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: Final Text
Developed in accordance with work item 2014-04-A.

--- Draft ---
TODO:

1. Create Supplement Generalized Worklist and Notifications
   a. Create Notifications (NADO) CP
   b. Create Worklist CP
      i. Create Workitem Location Header Field and create UID
   c.

2. CPs
   a. Create CPs for URI parameter & errors
   b. Create CP for WS parameters & errors
   c. Create CP for RS parameter errors
   d. DICOM Media Type CP1509
   e. CP about "includefield" and "include" taking multiple arguments
   f. Create Status Details CP
   g. Metadata & Bulkdata CP
   h. CP about not using obs-text and obs-fold rules in HTTP
      i. Create Conditional CP
      j. Create Caching CP
      k. Create Lock Resource CP
      l.

3. Create Part 5 Conventions
   a. Generalize search parameters in Part 5
   b. Generalize Transactions in Part 5
   c. Search Query Parameters
   d. Rendering Query Parameters
   e. Update all sections to conventions
      i. Resource
      ii. Query Parameters
      iii. Headers
   f. 5

4. Re-Doc
   a. URL -> URI
   b. Standardize
      a. Header Tables
      b. Status code and payload phrasing and terminology
   c. Remove "<" and ">" where not absolutely required
   d. Add Sections to Header Tables Description column
   e. Move parts of Section 7 Intro to Section
   f. 5. Conventions
   g. Create syntax for:
      i. content-header-fields
      ii. payload-header-fields
   h. 5

5. Verify all sections conform to standard format
   a. Resource
   b. Query Parameters
   c. Request Headers
   d. Response Status codes
   e. Response Payload
   f. Add note to each definition that standard syntactic variables are defined in Section X.
   g. Add media type Annexes
   h.

6.

--- Draft ---
Editor’s Notes

Terminology and Structure
Throughout this document I have endeavored to use the following terminology to structure the document:

Service
A Service defines a set of related transactions on a resource or resource group

Transaction
A transaction is a request/response message pair using a specific HTTP method on a resource or resource group

Resource
A Resource refers to a thing. The thing might be abstract like an idea. In PS3.18 it typically refers to a DICOM Information Object. Resources can have more than one representation.

Resource Group
A collection of related Resources. For example the Studies resource.

Target Resource
A target resource is the resource that is the focus of the transaction. It is referenced by a target URI in the request message of a transaction.

Target URI
The URI in the first line of a request message. It references the target resource.

Representation
A data object that is a representation of a resource.

Media Type
The name, possibly parameterized, of a specific encoding of a representation. See IETF Media Type Registry.

Editorial Issues and Decisions

<table>
<thead>
<tr>
<th>#</th>
<th>Issue</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>It seems to be common practice to use relative URIs (starting with '/') to describe resource templates. This would eliminate {+svc} everywhere. Should we make this change?</td>
<td>Closed</td>
</tr>
<tr>
<td>Decision: Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Use full syntax for URI template components?</td>
<td>Closed</td>
</tr>
<tr>
<td>Decision: yes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Closed Issues

<table>
<thead>
<tr>
<th>#</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>85</td>
<td></td>
</tr>
</tbody>
</table>

Open Issues

<table>
<thead>
<tr>
<th>#</th>
<th>Issues</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Should we define our own &quot;reason phrases&quot;?</td>
<td>Open</td>
</tr>
</tbody>
</table>

CPs Needed before publishing

<table>
<thead>
<tr>
<th>#</th>
<th>CP</th>
<th>Status</th>
</tr>
</thead>
</table>

--- Draft ---
Scope and Field of Application of this Supplement

This supplement re-documents (rewrites) PS3.18 Web Services.

The goals of this re-write are:

- Bring the Standard into conformance with the latest Web Standards, especially [RFC7230 – 7234], and [RFC3986 – 3987].
- Use the ABNF defined in [RFC5234] and [RFC7405] to specify the syntax of request and response messages.
- Use standard terminology throughout the standard.
- Use a standard format for documenting services and transactions.
- Clarify ambiguities and underspecified aspects of the Standard using the CP process.

