but the last attribute shall have a VR of SQ, and the last attribute shall not have a VR of SQ.

```
sequence-attribute = (keyword / tag) *("." attribute)
```

The following are examples of valid values for attribute:

```
0020000D
StudyInstanceUID
00101002.00100020
OtherPatientIDsSequence.PatientID
00101002.00100024.00400032
OtherPatientIDsSequence.IssuerOfPatientIDQualifiersSequence.UniversalEntityID
```

Some Query Parameters have a qp-name, which is an attribute, and a value that is a comma-separated list of one or more qp-values. The qp-values of an attribute parameter shall satisfy its Value Representation and Value Multiplicity, as defined in PS3.5 and PS3.6, of the corresponding attribute.

Unlike the Value Representations defined in PS3.5, Query Parameters:

- shall not be padded to an even length
- shall not contain any NULL (%x00) characters
- shall encode UIDs as specified in PS3.5, except that they shall not be padded to an even length

8.3.1.1 Query Parameter Syntax

The syntax and semantics of valid qp-names, qp-values and attributes are specified by the defining Service or Transaction; however, they shall conform to the rules in this Section.

Table 8.3.1-1 contains the collected syntax of Query Parameters. The Services and Transactions defined elsewhere in PS3.18 may further refine the qp-name, attribute, and qp-value rules.

All qp-names are case sensitive.

Table 8.3.1-1. ABNF for Query Parameter

Name	Rule
query-parameters	= "?" parameter *("&" parameter)
parameter	= qp-name ; a name only / qp-name "=" 1#qp-value ; a name with one or more values / qp-name "=" 1#attribute ; a name with one or more attributes / attribute ; an attribute only / attribute "=" 1#qp-value ; an attribute with one or more values
qp-name	= %s (ALPHA / "_") *(ALPHA / DIGIT / "_")
qp-value	= int / uint / pos-int / decimal / float /string / base64 / uid / %s 1*qp-char / %s DQ 1*qp-special DQ ; See Section 5.2.1
qp-char	= unreserved / pct-encoded
qp-special	= "!" / "$" / "'" / "(" / ")" / "*" / "+" / ";" /":" / "@" / "/" / "?"
simple-attribute	= keyword / tag
sequence-attribute	= keyword *("." attribute) / tag *("." attribute)
keyword	= %s uppercase *(ALPHA / DIGIT)
uppercase	= %x41-5A
tag	= 8HEXDIG

Note

The syntax of qp-values is defined in Section 5.2.1.

The qp-char (Query Parameter characters) rule defined above is the query rule of [RFC3986], which defines the legal characters for the query component, minus the equal sign ("="), comma (","), and ampersand ("&"). So, the qp-char rule is the VCHAR rule minus "#", "[", "]", "=", "&", and ",".

All DICOM keywords are case sensitive. See PS3.6.

8.3.2 Query Parameter Usage

An implementation's support for Query Parameters is either Mandatory or Optional. Each Query Parameter section contains a table specifying Query Parameter keys, values, user agent usage requirement, and origin server usage requirements. Table 8.3.2-1 specifies the usage symbols, types, and definitions.

Table 8.3.2-1. Query Parameter Usage

Symbol	Type
M	Mandatory

C	Conditional
O	Optional

For example, Table 8.3.2-2 shows an example Query Parameter table.

Table 8.3.2-2. Example Query Parameter Table

		Usage		
Name	**Values**	**User Agent**	**Origin Server**	**Section**
requestType	"WADO"	M	M	9.1.2.1.1
studyUID	uid	M	M	9.1.2.1.3
seriesUID	uid	M	M	9.1.2.1.3
objectUID	uid	M	M	9.1.2.1.4

8.3.3 Content Negotiation Query Parameters

The parameters defined in this section are primarily designed for use in hyperlinks, i.e. URIs embedded in documents, where the Content Negotiation header fields (See Section 8.3.3) are not accessible.

8.3.3.1 Accept Query Parameter

The Accept Query Parameter is designed to be used in hyperlinks, which may be dereferenced and retrieved by an unknown user agent.

It shall be supported by the origin server. It is optional for the user agent.

The Accept Query Parameter has the following syntax:

```
accept        = accept-name "=" 1#(media-type [weight])
accept-name   = "accept"
```

The Accept Query Parameter has the same syntax as the Accept header field (see Section 8.4.3), except that it shall not have wildcards (<type>/* or */*). See Section 8.7.

Note

1. The normal name of this parameter is "accept"; however, the URI Service uses an accept-name of "contentType". See Section 9.1.2.2.1.
2. The "%s" that prefixes the accept-name specifies that it is a case sensitive token. See [RFC7405].

The parameter value is a comma-separated list of one or more media-types.

The Accept Query Parameter should not be used when the user agent can specify the values by using 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 */*.

If none of the Acceptable Media Types (see Section 8.7.5) are supported by the origin server, the origin server response shall be in the default media type for the Resource Category of the Target Resource(s). If there is no default media type defined for the Target Resource, the origin server response shall be 406 (Not Acceptable) and may include a payload containing an appropriate Status Report.

If a DICOM Media Type is present, non-DICOM Media Types shall not be present. If both DICOM and non-DICOM Media Types are present, the response shall be 400 (Bad Request), and may include a payload containing an appropriate Status Report.

8.3.3.2 Character Set Query Parameter

The character set Query Parameter is designed for use in hyperlinks, that is URIs embedded in documents, where the Accept-Charset header field is not accessible.

It shall be supported by the origin server. It is optional for the user agent.

The character set Query Parameter has the following syntax:

```
character-set = "charset" "=" 1#(charset [weight])
```

The character set Query Parameter value is a comma-separated list of one or more charsets. It is like the Accept-Charset header field, except that it shall not have wildcards.

Note

Charsets present in the character set Query Parameter typically have a corresponding character set in the Accept-Charset header field, either explicitly or implicitly through wildcards.

If this parameter has a value that is not a valid or supported character set, the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate Status Report. See Section 8.6.3.

8.3.4 Search Query Parameters

Table 8.3.4-1 contains the syntax for the names and values of search parameters, along with a reference to the section where their meaning is defined. Search transactions shall support these parameters. The ABNF for the various search parameters is:

Table 8.3.4-1. Query Parameter Syntax

Term	Value	Usage		Description
		User Agent	Origin Server	
search	= match / fuzzy / include / limit / offset			
match	; See attribute matching rules below	O	M	8.3.4.1
fuzzy	= "fuzzymatching" "=" true / false	O	M	8.3.4.2
includefield	= "includefield" "=" 1#attribute / "all"	O	M	8.3.4.3
limit	= "limit" "=" uint ; Maximum number of results	O	M	8.3.4.4
offset	= "offset" "=" uint ; Number of skipped results	O	M	8.3.4.4

The following sections describe these parameters in detail.

8.3.4.1 Attribute Matching

The syntax of the `match` Query Parameter shall be:

```
match          = normal-match / uid-list-match
normal-match   = 1*("&" attribute "=" value)
uid-list-match = 1*("&" attribute "=" 1#value)
attribute      = (attribute-id) *("." attribute-id)
attribute-id   = tag *("." tag) / keyword *("." keyword)
tag            = 8HEXDIG
keyword        = ; A keyword from PS3.6 Table 6-1.
```

Zero or more DICOM attribute/values pairs specify the matching criteria for the search.

Each search transaction defines which attributes are required or permitted.

Note

DICOM attributes should not be confused with XML attributes. The Tags and Keywords for DICOM attributes are defined in PS3.6, Table 6-1.

DICOM Attribute/Values pairs shall satisfy the following requirements:

1. Each attribute-id shall be a Data Element Tag or Keyword.
2. Each attribute in the Query Parameter shall be not be repeated
3. Each attribute in the Query Parameter shall have a single value, unless the associated DICOM Attribute allows UID List matching (see PS3.4 Section C.2.2.2.2), in which case the value is a comma-separated list of UIDs.
4. The acceptable values are determined by the types of matching allowed by C-FIND for its associated attribute. See PS3.4 Section C.2.2.2. All characters in values that are not qp-chars shall be percent-encoded. In particular, all non-ASCII characters shall be percent encoded. See [RFC3986] for details.

The following US-ASCII characters"#", "[", "]", "&", "=", and "," shall be percent encoded in any Query Parameter.

8.3.4.1.1 Matching Rules

The matching semantics for each attribute are determined by the types of matching allowed by C-FIND. See PS3.4, Section C.2.2.2 .

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

Combined date-time matching shall be performed as specified in PS3.4, Section C.2.2.2.5.

Note

If an origin server is acting as a proxy for a C-FIND SCP that does not support combined date-time matching, it shall perform a C-FIND request using only the date and filter any results that are outside the time range before returning a response.

If the Timezone Offset From UTC (0008,0201) attribute is specified in the request, dates and times in the request are to be interpreted in the specified time zone. See PS3.4 Section C.4.1.1.

8.3.4.2 Fuzzy Matching of Person Names

A single parameter specifies whether Fuzzy Matching of Person Names is to be performed. This parameter is optional. If this parameter is not present its value is "false".

```
fuzzy = "fuzzymatching" "=" ("true" / "false")
```

If the value is "false", then the search shall be performed without Fuzzy Matching.

If the value is "true" and the origin server supports Fuzzy Matching, then the search shall be performed with fuzzy matching of Person Name attributes as specified in PS3.4 Section C.2.2.2.1.1 and described in the DICOM Conformance Statement for the origin server.

If the value is "true" and the origin server does not support Fuzzy Matching, then the search shall be performed without Fuzzy Matching, and the response may include a payload containing an appropriate Status Report.

If the value is "true" and Fuzzy Matching 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 URI for the origin server. This may be a combination of scheme, authority, and path.

8.3.4.3 Attributes Included in the Response

A parameter specifies the attributes that should be included in the response. The value is either a comma-separated list of attributes, or the single keyword "all", which means that all available attributes of the object should be included in the response.

```
include = *("includefield" "=" (1#attribute / "all"))
```

The request may contain one or more include parameters; however, if a parameter with the value of "all" is present, then no other `include` parameters shall be present.

If an attribute is a value of an includefield parameter, it is equivalent to C-FIND Universal matching for that attribute. See PS3.4 Section C.2.2.2.3.

8.3.4.4 Response Pagination

The following two parameters can be used to paginate a search response that might contain more matches than can readily be handled. If either is present, then both shall be present.

```
offset = "offset" "=" uint
```

A single parameter specifies the number of matches the origin server shall skip before the first returned match. The "offset" parameter value is an unsigned integer (uint). If this Query Parameter is not present, its value defaults to 0.

```
limit = "limit" "=" uint
```

A single parameter specifies the maximum number of matches the origin server shall return in a single response. The "limit" parameter value is an unsigned integer. If this parameter is not present, its value is determined by the origin server.

8.3.4.4.1 Paging Behavior

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 matches, if the set of matches on the origin server has not changed.

Given the following definitions:

`Offset` 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.

`maxResults` the maximum number of results the origin server allows in a single response.

`matches` the number of matches resulting from the search.

`results` The number of results returned in the response. It is equal to the minimum of:

- `matches - offset`, where if the result is less than zero, the result is zero
- `maxResults`
- `limit`

`remaining` the number of `candidates` that were not returned. It is equal to:
`(remaining = candidates - results)`

The results returned in the response are determined as follows:

- If (`results` <= 0) then there are 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
```

and may include a payload containing an appropriate Status Report.

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 "offset" and "limit" parameters may be inconsistent.

The response shall be in an acceptable media type.

8.3.5 Rendering Query Parameters

This section defines the Query Parameter syntax and behavior for Retrieve requests for Rendered Media Types. All Retrieve transactions for Rendered Media Types shall support these parameters.

8.3.5.1 Query Parameters for Rendered Resources

The Query Parameters defined in this section specify various rendering transformations to be applied to the DICOM images, video, and text contained in the parent DICOM Resource.

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. Instances that are not images will be rendered in an Acceptable Media Type, if one exists; otherwise, they will not be rendered.
4. The set of transformations specified by the parameters in this section shall be applied to the images as if the parameters were a Presentation State, that-is, in the order specified by the applicable image rendering pipeline specified in PS 3.4.

Table 8.3.5-1 shows the Query Parameters that may be used when requesting a Rendered Representation.

Table 8.3.5-1. Retrieve Rendered Query Parameters

Key	Values	Target Resource	Section
accept	Rendered Media Type	All	8.3.2.1

annotation	"patient" and/or "technique"	Image or Video	8.3.2.1
charset	character set token	All	8.3.4.1.2
quality	Integer	Image or Video	8.3.4.1.2
viewport	vw, vh, [sx, sy, sw, sh]	Non-Presentation States	8.3.4.1.3
viewport	vw, vh,	Presentation States	8.3.4.1.3
window	center, width, shape	Non-Presentation States	8.3.4.1.4

8.3.5.1.1 Image Annotation

This parameter specifies that the rendered images or video will have annotations. Its name is "annotation" and 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 image rendering pipelines specified in PS3.4 require that annotations be applied after all other parameters have been applied and the image or video has been rendered. The exact nature and presentation of the annotations is determined by the origin server and is "burned-in" to the rendered content.

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

If any of the parameter values are not keywords, or there are no parameter values, the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

The origin server shall ignore any unsupported parameter values. If unsupported values are present, the origin server shall include the following header field:

```
Warning 299 <service>: The following annotation values are not supported: <values>
```

And may include a payload containing an appropriate warning message.

Note

1. A user agent wanting more control over annotations may retrieve an image, omitting the "annotation" parameter; separately retrieve the metadata; and create customized annotations on the image.

2. The Target Resource may already contain "burned-in" text that is beyond the control of this parameter.

8.3.5.1.2 Image Quality

The "quality" parameter specifies the requested quality of the rendered images or video. 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 may include a payload containing an appropriate error message.

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

The specific interpretation of the meaning of this parameter is determined by the origin server but shall be documented in the conformance statement and the Retrieve Capabilities response.

Note:

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.

The specific interpretation of the meaning of this parameter is determined by the origin server. For example, the origin server could choose to disregard the quality parameter if the resultant image quality would be too low.

8.3.5.1.3 Viewport Scaling

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

The syntax of this parameter for a Presentation State Instance or a Thumbnail 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 or video. Both values are required.
`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. A value of 0,0 specifies the top-left corner of the source image(s).
`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, it defaults to the right edge of the source image. If `sh` is not specified, it defaults 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 origin server shall crop the source images or video to the region specified by `sx, sy, sw,` and `sh`. It shall then scale the source content, maintaining the original aspect ratio, until either the rendered content width or height is the same as the viewport width or 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, i.e., the image(s) shall not be cropped, i.e., the full image is rendered.

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

If any of the following are true:

- This parameter specifies viewport dimensions that are either ill-defined or not supported
- The Target Resource is a Presentation State or Thumbnail and there are not exactly two positive integer parameters

then the response shall be 400 (Bad Request) and may include a payload containing an appropriate Status Report.

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 PS3.3 Section C.10.4.

8.3.5.1.4 Windowing

The "window" parameter controls the windowing of the images or video as defined in PS3.3 Section C.8.11.3.1.5. It has the following syntax:

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

Where

`center` is a decimal number containing the window-center value

`width` is a decimal number containing 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 PS3.3 Section C.11.2.1.2.

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 may 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).

8.3.5.2 Query Parameters for Thumbnails

Table 8.3.5-2 shows the Query Parameters that may be used when requesting a Thumbnail representation.

Table 8.3.5-2. Retrieve Rendered Query Parameters

Key	Values	Target Resource	Section
accept	Rendered Media Type	All	11.4.1.2.1.1
charset	character set token	All	11.4.1.2.1.2
viewport	viewport width, viewport-height	All	11.4.1.2.2.3

The Viewport parameter only has width and height values. If no viewport parameter is provided the origin server will determine the size of the thumbnail.

The charset parameter is only used if the Target Resources is a text/* media type.

8.4 Header Fields

The following sections specify important header fields, some of which have stronger requirements than those specified in the HTTP Standard.

8.4.1 Content Negotiation Header Fields

HTTP uses the Accept and Content-Type header fields for content negotiation and data typing. The values of these header fields are media types. The media types in the Accept header field of a request define the media types that the user agent would find acceptable in the response. The media type in the Content-Type header field of a message, or payload part, describes the format of the representation contained in the payload or part.

Content Negotiation header fields in requests allow the user agent to specify acceptable representations for the response. Table 8.4.1-1 lists the content negotiation header fields. The values in these fields apply to any content in the response, including representations of the Target Resource, representations of error or processing status, and potentially even the miscellaneous text strings that might appear within the HTTP protocol. See [RFC7231] Section 5.3 <https://tools.ietf.org/html/rfc7231#section-5.3>.

Table 8.4.1-1. Content Negotiation Header Fields

Name	Value	Usage	Description
Accept	1#media-	M	All requests that expect to receive a response with a payload shall contain an Accept

	range		header field. See Section 6.6.1.1.
Accept-Charset	1#charset	O	The Accept-Charset header field can be sent by a user agent to indicate what charsets are acceptable in response content. See [RFC7231] Section 5.3.3< https://tools.ietf.org/html/rfc7231#section-5.3.3>.
Accept-Encoding	1#encoding	O	The Accept-Encoding header field is used to indicate the (content-codings<https://tools.ietf.org/html/rfc7231#section-3.1.2.1>). Section 3.1.2.1) acceptable in the response. See [RFC7231] Section 5.3.4 <https://tools.ietf.org/html/rfc7231#section-5.3.4>.
Accept-Language	1#language	O	The Accept-Language header field can be used by user agents to indicate the set of natural languages that are preferred in the response. See [RFC7231] Section 5.3.5<https://tools.ietf.org/html/rfc7231#section-5.3.5>.

8.4.1.1 Accept

User agents use the Accept header field to specify Acceptable Media Types for the response payload. Accept header field can be used to indicate that the request is specifically limited to a small set of desired media types It has the following syntax:

```
Accept        = "Accept:" 1#media-range
media-range   = ("*/*" / (type "/" "*") / (type "/" subtype)) *(OWS ";" OWS accept-params)
accept-params = weight *(accept-ext)
```

Most requests have an Accept header field that contains a comma-separated list of one or more media ranges. A media-range extends media-type with wildcards (*/* or type/*) and parameters that are not defined for media-types. See [RFC7231] Section 5.3.2. For example, if the user agent is willing to accept any media type in the response it should include */* as a value of the Accept header field.

Many of the content negotiation header fields use a weight parameter, named "q" (case-insensitive), to assign a relative "weight" to the preference for that associated kind of content.

The media types in the Accept header can be given a priority ordering by using Weights.

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

This weight is often referred to as "quality value" or "qvalue". See [RFC7231] Section 5.3.1 <https://tools.ietf.org/html/rfc7231#section-5.3.1>.

All requests that might have a response containing a payload shall provide an Accept header field.

See Section 8.7.5 for Acceptable Media Types.

See Section 8.8.1 for Acceptable Character Sets.

8.4.1.1.1 Charset Media Type Parameter

Many media types, especially text/* types, define a "charset" parameter that specifies the character set for the representation. See [RFC7231] Section 3.1.1.2 <https://tools.ietf.org/html/rfc7231#section-3.1.1.2.

DICOM Media Types define a "charset" parameter. See Section 8.7.3.4.3.

For example,