The most important aspect of the re-documentation is that no semantic content of the Standard should be changed. Errors, ambiguities, and underspecified aspects of the existing standard have been corrected through the CP process prior to the finalization of this supplement.
PS3.18

DICOM PS3.18 201Xy - Web Services APIs
# Table of Contents

Notice and Disclaimer                               17  
Foreword                                           18  
1 Scope                                           19  
2 Normative References                            20  
3 Terms and Definitions                            22  
   3.1 DICOM Conformance Definitions               22  
   3.2 DICOM Information Object                    22  
   3.3 DICOM Service Class Specifications          22  
   3.4 DICOM Data Structures and Encoding          22  
   3.5 DICOM Message Exchange Definitions          22  
   3.6 DICOM Web Services                          23  
   3.7 HyperText Transfer Protocol                 24  
4 Symbols and Abbreviated Terms                    25  
5 Conventions                                      26  
   5.1 URIs versus URLs                            26  
   5.2 Message Syntax                              26  
      5.2.1 Common Syntactic Rules for Data Types   26  
      5.2.2 URI Templates                          27  
      5.2.3 List Extension Rule (‘#’)              27  
   5.3 Standard Query Parameters                   27  
      5.3.1 Image Rendering Parameters              27  
      5.3.2 Search Query Parameters                27  
      5.3.2.1 Attribute / Values Pair Requirements  27  
   5.4 Header Field Documentation                  28  
      5.4.1 Header Field Usage                      28  
   5.5 Format for Web Service Definitions          28  
      5.5.1 Example Service Description with Preferred Phrases 28
6 Conformance                                      31  
7 Overview of Web Services APIs                    32  
   7.1 Services, and Transactions                   32  
   7.2 Resources, Target URIs and Representations   32  
   7.3 Transactions                                 32  
      7.3.1 General Format of Request Messages      33  
         7.3.1.1 Method                            34  
         7.3.1.2 Target Resource URI                34  
         7.3.1.3 Query Parameters                   34  
         7.3.1.4 Request Header Fields              34  
         7.3.1.5 Request Payload                    34  
      7.3.2 General Format of Response Messages      34  
         7.3.2.1 Status Codes                       34  
         7.3.2.2 Response Header Fields              35  
         7.3.2.3 Response Payload                    35  
   7.4 Target Resource URI                          35  
      7.4.1 The Base URI of a Service               35  
      7.4.2 Target Resource Path                    36  
   7.5 Query Parameters                             36  
      7.5.1 Query Parameter Syntax                  37  
   7.6 Header Fields                                38  
      7.6.1 Header Field Usage                      38  