```
application/dicom; charset=ISO-8859-1
```

8.4.2 Content Representation Header Fields

The media type in the Content-Type header field of a message, or payload part, describes the format of the representation contained in the payload or part.

When a message has a payload, the Content Representation Header Fields provide metadata describing how to interpret the representation(s) contained in the payload. Table 8.4.2-1 describes the Content Representation Header Fields, and the usage requirements (Mandatory, Conditional, or Optional) for when they shall be present.

Table 8.4.2-1. Content Representation Header Fields

Name	Value	Usage	Requirement
Content-	media-	C	Specifies the media type of the representation contained in the payload.

Type	type		If a message has a payload, it shall have a Content-Type header field specifying the media type of the payload. See [RFC7231] Section 3.1.1.5<https://tools.ietf.org/html/rfc7231#section-3.1.1.5 >.
Content-Encoding	encoding	C	Specifies any content encodings applied to the representation (beyond those inherent in the media type), and thus what decoding to apply to obtain a representation in the media type specified by the Content-Type. See [RFC7230] Section 3.1.2.2<https://tools.ietf.org/html/rfc7231#section-3.1.2.2>. Content-Encoding allows compression, encryption, and/or authentication of representations. Shall be present if a content encoding has been applied to the representation in the payload.
Content-Language	language	O	Specifies the natural language(s) of the intended audience used in representation. See RFC7231] Section 3.1.3.2 <https://tools.ietf.org/html/rfc7231#section-3.1.3.2>.
Content-Location	uri	C	Contains a URI that references the specific resource corresponding to the representation in the payload. Shall be present if the payload contains a representation of a resource.

8.4.3 Payload Header Fields

When a message has a payload, the Payload Header Fields contain metadata describing the payload, rather than the representation, it contains. Table 8.4.3-1 describes the payload header fields, and the usage requirements (Mandatory, Conditional, or Optional) for when they shall be present.

Table 8.4.3-1. Payload Header Fields

Name	Value	Usage	Description
Content-Length	uint	C	Specifies the decimal number of octets in the payload. If the response message has a payload and does not have a Content-Encoding header field, it shall have a Content-Length header field specifying the length in octets (bytes) of the payload. Shall not be present if the message has a Content-Encoding header field. Shall be present otherwise, even is the size of the payload is zero.
Content-Range	range	C	Specifies the range of a partial representation contained in a payload. See [RFC7233] Section 4.2 <https://tools.ietf.org/html/rfc7233#section-4.2>. The Content-Range header field is sent in a single part 206 (Partial Content) response to indicate the partial range of the selected representation enclosed as the message payload. It is sent in each part of a multipart 206 response to indicate the range enclosed within each body part. It is sent in 416 (Range Not Satisfiable) responses to provide information about the selected representation.
Transfer-Encoding	encoding	C	See [RFC7230] Section 3.3.1<https://tools.ietf.org/html/rfc7230#section-3.3.1>. Shall be present if transfer-encodings have been applied to the payload.

8.5 Status Codes

Each response message contains a status-code.

The most common HTTP status codes used are listed in Table 8.5-1. Most of these codes are described in detail in [RFC7231]. IANA maintains the HTTP Status Code Registry <http://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml>, which contains a complete list of registered status codes.

Table 8.5-1: Status Code Meaning

Status	Code	Description

Success	The 2xx (Successful) class of status code indicates that the client's request was successfully received, understood, and accepted.	
	200 (Success)	All Target Resource representations are contained in the payload. See [RFC7231] Section 6.3.1. <https://tools.ietf.org/html/rfc7231#section-6.3.1>
	201 (Created)	The request has been fulfilled and has resulted in one or more new resources being created. See [RFC7231] Section 6.3.2. <https://tools.ietf.org/html/rfc7231#section-6.3.2>
	202 (Accepted)	The request has been accepted for processing, but the processing has not been completed. The payload of this response should contain a Status Report. [RFC7231] Section 6.3.3 <https://tools.ietf.org/html/rfc7231#section-6.3.3>. The user agent may be able to inspect relevant resources to determine the status at some later time.
	203 (Non-Authoritative Information)	The request was successful, but the enclosed payload has been modified from that of the origin server's 200 (OK) response by a transforming proxy. See [RFC7230] Section 5.7.2https://tools.ietf.org/html/rfc7230#section-7.5.2 and [RFC7231] Section 6.3.4. <https://tools.ietf.org/html/rfc7231#section-6.3.4>
	204 (No-Content)	The server has successfully fulfilled the request and there is no additional content to send in the response payload body. This should be the response when content is successfully uploaded, and the response has no payload. For example, this status code is used in the response to a Conditional Retrieve request), when the Target Resource has not been modified. See [RFC7231] Section 6.3.5. <https://tools.ietf.org/html/rfc7231#section-6.3.5>
	205 (Reset Content)	The server has fulfilled the request and desires that the user agent reset the "document view", which caused the request to be sent, to its original state as received from the origin server.
	206 (Partial Content)	The 206 (Partial Content) status code indicates that the server is successfully fulfilling a range request for the Target Resource by transferring one or more parts of the selected representation that correspond to the satisfiable ranges found in the request's Range header field. This status code shall only be used with Range Requests. See [RFC7233]. Note: This status code was previously (erroneously) used to indicate that only some of a payload was stored.
Redirection	The 3xx (Redirection) class of status code indicates that further action needs to be taken by the user agent to fulfill the request.	
	301 (Moved Permanently)	The origin server has assigned the Target Resource to a new permanent URI, indicated in a Location header field. This status is typically needed when the resource has been moved from one service to another, for example during a migration.
	303 (See Other)	The origin the server is redirecting the user agent to a different resource, as indicated by a URI in the Location header field, which will provide a response to the original request.
	304 (Not Modified)	The origin server has received a conditional GET or HEAD request that would have resulted in a 200 (OK) response if it were not for the fact that the condition evaluated to false.
Client Error	The 4xx (Client Error) class of status code indicates that the user agent has erred. For all these error codes, the origin server should return a payload containing an explanation of the error situation, and whether it is a temporary or permanent condition, except when responding to a HEAD request.	
	400 (Bad Request)	The server cannot or will not process the request due to something that is perceived to be a client error (e.g., malformed request syntax, invalid request …).
	401 (Unauthorized)	The request has not been fulfilled because it lacks valid authentication credentials for the service or Target Resource. The server generating a 401 response shall send a WWW-Authenticate header field (Section 4.1) containing at least one challenge applicable to the server or Target Resource.

	403 (Forbidden)	The origin server understood the request, but refused to authorize it (e.g., an authorized user with insufficient privileges). If authentication credentials were provided in the request, the server considers them insufficient to grant access. The origin server may respond with a 404 (Not Found) if not permitted to use this status code.
	404 (Not Found)	The origin server did not find a representation for the Target Resource or is not willing to disclose that one exists. This might be a temporary condition. If the origin server knows that the resource has been deleted, the 410 (Gone) status code shall be returned rather than 404.
	405 (Method Not Allowed)	The method in the request is known by the origin server but not supported by the target service or resource. The origin server shall include an Allow header field in a 405 response containing a list of the target service or resource's currently supported methods.
	406 (Not Acceptable)	The Target Resource does not have a representation that would be acceptable to the user agent, per the content negotiation header fields in the request, and the server is unwilling to supply a default representation. The origin server should return a payload that lists the available media types and corresponding resource identifiers.
	409 (Conflict)	The request could not be completed due to a conflict with the current state of the Target Resource. This code is used in situations where the user agent might be able to resolve the conflict and resubmit the request. The origin server should return a payload containing enough information for the user agent to recognize the source of the conflict. In the DICOM context, this code might indicate that the origin server was unable to store any Instances due to a conflict in the request (e.g., unsupported SOP Class or SOP Instance mismatch).
	410 (Gone)	Access to the Target Resource is no longer available at the origin server and this condition is likely to be permanent. If the origin server does not know, or has no facility to determine, whether the condition is permanent, the 404 (Not Found) status code should be used instead.
	411 (Length Required)	The origin server refuses to accept the request because the Content-Length header field was not specified.
	413 (Payload Too Large)	The server is refusing to process the request because the request payload is larger than the server is willing or able to process.
	414 (URI Too Long)	The server is refusing to service the request because the request-target (Section 5.3 of [RFC7230]) is longer than the server is willing to interpret.
	415 (Unsupported Media Type)	The origin server does not support the Content-Type in the request payload. This error typically occurs when the user agent is trying to create or update a resource. The origin server should return a payload that lists the available media types and corresponding resource identifiers. Note This is different from 406 (Not Acceptable).
Server Error	The 5xx (Server Error) class of status code indicates that the server is aware that it has erred or is incapable of performing the requested method. For all these error codes, the server should send an explanation of the error situation, and whether it is a temporary or permanent condition, except when responding to a HEAD request.	
	500 (Internal Server Error)	The server encountered an unexpected condition that prevented it from fulfilling the request.
	501 (Not Implemented)	The server does not support the functionality required to fulfill the request. In the DICOM context, this status code shall be used for SOP Class Not Supported errors.

	503 (Service Unavailable)	The origin server is currently unable to handle the request due to a temporary overload or scheduled maintenance, which will likely be alleviated after some delay.
	505 (HTTP Version Not Supported)	The origin server does not support, or refuses to support, the major version of HTTP that was used in the request message.

When a web server determines that a user agent should not receive certain information, the web server must choose the status code and the contents of a Status Report carefully. For example, local policy may dictate that the web service returns a 404 (Not Found) rather than a 401 (Unauthorized) status code to avoid allowing the user agent to infer the existence of a resource. To balance usability of the returned result against appropriate protection, the status code and payload of the response needs to be controlled by policy and context. See also http://hl7.org/fhir/security.html#AccessDenied and https://www.owasp.org/index.php/Top_10_2007-Information_Leakage_and_Improper_Error_Handling.

8.6 Payloads

Both request and response messages may have message bodies. The message body (if any) of an HTTP message is used to carry the payload of the message. The message body is identical to the payload unless a content coding has been applied, as described in [RFC7230] Section 3.3.1 <https://tools.ietf.org/html/rfc7230#section-3.3.1>. PS3.18 uses the term 'payload' to denote the message body before any content coding has been applied to it.

A message may or may not have a payload. A payload may be empty; that is, its length is zero. If a message has no payload, then the message shall have neither Content-Encoding nor Content-Length header fields. If a message has a payload to which a transfer-coding has been applied, then the message shall have a Content-Encoding header field. If a message has a payload that has not had a transfer-coding applied, then the message shall have a Content-Length header field.

Any message containing a payload shall have appropriate Content Representation <https://tools.ietf.org/html/rfc7231#section-3.1> and Payload Header Fields <https://tools.ietf.org/html/rfc7231#section-3.3>. Any message with a payload shall have a Content-Type header field that specifies the media type of the representation contained in the payload. The media type specifies whether the payload is single part or multipart (see Section 8.7). Any message with a payload should include a Content-Location header field. See RFC7231] Section 3.1.2.2 <https://tools.ietf.org/html/rfc7231#section-3.1.2.2>.

8.6.1 Payload Format

Payloads may be in either single part or multipart format depending on the media type.

8.6.1.1 Single Part Payload

A single part payload contains one representation that is described by the Content Representation Header Fields (see Section 8.4.3) contained in the message header. A message with a single part payload shall have a Content-Type header field with a single part media-type.

8.6.1.2 Multipart Payload

A message with a multipart payload contains one or more representations. The media type of the root representation (see RFC2387) may be specified by the Content-Type header field of the message. If no root parameter is specified, then the root representation is the first representation in the payload.

Each part in a multipart payload shall start with a boundary string, followed by a Content-Type header field. See Table 8.6.1-1 for other header fields occurring in multipart payloads.

Table 8.6.1-1. Multipart Header Fields

Name	Value	Usage	Description
Content-Type	media-type	M	
Content-Encoding	encoding	C	Shall be present if the response payload has a content encoding
Content-Length	int	C	Shall be present if the response payload does not have a content encoding
Content-Location	uri	C	Shall be present if the response payload contains a representation of a resource. See [RFC7231] Section 3.1.4.2
Location	url	C	See [RFC7231] Section 7.1.2

See Section 8.7.1 and [RFC7031].

The following is an example template of a multipart request or response message that has a multipart payload:

```
request-line / response-line
Content-Type: multipart-media-type CRLF
Content-Location: "/" {/url} CRLF
*(header-field CRLF)
CRLF
multipart-payload
```

The Content-Type header field shall have a multipart media-type such as:

```
Content-Type: multipart/related; type=DQ root-media-type DQ; boundary="---boundary---"
```

Where,

`multipart-media-type` is a media type defined by [RFC2387] <https://tools.ietf.org/html/rfc2387>.

`root-media-type` is a single part media type that specifies the media type of the root, typically the first part, in the payload. If the value of the type parameter and the root body part's content-type differ then the user agent's behavior is undefined.

`boundary` specifies a string that acts as a boundary between message parts.

Each part in a multipart payload shall start with a Boundary header field, followed by a Content-Type header field. Other header fields may be included, such as Content-Location, and either the Content-Length or Content-Encoding header field, optionally followed by other header fields.

If a multipart payload contains Metadata (see Section 8.7.3.3.1), and Bulkdata (see Section 8.7.3.3.2), then all Metadata message parts that reference a Bulkdata part shall precede the referenced Bulkdata part.

8.6.1.2.1 Multipart Payload Syntax

The syntax of a multipart payload is:

```
multipart-payload = 1*(DASH boundary CRLF part CRLF) DASH boundary DASH
```

Where

```
DASH = "--"
boundary = 0*69(bchar / SP) bchar
bchar = DIGIT / ALPHA / "'" / "(" / ")" / "+" / "_"    ; The legal boundary characters
      / "," / "-" / "." / "/" / ":" / "=" / "?"
part = Content-Type: media-type CRLF
       Content-Location: uri CRLF
       (Content-Length: uint CRLF / Content-Encoding: encoding CRLF)
       [Content-Description: text CRLF]
       *(header-field CRLF)
       CRLF
       part-payload
part-payload = *OCTET
```

For example, if the boundary is "++++", then a message payload containing three parts would be structured as follows:

```
--++++CRLF
Content-Type: media-type CRLF
Content-Location: uri CRLF

(Content-Length: uint CRLF / Content-Encoding: encoding CRLF)

[Content-Description: {description} CRLF]
CRLF
payload CRLF
--++++CRLF
Content-Type: media-type CRLF
. . .
payload CRLF
--++++CRLF
Content-Type: media-type CRLF
```

```
. . .
payload CRLF
--++++--
```

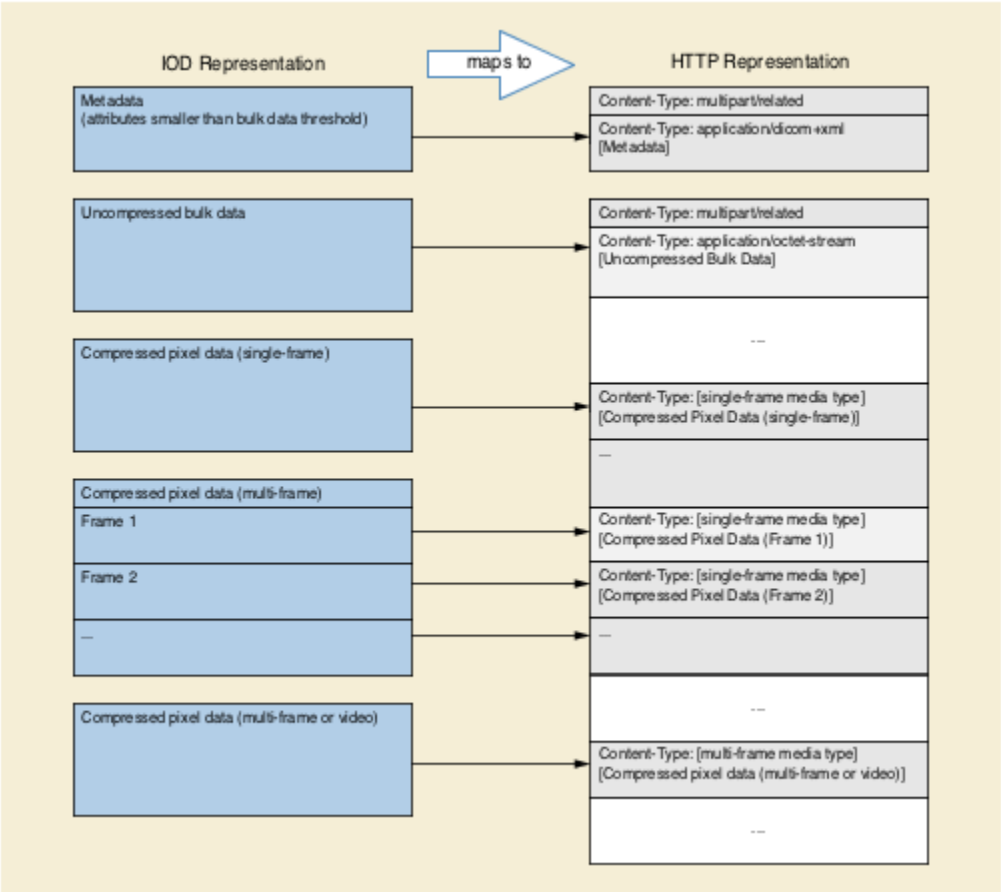


Figure 8.6-1. Mapping between IOD and Multipart Payload

8.6.2 DICOM Representations

All DICOM objects are defined by Information Object Definitions (IODs). See PS3.3. Representations of DICOM Resources are encodings of DICOM Information Objects into octet streams.

Each IOD has an associated set of Attributes, which define semantic concepts. Each Attribute has:

- a Tag, which identifies the attribute using an integer
- a Keyword, which identifies the attribute using a token
- a Type, which indicates whether it is required or optional
- a Value Representation, which defines the data type of its value(s)
- a Value Multiplicity, which specifies the number of values that it may have

A Data Element is a concrete representation of an Attribute See PS3.5. Each Data Element has:

- an Identifier, which would typically be its Tag, but could be its Keyword
- a Value Representation, which defines its data type
- a Value Field Length
- a Value Field, which is a sequence of bytes containing zero or more values

Each Instance contains Data Elements representing the Attributes from the Patient, Study, Series, and Instance levels of the IOD. For example, if a Series resource contains 12 Instances, then a transaction that retrieves that Series will contain a representation

of the Series and its 12 Instances, in a specific media type, and each Instance will have Patient, Study, Series, and Instance level attributes.

PS3.18 defines three distinct representations of DICOM Resources that can be encoded into DICOM Media Types: Composite SOP Instances, Metadata, and Bulkdata.

DICOM Media Types and their corresponding representations are defined in Section 8.7.3. Other media types used in PS3.18 are defined in Section 7.10.4

8.6.2.1 Composite SOP Instances

The traditional DICOM unit of communication and storage is the Composite SOP Instance, which is a concrete representation of an Information Object. Instances, and groups of related Instances, can be transmitted, stored, and retrieved. See PS3.3 and 3.4.

8.6.2.2 Web Service Constraints

DICOM Web Services only support representations with explicit Value Representations. Implicit Value Representations (see PS3.5, Section 7.1.3) shall not be used.

8.6.3 Status Report

A Status Report is a description of warnings or errors encountered by the origin server in processing a request. The contents should be clear and succinct. If the request does not include an Acceptable Media Type, the Status Report should use the default media type for the Text Resource Category, which is text/html.

8.7 Media Types

Media types are the basis for both content negotiation and data typing of message payloads. Each PS3.18 service, and/or transaction defines the media types and associated representations that are default, required and optional.

The media type also specifies whether the payload contains a single representation (single part), or multiple representations (multipart). Multipart payloads are only defined for the RESTful APIs. See Section 8.6.1.2 and Section 10.4.3.

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 to provide open and extensible data typing and type negotiation. The syntax of media types is:

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

Where

```
type           = token
subtype        = token
mt-parameter   = mtp-name "=" mtp-value
mtp-name       = token
mtp-value      = (token / quoted-string)
```

The ‘type/subtype’ may be followed by parameters in the form of ‘name "=" value’ pairs.

The type, subtype, and mtp-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.

An mtp-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 <https://tools.ietf.org/html/rfc7231#section-3.1.1.1>.

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

Many media types specify a character set parameter.

Note

The term "MIME Type" is not synonymous with "Media Type". MIME types are defined by *Multipurpose Internet Mail Extensions* [RFC2045] and used by email programs. Media Types are defined by *Media Type Specifications and Registration Procedures* [RFC6838].

8.7.1 Multipart Media Types

Some of the services defined in PS3.18 support the multipart media types [RFC2387] <https://tools.ietf.org/html/rfc2387>. The syntax is:

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

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

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

Where

```
boundary   ; See Section 7.8.1.2.1
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.

8.7.2 DICOM Resource Categories

Table 8.7.2-1 defines Resource Categories that correspond to different SOP Classes. The following sections map each Resource Category to appropriate DICOM and Rendered media types.

Table 8.7.2-1. 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 H.265), 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).

Resource Category	Definition
Other	This category includes all resources that are not included above, for example waveforms.

8.7.3 DICOM Media Types

This section defines the media types used to represent DICOM Instances, Metadata and Bulkdata. It describes:

- The media type and Transfer Syntax parameters for DICOM (PS3.10 File Format) Instances
- The media types that can be used for Metadata
- The media types and Transfer Syntaxes parameters for Bulkdata
- 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 8.7.4); some of the same media types are used for both rendered content and Bulkdata.

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.

The Implicit VR Little Endian (1.2.840.10008.1.2), and Explicit VR Big Endian (1.2.840.10008.1.2.2 - Retired) 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 8.7.3-5 or Table 8.7.3-6.

Table 8.7.3-1 specifies the definition of media type requirement terms used in the tables in this section and the tables in Section 8.7.3.1.

Table 8.7.3-1. Definition of Media Type Requirement

Requirement	Code	Definition
Default	D	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	R	The origin server shall support this media type.
Optional	O	The origin server may support this media types.

Table 8.7.3-2, Table 8.7.3-3, Table 8.7.3-4, Table 8.7.3-5, and Table 8.7.3-6 specify the media types used to encode different representations of DICOM Instances for the URI and RESTful services. These media types apply to all Resource Categories and have default encodings for images and video data elements contained in the Instances.

8.7.3.1 The application/dicom Media Type

The application/dicom media type specifies that the representation is encoded in the DICOM File Format specified in PS3.10, Section 7.

8.7.3-2. Media Types for DICOM PS3.10 Files

Media Type	Descriptions	URI	RESTful
application/dicom	Encodes Composite SOP Instances in the DICOM File Format	See Table 8.7.3-3	See Table 8.7.3-3

Table 8.7.3-2 specifies the default and optional Transfer Syntax UID combinations for each application/dicom Resource Category (see Table 8.7.2-1) for the URI and RESTful 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 media type Transfer Syntax parameter is specified, then the Explicit VR Little Endian Transfer Syntax "1.2.840.10008.1.2.1" shall be used.

Note:

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

The origin server may support additional Transfer Syntaxes.