      7.6.2 Content Negotiation Header Fields       38  
         7.6.2.1 Content Negotiation Header Field General Parameters 39  
         7.6.2.1.1 Quality Value Parameter            39  
         7.6.2.1.2 Charset Parameter                  39  
         7.6.2.2 Accept                              39
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.6.3</td>
<td>Content Representation Header Fields</td>
<td>39</td>
</tr>
<tr>
<td>7.6.4</td>
<td>Payload Header Fields</td>
<td>40</td>
</tr>
<tr>
<td><strong>7.6.5</strong></td>
<td>Conditional Request Header Fields</td>
<td>40</td>
</tr>
<tr>
<td>7.6.6</td>
<td>Caching Header Fields</td>
<td>41</td>
</tr>
<tr>
<td>7.7</td>
<td>Status Codes</td>
<td>41</td>
</tr>
<tr>
<td>7.8</td>
<td>Payloads</td>
<td>44</td>
</tr>
<tr>
<td>7.8.1</td>
<td>Payload Format</td>
<td>44</td>
</tr>
<tr>
<td>7.8.1.1</td>
<td>Single Part Payload</td>
<td>44</td>
</tr>
<tr>
<td>7.8.1.2</td>
<td>Multipart Payload</td>
<td>44</td>
</tr>
<tr>
<td>7.8.1.2.1</td>
<td>Multipart Payload Syntax</td>
<td>45</td>
</tr>
<tr>
<td>7.8.2</td>
<td>Representations</td>
<td>46</td>
</tr>
<tr>
<td>7.8.2.1</td>
<td>DICOM Representations</td>
<td>47</td>
</tr>
<tr>
<td>7.8.2.2</td>
<td>Composite SOP Instances</td>
<td>47</td>
</tr>
<tr>
<td>7.8.2.3</td>
<td>Metadata and Bulkdata</td>
<td>47</td>
</tr>
<tr>
<td>7.8.2.3.1</td>
<td>Metadata and Bulkdata Definitions</td>
<td>48</td>
</tr>
<tr>
<td>7.8.2.3.2</td>
<td>Converting SOP Instances to/from Metadata and Bulkdata</td>
<td>48</td>
</tr>
<tr>
<td>7.8.2.4</td>
<td>Error! Bookmark not defined.</td>
<td></td>
</tr>
<tr>
<td>7.9</td>
<td>Media Types</td>
<td>48</td>
</tr>
<tr>
<td>7.9.1</td>
<td>Multipart Media Types</td>
<td>49</td>
</tr>
<tr>
<td>7.9.2</td>
<td>DICOM Resource Categories</td>
<td>50</td>
</tr>
<tr>
<td>7.9.3</td>
<td>DICOM Media Types</td>
<td>50</td>
</tr>
<tr>
<td>7.9.4</td>
<td>Rendered Media Types</td>
<td>50</td>
</tr>
<tr>
<td>7.9.5</td>
<td>Acceptable Media Types</td>
<td>51</td>
</tr>
<tr>
<td>7.9.6</td>
<td>Accept Query Parameter</td>
<td>51</td>
</tr>
<tr>
<td>7.9.7</td>
<td>Accept Header Field</td>
<td>52</td>
</tr>
<tr>
<td>7.9.8</td>
<td>Selected Media Type</td>
<td>52</td>
</tr>
<tr>
<td>7.10</td>
<td>Character Sets</td>
<td>53</td>
</tr>
<tr>
<td>7.10.1</td>
<td>Acceptable Character Sets</td>
<td>54</td>
</tr>
<tr>
<td>7.10.2</td>
<td>Character Set Query Parameter</td>
<td>54</td>
</tr>
<tr>
<td>7.10.3</td>
<td>Character Set Media Type Parameters</td>
<td>54</td>
</tr>
<tr>
<td>7.10.4</td>
<td>Accept-Charset Header Field</td>
<td>55</td>
</tr>
<tr>
<td>7.10.5</td>
<td>Selected Character Set</td>
<td>55</td>
</tr>
<tr>
<td>7.11</td>
<td>Conformance</td>
<td>55</td>
</tr>
<tr>
<td>7.12</td>
<td>Security</td>
<td>56</td>
</tr>
<tr>
<td>8</td>
<td>WADO-URI Service API</td>
<td>57</td>
</tr>
<tr>
<td>8.1</td>
<td>General Query Parameters</td>
<td>57</td>
</tr>
<tr>
<td>8.1.1</td>
<td>Required Query Parameters</td>
<td>57</td>
</tr>
<tr>
<td>8.1.1.1</td>
<td>Request Type</td>
<td>57</td>
</tr>
<tr>
<td>8.1.1.2</td>
<td>Study UID</td>
<td>57</td>
</tr>
<tr>
<td>8.1.1.3</td>
<td>Series UID</td>
<td>57</td>
</tr>
<tr>
<td>8.1.1.4</td>
<td>Instance UID</td>
<td>57</td>
</tr>
<tr>
<td>8.2</td>
<td>WADO-URI Retrieve DICOM Instance Transaction</td>
<td>58</td>
</tr>
<tr>
<td>8.2.1</td>
<td>Request Message</td>
<td>58</td>
</tr>
<tr>
<td>8.2.1.1</td>
<td>Target Resource</td>
<td>58</td>
</tr>
<tr>
<td>8.2.1.2</td>
<td>Query Parameters</td>
<td>58</td>
</tr>
<tr>
<td>8.2.1.2.1</td>
<td>Acceptable Media Type</td>
<td>58</td>
</tr>
<tr>
<td>8.2.1.2.2</td>
<td>Character Set</td>
<td>58</td>
</tr>
<tr>
<td>8.2.1.2.3</td>
<td>Anonymize</td>
<td>59</td>
</tr>
<tr>
<td>8.2.1.2.4</td>
<td>Transfer Syntax</td>
<td>59</td>
</tr>
<tr>
<td>8.2.1.3</td>
<td>Request Header Fields</td>
<td>59</td>
</tr>
<tr>
<td>8.2.1.4</td>
<td>Request Payload</td>
<td>59</td>
</tr>
<tr>
<td>8.2.2</td>
<td>Behavior</td>
<td>59</td>
</tr>
<tr>
<td>8.2.3</td>
<td>Response Message</td>
<td>59</td>
</tr>
<tr>
<td>8.2.3.1</td>
<td>Status Codes</td>
<td>59</td>
</tr>
<tr>
<td>8.2.3.2</td>
<td>Response Header Fields</td>
<td>60</td>
</tr>
<tr>
<td>8.2.3.3</td>
<td>Response Payload</td>
<td>60</td>
</tr>
<tr>
<td>8.2.4</td>
<td>Acceptable Media Types</td>
<td>60</td>
</tr>
<tr>
<td>8.3</td>
<td>WADO-URI Retrieve Rendered Instance Transaction</td>
<td>60</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Request Message</td>
<td>60</td>
</tr>
<tr>
<td>8.3.1.1</td>
<td>Target Resource</td>
<td>61</td>
</tr>
</tbody>
</table>
8.3.1.2 Query Parameters 61
8.3.1.3 Common Query Parameters 61
  8.3.1.3.1 Acceptable Media Type 61
  8.3.1.3.2 Character Set 61
8.3.1.4 Image Related Query Parameters 61
  8.3.1.4.1 Frame Number 61
  8.3.1.4.2 Image Annotation 61
  8.3.1.4.3 Viewport 62
    8.3.1.4.3.1 Viewport Rows 62
    8.3.1.4.3.2 Viewport Columns 62
  8.3.1.4.4 Source Image Region 62
  8.3.1.4.5 Windowing 63
    8.3.1.4.5.1 Window Center 63
    8.3.1.4.5.2 Window Width 63
  8.3.1.4.6 Image Quality 63
  8.3.1.4.7 Presentation State 63
    8.3.1.4.7.1 Presentation Series UID 64
    8.3.1.4.7.2 Presentation UID 64
8.3.1.5 Request Header Fields 64
8.3.1.6 Request Payload 64
8.3.2 Behavior 64
  8.3.2.1 Without Presentation State 64
  8.3.2.2 With Presentation State 64
8.3.3 Response Message 65
  8.3.3.1 Status Codes 65
  8.3.3.2 Response Header Fields 65
  8.3.3.3 Response Payload 65
  8.3.3.4 Acceptable Media Types 65
8.4 Media Types 65
8.5 Conformance 65
9 WADO-WS Services API 67
  9.1 Conformance 67
10 RESTful Services APIs 68
  10.1 Services 68
  10.2 Target Resources 68
  10.3 Transaction Overview 68
  10.4 Generic Service Transactions 70
    10.4.1 Retrieve Information Transaction 70
      10.4.1.1 Request 70
        10.4.1.1.1 Resource 70
        10.4.1.1.2 Query Parameters 71
        10.4.1.1.3 Request Header Fields 71
        10.4.1.1.4 Request Payload 71
      10.4.1.2 Behavior 71
      10.4.1.3 Response 71
        10.4.1.3.1 Status Codes 71
        10.4.1.3.2 Response Header Fields 71
        10.4.1.3.3 Response Payload 71
      10.4.1.4 Media Types 71
  10.4.2 Retrieve Resource Status 71
10.5 Notification Sub-Service (or Subsystem) 72
10.6 Conditional Requests