Table 8.7.3-3. Transfer Syntax UIDs for application/dicom Media Types

Category	Transfer Syntax UID	Transfer Syntax Name	URI	RESTful
Single Frame Image	1.2.840.10008.1.2.1	Explicit VR Little Endian	D	D
	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	O	O
	1.2.840.10008.1.2.4.50	JPEG Baseline (Process 1): Default Transfer Syntax for Lossy JPEG 8 Bit Image Compression	O	O
	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)	O	O
	1.2.840.10008.1.2.4.57	JPEG Lossless, Non-Hierarchical (Process 14)	O	O
	1.2.840.10008.1.2.5	RLE Lossless	O	O
	1.2.840.10008.1.2.4.80	JPEG-LS Lossless Image Compression	O	O
	1.2.840.10008.1.2.4.81	JPEG-LS Lossy (Near-Lossless) Image Compression	O	O
	1.2.840.10008.1.2.4.90	JPEG 2000 Image Compression (Lossless Only)	O	O
	1.2.840.10008.1.2.4.91	JPEG 2000 Image Compression	O	O
	1.2.840.10008.1.2.4.92	JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only)	O	O
	1.2.840.10008.1.2.4.93	JPEG 2000 Part 2 Multi-component Image Compression	O	O
Multi-Frame Image	1.2.840.10008.1.2.1	Explicit VR Little Endian	D	D
	1.2.840.10008.1.2.4.90	JPEG 2000 Image Compression (Lossless Only)	O	O
	1.2.840.10008.1.2.4.91	JPEG 2000 Image Compression	O	O
	1.2.840.10008.1.2.4.92	JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only)	O	O
	1.2.840.10008.1.2.4.93	JPEG 2000 Part 2 Multi-component Image Compression	O	O
Video	1.2.840.10008.1.2.1	Explicit VR Little Endian	D	D
	1.2.840.10008.1.2.4.100	MPEG2 Main Profile @ Main Level	O	O
	1.2.840.10008.1.2.4.101	MPEG2 Main Profile @ High Level	O	O
	1.2.840.10008.1.2.4.102	MPEG-4 AVC/H.264 High Profile / Level 4.1	O	O
	1.2.840.10008.1.2.4.103	MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1	O	O
	1.2.840.10008.1.2.4.104	MPEG-4 AVC/H.264 High Profile / Level 4.2 For 2D Video	O	O
	1.2.840.10008.1.2.4.105	MPEG-4 AVC/H.264 High Profile / Level 4.2 For 3D Video	O	O
	1.2.840.10008.1.2.4.106	MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2	O	O
Text	1.2.840.10008.1.2.1	Explicit VR Little Endian	D	D

Other	1.2.840.10008.1.2.1	Explicit VR Little Endian	D	D

8.7.3.2 DICOM Metadata Media Types

Table 8.7.3-4 specifies the media types that may be used to encode representations of Metadata for the URI and RESTful services. Only the RESTful Services support Metadata representations.

8.7.3-4. Media Types for Metadata

Media Type	Descriptions	URI	RESTful
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

8.7.3.3 DICOM Bulkdata Media Types

Bulkdata representations are only supported by RESTful services. There are two categories of Bulkdata: uncompressed and compressed.

The default media type for the Resource Category shall be returned when the origin server supports none of the Acceptable Media Types.

The origin server may support additional Transfer Syntaxes.

If no media type Transfer Syntax parameter is specified, then the Explicit VR Little Endian Transfer Syntax "1.2.840.10008.1.2.1" shall be used.

Note:

The tables in this section have no entries for the URI service, since they do not support separate retrieval of Bulkdata.

8.7.3.3.1 Uncompressed Bulkdata

Table 8.7.3-5 specifies the default media type and Transfer Syntax UIDs, by Resource Category (see Table 8.7.2-1) that can be used with uncompressed Bulkdata for the RESTful services. Uncompressed Bulkdata is encoded as a stream of uncompressed bytes (octets) 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 Bulkdata.

Table 8.7.3-5. Transfer Syntax UIDs for Uncompressed Data in Bulkdata

Category	Media Type	Transfer Syntax UID	Transfer Syntax Name	RESTful
Single Frame Image	application/octet-stream	1.2.840.10008.1.2.1	Explicit VR Little Endian	D
Multi-Frame Image	application/octet-stream	1.2.840.10008.1.2.1	Explicit VR Little Endian	D
Video	application/octet-stream	1.2.840.10008.1.2.1	Explicit VR Little Endian	D
Text	application/octet-stream	1.2.840.10008.1.2.1	Explicit VR Little Endian	D
Other	application/octet-stream	1.2.840.10008.1.2.1	Explicit VR Little Endian	D

Note

Even though the Transfer Syntax is Explicit VR Little Endian, the Value Representation is not actually encoded at the beginning of the octet-stream. The Value Representation is contained in the Metadata that references the Bulkdata.

8.7.3.3.2 Compressed Bulkdata

Table 8.7.3-6 specifies the default and optional media types and Transfer Syntax UID combinations for each Resource Category (see Table 8.7.2-1) of compressed Bulkdata for the RESTful services.

Note:

Some of the Transfer Syntax Names include text about Default Transfer Syntax, however this applies to its role in DIMSE transactions, rather than the default for RESTful services (which is specified in the RESTful column of the table).

These media types can be used to retrieve Bulkdata, such as images or video, encoded in a specific Transfer Syntax.

Bulkdata containing compressed Pixel Data will have each frame encoded as a separate part of a multipart response and identified by an appropriate Content-Type header and Content-Location header field.

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).

Table 8.7.3-6. Media Types and Transfer Syntax UIDs for Compressed Data in Bulkdata

Category	Media Type	Transfer Syntax UID	Transfer Syntax Name	RESTful
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	D
		1.2.840.10008.1.2.4.50	JPEG Baseline (Process 1): Default Transfer Syntax for Lossy JPEG 8 Bit Image Compression	O
		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)	O
		1.2.840.10008.1.2.4.57	JPEG Lossless, Non-Hierarchical (Process 14)	O
	image/x-dicom-rle	1.2.840.10008.1.2.5	RLE Lossless	D
	image/x-jls	1.2.840.10008.1.2.4.80	JPEG-LS Lossless Image Compression	D
		1.2.840.10008.1.2.4.81	JPEG-LS Lossy (Near-Lossless) Image Compression	O
	image/jp2	1.2.840.10008.1.2.4.90	JPEG 2000 Image Compression (Lossless Only)	D
		1.2.840.10008.1.2.4.91	JPEG 2000 Image Compression	O
	image/jpx	1.2.840.10008.1.2.4.92	JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only)	D
		1.2.840.10008.1.2.4.93	JPEG 2000 Part 2 Multi-component Image Compression	O
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	D
		1.2.840.10008.1.2.4.50	JPEG Baseline (Process 1): Default Transfer Syntax for Lossy JPEG 8 Bit Image Compression	O
		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)	O
		1.2.840.10008.1.2.4.57	JPEG Lossless, Non-Hierarchical (Process 14)	O
	image/x-dicom-rle	1.2.840.10008.1.2.5	RLE Lossless	D
	image/x-jls	1.2.840.10008.1.2.4.80	JPEG-LS Lossless Image Compression	D
		1.2.840.10008.1.2.4.81	JPEG-LS Lossy (Near-Lossless) Image Compression	O
	image/jp2	1.2.840.10008.1.2.4.90	JPEG 2000 Image Compression (Lossless Only)	D
		1.2.840.10008.1.2.4.91	JPEG 2000 Image Compression	O

	image/jpx	1.2.840.10008.1.2.4.92	JPEG 2000 Part 2 Multi-component Image Compression (Lossless Only)	D
		1.2.840.10008.1.2.4.93	JPEG 2000 Part 2 Multi-component Image Compression	O
Video	video/mpeg2	1.2.840.10008.1.2.4.100	MPEG2 Main Profile @ Main Level	O
		1.2.840.10008.1.2.4.101	MPEG2 Main Profile @ High Level	D
	video/mp4	1.2.840.10008.1.2.4.102	MPEG-4 AVC/H.264 High Profile / Level 4.1	D
		1.2.840.10008.1.2.4.103	MPEG-4 AVC/H.264 BD-compatible High Profile / Level 4.1	O
		1.2.840.10008.1.2.4.104	MPEG-4 AVC/H.264 High Profile / Level 4.2 For 2D Video	O
		1.2.840.10008.1.2.4.105	MPEG-4 AVC/H.264 High Profile / Level 4.2 For 3D Video	O
		1.2.840.10008.1.2.4.106	MPEG-4 AVC/H.264 Stereo High Profile / Level 4.2	O
Text		N/A (no defined compression transfer syntaxes for Text)		
Other		N/A (no defined compression transfer syntaxes for Other)		

The origin server may support additional Transfer Syntaxes.

Note

1. The compressed Bulkdata of each part of a multipart payload contains only the compressed bit stream and not the DICOM PS3.5 Encapsulated Sequence or Delimiter Items.
2. For the media type image/dicom+jpeg Transfer Syntaxes, the image may or may not include the JFIF marker segment. See PS3.5 Section 8.2.1.
3. For the media type image/dicom+jp2 and image/dicom+jpx Transfer Syntaxes, the image does not include the jp2 marker segment. See PS3.5 Section 8.2.4 and A.4.4.
4. The resource on the origin server may have been encoded in the Deflated Explicit VR Little Endian (1.2.840.10008.1.2.1.99) Transfer Syntax. If so, the origin server may inflate it, and then convert it into an Acceptable Transfer Syntax. Alternatively, if the user agent allowed a Content-Encoding header field of 'deflate', then the deflated bytes may be transferred unaltered, but the Transfer Syntax parameter in the response should be the Explicit VR Little Endian Transfer Syntax.
5. Compressed multi-frame image Bulkdata 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 (RFC3745) or uncompressed application/octet-stream media type.
6. Video Bulkdata 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 (RFC4337) media type.
7. Many of the media types used for compressed Pixel Data transferred as Bulkdata values are also used for consumer format media types. A web 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 8.7.4.

   For example, the media type for Bulkdata values of lossless 16-bit JPEG 10918-1 encoded Pixel Data is "image/jpeg", the same media type as might be used for 8-bit JPEG 10918-1 encoded Pixel Data, whether extracted as Bulkdata, or rendered. The Transfer Syntax parameter of the Content-Type header field is useful to signal the difference.

8.7.3.4 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.

The application/dicom+xml and application/dicom+json Media Types may have a Transfer Syntax parameter in order to specify the encoding of base64 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.

8.7.3.4.1 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 8.6.1.2.1 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.

8.7.3.4.2 Transfer Syntax Parameter

For a given DICOM Media Type, a single Transfer Syntax parameter value may be specified, but its usage may be constrained by the service for which they are used.

RESTful origin servers shall support the Transfer Syntax parameter.

Transfer syntax media type parameters are forbidden in URI Service 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 [RFC6838] 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; type="application/dicom";transfer-syntax=1.2.840.10008.1.2.4.50, multipart/related;
type="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.

8.7.3.4.3 Character Set Parameter

All DICOM Media Types may have a single Character Set parameter that to specifies the Acceptable Character Set 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 8.8 for character set details.

8.7.3.5 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 for the user agent and required for the origin server.

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 RESTful service uses the Transfer Syntax parameter in the "accept" Query Parameter (see 8.3.3.1) and the Transfer Syntax Query Parameter is not supported.

8.7.3.6 Acceptable Transfer Syntaxes

Each DICOM Media Type in the Acceptable Media Types has an Acceptable Transfer Syntax, which is explicitly specified or has a default value.

The Acceptable Transfer Syntax for a DICOM 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 8.3.3.1)
2. The value(s) contained in the Transfer Syntax Query Parameter (see Section 8.7.3.5)
3. The Transfer Syntax media type parameter contained in the Accept header field.

8.7.3.7 Support for DICOM Media Types by Service

The URI and RESTful APIs support the following DICOM Media Types:

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

Support for the Transfer Syntax and Character Set media type parameters is required for RESTful services.

Support for the Transfer Syntax and Character Set parameters is forbidden for URI Services (i.e. they may not be present in the request or the response).

8.7.4 Rendered Media Types

DICOM Instances may be converted by a rendering process into non-DICOM Media Types. This can be useful to display or process them using 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 Bulkdata Media Types, that is, for encoding Bulkdata extracted from Encapsulated Pixel Data (used with compressed Transfer Syntaxes), without applying a rendering process. See Section 8.7.4.

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

The definitions of media type requirements are provided in Table 8.7.3-1.

Origin servers that support URI or RESTful services shall support rendering Instances of different Resource Categories into Rendered Media Types as specified in Table 8.7.4-1.

Table 8.7.4-1. Rendered Media Types by Resource Category

Category	Media Type	URI	RESTful
Single Frame Image	image/jpeg	D	D
	image/gif	O	R
	image/png	O	R
	image/jp2	O	O
Multi-Frame Image	image/gif	O	O
Video	video/mpeg	O	O
	video/mp4	O	O
	video/H265	O	O
Text	text/html	D	D
	text/plain	R	R
	text/xml	O	R
	text/rtf	O	O
	application/pdf	O	O

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.

The origin server may support additional Rendered Media Types.

A Transfer Syntax media type parameter is not permitted for Rendered Media Types.

8.7.5 Acceptable Media Types

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

- The Accept Query Parameter, which may or may not be present.
- The Accept header field, which shall be present.

The response to a request without an Accept header field shall be 406 (Not Acceptable). The presence of an Accept Query Parameter does not eliminate the need for an Accept header field. For details see Section 8.3.3.1.

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 400 (Bad Request).

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 <https://tools.ietf.org/html/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.

8.7.6 Accept Query Parameter

The Accept Query Parameter can be used to specify Acceptable Media Types. See Section 8.7.5.

8.7.7 Accept Header Field

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

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

The Accept header field value shall be a comma-separated list of one or more media ranges acceptable in the response. See [RFC7231] Section 5.3.2 <https://tools.ietf.org/html/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. 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. DICOM specified that the */* media range matches the default media type for the target's Resource Category. If no default media type is defined for a Resource Category, then any media type from the Resource Category is acceptable.

If the response might contain a payload , an Accept header field shall be present in the request.

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 by the origin server.

8.7.8 Selected Media Type and Transfer Syntax

The selection of the media type and transfer syntax by the origin server are interrelated.

8.7.8.1 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. Identify 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 which 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 qvalues, based on the media ranges above:

Table 8.7.8-: Media Type QValue Example

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.

8.7.8.2 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 8.7.8 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, if any;
3. Otherwise select the default Transfer Syntax (see Table 8.7.3-3, Table 8.7.3-5 or Table 8.7.3-6) 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.

8.7.9 Content-Type Header Field

The Content-Type header field specifies the media type of the payload. It shall 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 8.7.3.4.2), and zero or one charset parameter (see Section 8.7.3.4.3), 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.

8.8 Character Sets

HTTP uses charset names to indicate or negotiate the character encoding of textual content in representations [RFC6365] Section 3.3 <https://tools.ietf.org/html/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 also be identified by using the DICOM Defined Terms for the character set (see Annex D, PS3.3, Section C.12.1.1.2, and PS3.5, Section 6.1.2.3), which shall be quoted strings since they contain the space (' ') character

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

The syntax is:

```
charset = token / defined-term / DQ defined-term DQ
```

Where

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

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 URIs.

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

A request without any Character Set Query Parameter or Accept-Charset header field implies that the user agent will accept the Default Character Set, UTF-8, in the response.

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

8.8.1 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 PS3.3, Section C.12.1.1.2, and PS3.5, Section 6.1.2.3. This is especially important for ISO 2022 character sets.

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

8.8.2 Character Set Query Parameter

See Section 8.3.3.2

8.8.3 Character Set Media Type Parameters

DICOM Media Types accept character set (charset) parameters. See Section 8.7.3.1.3.

Many other media types also accept character set (charset) parameters. See the IANI Media Type Registry <http://www.iana.org/assignments/media-types/media-types.xhtml>.

8.8.4 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 charsets 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.

If this parameter has a value that is not a valid or supported character set, the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate Status Report. See Section 8.6.3.

8.8.5 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 Character Set:

- 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 RESTful 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 PS3.5 Section 6.1.2.3).

8.9 Retrieve Capabilities Transaction

This transaction retrieves a Capabilities Description (see Annex I), which is a machine-readable description of the service(s) implemented by an origin server. All RESTful services defined by in PS3.18 shall implement this transaction.

The Target Resource for this transaction is an origin server. The response contains a Capabilities Description, which describes the transactions, resources, representations, etc. that are supported by the service(s).

8.9.1 Request

The request shall have the following syntax:

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

8.9.1.1 Resource

The Target Resource for this transaction is the Base URI ("/") for the Service. See Section 8.2.

8.9.1.2 Query Parameters

This transaction has no Query Parameters.

8.9.1.3 Request Header Fields

Table 8.9.1-1. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	media-type	M	M	The Acceptable Media Types for the response payload
Accept-Charset	charset	O	O	The Acceptable Character Sets of the response payload

See also Section 8.4.

8.9.1.4 Request Payload

The request has no payload.

8.9.2 Behavior

The origin server shall return a Capabilities Description in an Acceptable Media Type as defined in Annex I.

8.9.3 Response

The format of the response is as follows:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
*(header-field CRLF)
CRLF
[payload / status-report]
```

8.9.3.1 Status Codes

Table 8.9.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 8.9.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	All Instances were successfully retrieved.
	304 (Not Modified)	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 request containing an If-None-Match header field.
Failure	400 (Bad Request)	There was a problem with the request.

8.9.3.1.1 Response Header Fields

Table 8.9.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	dicom-media-type	M	The media-type of the payload
Content-Length	uint	C	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

8.9.3.2 Response Payload

A success response shall have a payload containing a Capabilities Description in the Selected Media Type. The Capabilities Description shall conform to the requirements and structure defined Annex H.

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

8.9.4 Media Types

The media types supported by the Retrieve Capabilities service are application/vnd.sun.wadl+xml (Web Application Description Language) or application/json.

8.10 Notifications

Some of the Restful Services support Notifications. The following transactions must be supported if Notifications are supported.

8.10.1 Overview

8.10.2 Conformance

An implementation's Conformance Statement shall document whether or not it supports notifications.

8.10.3 Transaction Overview

Any service that supports the Notifications must support the following transactions:

Table 8.10.3: Notification Sub-System Transactions

Name	Method	Description
Open Notification Connection	GET	The user agent requests that the origin server create a Notification Connection between them.
Send Event Report	N/A	The origin server sends an Event Report to an End-Point

8.10.4 Open Notification Connection Transaction

This transaction creates a connection between the user agent and the origin server over which the origin server can send Event Reports to the user agent.

The connection uses the WebSocket protocol. The connection can use the same TCP port as the HTTP connection, but they are separate connections.

See [RFC-6455] <https://tools.ietf.org/html/rfc6455#page-38> for details of the WebSocket protocol.

8.10.4.1 Request

There is more than one way to establish a WebSocket connection. One typical way has the following syntax:

```
GET SP / SP version CRLF
Host: host CRLF
Upgrade: "WebSocket" CRLF
Connection: "Upgrade" CRLF
Origin: url CRLF
Sec-WebSocket-Key: nonce CRLF
Sec-WebSocket-Protocol: protocols CRLF
Sec-WebSocket-Version: "13" CRLF
*(<header-field> CRLF)
CRLF
```

8.10.4.1.1 Target Resources

The Target Resource is an origin server implementing a DICOM RESTful Service.

8.10.4.1.2 Query Parameters

This transaction has no query parameters.

8.10.4.1.3 **Request Header Fields**

Table 8.10.4-1. Request Header Fields

Name	Value	Usage
Content-Type	media-type	M
Upgrade	"WebSocket"	M
Connection	"Upgrade"	M
Origin	url	M
Sec-WebSocket-Key	accept-key	M
Sec-WebSocket-Protocol	protocols	O
Sec-WebSocket-Version	version	M

For details of the request header field values see [RFC6455]<https://tools.ietf.org/html/rfc6455#page-38>.

8.10.4.1.4 **Request Payload**

The request has no payload.

8.10.4.2 Behavior

When the origin server receives this request, it shall open and maintain a WebSocket connection between itself and the user agent.

If the connection is lost at any point, the user agent can re-establish it by repeating this transaction.

8.10.4.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
Upgrade: "WebSocket" CRLF
Connection: "Upgrade" CRLF
Sec-WebSocket-Accept: response-key CRLF
Sec-WebSocket-Protocol: protocol CRLF
*(header-field CRLF)
```

8.10.4.3.1 **CRLFStatus Codes**

Table 8.10.4-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 8.10.4-1. Status Code Meaning

Status	Code	Meaning
Success	101 (Switching Protocols)	The protocol was successfully changed to WebSocket.
Failure	400 (Bad Request)	There was a problem with the request.

8.10.4.3.2 **Response Header Fields**

Table 8.10.4-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Upgrade	"WebSocket"	M	
Connection	"Upgrade"	M	
Origin	url	M	
Sec-WebSocket-Accept	response-key	M	See [RFC6455]
Sec-WebSocket-Protocol	protocol	M	See [RFC6455]

See also Section 8.4.

8.10.4.3.3 Response Payload

The response has no payload.

8.10.5 Send Event Report Transaction

The origin server uses this transaction to notify a user agent of Events.

This transaction sends a notification, containing an Event Report, over an established Notification Connection between a user agent and an origin server. Unlike most transactions, this transaction is initiated by the origin server.

This transaction corresponds to a DIMSE N-EVENT-REPORT action.

Each service may define Events, and the corresponding Event Report messages and their contents, related to its resources.