10.6.1 Origin Server Requirements

10.6.2 User Agent Requirements

10.6.3 Conditional Retrieve

10.6.4 Conditional Modifications (POST, PUT, PATCH, DELETE)

10.7 Caching and Proxies

11 RESTful Studies Service

11.1 Resources

11.2 Transactions

11.3 Retrieve Capabilities Transaction

11.3.1 Request

11.3.2 Behavior

11.3.3 Response

11.3.4 Conformance

11.4 Retrieve DICOM Transaction

11.4.1 Request

11.4.2 Behavior

11.4.3 Response

11.4.4 Media Types

11.4.5 Conformance

11.5 RS Retrieve Rendered Transaction

11.5.1 Request

11.5.2 Behavior

11.5.3 Response

11.5.4 Media Types

11.5.5 Conformance

11.6 Store Transaction

11.6.1 Request

11.6.2 Behavior

11.6.3 Response

11.6.4 Media Types

11.6.5 Conformance

11.7 Search Transaction

11.7.1 Request
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.7.1.1</td>
<td>Resources</td>
</tr>
<tr>
<td>11.7.1.2</td>
<td>Search Query Parameters</td>
</tr>
<tr>
<td>11.7.1.2.1</td>
<td>Attribute / Values Pair Requirements</td>
</tr>
<tr>
<td>11.7.1.2.2</td>
<td>Required Attributes for Query Parameters</td>
</tr>
<tr>
<td>11.7.1.3</td>
<td>Request Header Fields</td>
</tr>
<tr>
<td>11.7.1.4</td>
<td>Request Payload</td>
</tr>
<tr>
<td>11.7.2</td>
<td>Behavior</td>
</tr>
<tr>
<td>11.7.2.1</td>
<td>Matching Rules</td>
</tr>
<tr>
<td>11.7.2.2</td>
<td>Number of Results and Paging</td>
</tr>
<tr>
<td>11.7.3</td>
<td>Response</td>
</tr>
<tr>
<td>11.7.3.1</td>
<td>Status Codes</td>
</tr>
<tr>
<td>11.7.3.2</td>
<td>Response Header Fields</td>
</tr>
<tr>
<td>11.7.3.3</td>
<td>Response Payload</td>
</tr>
<tr>
<td>11.7.3.3.1</td>
<td>Search for Study Payload</td>
</tr>
<tr>
<td>11.7.3.3.2</td>
<td>Search for Series Payload</td>
</tr>
<tr>
<td>11.7.3.3.3</td>
<td>Search for Instance Payload</td>
</tr>
<tr>
<td>11.7.4</td>
<td>Media Types</td>
</tr>
<tr>
<td>11.7.5</td>
<td>Conformance</td>
</tr>
<tr>
<td>11.8</td>
<td>Retrieve Capabilities Transaction</td>
</tr>
<tr>
<td>11.9</td>
<td>Studies Service Notification Transactions</td>
</tr>
<tr>
<td>11.9.1</td>
<td>Resources</td>
</tr>
<tr>
<td>11.9.2</td>
<td>Open Notification Channel Transaction</td>
</tr>
<tr>
<td>11.9.3</td>
<td>Subscribe Transaction</td>
</tr>
<tr>
<td>11.9.3.1</td>
<td>Request</td>
</tr>
<tr>
<td>11.9.3.1.1</td>
<td>Request Header Fields</td>
</tr>
<tr>
<td>11.9.3.1.2</td>
<td>Payload</td>
</tr>
<tr>
<td>11.9.3.2</td>
<td>Behavior</td>
</tr>
<tr>
<td>11.9.3.3</td>
<td>Response</td>
</tr>
<tr>
<td>11.9.3.3.1</td>
<td>Status Codes</td>
</tr>
<tr>
<td>11.9.3.3.2</td>
<td>Header Fields</td>
</tr>
<tr>
<td>11.9.3.3.3</td>
<td>Payload</td>
</tr>
<tr>
<td>11.9.3.4</td>
<td>Media Types</td>
</tr>
<tr>
<td>11.9.4</td>
<td>Unsubscribe Transaction</td>
</tr>
<tr>
<td>11.9.4.1</td>
<td>Request</td>
</tr>
<tr>
<td>11.9.4.1.1</td>
<td>Request Header Fields</td>
</tr>
<tr>
<td>11.9.4.1.2</td>
<td>Request Payload</td>
</tr>
<tr>
<td>11.9.4.2</td>
<td>Behavior</td>
</tr>
<tr>
<td>11.9.4.3</td>
<td>Response</td>
</tr>
<tr>
<td>11.9.4.3.1</td>
<td>Status Codes</td>
</tr>
<tr>
<td>11.9.4.3.2</td>
<td>Response Header Fields</td>
</tr>
<tr>
<td>11.9.4.3.3</td>
<td>Response Payload</td>
</tr>
<tr>
<td>11.9.5</td>
<td>Notify</td>
</tr>
<tr>
<td>11.9.5.1</td>
<td>Request</td>
</tr>
<tr>
<td>11.9.5.1.1</td>
<td>Payload</td>
</tr>
<tr>
<td>11.9.5.1.1.1</td>
<td>Event Report Format</td>
</tr>
<tr>
<td>11.9.5.2</td>
<td>Behavior</td>
</tr>
<tr>
<td>11.9.5.3</td>
<td>Behavior</td>
</tr>
<tr>
<td>11.9.5.4</td>
<td>Response</td>
</tr>
<tr>
<td>11.10</td>
<td>Conformance</td>
</tr>
<tr>
<td>12</td>
<td>RS Unified Worklist Service (UPS)</td>
</tr>
<tr>
<td>12.1.1</td>
<td>Workitems</td>
</tr>
<tr>
<td>12.1.1.1</td>
<td>Deletion Locks</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>12.1.2</td>
<td>102</td>
</tr>
<tr>
<td>12.1.2.1</td>
<td>103</td>
</tr>
<tr>
<td>12.1.2.2</td>
<td>103</td>
</tr>
<tr>
<td>12.1.2.2.1</td>
<td>103</td>
</tr>
<tr>
<td>12.2</td>
<td>103</td>
</tr>
<tr>
<td>12.2.1</td>
<td>103</td>
</tr>
<tr>
<td>12.2.2</td>
<td>103</td>
</tr>
<tr>
<td>12.2.3</td>
<td>104</td>
</tr>
<tr>
<td>12.2.4</td>
<td>105</td>
</tr>
<tr>
<td>12.2.4.1</td>
<td>105</td>
</tr>
<tr>
<td>12.2.5</td>
<td>106</td>
</tr>
<tr>
<td>12.2.6</td>
<td>106</td>
</tr>
<tr>
<td>12.2.7</td>
<td>107</td>
</tr>
<tr>
<td>12.2.8</td>
<td>107</td>
</tr>
<tr>
<td>12.2.9</td>
<td>107</td>
</tr>
<tr>
<td>12.3</td>
<td>108</td>
</tr>
<tr>
<td>12.3.1</td>
<td>108</td>
</tr>
<tr>
<td>12.3.1.1</td>
<td>108</td>
</tr>
<tr>
<td>12.3.1.1.1</td>
<td>108</td>
</tr>
<tr>
<td>12.3.1.1.2</td>
<td>108</td>
</tr>
<tr>
<td>12.3.1.1.3</td>
<td>108</td>
</tr>
<tr>
<td>12.3.1.1.4</td>
<td>109</td>
</tr>
<tr>
<td>12.3.1.2</td>
<td>109</td>
</tr>
<tr>
<td>12.3.1.3</td>
<td>109</td>
</tr>
<tr>
<td>12.3.1.3.1</td>
<td>109</td>
</tr>
<tr>
<td>12.3.1.3.2</td>
<td>109</td>
</tr>
<tr>
<td>12.3.1.3.3</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2.1</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2.1.1</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2.1.2</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2.1.3</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2.1.4</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2.2</td>
<td>110</td>
</tr>
<tr>
<td>12.3.2.3</td>
<td>111</td>
</tr>
<tr>
<td>12.3.2.3.1</td>
<td>111</td>
</tr>
<tr>
<td>12.3.2.3.2</td>
<td>111</td>
</tr>
<tr>
<td>12.3.2.3.3</td>
<td>111</td>
</tr>
<tr>
<td>12.3.3</td>
<td>111</td>