8.10.5.1 Request

The request shall use the WebSocket Data Frame transmission protocol.

8.10.5.1.1 Request Payload

The data frames shall have an opcode of "%x1" (text).

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

Note:

The WebSocket protocol does not currently allow content negotiation so it is not possible to support both XML and JSON encoding of Event Report messages.

If the Event Reports are being proxied into DIMSE N-EVENT Reports, a Message ID (0000,0110) must be managed by the proxy.

8.10.5.2 Behavior

The user agent shall accept all Attributes included in any Event Report. No requirements are placed on what the user agent does with this information.

8.10.5.3 Response

The user agent shall send an acknowledgement response containing a Status Code. The response Status Code can be either general (See PS3.7, Section C) or specific to the service. Each service may define response codes specific to that service, which should be the same codes used by the corresponding DIMSE service, if any. The response is encoded as a WebSocket (binary) data frame with an opcode of "%x2" (binary). See [RFC6455]<https://tools.ietf.org/html/rfc6455#page-27>.

8.10.5.3.1 Response Payload

The data frame payload data shall be a US-ASCII string "success" for a success acknowledgement (ACK), or "failure" for a failure acknowledgement (NAK).

8.11 Security and Privacy

It is very likely that DICOM objects contain Protected Health Information. Privacy regulations in the United States (HIPAA), Europe (GDPR), and elsewhere, require that Individually Identifiable Information be kept private. It is the responsibility of implementers of the DICOM Standard to ensure that governmental regulations for security and privacy are satisfied.

See Guide to Privacy and Security of Electronic Health Information <https://www.healthit.gov/sites/default/files/pdf/privacy/privacy-and-security-guide.pdf>.

9 URI Web Service

9.1 Overview

The URI Service, also referred to as WADO-URI, enables a user agent to retrieve DICOM instances over web protocols.

9.1.1 Resource Descriptions

The URI Service does not define resources in the form of a Target Resource Path, such as {/resource}. The Target URI of each transaction is a reference to the Base URI ("/") and the Target Resource is identified using query parameter values. The resources for the URI Service are instances of DICOM Composite Storage SOP Classes defined in PS3.4.

9.1.2 Common Query Parameters

The Query Parameters specified in this section may be used with either the Retrieve DICOM Instance or Retrieve Rendered Instance transactions and are applicable to all supported DICOM SOP Classes.

9.1.2.1 Mandatory Query Parameters

The origin server shall support Query Parameters as required in Table 9.1.2-1.

The user agent shall supply in the request Query Parameters as required in Table 9.1.2-1.

The Query Parameters may appear in any order.

Table 9.1.2-1. Mandatory Query Parameters

Name	Values	Usage		Section
		User Agent	Origin Server	
requestType	"WADO"	M	M	9.1.2.1.1
studyUID	uid	M	M	9.1.2.1.3
seriesUID	uid	M	M	9.1.2.1.3
objectUID	uid	M	M	9.1.2.1.4

See Section 8.4.

Note

To identify a DICOM SOP Instance, only a SOP Instance UID is required, because any UID is globally unique. However, the Standard requires that the UIDs of the higher levels in the DICOM Information Model (i.e., series and study) are specified, 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.

9.1.2.1.1 Request Type

```
requestType = %s"requestType=" token
token = "WADO"
```

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.

9.1.2.1.2 Study UID

```
study = %s"studyUID=" uid
```

The value of this parameter is a Study Instance UID. It is REQUIRED.

The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded.

9.1.2.1.3 Series UID

```
series = %s"seriesUID=" uid
```

The value of this parameter is a Series Instance UID. It is REQUIRED.

The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded.

9.1.2.1.4 Instance UID

```
instance   = %s"objectUID=" uid
```

The value of this parameter is a SOP Instance UID. It is REQUIRED.

The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded.

9.1.2.2 Optional Query Parameters

The parameters defined in this section are optional for all URI requests.

Table 9.1.2-2. Optional Query Parameters

		Usage		
Key	**Values**	**User Agent**	**Origin Server**	**Section**
contentType	media-type	O	O	8.3.3.1
charset	token	O	O	8.3.3.2

See Section 8.4.

9.1.2.2.1 Acceptable Media Types

The Accept Query Parameter specifies the Acceptable Media Types for the response payload. See Section 8.7.5. The case-sensitive name of the parameter is "contentType". Its syntax is:

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

The value of this parameter, if present, shall be either application/dicom, or one or more of the Rendered Media Types.

The DICOM Media Type transfer-syntax and character set parameters are forbidden in the request. If either are present, the response shall be 400 (Bad Request), and may include a payload containing an appropriate error message.

See Section 8.7.5 for other errors related to this parameter.

Note:

URI origin servers may support Transfer Syntax and charset Query Parameters. This is different from the approach used by the DICOM RESTful services, which uses transfer-syntax and charset media type parameters.

9.1.2.2.2 Acceptable Character Sets

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

The value of this parameter is a comma-separated list of one or more character-set identifiers. See Section 8.8.1.

9.1.3 Common Media Types

The URI resource supports both DICOM Media Types (see Sections 8.7.3) and Rendered Media Types (see Section 8.7.4).

9.2 Conformance

An implementation conforming to the URI service shall support retrieval of one or more of the Information Objects specified for the Storage Service Class, as specified in PS3.4 Annex B.5.

An implementation's Conformance Statement shall document the Information Objects supported for the URI service, and whether it plays the role of origin server or user agent, or both.

9.3 Transactions Overview

The URI Service supports two transactions:

Retrieve DICOM Instance

This transaction retrieves a single SOP Instance in the application/dicom media type.

Retrieve Rendered Instance

This transaction retrieves a single SOP Instance in a Rendered Media Type.

These two transactions have the same "requestType" type, but are differentiated by their Selected Media Type.

If there is no "contentType" Query Parameter and the Accept header field is '*/*', then the Selected Media Type defaults to 'image/jpeg' media type and the transaction defaults to Retrieve Rendered Instance.

9.4 Retrieve DICOM Instance Transaction

This transaction retrieves a single DICOM SOP Instance in the application/dicom media type. See Section 8.7.3.

9.4.1 Request

The request shall have the following syntax:

```
GET SP / ?{requestType}&{study}&{series}&{instance}
         {&accept}
         {&charset}
         {&anonymize}
         {&transferSyntax}
         SP HTTP/1.1 CRLF
Accept: uri-media-type CRLF
*(header-field CRLF)
CRLF
```

9.4.1.1 Target Resource

The Target Resource shall be an Instance of a Composite Storage SOP class defined in PS3.4.

9.4.1.2 Query Parameters

The origin server shall support Query Parameters as required in Table 9.4.1-1.

The user agent shall supply in the request Query Parameters as required in Table 9.4.1-1.

Table 9.4.1-1. Optional Query Parameters

Key	Values	Usage		Section
		User Agent	Origin Server	
anonymize	"yes"	O	O	9.4.1.2.1
annotation	"patient" "technique"	O	O	9.4.1.2.2
transferSyntax	uid	O	O	9.4.1.2.3

9.4.1.2.1 **Anonymize**

```
anonymize = %s"anonymize=" token
token = "yes"
```

This parameter specifies that the returned representations shall have all Individually Identifiable Information (III), removed, as defined in PS3.15, Annex E Basic Profile with Clean Pixel Data Option. Its name is "anonymize" and its value is a token. The defined token is "yes". If this parameter is not present, no anonymization is requested.

9.4.1.2.2 **Annotation**

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

This 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.

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.).

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

9.4.1.2.3 **Transfer Syntax**

```
transfer-syntax = %s"transferSyntax" "=" transfer-syntax-uid
```

This parameter specifies a Transfer Syntax UID. Its name is "transferSyntax" and its value is a single Transfer Syntax UID. It is optional for both the user agent and origin server. See Section 8.7.3.5 for details.

Implicit VR Little Endian and Explicit VR Big Endian Transfer Syntaxes shall not be used.

9.4.1.3 Request Header Fields

The origin server shall support header fields as required in Table 9.4.1-2 in the request.

The user agent shall supply in the request header fields as required in Table 9.4.1-2.

Table 9.4.1-2. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	media-type	O	M	The Acceptable Media Types for the response payload
Accept-Charset	charset	O	M	The Acceptable Character Sets of the response payload

See also Section 8.4.

9.4.1.4 Request Payload

The request has no payload.

9.4.2 Behavior

A success response shall contain the Target Resource(s) in an Acceptable DICOM Media Type. See Section 8.7.5.

9.4.2.1 Request Type

If the Query Parameter is not present; or if it is present with a value other than "WADO" and the origin server does not support the value, then the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

9.4.2.2 Study, Series, and Instance UIDs

If the Study, Series, or Instance UID Query Parameters are not present, if the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

9.4.2.3 Anonymize

If the Query Parameter is supported and present, and if any of the following are true:

- The number of parameter values is not equal to one
- The parameter value is not "yes"

the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

If the Target Resource has not already been de-identified, the returned Instance shall have a new SOP Instance UID.

If the origin server is either unable or refuses to anonymize the Target Resource, it may return an error response.

9.4.2.4 Annotation

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 shall ignore any unsupported parameter values.

Note

The exact nature and presentation of the annotation is determined by the origin server. The annotation is burned into the rendered image pixels.

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.

9.4.2.5 Transfer Syntax UID

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

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.

If the Query Parameter is supported and present, and if any of the following are true:

- The number of parameter values is not equal to one
- The parameter value is not a valid Transfer Syntax UID

the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

If the parameter value is a valid Transfer Syntax UID, but is not supported by the origin server, the response shall be 406 (Not Acceptable), and may include a payload containing a list of the supported transfer syntaxes.

9.4.3 Response

```
version SP status-code SP reason-phrase
[Content-Type: media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
Content-Location: uri CRLF
*(header-field CRLF)
CRLF
[payload / status-report]
```

9.4.3.1 Status Codes

Table 9.4.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 9.4.3-1 Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The Instance was successfully retrieved.
Failure	400 (Bad Request)	There was a problem with the request.
	404 (Not Found)	The resource corresponding to the UIDs in the Query Parameters was not found.

	410 (Gone)	The resource corresponding to the UIDs in the Query Parameters, once existed, but no longer exists.

9.4.3.2 Response Header Fields

Table 9.4.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	dicom-media-type	M	The media-type of the payload
Content-Length	uint	M	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	M	Shall be present if a content encoding has been applied to the payload

See Section 8.4.

9.4.3.3 Response Payload

A successful response shall have a payload containing the Target Resource in the application/dicom media type.

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

9.5 Retrieve Rendered Instance Transaction

This transaction returns a single Composite SOP Instance in a Rendered Media Type. See Section 8.7.4.

The Acceptable Media Types shall not be application/dicom. If it is, the response should be 406 (Not Acceptable) response.

9.5.1 Request

The request shall have the following syntax:

```
GET SP /?{requestType}&{study}&{series}&{instance}{&frameNumber}
        {&accept}
        {&charset}
        {&annotation}
        {&rows}
        {&columns}
        {&region}
        {&windowCenter}
        {&windowWidth}
        {&imageQuality}
        {&annotation}
        {&presentationSeriesUID}
        {&presentationUID}
         SP HTTP/1.1 CRLF
Accept: 1#media-type CRLF
*(header-field CRLF)
CRLF
```

9.5.1.1 Target Resource

The Target Resource shall be an Instance of a Composite SOP Class as defined in PS.3.3.

9.5.1.2 Query Parameters

The Query Parameters in this section shall only be included in a request if the DICOM Category of the Target Resource is Single Frame, Multi-Frame, or Video as defined in Section 8.7.2.

The origin server shall support Query Parameters as required in Table 9.5.1-1.

The user agent shall supply in the request Query Parameters as required in Table 9.5.1-1.

Table 9.5.1-1. Query Parameters

Key	Values	Usage		Section
		User Agent	Origin Server	
contentType	rendered-media-type	O	M	8.1.2.1
charset	charset	O	M	8.1.2.2
frameNumber	uint	O	O	8.3.1.2.1
imageAnnotation	"patient" / "technique"	O	O	8.3.1.2.2
rows	uint	O	O	8.3.1.2.3.1
columns	uint	O	O	8.3.1.2.3.2
region	4decimal	O	O	8.3.1.2.4
windowCenter	decimal	O	O	8.3.1.2.5.1
windowWidth	decimal	O	O	8.3.1.2.5.2
imageQuality	uint	O	O	8.3.1.2.6
presentationSeriesUID	uid	O	O	8.3.1.2.7.1
presentationUID	uid	O	O	8.3.1.2.7.2

9.5.1.2.1 Frame Number

```
frame-number = %s"frameNumber" "=" uint
```

This parameter specifies a single frame within a multi-frame image Instance, as defined in PS3.3 that shall be returned. Its name is "frameNumber" and its value shall be a positive integer (i.e. starts at 1 not 0).

9.5.1.2.2 Image Annotation

See Section 8.3.5.1.1.1.

9.5.1.2.3 Image Quality

See Section 8.3.5.1.2.

9.5.1.2.4 Viewport

The Viewport Query 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. Both parameters shall be present.

The Viewport parameters syntax in this Section override those described in Section 8.3.5.1.3.

9.5.1.2.4.1 Viewport Rows

```
rows = %s"rows" "=" uint
```

This parameter specifies the number of pixel rows in the returned image. It corresponds to the height in pixels of the user agent's viewport. Its name is "rows" and its value shall be a positive integer.

9.5.1.2.4.2 Viewport Columns

```
columns = %s"columns" "=" uint
```

This parameter specifies the number of pixel columns in the returned image. It corresponds to the width, in pixels, of the user agent's viewport. Its name is "columns" and its value shall be a positive integer.

9.5.1.2.5 Source Image Region

```
region = %s"region" "=" xmin "," ymin "," xmax "," ymax
    xmin = decimal
    ymin = decimal
    xmax = decimal
    ymax = decimal
```

This parameter specifies a rectangular region of the Target Resource. Its name is "region" and its values shall be a comma-separated list of four positive decimal numbers:

xmin the left column of the region

ymin the top row of the region

xmax the right column of the region

ymax the bottom row of the region

The region is specified using a normalized coordinate system relative to the size of the original image matrix, measured in rows and columns. Where

0.0, 0.0 corresponds to the top row and left column of the image, and

1.0, 1.0 corresponds to the bottom row and right column of the image.

and

0.0 <= xmin < xmax <= 1.0

0.0 <= ymin < ymax <= 1.0

This parameter when used in conjunction with one of the viewport parameters, allows the user agent to map a selected area of the source image into its viewport.

9.5.1.2.6 Windowing

The Windowing parameters are optional; however, if either is present, both shall be present.

The Windowing and Presentation State parameters shall not be present in the same request.

9.5.1.2.6.1 Window Center

```
window-center = %s"windowCenter" "=" decimal
```

This parameter specifies the Window Center of the returned image as defined in PS3.3. Its name is "windowCenter" and its value shall be a decimal number.

9.5.1.2.6.2 Window Width

```
window-width = %s"windowWidth" "=" decimal
```

This parameter specifies the Window Width of the returned image as defined in PS3.3. Its name is "windowWidth" and its value shall be a decimal number.

9.5.1.2.7 Presentation State

The parameters below specify the Series and Instance UIDs of a Presentation State. They are optional; however, if one is present, they shall both be present.

If the Presentation State parameters are present, then only the Annotation, Image Quality, and Viewport parameters may also be present.

9.5.1.2.7.1 Presentation Series UID

```
presentation-series = %s"presentationSeriesUID" "=" uid
```

This parameter specifies the Series containing the Presentation State Instance to be used to render the image. Its name shall be "presentationSeriesUID" and its value shall be a Series Instance UID.

9.5.1.2.7.2 Presentation UID

```
presentation-instance = %s"presentationUID" "=" uid
```

This parameter identifies the Presentation State Instance, which is used to render the image. Its name is "presentationUID" and its value shall be a Presentation State SOP Instance UID of a Presentation State Instance.

9.5.1.3 Request Header Fields

The origin server shall support header fields as required in Table 9.5.1-2.

The user agent shall supply in the request header fields as required in Table 9.5.1-2.

Table 9.5.1-2 Request Header Fields

Name	Value	Usage	Description

		User Agent	Origin Server	
Accept	media-type	M	M	The Acceptable Media Types for the response payload
Accept-Charset	charset	O	M	List of one or more character sets

The Acceptable Media Types shall contain only Rendered Media Types. See Section 8.7.4.

9.5.1.4 Request Payload

The request message has no payload.

9.5.2 Behavior

A success response shall contain the Target Resource(s) in an Acceptable Rendered Media Type. See Section 8.7.4.

The Target Resource shall be rendered as specified in the Query Parameters, by applying the transformations using the appropriate rendering pipeline specified in PS3.4, Section N.2. See Section 7.9.4 for details.

Even if the output of the image is defined in P-Values (grayscale values intended for display on a device calibrated to the DICOM Grayscale Standard Display Function PS3.14), or contains an ICC profile, the grayscale or color space for the rendered image is not defined by this Standard.

9.5.2.1 Frame Number

If the Query Parameter is supported and is present in the request, and if any of the following are true:

- The Target Resource is not a multi-frame image
- The number of parameter values is not equal to one
- The parameter value is not an unsigned integer greater than 0 and less than or equal to the number of frames in the Instance.

the origin server shall return a 400 (Bad Request) response, and may include a payload containing an appropriate error message.

9.5.2.2 Viewport

If both rows and columns Query Parameters are specified, then each shall be interpreted as a maximum, and a size will be chosen for the returned image within these constraints, maintaining the original aspect ratio.

If the rows Query Parameter is absent and the columns Query Parameter is present, the number of rows in the returned image shall be chosen to maintain the original aspect ratio.

If the columns Query Parameter is absent and the rows Query Parameter is present, the number of columns in the returned image shall be chosen to maintain the original aspect ratio.

If both Query Parameters are absent, the image (or selected region) is returned with its original size (or the size of the presentation state applied to the image), resulting in one pixel in the returned image for each pixel in the original image.

9.5.2.3 Source Image Region

If the Query Parameter is not supported or is not present, the original image shall be returned.

If the Query Parameter is supported:

- An image matrix corresponding to the specified region shall be returned with its size corresponding to the specified normalized coordinate values.
- If the Presentation UID Query Parameter is present, the corresponding presentation state shall be applied before selecting the region.

If the Query 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.

If there are greater or fewer than four parameter values present or if any of the parameters do not conform to the requirements specified in 9.5.1.2.5, the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

9.5.2.4 Windowing

If any of the following are true:

- Only one of the parameters is present
- If either of the parameter values is not a decimal number
- If Presentation Series UID or the Presentation UID Query Parameters are present

the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

9.5.2.5 Presentation State

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.

If any of the following are true:

- The Frame Number, Source Image Region, or Windowing parameters are present
- The Presentation Series UID does not correspond to an existing Presentation Series on the origin server
- The Presentation UID does not correspond to an existing Presentation SOP Instance on the origin server

the origin server shall return a 400 (Bad Request) response and may include a payload containing an appropriate error message.

9.5.3 Response

```
version SP status-code SP reason-phrase
[Content-Type: rendered-media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: uri CRLF]
*(header-field CRLF)
CRLF
[payload / status-report]
```

9.5.3.1 Status Codes

Table 9.5.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 9.5.3-1: Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	All Instances were successfully retrieved.
Failure	400 (Bad Request)	There was a problem with the request.

9.5.3.2 Response Header Fields

Table 9.5.3-2. Response Header Fields

Name	Value	Origin Server	Description

		Usage	
Content-Type	media-type	C	Shall be present if the response contains a payload. See Section 8.4.3.
Content-Encoding	encoding	C	Shall be present if the response payload has a content encoding. See Section 8.4.3.
Content-Length	uint	C	Shall be present if the response payload does not have a content encoding. See Section 8.4.3.
Content-Location	uri	C	Shall be present if the response has a payload containing a resource. See Section 8.4.3.

See also Section 8.4.

9.5.3.3 Response Payload

A success response shall contain a single rendered image encoded in the Selected Media Type.

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

10 Studies Service and Resources

10.1 Overview

The Studies Resource enables a user agent to store, retrieve, update, and search an origin server for DICOM Studies, Series, and Instances – along with their /metadata, /rendered, and /thumbnail variants; as well as Frames and Bulkdata.

The Retrieve transaction of this Service is referred to as WADO-RS. The Store transaction of this Service is referred to as STOW-RS. The Search transaction of this Service is referred to as QIDO-RS. See Section 10.3.

10.1.1 Resource Descriptions

The Studies Service manages a collection of DICOM Study resources. Each Study is organized in a hierarchy of sub-resources that correspond to the DICOM Information Model. See PS3.3, Section 7.

There are three top level resources:

`/studies` references all studies managed by the service.

`/series` references all Series managed by the service.

`/instances` references all Instances managed by the service.

The following URI Template variables are used in resource definitions in this Section.

`{study}` the Study UID of a Study managed by the Studies Service.

`{series}` the Series UID of a Series contained within a Study resource.

`{instance}` the SOP Instance UID of an Instance contained within a Series resource.

`{frames}` a comma-separated list of frame numbers, in ascending order, contained within an Instance.

`{/bulkdata}` an opaque URI that references a Bulkdata Value.

The Studies Service defines the following resources:

Table 10.1-1. Resources and Descriptions

Resource	Description
Studies Service	The Base URI of the Studies Service.
All Studies	The All Studies resource references the entire collection of studies contained in the Studies Service. All Studies Service resources begin with this resource.
Study	The Study resource references a single Study.
Study Metadata	The Study Metadata resource references the Metadata of a single Study.
Rendered Study	The Study Rendered resource references a Study to be rendered.
Study Thumbnail	The Study Thumbnail resource references a thumbnail image of a Study.
Study's Series	The Study's Series resource references the collection of all Series contained in a Study.
Study's Instances	The Study's Instances resource references the collection of all Instances in a single Study.
All Series	The All Series resource references the collection of all Series in all studies contained in the Studies Service.
Series	The Series resource references a single Series.
Series Metadata	The Series Metadata resource contains the Metadata of a single Series in a Study.
Rendered Series	The Series Rendered resource references a Series to be rendered.
Series Thumbnail	The Series Thumbnail resource references a thumbnail image of a Series.
Series' Instances	The Series' Instances resource references the collection of all Series in a single Study.
All Instances	The All Instances resource references the collection of all Instances in all Series in all studies contained in the Studies Service.
Instance	The Instance resource references a single Instance.
Instance Metadata	The Instance Metadata resource contains the Metadata of a single Instance in a Series.
Rendered Instance	The Instance Rendered resource references an Instance to be rendered.

Instance Thumbnail	The Instance Thumbnail resource references a thumbnail image of an Instance.
Frames	The Frames resource references an ordered collection of frames in a single multi-frame Instance.
Rendered Frames	The Frames resource references an ordered collection of frames in a single multi-frame Instance.
Frame Thumbnail	The Frame Thumbnail resource references a thumbnail image for frames within an Instance.
Bulkdata	The Bulkdata resource contains one or more Bulkdata Values.

10.1.2 Common Query Parameters

The origin server shall support Query Parameters as required in Table 10.1.2-1.

The user agent shall supply in the request Query Parameters as required in Table 10.1.2-1.

Table 10.1.2-1. Common Query Parameters

Name	Value	Usage		Section
		User Agent	Origin Server	
Accept	media-type	M	M	8.3.3.1
Accept-Charset	charset	O	M	8.3.3.2

See also Section 8.4.

10.1.3 Common Media Types

Each Transaction in this service supports a different set of media types. See each transaction section for details.

10.2 Conformance

An origin server claiming conformance to the Retrieve Transaction of the Studies Service:

- shall support the Retrieve Capabilities Transaction (See 8.9.1)
- shall support the Retrieve Transaction for all mandatory resources in Table 10.1-1

An origin server claiming conformance to the Store Transaction of the Studies Service:

- shall support the Retrieve Capabilities Transaction (See 8.9.1)
- shall support the Store Transaction for all mandatory resources in Table 10.1-1

An origin server claiming conformance to the Search Transaction of the Studies Service:

- shall support the Retrieve Capabilities Transaction (See 8.9.1)
- shall support the Search Transaction for all mandatory resources in Table 10.1-1

The user agent may support any of the transactions for any of the corresponding resources in Table 10.1-1.

10.3 Transactions Overview

The Studies Service consists of the following transactions:

Table 10.3-1. Studies Service Transactions

Transaction Name	Method	Payload		Description
		Request	Success Response	
Retrieve	GET	N/A	Instance(s)	Retrieve one or more representations of DICOM Resources.
Store	POST	Instance(s)	Store Instances Response Module	Stores one or more representations of DICOM Resources, contained in the request payload, in the location referenced by the Target Resource.
Search	GET	N/A	Result(s)	Searches the Target Resource for DICOM objects that match the search parameters and returns a list of matches in an Acceptable Media Type.

In Table 10.3-2, the Target Resources permitted for each transaction are marked with M if support is mandatory for the origin server and O if it is optional". A blank cell indicates that the resource is not allowed in the transaction.

Table 10.3-2. Resources by Transaction

Resource	Retrieve	Store	Search
Studies Service			
All Studies		M	M
Study	M	M	M
Study Metadata	M		
Study Bulkdata	M		
Rendered Study	M		
Study Thumbnail	O		
Study's Series			M
Study's Instances			M
All Series			M
Series	M		M
Series Metadata	M		
Series Bulkdata	M		
Series' Instances			M
Rendered Series	M		
Series Thumbnail	O		
All Instances			M
Instance	M		M
Instance Metadata	M		
Instance Bulkdata	M		
Rendered Instance	M		
Instance Thumbnail	O		
Frames	M		
Rendered Frames	M		
Frames Thumbnail	O		
Bulkdata	M	M	

10.4 Retrieve Transaction

This Transaction uses the GET method to retrieve the Target Resource. The media type in the response payload will depend on the Target URI and the Query Parameters; for example, binary DICOM instances, Metadata in DICOM+JSON, or rendered JPEG images.

The retrieve transaction supports DICOM, Rendered, and Thumbnail resources.

10.4.1 Request

The request shall have the following syntax:

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

Where `parameter` is one of the Query Parameters defined for the Target Resource in Section 11.4.1.2.

10.4.1.1 Target Resources

10.4.1.1.1 DICOM Resources

Table 10.4.1-1 defines the resources used to retrieve DICOM SOP Instances.

Table 10.4.1-1. Retrieve Transaction DICOM Resources

Resource	URI Template
Study	`/studies/{study}`
Series	`/studies/{study}/series/{series}`

Instance	/studies/{study}/series/{series}/instances/{instance}
Frames	/studies/{study}/series/{series}/instances/{instance}/frames/{frames}
Bulkdata	/{/bulkdata}

10.4.1.1.2 **Metadata Resources**

Table 10.4.1-2 defines the resources used to retrieve the metadata contained in DICOM SOP Instances.

Table 10.4.1-2. Retrieve Transaction Metadata Resources

Resource	URI Template
Study Metadata	/studies/{study}/metadata
Series Metadata	/studies/{study}/series/{series}/metadata
Instance Metadata	/studies/{study}/series/{series}/instances/{instance}/metadata

The Metadata Resources are used to retrieve the DICOM instances with bulkdata removed. The Metadata returned for a study, series, or instance resource includes all attributes in the resource. For Data Elements having a Value Representation (VR) of DS, FL, FD, IS, LT, OB, OD, OF, OL, OW, SL, SS, ST, UC, UL, UN, US, and UT, the origin server is permitted to replace the Value Field of the Data Element with a Bulkdata URI. The user agent can use the Bulkdata URI to retrieve the original Value Field of that Data Element.

10.4.1.1.3 **Rendered Resources**

A Retrieve Transaction on a Rendered resource will return a response that contain representations of a DICOM Resource rendered as appropriate images, videos, text documents, or other representations. Its primary use case is to provide user agents with a simple means to display medical images and related documents, without requiring deep knowledge of DICOM data structures and encodings.

A Rendered resource contains one or more rendered representations, i.e., in a Rendered Media type, of its parent DICOM Resource. Table 10.4.1-3 shows the Rendered resources supported by the Retrieve transaction along with their associated URI templates.

Table 10.4.1-3. Retrieve Transaction Rendered Resources

Target Resource	URI Template
Rendered Study	/studies/{study}/rendered
Rendered Series	/studies/{study}/series/{series}/rendered
Rendered Instance	/studies/{study}/series/{series}/instances/{instance}/rendered
Rendered Frames	/studies/{study}/series/{series}/instances/{instance}/frames/{frames}/rendered

The origin server shall be able to render all valid Instances of the Composite SOP classes for which conformance is claimed, e.g., origin server shall be able to render all Photometric Interpretations that are defined in the IOD for that SOP class.

The content type of the response payload shall be a Rendered Media Type. See Section 8.7.4.

10.4.1.1.4 **Thumbnail Resources**

A Thumbnail URI references a Thumbnail that is a representation of its parent DICOM Resource.

Table 11.4.1-4 shows the Thumbnail resources supported by the Retrieve transaction along with their associated URI templates. The URL of the Thumbnail resource is created by adding "/thumbnail" to the URL of the parent DICOM Resource.

Table 11.4.1-4. Retrieve Transaction Thumbnail Resources

Target Resource	Resource URI Template
Study Thumbnail	/studies/{study}/thumbnail
Series Thumbnail	/studies/{study}/series/{series}/thumbnail
Instance Thumbnail	/studies/{study}/series/{series}/instances/{instance}/thumbnail
Frame Thumbnail	/studies/{study}/series/{series}/instances/{instance}/frames/{frames}/thumbnail

The representation returned in the response to a Retrieve Thumbnail resource request shall be in a Rendered Media Type. The Thumbnail shall not contain any Patient Identifying Information. Only a single image shall be returned.

If the origin server supports any of the Thumbnail resources, it shall support all of them.

The origin server will determine what constitutes a meaningful representation.

The origin server may return a redirection response (HTTP status code 302) to a rendered resource instead of returning a rendered image.

There is no requirement that Thumbnail resources be related to any Icon Image Sequence (0088,0200) encoded in SOP Instances or returned in query responses.

10.4.1.2 Query Parameters

The origin server shall support Query Parameters as required in Table 10.4.1-5.

The user agent shall supply in the request Query Parameters as required in Table 10.4.1-5.

10.4.151. Query Parameters by Resource

Key	Resource Category	Usage		Section
		User Agent	Origin Server	
accept	All	O	M	8.3.3.1
charset	Text	O	M	8.3.3.2
annotation	Rendered	O	M	8.3.5.1.1
quality	Rendered	O	M	8.3.5.1.2
viewport	Rendered	O	M	8.3.5.1.3
	Thumbnail	O	O	
window	Rendered	O	M	8.3.5.1.4

10.4.1.3 Request Header Fields

The origin server shall support header fields as required in Table 10.4.1-6 in the request.

The user agent shall supply in the request header fields as required in Table 10.4.1-6.

Table 10.4.1-6. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	media-type	M	M	The Acceptable Media Types of the response payload
Accept-Charset	charset	O	M	The Acceptable Character Sets of the response payload

See also Section 8.4.

10.4.1.4 Request Payload

The request shall have no payload.

10.4.2 Behavior

A success response shall contain the Target Resource(s) in an Acceptable Media Type. See Section 8.7.4.

10.4.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: uri CRLF]
*(header-field CRLF)
CRLF
+[payload / status-report]
```

10.4.3.1 Status Codes

Table 10.4.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 10.4.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	All Instances were successfully retrieved.
Failure	400 (Bad Request)	There was a problem with the request.

10.4.3.2 Response Header Fields

The origin server shall supply header fields as required by Table 10.4.3-2.

Table 10.4.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	C	The media type of the payload. Shall be present if the response has a payload.

See also Section 8.4.

10.4.3.3 Response Payload

A success response shall have a payload containing one or more representations of the Target Resource in an Acceptable Media Type. The payload may be single part or multipart depending on the media type. A success response may include a Status Report.

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

Table 10.4.3.-3 shows the media type category for each resource type. The origin server shall support the default and required media types in the media type category specified.

Table 10.4.3-3. Resource Media Types

Resource	Section	Media Type Category	Section
DICOM Resources	10.5.1.1.1	DICOM Media Types	8.7.3
Metadata Resources	10.5.1.1.2	DICOM Media Types	8.7.3
Rendered Resources	10.5.1.1.3	Rendered Media Types	8.7.4
Thumbnail Resources	10.5.1.1.4	Rendered Media Types	8.7.4

10.4.4 Conformance Statement

The creator of an implementation shall document in its Conformance Statement:

- the origin server and/or user agent role(s) played,
- the resources supported for this transaction,
- the media types supported for this transaction,
- the optional Query Parameters supported,
- the optional Header Fields supported.

The creator of an implementation shall also document:

- The Composite SOP classes supported, including:
  - The Image Storage SOP classes supported
  - The Image Storage SOP classes supported by Rendered Presentation States

The creator of an origin server implementation shall also document:

- If Thumbnails are supported:
  - A description of the method used to render thumbnails for the study, series, or instance
  - The minimum and maximum sizes for thumbnails
  - Character sets supported for Thumbnail resources (if other than UTF-8).

10.5 Store Transaction

This transaction uses the POST method to Store representations of Studies, Series, and Instances contained in the request payload.

The retrieve transaction supports only DICOM resources. The DICOM resource can be supplied as Metadata along with a rendered representation.

10.5.1 Request

The request shall have the following syntax:

```
POST SP "/" {/resource} SP version CRLF
Accept: 1#media-type CRLF
Content-Type: dicom-media-type CRLF
(Content-Length: uint / Content-Encoding: encoding) CRLF
*(header-field CRLF)
CRLF
payload
```

10.5.1.1 Target Resources

10.5.1.1.1 DICOM Resources

Table 10.5.1-1 defines the resources used to store DICOM SOP Instances.

Table 10.5.1-1. Store Transaction DICOM Resources

Resource	URI Template	Description
Studies	/studies	Stores a set of representations that may have different Study Instance UIDs.
Study	/studies/{study}	Stores a set of representations that belong to the same Study, i.e., each representation shall have the same Study Instance UID.

10.5.1.2 Query Parameters

The Store transaction has no Query Parameters.

10.5.1.3 Request Header Fields

The origin server shall support Query Parameters as required in Table 10.5.1-2.

The user agent shall supply in the request Query Parameters as required in Table 10.5.1-2.

Table 10.5.1-2. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Content-Type	media-type	M	M	The DICOM Media Type of the request payload Shall be present if the response has a payload
Content-Length	uint	C	M	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	M	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

10.5.1.4 Request Payload

The request payload shall be present and shall contain one or more representations specified by the Content-Type header field.

The payload may contain Instances from more than one Study, if the Study Instance UID is not specified in the Target URI.

The request payload shall consist of either:

- DICOM PS3.10 SOP Instances, or
- bulk data accompanied by DICOM Metadata.

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

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

- 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 10.5.2-1.
- 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 10.5.2-1.
- 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)

- Uncompressed bulk data shall be encoded as application/octet-stream.

- 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.

10.5.2 Behavior

The origin server creates resources from the representations contained in the request payload.

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

This Transaction can implicitly create a new Study, Series and Instances; or add Series and/or Instances to an existing Study.

While creating resources from the representations, the origin server may coerce or replace the values of data elements. For example, Patient Name, Patient ID, and Accession Number might be coerced when importing media from an external institution, reconciling the Instances against a master patient index, or reconciling them against an imaging procedure order. The origin server may also fix incorrect values, such as Patient Name or Patient ID; for example, because an incorrect work list item was chosen, or an operator input error has occurred.

If any Attribute 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 Store Instances Response Module (see Annex I) in the response.

Note

For more information on populating the Original Attribute Sequence see PS3.3, Section C.12.1.

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 10.5.2-1.

If the request message contains compressed bulk data with a Content Type that is one of the media types specified in Table 10.5.2-1, 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.

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. Table 10.5.2-1 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 10.5.2-1 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 10.5.2-1. 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 10.5.2-1.

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).

10.5.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: media-type CRLF]
[(Content-Length: uint CRLF / Content-Encoding: encoding CRLF)]
*(header-field CRLF)
CRLF
store-instances-response-module
```

The response shall contain an appropriate status code.

The response shall include a payload containing the Store Instances Response Module as defined in Annex I.

10.5.3.1 Status Codes

Table 10.5.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 10.5.3-1. Status Code Meaning

Status	Status Code	Meaning
Success	200 (OK)	The origin server has successfully received, processed, and stored at least one of the representations contained in the request payload. The response shall include a Store Instances Response Module detailing the disposition of all representations contained in the request.
	202 (Accepted)	The origin server successfully received the request message, but may not have validated, processed, or stored the representations in the request payload. The response shall include a Store Instances Response Module detailing the disposition of all representations contained in the request.
Failure	400 (Bad Request)	The was a problem with the request. For example, the origin server did not store any of the representations contained in the request payload because of unsupported SOP Class, or Study Instance UID mismatch, or representation specific errors.
	409 (Conflict)	This origin server was unable to store any instances due to a conflict in the request (e.g., unsupported SOP Class or Study Instance UID mismatch). This may also be used to indicate that a STOW-RS Service was unable to store any instances for other reasons. Additional information regarding the instance errors can be found in the Store Instances Response Module contained in the payload.
	415 (Unsupported Media Type)	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

10.5.3.2 Response Header Fields

The origin server shall support all header fields in Table 10.5.3-2.

Table 10.5.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	M	The media type of the response payload, if present.
Content-Encoding	encoding	C	Shall be present if the response payload has a content encoding. See Section 8.4.3.
Content-Length	uint	C	Shall be present if the response payload does not have a content encoding. See Section 8.4.3.
Content-Location	url	C	Shall be present if a new resource was created. The value is the URL of the representation contained in the request payload. May be present otherwise
Location	url	C	Shall be present if a new resource was created. The value is the URL of the created resource. May be present otherwise

All success responses shall also contain the Content Representation (see Section 8.4.3) and Payload header fields (see Section 8.4.4) with appropriate values.

It is recommended that the text returned in the Warning header field (see RFC7234, Section 5.5 <https://tools.ietf.org/html/rfc7234#section-5.5>) contain a DICOM Status Code (see PS3.4 and PS3.7 Section C) and descriptive reason. For example:

```
Warning: A700 <service>: Out of memory
```

See also Section 8.4.

10.5.3.3 Response Payload

A success response payload shall contain a Store Instances Response Module. See Annex X.

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

10.5.4 Media Types

The origin server shall support the default and required media types in the media type category specified in Table 10.5.3-3.

Table 10.5.3-3. Resource Media Types

Resource Type	Media Type Category	Section
DICOM Resources	DICOM Media Type	8.7.3

10.5.5 Conformance Statement

An implementation conforming to the Store transaction shall support the resources and media types specified in this Section.

An implementation shall declare in its Conformance Statement the Information Objects supported for the Store transaction, and whether it plays the role of origin server or user agent, or both.

10.6 Search Transaction

This Transaction uses the GET method to Search for Studies, Series, and Instances managed by the origin server.

10.6.1 Request

The request shall have the following syntax:

```
GET SP "/" {/resource} {?search*} SP version CRLF
Accept: 1#search-media-type CRLF
*(header-field CRLF)
CRLF
```

Where

```
search-media-type = multipart/related; type="application/dicom+xml" / dicom-json
```

10.6.1.1 Target Resources

The Target Resource Path component of the Target URI specifies the collection of resources that is the target of the search.

An origin server that is a native implementation shall support all Mandatory (M) resources specified in the Native column in Table 10.6.1-1.

An origin server that is a DIMSE Proxy implementation shall support all Mandatory (M) resources specified in the Proxy column in Table 10.6.1-1.

Table 10.6.1-1. Search Transaction Resources

Resource	URI Template	Native	Proxy	Query Type
All Studies	`/studies{?search*}`	M	M	hierarchical
Study's Series	`/studies/{study}/series{?search*}`	M	M	hierarchical
Study's Instances	`/studies/{study}/instances{?search*}`	M	O	relational
All Series	`/series{?parameter*}`	M	O	relational
Study Series' Instances	`/studies/{study}/series/{series}/instances{?search*}`	M	M	hierarchical
All Instances	`/instances{?search*}`	M	O	relational

For more information about Hierarchical Queries see PS3.4, Section C.4.1.3.1.1. For more information about Relational Queries see PS3.4, Section C.4.1.2.2.1 and Section C.4.1.3.2.1.

Table 10.6.1-2 shows the resources supported by the Search transaction along with a description of the search performed and the results returned.

Table 10.6.1-2. Search Resource Descriptions

Resource	Description
All Studies	Searches the entire service for Studies that match the search parameters, and returns a list of matching Studies, including the default and requested attributes that are supported for each Study.
Study's Series	Searches for all Series in the specified Study that match the search parameters, and returns a list of matching Series, including the default and requested attributes that are supported for each Series.
Study's	Searches for all Instances in the specified Study that match the search parameters, and returns a list of

Instances	matching Instances, including the default and requested attributes that are supported for each Instance.
All Series	Searches the entire service for Series that match the search parameters, and returns a list of matching Series, including the default and requested attributes that are supported for each Series.
Study Series' Instances	Searches for all Instances in the specified Study and Series that match the search parameters, and returns a list of matching Instances, including the default and requested attributes that are supported for each Series.
All Instances	Searches the entire service for Instances that match the search parameters, and returns a list of matching Instances, including the default and requested attributes that are supported for each Series.

10.6.1.2 Query Parameters

The origin server shall support Query Parameters as required in Section 8.3.4-1 for the corresponding Resource Categories.

The origin server shall support Query Parameters as required in Section 8.3.4-1 for the supported Resource Categories.

10.6.1.2.1 Attribute / Value Pair Requirements

DICOM Attribute/Value pairs included as Query Parameters in the request shall satisfy the requirements in Section 8.3.4.1.

The user agent may include the following attributes in the request:

- Patient IE attributes
- Study IE attributes (only allowed if the resource is All Studies, All Series, All Instances)
- Series IE attributes (only allowed if the resource is Study's Series, All Series, Study's Instances, or All Instances)
- Composite Instance IE attributes (only allowed if the resource is Study's Instances, Study Series' Instances, or All Instances)
- Additional Query/Retrieve Attributes (see PS3.4 Section C.3.4)
- Timezone Offset From UTC (0008,0201)

The following are examples of Search URIs with valid attribute/value pairs:

```
/studies?PatientID=11235813
/studies?PatientID=11235813&StudyDate=20130509
/studies?00100010=SMITH*&00101002.00100020=11235813&limit=25
/studies?00100010=SMITH*&OtherPatientIDsSequence.00100020=11235813
/studies?PatientID=11235813&includefield=00081048,00081049,00081060
/studies?PatientID=11235813&includefield=00081048&includefield=00081049
   &includefield=00081060
/studies?PatientID=11235813&StudyDate=20130509-20130510
/studies?StudyInstanceUID=1.2.392.200036.9116.2.2.2.2162893313.1029997326.94587,
   1.2.392.200036.9116.2.2.2.2162893313.1029997326.94583
```

10.6.1.2.2 Search Key Types and Requirements

The following table defines the Search Key Types and their requirements.

Table 10.6.1-3. Search Key Types

Type	Requirement
U	Unique and Required Key
R	Required Key
C	Conditional Key
O	Optional Key

10.6.1.2.3 Required Matching Attributes

The origin server shall support the IE Levels specified in Table 10.6.1-4.

Table 10.6.1-4. Required IE Levels by Resource

	IE Level		
Resource	**Study**	**Series**	**Instance**
All Studies	X		
Study's Series		X	
Study's Instances		X	X
All Series	X	X	

Study Series' Instances			X
All Instances	X	X	X

The origin server shall support the matching attributes specified in Table 10.6.1-5 for each supported IE Level.

Table 10.6.1-5. Required Matching Attributes

IE Level	Keyword	Tag
Study	StudyDate	(0008,0020)
	StudyTime	(0008,0030)
	AccessionNumber	(0008,0050)
	ModalitiesInStudy	(0008,0061)
	ReferringPhysicianName	(0008,0090)
	PatientName	(0010,0010)
	PatientID	(0010,0020)
	StudyInstanceUID	(0020,000D)
	StudyID	(0020,0010)
Series	Modality	(0008,0060)
	SeriesInstanceUID	(0020,000E)
	SeriesNumber	(0020,0011)
	PerformedProcedureStepStartDate	(0040,0244)
	PerformedProcedureStepStartTime	(0040,0245)
	RequestAttributeSequence	(0040,0275)
	>ScheduledProcedureStepID	(0040,0009)
	>RequestedProcedureID	(0040,1001)
Instance	SOPClassUID	(0008,0016)
	SOPInstanceUID	(0008,0018)
	InstanceNumber	(0020,0013)

Note

While some of the Data Elements in Table 10.6.1-5 in are optional in PS3.4 Section C.6.2.1, the above list is consistent with those required for IHE RAD-14. See Table 4.14-1 in http://www.ihe.net/Technical_Framework/upload/IHE_RAD_TF_Vol2.pdf.

10.6.1.3 Request Header Fields

The origin server shall support header fields as required in Table 10.6.1-6 in the request.

The user agent shall supply in the request header fields as required in Table 10.6.1-6.

Table 10.6.1-6. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	media-type	M	M	The Acceptable Media Types for the response payload
Accept-Charset	charset	O	M	The Acceptable Character Sets of the response payload

See also Section 8.4.

10.6.1.4 Request Payload

The request has no payload.

10.6.2 Behavior

The origin server shall perform the search indicated by the request, using the matching rules in Section 8.3.4.

10.6.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: media-type CRLF]
[Content-Location: uri CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
*(header-field CRLF)
CRLF
[payload / status-report]
```

10.6.3.1 Status Codes

Table 10.6.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 10.6.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The search completed successfully, and the results are contained in the payload. If there are additional results available or there are warnings the Warning header field shall contain a URL referencing a Search Status report.
Failure	400 (Bad Request)	The was a problem with the request. For example, For example, the Query Parameter syntax is incorrect
	413 (Payload Too Large)	The search was too broad, and the body of the response should contain a Status Report with additional information about the failure.

10.6.3.2 Response Header Fields

The origin server shall support all header fields in Table 10.6.3-2.

Table 10.6.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	C	The DICOM Media Type of the response payload Shall be present if the response has a payload
Content-Length	Uint	C	Shall be present if no content coding has been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload

All responses with payloads shall contain the Content Representation and Payload header fields with appropriate values. See Section 78.3.

10.6.3.3 Response Payload

A success response shall contain a list of matching results in an Acceptable Media Type. See Section 8.7.4.

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

10.6.3.3.1 Study Resource

For each matching Study, the origin server response shall contain attributes in accordance with Table 10.6.3-3. The "Type" column in the table below refers to the Query/Retrieve Attribute Types defined in PS3.4, Section C.2.2.1. The unique key for a Study resource search response is the Study Instance UID (0020,000D).

Table 10.6.3-3. Study Resource Search Response Payload

Attribute Name	Tag	Type	Condition

Attribute Name	Tag	Type	Condition
Specific Character Set	(0008,0005)	C	Shall be present if known
Study Date	(0008,0020)	R	
Study Time	(0008,0030)	R	
Accession Number	(0008,0050)	R	
Instance Availability	(0008,0056)	C	Shall be present if known
Modalities in Study	(0008,0061)	R	
Referring Physician's Name	(0008,0090)	R	
Timezone Offset From UTC	(0008,0201)	C	Shall be present if known
Retrieve URL	(0008,1190)	C	Shall be present if the Instance is retrievable by the Retrieve transaction
Patient's Name	(0010,0010)	R	
Patient ID	(0010,0020)	R	
Patient's Birth Date	(0010,0030)	R	
Patient's Sex	(0010,0040)	R	
Study Instance UID	(0020,000D)	U	
Study ID	(0020,0010)	R	
Number of Study Related Series	(0020,1206)	R	
Number of Study Related Instances	(0020,1208)	R	

Note

While some of the above Attributes are optional in PS3.4 Table C.6-1, they are consistent with those required in IHE Radiology Technical Framework Vol. 2, Table 4.14-1 (see http://www.ihe.net/Technical_Framework/upload/IHE_RAD_TF_Vol2.pdf).

In addition, the response shall contain:

- All other Study level attributes passed as match or include parameters in the request that are supported by the origin server
- If the include parameter has been specified in the request, and its value is "all", all available Study Level attributes

The origin server shall not return any Series or Instance Level attributes contained in include parameters.

10.6.3.3.2 Series Resources

For each matching Series, the origin server shall return all attributes listed in Table 10.6.3-4. The "Type" column in the table below refers to the Query/Retrieve Attribute Types defined in PS3.4, Section C.2.2.1. The unique key for a Series resource search response is the Series Instance UID (0020,000E).

Table 10.6.3-4. Series Resources Search Response Payload

Attribute Name	Tag	Type	Condition
Specific Character Set	(0008,0005)	C	Shall be present if known
Modality	(0008,0060)	R	
Timezone Offset From UTC	(0008,0201)	C	Shall be present if known
Series Description	(0008,103E)	C	Shall be present if known
Retrieve URL	(0008,1190)	R	Shall be present if the Instance is retrievable by the Retrieve transaction
Series Instance UID	(0020,000E)	U	
Series Number	(0020,0011)	R	
Number of Series Related Instances	(0020,1209)	R	

Attribute Name	Tag	Type	Condition
Performed Procedure Step Start Date	(0040,0244)	C	Shall be present if known
Performed Procedure Step Start Time	(0040,0245)	C	Shall be present if known
Request Attribute Sequence	(0040,0275)	C	Shall be present if known
>Scheduled Procedure Step ID	(0040,0009)	R	
>Requested Procedure ID	(0040,1001)	R	

Note

While some of the above Attributes in are optional in PS3.4, Table C.6-1, they are consistent with the those required in IHE Radiology Technical Framework Vol. 2, Table 4.14-1 (see http://www.ihe.net/Technical_Framework/upload/IHE_RAD_TF_Vol2.pdf).

In addition, the response shall contain:

- All other Series Level attributes passed as match or include parameters in the request that are supported by the origin server.
- If the "includefield" parameter has been specified in the request and its value is "all",
  - If the resource is Study's Series include all available Series Level attributes
  - If the resource is All Series, include all available Study and Series Level attributes
- If the Target Resource is the All Series resource, then include all Study level attributes specified in Section 11.6.3.3.1.

Instance Level attributes contained in include parameters shall not be returned.

10.6.3.3.3 Instance Resources

For each matching Instance, the origin server shall return all attributes listed in Table 10.6.3-5, if present in the Instance. The Type column in the table below refers to the Query/Retrieve Attribute Types defined in PS3.4, Section C.2.2.1. The unique key for an Instance resource search response is the SOP Instance UID (0008,0018).

Table 10.6.3-5. Instance Resources Search Response Payload

Attribute Name	Tag	Type	Condition
Specific Character Set	(0008,0005)	C	Shall be present if known
SOP Class UID	(0008,0016)	R	
SOP Instance UID	(0008,0018)	U	
Instance Availability	(0008,0056)	C	Shall be present if known
Timezone Offset From UTC	(0008,0201)	C	Shall be present if known
Retrieve URL	(0008,1190)	R	Shall be present if the Instance is retrievable by the Retrieve transaction
Instance Number	(0020,0013)	R	
Rows	(0028,0010)	C	Shall be present if known
Columns	(0028,0011)	C	Shall be present if known
Bits Allocated	(0028,0100)	C	Shall be present if known
Number of Frames	(0028,0008)	C	Shall be present if known

Note

While some of the above Attributes in are optional in PS3.4, Table C.6-1, they are consistent with the those required in IHE Radiology Technical Framework Vol. 2, Table 4.14-1 (see http://www.ihe.net/Technical_Framework/upload/IHE_RAD_TF_Vol2.pdf).

In addition, the response shall contain:

- All other Instance Level attributes passed as match or include parameters that are supported by the origin server.
- if the "includefield" parameter has been specified in the request and its value is "all",
  - If the resource is Study's Instances include all available Series Level attributes
  - If the resource is Study Series' Instances, include all Instance Level attributes
  - If the resource is All Instances, include all available Study and Series Level attributes

- If the Target Resource is the All Instances or Series' Instances resource, then include all Study level attributes specified in Section 10.6.3.3.1.
- If the Target Resource is the All Instances or Study's Instances resource, then include all Series level attributes specified in Section 10.6.3.3.2.

10.6.4 Media Types

The origin server shall support the following media types:

Table 10.6.4-1. Default, Required, and Optional Media Types

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

10.6.5 Conformance Statement

An implementation supporting the Search transaction shall declare it in its Conformance Statement and include whether it plays the role of origin server or user agent, or both.

An implementation playing the role of origin server shall declare the maximum number of matches supported for a single query.

An implementation playing the role of origin server shall declare its support for the following in its Conformance Statement:

- Whether it is a native or proxy implementation
- Fuzzy -matching
- Paging limit / offset
- Optional resources supported
- Optional Attributes supported

11 Worklist Web Service and Resources

11.1 Overview

The Worklist Service, also referred to as UPS-RS, defines a RESTful interface to the Unified Procedure Step Service SOP Classes defined in PS3.3, Section B.26 and PS3.4, Section CC.

The Worklist Service manages a single Worklist containing one or more Workitems. Each Workitem represents a procedure step. User agents and origin servers can create, retrieve, update, search for, and change the state of Workitems. See PS3.17, Section GGG for an in-depth overview of Worklists and Workitems (UPS Instances).

11.1.1 Resource Description

There are three resources defined by this service:

workitems	A list of Workitems managed by the origin server.
workitem	A dataset containing the attributes specified in PS3.4, Section 2.5.1.3.
subscription	A resource that specifies a Subscriber, to whom notifications about changes in the resource's state should be sent.

The Affected SOP Instance UID and Requested SOP Instance UID are not part of the dataset and shall not be contained in the payload.

Note:

The Workitem UID is the same as the Affected SOP Instance UID and Requested SOP Instance UID.

The URI Templates defined by this service are specified in Table 11.1.1-1.

Table 11.1.1-1. Worklist Resources

Resource	URI Template	Description
Worklist	/workitems	The URL of the root Worklist Service
Workitem	/workitems/?{workitem}	The URL of a Workitem
Subscription	/workitems/{resource}/subscribers/{aetitle}	The URL of a Subscription

The Worklist Service manages a DICOM UPS Worklist and the Workitems. It contains the following resources:

Table 11.1.1-2. Resources, URI Templates and Descriptions

Resource	URI Template and Description
Worklist	/ The Base URI of the Worklist Service. A Worklist contains and manages a list of Workitems. There is only one Worklist per service.
Workitems	`/workitems` The Workitems resource contains the entire collection of workitems in the Worklist.
Workitem	`/workitems/{workitem}` The Workitem resource contains a single Workitem that is identified by its SOP Instance UID. A Workitem describes a unit of work (aka a Procedure Step).
Workitem State	`/workitems/{workitem}/state` The Workitem State resource is used to change the state of a Workitem.
Workitem Request Cancellation	`/workitems/{workitem}/cancelrequest` The Workitem Cancel resource is used to request the cancellation of a Workitem.
Workitem Subscription	`/workitems/{workitem}/subscribers/{aetitle}` The Subscription to a Workitem.
Worklist Subscription	`/workitems/1.2.840.10008.5.1.4.34.5/subscribers/{aetitle}` The Workitem Subscription resource contains a Subscription to the Worklist

Filtered Worklist Subscription	`/workitems/1.2.840.10008.5.1.4.34.5.1/subscribers/{aetitle}` The Workitem Subscribers resource contains a single Worklist Subscriber.

The following URI Template variables are used in the definitions of the resources throughout Section 11.

`{workitem}` the Instance UID of a Workitem. Where uid corresponds to the Affected SOP Instance UID.

`{aetitle}` The Application Entity Title of a Subscriber.

11.1.1.1 Workitems

The Workitem embodies a one to one relationship between the scheduled procedure step and the performed procedure step performed. See PS3.3, Annex CC.

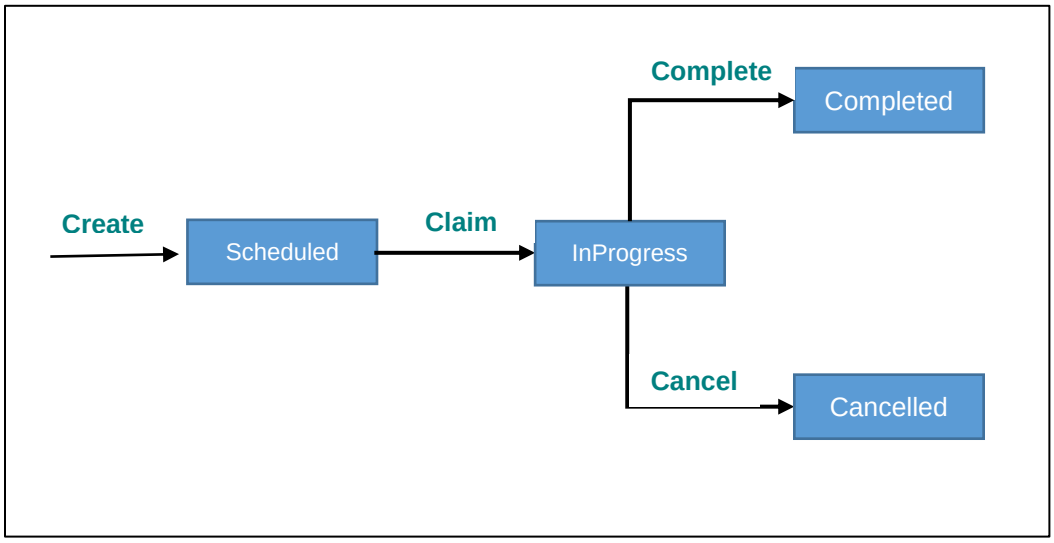
Workitems may be used to represent a variety of tasks such as: Image Processing, Quality Control, Computer Aided Detection, Interpretation, Transcription, Report Verification, or Printing. The tasks may or may not be formally scheduled.

11.1.1.2 Workitem States and Transitions

Figure 11.1.1-1, shows the four states of a Workitem, along with the legal state transitions. All state transitions are atomic, that is, the transition from one state to another happens instantaneously from the point of view of any user agent. Once created a Workitem is always in some state. It is never between one state and another.

The origin server shall process all requests atomically, that is, once the origin server begins processing the request, all the updates shall be applied to the target Workitem before any other transaction may access the modified Workitem.

Figure 11.1.1-1. Workitem Transactions and State Transitions



The details of all state transition requirements can be found in PS3.4, Section CC.1.1<http://dicom.nema.org/medical/dicom/current/output/html/part04.html#sect_CC.1.1>. See PS3.4 Table CC.1.1-2<http://dicom.nema.org/medical/dicom/current/output/html/part04.html#table_CC.1.1-2> for the complete state table including error codes.

All the Change State transactions require a Transaction UID in the payload.

All N-SET Actions are applied before the state is changed.

The requirements for moving from one state to another are specified in PS3.4, Section CC.2.5.1.1 UPS Final State Requirements.

11.1.1.3 Deletion Locks and Workitem Lifetime

The Worklist Service supports Reliable Watchers (see PS3.17, Section GGG.1) by allowing a user agent to request a Deletion Lock, using the Deletion Lock Query Parameter (see Section 11.1.1.3), when it Subscribes to a Worklist or Workitem.

The Deletion Lock is released when the user agent Unsubscribes from the Worklist or Workitem.

The origin server will not delete a Workitem in the Completed or Cancelled state until all Deletion Locks on that Workitem have been released. A Workitem that has moved to the Completed or Cancelled state and has no Deletion Locks may be deleted by the origin server. If the origin server has determined that the user agent that owns the deletion lock no longer has a connection, then the origin server may remove the lock if the user agent doesn't reconnect in a origin server defined amount of time. The amount of time should be specified in the Conformance Statement.

Once a Workitem has been deleted a request with that deleted Workitem as the Target Resource will receive a 410 (Gone) response, if the origin server knows that the Workitem did exist, but has been deleted; otherwise, a 404 (Not Found) response. See Section CC.2.1.3 in PS3.4.

11.1.1.4 Subscriptions and Notifications

A user agent can request notification of certain events related to the Worklist or Workitems. To receive notifications, the user agent shall first create a Notification Connection between itself and the origin server using the Open Notification Connection transaction. See Section 8.9.2.

Once the Notification Connection has been opened the user agent can create Subscriptions to the Worklist as a whole, or to individual Workitems. Once a Subscriber has created a Subscription, the origin server will notify the Subscriber about events related to that Subscription. Each notification contains an Event Report related to the Subscription's resource.

While a Workitem Subscription provides the Subscriber with notifications related to the Workitem, a Worklist Subscription is different; it is a Subscription Generator, which specifies that the origin server should create a Subscription, on the Subscriber's behalf, to each new Workitem as it is created. When creating a Worklist Subscription, the user agent may specify a Filter Query Parameter. The origin server applies the Filter to each newly created Workitem, and if the Filter is satisfied, the origin server creates a Subscription to the new Workitem on behalf of the Subscriber. The Filter Query Parameter specifies a comma-separated list of attribute/value pairs. See Section 8.3.4.1.

For each Worklist Subscription, the origin server will continue to create Subscriptions for new Workitems as they are created until the Subscriber Unsubscribes from the Worklist or closes its connection.

If the Notification Connection is lost at any point, the user agent can re-establish it by repeating the Open Notification Connection Transaction.

The state of the Notification Connection does not affect Subscriptions. An origin server may queue Event Reports when the connection is down but is not required to do so.

Note

A user agent will only receive the initial state of a newly-subscribed Workitem if the Notification Connection was initiated before creating the subscription

11.1.1.5 Web Services and DIMSE Terminology

Table 11.1.1-3. Correspondence between RESTful and DIMSE Terminology

RESTful Term	DIMSE Term
Worklist	Worklist
Workitem	UPS Instance
Deletion Lock	Deletion Lock
Filter	Matching Keys
Matching Key	Matching Key
Subscribe	Subscribe
Unsubscribe	Unsubscribe
Subscription	Subscription
Subscription Generator	Global Subscription Filtered Global Subscription
Subscriber	Subscriber
Suspend Subscription	Suspend Global Subscription
Notification Connection	Association
Transaction	N-GET, N-SET, N-ACTION
Notification	N-EVENT-REPORT

11.1.2 Common Query Parameters

The origin server shall support Query Parameters as required in Table 11.1.2-1.

The user agent shall supply in the request Query Parameters as required in Table 11.1.2-1.

Table 11.1.2-1. Common Query Parameters

Name	Value	Usage		Section
		User Agent	Origin Server	
Accept	media-type	M	M	8.3.3.1
Accept-Charset	charset	O	M	8.3.3.2

See also Section 8.4.

11.1.3 Common Media Types

The following media types shall be supported by all transactions in the Worklist Service:

- application/dicom+json
- application/dicom+xml

The transactions shall not support Metadata or Bulkdata objects.

The origin server shall support the following media types:

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

  Specifies that the payload is a multipart message body where each part is a DICOM PS3.19 XML DICOM Native Model element containing the appropriate Workitem attributes. See PS3.19 Section A.1.

- application/dicom+json

  Specifies that the payload is a JSON array containing Workitems, and each Workitem contains the appropriate attributes. See Section F.2.

11.2 Conformance

An origin server shall support all transactions of this service. Additional requirements for an origin server that is also a Unified Worklist and Procedure Step SCP are described in Section CC.1 in PS3.4.

A user agent or origin server implementing the Worklist Service shall comply with all requirements placed on the SCU and/or SCP for the corresponding services in PS3.4 Section CC including Conformance Statement requirements.

An implementation supporting the Worklist Service shall describe its support in its Conformance Statement and in its response to the Retrieve Capabilities transaction, and whether it plays the role of origin server, user agent, or both.

11.3 Transactions Overview

The Worklist Service consists of the Transactions in Table 11.3-1.

Table 11.3-1. Worklist Service Methods and Resource Templates

Transaction	Method	Payload		Description
		Request	Success Response	
Create	POST	dataset		Creates a new Workitem
Retrieve	GET		dataset	Retrieves the Target Workitem
Update	POST	dataset		Updates the Target Workitem
Change State	PUT			Changes the state of the Target Workitem
Request Cancellation	POST	dataset		Request that the origin server cancel a Workitem

Search	GET		results	Searches for Workitems
Subscribe	POST			Creates a Subscription to the Target Worklist or Target Workitem
Unsubscribe	DELETE			Cancels a Subscription from the Target Worklist or Target Workitem

The details of all state transition requirements can be found in PS3.4, Section CC.1.1.

The Request Cancellation transaction does not perform an actual state transition, but it might cause a state transition.

When creating a new workitem, to convey the Workitem UID that is to be assigned, DIMSE uses the Affected SOP instance UID in the DIMSE header. In the Web Services, the Workitem UID is included as a Query Parameter to the Create request. All attributes in the HTTP transaction payloads are the same as those in the DIMSE payload.

11.4 Create Workitem Transaction

This transaction creates a Workitem on the target Worklist. It corresponds to the UPS DIMSE N-CREATE operation.

11.4.1 Request

The request shall have the following syntax:

```
POST SP /workitems ?workitem SP version CRLF
Accept: dicom-media-type CRLF
Content-Type: dicom-media-type CRLF
(Content-Length: uint / Content-Encoding: encoding) CRLF
*(header-field CRLF)
CRLF
Workitem
```

The user agent shall conform to the SCU behavior specified in PS3.4, Section CC.2.5.2.

11.4.1.1 Target Resources

The Target Resource is either the Worklist, or a Workitem.

Table 11.4.1-1. Create Transaction Resources

Resource	URI Template
Worklist	/workitems
Workitem	/workitems?{workitem}

If the Target Resource is the Worklist, then the payload shall contain one Workitem dataset.

The value of the `workitem` Query Parameter is the Workitem UID.

11.4.1.2 Query Parameters

The origin server shall support Query Parameters as required in Table 11.1.2.

The user agent shall supply in the request Query Parameters as required in Table 11.1.2.

11.4.1.3 Request Header Fields

The origin server shall support all header fields in Table 11.4.1-3.

The user agent shall supply in the request header fields as defined in Table 11.4.1-3.

Table 11.4.1-3. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Content-Type	dicom-media-type	M	M	The media-type of the payload
Content-Length	uint	C	M	Shall be present if a content encoding has not been applied to the payload

Content-Encoding	encoding	C	M	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

11.4.1.4 Request Payload

The payload shall have a single part, containing a Workitem encoded in the media type specified in the Content-Type header field. The payload shall contain all data elements to be stored. The Affected SOP Instance UID shall not be present in the Workitem dataset.

The Workitem in the payload shall comply with all Instance requirements in the Req. Type N-CREATE column of PS3.4, Table CC.2.5-3.

11.4.2 Behavior

The origin server shall create a new Workitem in the Scheduled state and return a URL referencing the newly created Workitem in the Location header field of the response. A Workitem will only be added to a Worklist once.

The origin server shall create and maintain the Workitem as specified by the SCP behavior defined in PS3.4 Section CC.2.5.3.

11.4.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
Content-Location: representation CRLF
Location: resource CRLF
*(header-field CRLF)
CRLF
[status-report]
```

11.4.3.1 Status Codes

Table 11.4.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.4.3-1. Status Code Meaning

Status	Code	Meaning
Success	201 (Created)	The Target Workitem was successfully added to the Worklist.
Failure	400 (Bad Request)	There was a problem with the request. For example, the request payload did not satisfy the requirements of the Req. Type N-CREATE column of PS3.4, Table CC.2.5-3.
	409 (Conflict)	The Target Workitem already exists.

11.4.3.2 Response Header Fields

Table 11.4.3-2. Response Header Fields

Names	Value	Origin Server Usage	Condition
Content-Type	media-type	C	Shall be present if the response has a payload
Content-Length	uint	C	Shall be present if a content coding has not been applied to the payload
Content-Encoding	encoding	C	Shall be present if content encoding has been applied to the payload
Content-Location	url	O	Shall be present if the response has a payload containing a resource. See Section 8.4.3. May be present otherwise
Location	url	C	A URL-reference to the created Workitem. Shall be present if a Workitem was created. May be present if the payload contains a resource

Warning	*see below*	C	Shall be present if the Target Workitem was modified by the origin server and shall include the warning below

If the Target Workitem was modified by the origin server, the response shall also have the following Warning header:

```
Warning: 299 <service>: The Workitem was created with modifications. CRLF
```

See also Section 8.4.

11.4.3.3 Response Payload

A success response should have no payload.

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

11.5 Retrieve Workitem Transaction

This transaction retrieves a Workitem. It corresponds to the UPS DIMSE N-GET operation.

Note:

The requirement for the origin server to respond to Retrieve Workitem requests for UPS Instances that have moved to the COMPLETED or CANCELED state is limited. See Section CC.2.1.3 in PS3.4 .

11.5.1 Request

The request shall have the following syntax:

```
GET SP /workitems/{workitem} SP version CRLF
Accept dicom-media-type CRLF
[Cache-Control: no-cache CRLF]
*(header-field CRLF)
CRLF
```

The user agent shall conform to the SCU behavior specified in PS3.4, Section CC.2.7.2.

11.5.1.1 Target Resources

The Target Resource of this transaction is a Workitem.

11.5.1.2 Query Parameters

The user agent shall supply, and the origin server shall support, the Common Query Parameters in Section 11.1.2.

11.5.1.3 Request Header Fields

The origin server shall support all header fields in Table 10.5.1-1.

The user agent shall supply in the request header fields as defined in Table 10.5.1-1.

Table 11.5.1-1. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	`1#-dicom-media-type`	M	M	The Acceptable Media Types of the response payload

See also Section 8.4.

11.5.1.4 Request Payload

The request shall have no payload.

11.5.2 Behavior

If the Workitem exists on the origin server, the Workitem shall be returned in an Acceptable Media Type (See Section 8.7.4); however, the origin server may send a failure response to requests for Workitems that have moved to the Completed or Cancelled state. See Section 11.2.2 and PS3.4, Section CC.2.1.3.

The returned Workitem shall not contain the Transaction UID (0008,1195) Attribute. This is necessary to preserve this Attribute's role as an access lock.

11.5.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: dicom-media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF)
CRLF
[workitem / status-report]
```

11.5.3.1 Status Codes

Table 11.5.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.5.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	All Instances were successfully retrieved.
Failure	400 (Bad Request)	There was a problem with the request. For example, the Acceptable Media Types include non-Dicom Media Types.
	404 (Not Found)	The origin server has no knowledge of the Target Workitem. See PS3.4 Section CC.2.1.3.
	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
	410 (Gone)	The origin server knows that the Target Workitem did exist but has been deleted.

11.5.3.2 Response Header Fields

Table 11.5.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	M	media type of the Target Workitem or Status Report in payload
Content-Location	url	O	Shall be present if the response has a payload containing a resource. See Section 8.4.3. May be present otherwise
Content-Length	uint	C	Shall be present if no content encoding has been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload

See Also Section 8.4.

11.5.3.3 Response Payload

A success response has a single part payload containing the requested Workitem in the Selected Media Type.

If the Workitem is in the InProgress state, the returned Workitem shall not contain the Transaction UID (0008,1195) attribute of the Workitem, since that should only be known to the Owner.

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

11.6 Update Workitem Transaction

This transaction modifies attributes of an existing Workitem. It corresponds to the UPS DIMSE N-SET operation.

11.6.1 Request

The request shall have the following syntax:

```
POST SP /workitems/{workitem}?{transaction-uid} SP version CRLF
Content-Type: dicom-media-type CRLF
(Content-Length: uint / Content-Encoding: encoding) CRLF
Content-Location: url CRLF
*(header-field CRLF)
CRLF
Payload
```

The user agent shall conform to the SCU behavior specified in PS3.4, Section CC.2.6.2.

11.6.1.1 Target Resources

The Target Resource for this transaction is a Workitem.

11.6.1.2 Query Parameters

The origin server and user agent shall supply the Common Query Parameters in Section 11.1.2.

The origin server shall also supply the Transaction UID Query Parameter, which specifies the Transaction UID of the Workitem to be updated.

11.6.1.3 Request Header Fields

The origin server shall support all header fields in Table 10.6.1-1.

The user agent shall supply in the request header fields as defined in Table 11.6.1-1.

Table 11.6.1-1. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Content-Type	dicom-media-type	M	M	The media-type of the payload
Content-Length	uint	C	M	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	M	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

11.6.1.4 Request Payload

The request payload contains a dataset with the changes to the target Workitem. The dataset shall include all elements that are to be modified. All modifications to the Workitem shall comply with all requirements described in PS3.4 Section CC.2.6.2.

11.6.2 Behavior

The origin server shall modify the target Workitem as specified by the request, and in a manner consistent with the SCP behavior specified in PS3.4 Section CC.2.6.3.

If the Workitem is in the Completed or Cancelled state, the response shall be a 400 (Bad Request) failure response.

11.6.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
```

```
[Content-Type: media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: workitem CRLF
*(header-field CRLF)
CRLF
[status-report]
```

11.6.3.1 Status Codes

The response shall contain an appropriate status code.

Table 11.6.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.6.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The Target Workitem was updated.
Failure	400 (Bad Request)	There was a problem with the request. For example: • the Target Workitem was in the Completed or Cancelled state • the Transaction UID is missing • the Transaction UID is incorrect, or • the dataset did not conform to the requirements
	404 (Not Found)	The Target Workitem was not found.
	409 (Conflict)	The request is inconsistent with the current state of the Target Workitem
	410 (Gone)	The Target Workitem once existed, but no longer exists.

11.6.3.2 Response Header Fields

The origin server shall support all header fields in Table 11.6.3-2.

Table 11.6.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	M	The media-type of the payload
Content-Length	uint	C	Shall be present if no content encoding has been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload
Content-Location	url	O	Shall be present if the response has a payload containing a resource. See Section 8.4.3. May be present otherwise
Warning	see below	O	If the Target Workitem was modified by the origin server shall include one of the Warning header fields below

If the Workitem was successfully updated but with modifications made by the origin server, the response shall include the following in the Warning header field:

```
Warning: 299 <service>: The Workitem was updated with modifications.
```

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

```
Warning: 299 <service>: Requested optional attributes are not supported.
```

If the request was rejected with a failure status code, the response shall include a Warning header field with one of following messages that best describes the nature of the conflict:

```
Warning: 299 <service>: The target URL did not reference a claimed Workitem.
```

```
Warning: 299 <service>: The submitted request is inconsistent with the current state of the Workitem.
```

See also Section 8.4.

11.6.3.3 Response Payload

A success response shall either have no payload, or a payload containing a Status Report document.

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

11.7 Change Workitem State

This transaction is used to change the state of a Workitem. It corresponds to the UPS DIMSE N-ACTION operation "Change UPS State". State changes are used to claim ownership, complete, or cancel a Workitem.

11.7.1 Request

The request shall have the following syntax:

```
PUT SP /workitems/{workitem}/state SP version CRLF
Content-Type: dicom-media-type
(Content-Length: uint / Content-Encoding: encoding) CRLF
*(header-field CRLF)
CRLF
Payload
```

The user agent shall conform to the SCU behavior specified in PS3.4, Section CC.2.1.2.

11.7.1.1 Target Resources

The Target Resource for this transaction is a Workitem.

11.7.1.2 Query Parameters

The user agent shall supply, and the origin server shall support, the Common Query Parameters in Section 11.1.2.

11.7.1.3 Request Header Fields

The origin server shall support all header fields in Table 11.7.1-1.

The user agent shall supply in the request header fields as defined in Table 11.7.1-1.

Table 11.7.1-1. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Content-Type	dicom-media-type	M	M	The Acceptable Media Types of the response payload
Content-Length	uint	C	M	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	M	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

11.7.1.4 Request Payload

The request payload shall contain the Change UPS State Data Elements as specified in PS3.4, Table CC.2.1-1. These data elements are:

- Transaction UID (0008,1195)

  The request payload shall include a Transaction UID. The user agent creates the Transaction UID when requesting a transition to InProgress for a given Workitem. The user agent provides that Transaction UID in subsequent transactions with that Workitem.

- Procedure Step State (0074,1000)

The legal values correspond to the requested state transition. They are: "IN PROGRESS", "COMPLETED", or "CANCELLED".

11.7.2 Behavior

The origin server shall support the state changes to the Workitem specified in the request as described by the SCP behavior in Section CC.2.1.3 in PS3.4.

The origin server shall perform the following actions depending on the state of the Workitem:

- Claim Ownership

  A successful change from Scheduled to InProgress gives the user agent exclusive write access to the Workitem.

- Complete

  A successful change from InProgress to Completed is only permitted if the Workitem meets all the Final State requirements specified in PS3.4 Table CC.2.5-3.

- Cancel

  A successful change from InProgress to Cancelled is only permitted if the Workitem meets all the Final State requirements specified in PS3.4 Table CC.2.5-3.

Only the Owner can perform the Complete or Cancel transitions. A non-owner can use the Request Cancellation transaction (See Section 11.8) to request that the Owner cancel the Workitem.

Once the Workitem has moved to the Completed or Cancelled state, it shall not be modified. It shall remain accessible at least until all Deletion Locks have been released or abandoned by the owning user agent.

11.7.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: dicom-media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
*(header-field CRLF) CRLF
[status-report]
```

11.7.3.1 Status Codes

Table 11.7.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.7.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The update was successful, and the response payload contains a Status Report document.
Failure	400 (Bad Request)	The request cannot be performed for one of the following reasons: • the request is invalid given the current state of the Target Workitem • the Transaction UID is missing • the Transaction UID is incorrect
	404 (Not Found)	The Target Workitem was not found.
	409 (Conflict)	The request is inconsistent with the current state of the Target Workitem
	410 (Gone)	The Target Workitem once existed, but no longer exists.

11.7.3.2 Response Header Fields

The origin server shall support all header fields in Table 11.7.3-2.

Table 11.7.3-2. Response Header Fields

Name	Value	Origin Server	Description

		Usage	
Content-Type	media-type	M	The media-type of the payload
Content-Length	uint	C	Shall be present if no content encoding has been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload
Content-Location	uri	O	Shall be present if the response has a payload containing a resource. See Section 8.4.3. May be present otherwise
Warning	text	C	See below.

If the user agent specifies a Procedure Step State (0074,1000) attribute with a value of "CANCELED" and the workitem 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 a failure 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 state of the UPS Instance.
```

See also Section 8.4.

11.7.3.3 Response Payload

A success response shall have no payload.

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

11.8 Request Cancellation

This transaction allows a user agent that does not own a Workitem to request that it be cancelled. It corresponds to the UPS DIMSE N- ACTION operation "Request UPS Cancel". See PS3.4 Section CC.2.2.

To cancel a Workitem in the IN-PROGRESS state, which the user agent owns, the user agent shall use the Change Workitem State transaction as described in Section 11.7.

11.8.1 Request

The request shall have the following syntax:

```
POST SP /workitems/{workitem}/cancelrequest SP version CRLF
Content-Type: dicom-media-type
(Content-Length: uint / Content-Encoding: encoding) CRLF
*(header-field CRLF)
CRLF
[Payload]
```

The user agent shall conform to the SCU behavior specified in PS3.4, Section CC.2.2.2.

11.8.1.1 Target Resources

The Target Resource for this transaction is a Workitem.

11.8.1.2 Query Parameters

The user agent shall supply, and the origin server shall support, the Common Query Parameters in Section 11.1.2.

11.8.1.3 Request Header Fields

The origin server shall support all header fields in Table 11.8.1-1

The user agent shall supply in the request header fields as defined in Table 11.8.1-1.

Table 11.8.1-1. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Content-Type	dicom-media-type	M	M	The media-type of the payload
Content-Length	uint	C	M	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	M	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

11.8.1.4 Request Payload

The request payload, if present, may describe the reason for requesting the cancellation of the Workitem, a Contact Display Name, and/or a Contact URI for the person with whom the cancel request may be discussed.

The Request UPS Cancel Action Information is specified in PS3.4, Table CC.2.2-1.

11.8.2 Behavior

The origin server shall process the request as described by the SCP behavior in Section CC.2.2.3 in PS3.4.

11.8.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type dicom-media-type CRLF]
[Content-Type: dicom-media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF) CRLF
[status-report]
```

11.8.3.1 Status Codes

Table 11.8.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.8.3-1. Status Code Meaning

Status	Code	Meaning
Success	202 (Accepted)	The request was accepted by the origin server, but the Target Workitem state has not necessarily changed yet. Note: The system performing the Workitem 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.
Failure	400 (Bad Request)	There was a problem with the syntax of the request.
	404 (Not Found)	The Target Workitem was not found.
	409 (Conflict)	The request is inconsistent with the current state of the Target Workitem. For example, the Target Workitem is in the Scheduled or Completed state.

11.8.3.2 Response Header Fields

Table 11.8.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	C	The media type of the Status Report document Shall be present if the response contains a payload
Content-Length	uint	C	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload

If the Workitem Instance is already in a cancelled 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.
```

See also Section 8.4.

11.8.3.3 Response Payload

The response may include a payload containing an appropriate Status Report.

11.9 Search Transaction

This transaction searches a Worklist for Workitems that match the specified Query Parameters and returns a list of matching Workitems. Each Workitem in the returned list includes return attributes specified in the request. The transaction corresponds to the UPS DIMSE C-FIND operation.

11.9.1 Request

The request shall have the following syntax:

```
GET SP /workitems?{&match*}{&includefield}{&fuzzymatching}{&offset}{&limit} SP version CRLF
Accept: dicom-media-types CRLF
*(header-field CRLF)
CRLF
```

The user agent shall conform to the SCU behavior specified in PS3.4, Section CC.2.8.2.

11.9.1.1 Target Resources

The Target Resource for this transaction is the Worklist.

11.9.1.2 Query Parameters

The origin server shall support Query Parameters as required in Section 8.3.4-1.

The user agent shall supply in the request Query Parameters as required in Table 8.3.4-1.

11.9.1.3 Request Header Fields

The origin server shall support all header fields in Table 11.9.1-1.

The user agent shall supply in the request header fields as defined in Table 11.9.1-1.

Table 11.9.1-1. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	`1#-dicom-media-type`	M	M	The Acceptable Media Types of the response payload
Cache-Control	"no-cache"	O	M	If included, specifies that search results returned should be current and not cached.

See also Section 8.4.

11.9.1.4 Request Payload

The request payload shall be empty.

11.9.2 Behavior

The origin server shall perform a search according the requirements specified in Section 8.3.4.

For each matching Workitem, the origin server shall include in the results:

- All Unified Procedure Step Instance Attributes in PS3.4 Table CC.2.5-3 with a Return Key Type of 1 or 2.
- All Unified Procedure Step Instance Attributes in PS3.4 Table CC.2.5-3 with a Return Key Type of 1C for which the conditional requirements are met.
- All other Workitem Attributes passed as match parameters that are supported by the origin server as either matching or return attributes.
- All other Workitem Attributes passed as includefield parameter values that are supported by the origin server as return attributes.

11.9.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: dicom-media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF)
CRLF
[search-results / status-report]
```

11.9.3.1 Status Codes

Table 11.9.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.9.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The search completed successfully, and the matching results are returned in the message body.
	204 (No Content)	The search completed successfully, but there were no matching results.
	206 (Partial Content)	Only some of the search results were returned, and the rest can be requested through the appropriate request.
Failure	400 (Bad Request)	The was a problem with the request. For example, invalid Query Parameter syntax.
	413 (Payload Too Large)	The size of the results exceeds the maximum payload size supported by the origin server. The user agent may repeat the request with paging or with a narrower query to reduce the size of the result.

11.9.3.2 Response Header Fields

Table 11.9.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	M	The media-type of the payload
Content-Length	Uint	C	Shall be present if a content coding has not been applied to the payload
Content-	Encoding	C	Shall be present if a content encoding has been applied to

Encoding			the payload
Content-Location	url	C	Shall be present if the response has a payload containing a resource. See Section 8.4.3. May be present otherwise

See also Section 8.4.

11.9.3.3 Response Payload

A success response payload shall contain the search results in an Acceptable Media Type. See Section 8.7.5. If there are no matching results the payload will be empty (see 7.9.1).

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

11.10 Subscribe Transaction

This transaction creates a Subscription to a Worklist or Workitem resource. It corresponds to the UPS DIMSE N-ACTION operation "Subscribe to Receive UPS Event Reports".

Once a Subscription has been created the user agent will receive notifications containing Event Reports for events associated with the Subscription's resource. To receive the notifications generated by Subscriptions, the user agent must have first opened a Notification Connection between itself and the origin server using the Open Notification Connection transaction, see Section 8.9.2.

11.10.1 Request

The request shall have the following syntax:

```
POST SP /workitems/{resource}/subscribers/{aetitle}{?deletionlock}{&filter} SP version CRLF
*(header-field CRLF)
CRLF
```

The user agent shall conform to the SCU behavior specified in PS3.4, Section CC.2.3.2.

11.10.1.1 Target Resources

The origin server shall support the resources in Table 11.10.1-1.

Table 11.10.1-1. Subscribe Transaction Resources

Resource	URI Template
Worklist	/workitems/1.2.840.10008.5.1.4.34.5/subscribers/{aetitle}
Filtered Worklist	/workitems/1.2.840.10008.5.1.4.34.5.1/subscribers/{aetitle}
Workitem	/workitems/{workitem}/subscribers/{aetitle}

Where

`aetitle` is the Application Entity Title associated with the user agent.

11.10.1.2 Query Parameters

The origin server shall support Query Parameters as required in Table 11.10.1-2.

The user agent shall supply in the request Query Parameters as required in Table 11.10.1-2.

11.10.1-2. Query Parameters by Resource

Key	Value	Resource	Usage		Description
			User Agent	Origin Server	
Accept	media type	Worklist, Filtered Worklist, Workitem	O	M	
Charset	charset	Worklist, Filtered Worklist, Workitem	O	M	
deletionlock	true/false	Worklist, Filtered Worklist, Workitem	O	M	

Filter	1#(attribute"=" value)	Filtered Worklist	C	M	Shall be present if the Target Resource is the Filtered Worklist.

The Deletion Lock Query Parameter has a value of either true or false. If present with a value of true the Subscription will be created with a Deletion Lock (see PS3.4, Section CC.2.3.1).

```
deletionlock = "deletionlock=" true / false
```

The Filter Query Parameter has a value that is a comma-separated list of one or more matching keys (attribute/value pairs). A Workitem Subscription will be created for any existing and future Workitem that matches the attribute/value pairs. The valid attributes for this Query Parameter are defined by the UPS IOD (see PS3.3, Section B.26.2).

```
filter = 1#(attribute "=" value)
```

See Section 8.3.4.1 for the syntax of matching keys.

11.10.1.3 Request Header Fields

The request has no mandatory header fields. See Section 8.4.

11.10.1.4 Request Payload

The request shall have no payload.

11.10.2 Behavior

The origin server shall create and manage a Subscription to the Target Resource for the user agent. The origin server shall conform to the SCP behavior specified in PS3.4, Section CC.2.3.3. If the Target Resource is the Worklist resource, then a Worklist Subscription will be created with the Deletion Lock and Filter specified.

Upon receipt of the Subscribe request, the origin server shall attempt to update the state of the Global Subscription, Filtered Global Subscription, and/or Workitem of the specified Application Entity with respect to the SOP Instance UID as described in PS3.4 Table CC.2.3-2 and then return an appropriate response.

The effect of having a Worklist Subscription, is that when a new Workitem is created, the origin server will create a Subscription to that Workitem for the Subscriber.

The effect of having a Filtered Worklist Subscription, is that when a new Workitem is created, the origin server will create a Subscription to that Workitem for the Subscriber, but only if the filter is satisfied.

11.10.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF)
CRLF
[status-report]
```

11.10.3.1 Status Codes

Table 11.10.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.10.3-1. Status Code Meaning

Status	Code	Meaning
Success	201 (Created)	The Subscription was created.
Failure	400 (Bad Request)	There was a problem with the syntax of the request.
	403 (Forbidden)	The 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	The Target Resource was not found.

	Found)	

11.10.3.2 Response Header Fields

Table 11.10.3-2. Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	C	Shall be present if the response contains a payload.
Content-Location	url	C	A URL-reference to the WebSocket Connection. Shall be present if a Subscription was created. The URL shall include the WebSocket protocol (either WS or WSS) and may include a combination of authority and path.
Warning	String	C	See below

If the Create Subscription request was accepted but the Deletion Lock was not, the response shall include the following Warning header field:

```
Warning: 299 <service>: Deletion Lock not granted.
```

and may include a Status Report.

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

```
Warning: 299 <service>: Filtered Global Subscriptions are not supported.
```

See also Section 8.4.

11.10.3.3 Response Payload

A success response payload may contain a Status Report.

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

11.11 Unsubscribe Transaction

This transaction is used to stop the origin server from sending new Event Reports to the user agent or to stop the origin server from subscribing the user agent to new Workitems.

11.11.1 Request

The request shall have the following syntax:

```
DELETE SP {/resource} SP version CRLF
*(header-field CRLF)
CRLF
```

11.11.1.1 Target Resources

Table 11.11.1-1. Unsubscribe Transaction Resources

Resource	URI Template
Workitem	/workitems/{workitem}/subscribers/{aetitle}
Worklist	/workitems/1.2.840.10008.5.1.4.34.5/subscribers/{aetitle}/suspend
Filtered Worklist	/workitems/1.2.840.10008.5.1.4.34.5.1/subscribers/{aetitle}/suspend

11.11.1.2 Query Parameters

The request has no Query Parameters.

11.11.1.3 Request Header Fields

The request has no Mandatory header fields.

11.11.1.4 Request Payload

The request payload shall be empty.

11.11.2 Behavior

The origin server shall process the request as described in PS3.4 Section CC.2.3.3.

Upon receipt of an Unsubscribe request, the origin server shall attempt update the state of Global Subscription, Filtered Global Subscription, or Workitem Subscription of the specified Application Entity with respect to the specified SOP Instance UID as described in PS3.4 Table CC.2.3-2 and then return an appropriate response.

For a Workitem resource, the origin server will no longer send Event Reports to the Subscriber related to the Workitem. This corresponds to the UPS DIMSE N-ACTION operation "Unsubscribe from Receiving UPS Event Reports".

For a Worklist or a Filtered Worklist resource, the origin server will no longer create new subscriptions to Workitems for the Subscriber and will no longer send Event Reports to the Subscriber related to any Workitems. This transaction corresponds to the UPS DIMSE N-ACTION operation "Suspend Global Subscription".

11.11.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF)
CRLF
[status-report]
```

11.11.3.1 Status Codes

Table 11.11.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 11.11.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The Global Subscription was suspended, or the Workitem Subscription was removed.
Failure	400 (Bad Request)	There was a problem with the request. For example, • the request is invalid given the current state of the Subscription • the Target Workitem UID is missing
	404 (Not Found)	The target Subscription was not found.

11.11.3.2 Response Header Fields

Table 11.11.3-2: Response Header Fields

Name	Value	Origin Server Usage	Description
Content-Type	media-type	C	The media-type of the response payload. Shall be present if the response has a payload.
Content-Length	uint	C	Shall be present if no content encoding has been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload
Warning	text	O	A warning message

See also Section 8.4.

11.11.3.3 Response Payload

A success response shall have no payload.

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

11.12 Workitem Event Reports

The origin server uses the Send Event Report Transaction (see Section 8.9.3) to send a Workitem Event Report, containing the details of any state change in the Workitem to the user agent.

The origin server shall send Workitem Event Reports as described in PS3.4 Section CC.2.4.3.

The Event Report shall contain all mandatory attributes described in PS3.4 Table CC.2.4-1 and PS3.7 Table 10.3-2.

The following is an example application/dicom+json Workitem Event Report payload:

```
{
"00000002": {"vr": "UI", "Value": ["1.2.840.10008.5.1.4.34.6.4"] },
"00000110": {"vr": "US", "Value": [23] },
"00001000": {"vr": "UI", "Value": ["1.2.840.10008.5.1.4.34.6.4.2.3.44.22231"] },
"00001002": {"vr": "US", "Value": [1] },
"00404041": {"vr": "US", "Value": ["READY"] }
"00741000": {"vr": "LT", "Value": ["SCHEDULED"] },
} CRLF
```

12 Non-Patient Instance Web Service and Resources

12.1 Overview

The Non-Patient Instance (NPI) Storage Service enables a user agent to retrieve, store, and search an origin server for instances that are not related to a patient.

An NPI Storage Service manages a collection of resources belonging to the categories specified in Table 12.1.1-1.

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 support one or more of the Retrieve Capabilities, Retrieve, Store, or Search transactions.

12.1.1 Resource Descriptions

An NPI Service manages resources from the same NPI Category. Target 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 12.1.1-1 contains the templates for the NPI Resource Categories.

Table 12.1.1-1. Resource Categories, URI Templates and Descriptions

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

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

12.1.2 Common Query Parameters

The origin server shall support Query Parameters as required in Table 12.1.2-1.

The user agent shall supply in the request Query Parameters as required in Table 12.1.2-1.

Table 12.1.2-1. Common Query Parameters

		Usage		
Name	Value	User Agent	Origin Server	Section
Accept	media-type	M	M	8.3.3.1
Accept-Charset	charset	O	M	8.3.3.2

See also Section 8.4.

12.1.3 Common Media Types

The origin server shall support the media types listed as Default or Required in Table 12.1.3-1 for all NPI transactions.

Table 12.1.3-1. Default, Required, and Optional Media Types

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

12.2 Conformance

An origin server conforming to the Studies Service shall implement the Retrieve Capabilities Transaction (See 8.9.1).

The origin server shall support the transactions listed as Required in Table 12.2-1.

Table 12.2-1. 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 8.9 and Annex H):

- 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)

12.3 Transactions Overview

The NPI Service consists of the transactions listed in Table 12.3-1

Table 12.3-1. NPI Service Transactions

Transaction	Method	Resource	Payload		Description
			Request	Success Response	
Retrieve Capabilities	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 , 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 12.3-2 summarizes the Target Resources permitted for each transaction.

Table 12.3-2. Resources by Transaction

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

12.4 Retrieve Transaction

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

12.4.1 Request

The request shall have the following syntax:

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

12.4.1.1 Target Resources

The target URI shall reference one of the resources shown in Table 12.4.1-1.

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

Table 12.4.1-1. Retrieve Transaction Resources

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

12.4.1.2 Query Parameters

The user agent shall supply, and the origin server shall support, the Common Query Parameters in Section 12.1.2.

12.4.1.3 Request Header Fields

Table 12.4.1-2. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	`1#-dicom-media-type`	M	M	The Acceptable Media Types of the response payload

See also Section 8.4.

12.4.1.4 Request Payload

The request shall have no payload.

12.4.2 Behavior

The origin server shall try to locate the Target Resource and if found, return it in an Acceptable DICOM Media Type. See Section 8.7.5.

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

12.4.3 Response

The response has the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: dicom-media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF
CRLF
[payload / status-report]
```

12.4.3.1 Status Codes

Table 12.4.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 12.4.3-1. Status Code Meaning

Status	Code	Meaning
Success	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.
Failure	400 (Bad Request)	Indicates there was a problem with the request and 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	Indicates that the origin server did not find a current representation for the Target Resource or is not

Status	Code	Meaning
	Found)	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.

12.4.3.2 Response Header Fields

Table 12.4.3-2. Response Header Fields

Header Field	Value	Origin Server Usage	Requirements
Content-Type	dicom-media-type	M	The media-type of the response payload
Content-Length	uint	C	Shall be present if no content encoding has been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

12.4.3.3 Response Payload

A success response shall have a payload containing the DICOM instance specified by the Target Resource.

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

12.5 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.

12.5.1 Request

The request shall have the following syntax:

```
POST SP /{npi-name} {/uid} SP version CRLF
Content-Type: dicom-media-type CRLF
(Content-Length: uint / Content-Encoding: encoding) CRLF
CRLF
payload
```

12.5.1.1 Target Resources

The target URL shall reference one of the resources shown in Table 12.5.1-1.

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

Table 12.5.1-1. Store Transaction Resources

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.

12.5.1.2 Query Parameters

The user agent shall supply, and the origin server shall support, the Common Query Parameters in Section 12.1.2.

12.5.1.3 Request Header Fields

Table12.5.1-2. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Content-Type	media-type	M	M	The DICOM Media Type of the request payload
Content-Length	uint	C	M	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	M	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

12.5.1.4 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.

12.5.2 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.

12.5.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF)
CRLF
[Status Report]
```

12.5.3.1 Status Codes

Table 12.5.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 12.5.3-1. Status Code Meaning

Status	Code	Meaning
Success	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.
	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.
Failure	400 (Bad Request)	The was a problem with the request. For example,

Status	Code	Meaning
		• the origin server did not store any of the representations contained in the request payload because of errors in the request message. • the request contained an invalid Query Parameter • the request referenced 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.

12.5.3.2 Response Header Fields

Table 12.5.3-2. Response Header Fields

Header Field	Value	Origin Server Usage	Requirements
Content-Type	dicom-media-type	M	The media type of the response payload.
Content-Length	uint	C	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	encoding	C	Shall be present if a content encoding has been applied to the payload

See also Section 8.4.

12.5.3.3 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.

12.6 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.

12.6.1 Request

The request shall have the following syntax:

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

12.6.1.1 Target Resources

The target URI shall reference one of the resources shown in Table 12.6.1-1.

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

Table 12.6.1-1. Search Transaction Resources

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

12.6.1.2 Query Parameters

The user agent shall supply, and the origin server shall support, the Common Query Parameters in Section 12.1.2.

The origin server shall support Query Parameters as required in Table 8.3.4-1.

The user agent shall supply in the request Query Parameters as required in Table 8.3.4-1.

For each Resource Category the origin server supports, it shall support the behaviors and matching key attributes specified in the corresponding sections in Table 12.6.1-2.

Table 12.6.1-2. NPI Resource Search Attributes

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

12.6.1.3 Request Header Fields

Table 12.6.1-3. Request Header Fields

Name	Value	Usage		Description
		User Agent	Origin Server	
Accept	`1#-dicom-media-type`	M	M	The Acceptable Media Types of the response payload

See also Section 8.4.

12.6.1.4 Request Payload

The request has no payload.

12.6.2 Behavior

The origin server shall perform the search indicated by the request, using the matching behavior specified in Section 8.3.4.1.1 and in the corresponding sections in Table 8.3.4.1-1.

The rules for search results are specified in Section 8.3.4.

12.6.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Type: dicom-media-type CRLF]
[(Content-Length: uint / Content-Encoding: encoding) CRLF]
[Content-Location: url CRLF]
*(header-field CRLF
CRLF
[payload / status-report]
```

12.6.3.1 Status Codes

Table 12.6.3-1 shows some common status codes corresponding to this transaction. See also Section 8.5 for additional status codes.

Table 12.6.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	Indicates that the origin server found and returned at least one resource matching the request.
Failure	400 (Bad Request)	The request contained an error. For example, the Query Parameters were invalid
	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.

12.6.3.2 Response Header Fields

Table 12.6.3-2. Response Header Fields

Header Field	Value	Origin Server Usage	Requirement
Content-Type	dicom-media-type	M	The media type of the response payload.
Content-Length	Uint	C	Shall be present if a content encoding has not been applied to the payload
Content-Encoding	Encoding	C	Shall be present if a content coding has been applied to the payload

2865 See also Section 8.4.

12.6.3.3 Response Payload

A success response shall contain the search results in an Acceptable Media Type. See Section 8.7.5.

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

Annex A Collected ABNF

A machine readable collected ABNF for the syntax defined in PS3.18 can be found at ftp://medical.nema.org/medical/dicom/ABNF/part18_abnf.txt

Annex B Examples (Informative)

The following are example uses of the URI Web Service. See Section 9.

B.1 Retrieving a Simple DICOM Image in JPEG

```
http://www.hospital-stmarco.com/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
```

Annex C Applications (Informative) (Retired)

See PS3.18-2017x.

Annex D IANA Character Set Mappings

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

Annex E WADO-WS Schemas and Examples (Retired)

See PS3.18-2017x.

Annex 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 (see PS3.19), 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 “Evolution of WADO to Web and Rest 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 shall 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)

For Private Data Elements, the group and element numbers will follow the rules specified in PS3.5 Section 7.8.1<C://dicom/github/ps3.18_re_doc\\part05.pdf" \l "sect_7.8.1>.

The person name representation is more closely aligned with the DICOM Data Element representation than the PS3.19<C://dicom/github/ps3.18_re_doc\\part19.pdf> 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 in PS3.6 Section 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 BulkData URI

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 Inline Binary

If an attribute contains an "InlineBinary", this contains the base64 encoding of the enclosing attribute's Value Field.

There is a single Inline Binary 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 Inline Binary 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 shall 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 XML	DICOM JSON Model

<NativeDicomModel>
<DicomAttribute tag= **ggggee01** … />
<DicomAttribute tag= **ggggee02** … />
…
</NativeDicomModel>
	{
	ggggee01 : { … },
	ggggee02 : { … },
	…
	}

<DicomAttribute
tag= **gggggeee**
vr= **VR** >
<Value number="1"> **Value** </Value>
</DicomAttribute>
	gggggeee : {
	"vr": **VR** ,
	"Value": [**Value**]
	}

<DicomAttribute tag= **gggggeee** … >
<Value number="1"> **Value1** </Value>
<Value number="2"> **Value2** </Value>
…
</DicomAttribute>
	gggggeee : {
	…
	"Value": [**Value1** ,
	Value2 , …
	]
	}

<DicomAttribute tag= **gggggeee** … >
</DicomAttribute>
	gggggeee : {
	…
	}

<DicomAttribute tag= **gggggeee** vr="PN" … >
<PersonName number="1">
<Alphabetic>
<FamilyName> **SB1**
</FamilyName>
<GivenName> **SB2**
</GivenName>
<MiddleName> **SB3**
</MiddleName>
<NamePrefix> **SB4**
</NamePrefix>
	gggggeee : {
	…
	"vr": "PN",
	"Value": [
	{
	" Alphabetic " : **"SB1^SB2^SB3^SB4^SB5"**,
	"Ideographic": **"ID1^ID2^ID3^ID4^ID5"** ,
	"Phonetic": **"PH1^PH2^PH3^PH4^PH5"**
	},
	{
	"Alphabetic":

DICOM XML	DICOM JSON Model
<NameSuffix> **SB5** </NameSuffix> </Alphabetic> <Ideographic> <FamilyName> **ID1** </FamilyName> … </Ideographic> <Phonetic> <FamilyName> **PH1** </FamilyName> … </Phonetic> </PersonName> <PersonName number="2"> <Alphabetic> <FamilyName> **SB6** </FamilyName> </Alphabetic> </PersonName> </DicomAttribute>	" **SB6** " }] }

<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 Studies Service Search for Studies response (see Section 10.6.3)
// 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 ]
        }
    }
```

3305]

Annex H Capabilities Description

A Capabilities Description is a WADL Document. See http://www.w3.org/Submission/wadl/.

The Capabilities Description resource follows directly and unambiguously from the RESTful resources defined in Sections 10, 11, and 12.

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

The "application" element shall contain one "resources" element whose "base" attribute value 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 Target Resource specified in the request describing all methods and child resources for the specified resource and each of its children.

The full resource tree and the methods defined for each resource are described in Table H-1.

Note

The Retrieve Capabilities Transaction is excluded from this table because that transaction is used to retrieve this document and WADL is not self-describing.

Table H-1. Resources and Methods

Service	Resource	Transactions	Reference
Studies (see Section 10.1.1)			
	studies	Search for Studies Store Instances	10.6 10.5
	{StudyInstance}	Retrieve Study Store Study Instances	10.4 10.5
	metadata	Retrieve Study Metadata	10.4
	series	Search for Study Series	10.6
	{SeriesInstance}	Retrieve Series	10.4
	metadata	Retrieve Series Metadata	10.4
	instances	Search for Study Series Instances	10.4
	{SOPInstance}	Retrieve Instance	10.4
	metadata	Retrieve Instance Metadata	10.4
	frames	N/A	N/A
	{framelist}	Retrieve Frames	10.4
	instances	Search for Study Instances	10.6
	series	Search for Series	10.6
	{SeriesInstance}	N/A	N/A
	{instances}	Search for Instances	10.6
	instances	Search for Instances	10.6
	{BulkDataReference}	Retrieve Bulkdata	10.4
Worklist (see Section 11.1.1)			
	workitems	Search for Workitem Create Workitem	11.9 11.4
	{Workitem}	Retrieve Workitem Update Workitem	11.4 11.6
	state	Change Workitem State	11.7
	cancelrequest	Request Workitem Cancellation	11.8
	subscribers	N/A	N/A
	{AETitle}	Subscribe	11.10

Service	Resource	Transactions	Reference
		Unsubscribe	11.11
	1.2.840.10008.5.1.4.34.5	N/A	N/A
	subscribers	N/A	N/A
	{AETitle}	Subscribe Unsubscribe	11.10 11.11
	suspend	Unsubscribe	11.11
	1.2.840.10008.5.1.4.34.5.1	N/A	N/A
	subscribers	N/A	N/A
	{AETitle}	Subscribe Unsubscribe	11.10 11.11
	suspend	Suspend Global Subscription	11.11
Non-Patient Instances (see Section 12.1.1)			
	color-palettes	N/A	N/A
	{uid}	Retrieve Store Search	12.4 12.5 12.6
	defined-procedure-protocol	N/A	N/A
	{uid}	Retrieve Store Search	12.4 12.5 12.6
	hanging-protocol	N/A	N/A
	{uid}	Retrieve Store Search	12.4 12.5 12.6
	Implant Template	N/A	N/A
	{uid}	Retrieve Store Search	12.4 12.5 12.6

Annex 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 shall 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 shall 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"
  }
 ]
```