</tr>
<tr>
<td>12.3.3.1</td>
<td>111</td>
</tr>
<tr>
<td>12.3.3.1.1</td>
<td>112</td>
</tr>
<tr>
<td>12.3.3.1.2</td>
<td>112</td>
</tr>
<tr>
<td>12.3.3.1.3</td>
<td>112</td>
</tr>
<tr>
<td>12.3.3.1.4</td>
<td>112</td>
</tr>
<tr>
<td>12.3.3.2</td>
<td>113</td>
</tr>
<tr>
<td>12.3.3.3</td>
<td>113</td>
</tr>
<tr>
<td>12.3.3.3.1</td>
<td>113</td>
</tr>
<tr>
<td>12.3.3.3.2</td>
<td>113</td>
</tr>
<tr>
<td>12.3.3.3.3</td>
<td>114</td>
</tr>
<tr>
<td>12.3.4</td>
<td>114</td>
</tr>
<tr>
<td>12.3.4.1</td>
<td>114</td>
</tr>
<tr>
<td>12.3.4.1.1</td>
<td>114</td>
</tr>
<tr>
<td>12.3.4.1.2</td>
<td>114</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>12.3.4.2 Behavior</td>
<td>114</td>
</tr>
<tr>
<td>12.3.4.3 Response</td>
<td>115</td>
</tr>
<tr>
<td>12.3.4.3.1 Status Codes</td>
<td>115</td>
</tr>
<tr>
<td>12.3.4.3.2 Header Fields</td>
<td>115</td>
</tr>
<tr>
<td>12.3.4.3.3 Payload</td>
<td>115</td>
</tr>
<tr>
<td>12.3.5 Complete Workitem Transaction</td>
<td>115</td>
</tr>
<tr>
<td>12.3.5.1 Request</td>
<td>116</td>
</tr>
<tr>
<td>12.3.5.1.1 Header Fields</td>
<td>116</td>
</tr>
<tr>
<td>12.3.5.1.2 Payload</td>
<td>116</td>
</tr>
<tr>
<td>12.3.5.2 Behavior</td>
<td>116</td>
</tr>
<tr>
<td>12.3.5.3 Response</td>
<td>116</td>
</tr>
<tr>
<td>12.3.5.3.1 Status Codes</td>
<td>116</td>
</tr>
<tr>
<td>12.3.5.3.2 Header Fields</td>
<td>116</td>
</tr>
<tr>
<td>12.3.5.3.3 Payload</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6 Cancel Workitem Transaction</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.1 Request</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.1.1 Header Fields</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.1.2 Payload</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.2 Behavior</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.3 Response</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.3.1 Status Codes</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.3.2 Header Fields</td>
<td>117</td>
</tr>
<tr>
<td>12.3.6.4 Payload</td>
<td>117</td>
</tr>
<tr>
<td>12.3.7 Search Worklist Transaction</td>
<td>118</td>
</tr>
<tr>
<td>12.3.7.1 Request</td>
<td>118</td>
</tr>
<tr>
<td>12.3.7.1.1 Query Parameters</td>
<td>118</td>
</tr>
<tr>
<td>12.3.7.1.2 Header Fields</td>
<td>118</td>
</tr>
<tr>
<td>12.3.7.1.3 Payload</td>
<td>119</td>
</tr>
<tr>
<td>12.3.7.2 Behavior</td>
<td>119</td>
</tr>
<tr>
<td>12.3.7.3 Response</td>
<td>119</td>
</tr>
<tr>
<td>12.3.7.3.1 Status Codes</td>
<td>119</td>
</tr>
<tr>
<td>12.3.7.3.2 Header Fields</td>
<td>119</td>
</tr>
<tr>
<td>12.3.7.3.3 Payload</td>
<td>120</td>
</tr>
<tr>
<td>12.3.7.4 Media Types</td>
<td>120</td>
</tr>
<tr>
<td>12.4 Worklist Notification Sub-Service</td>
<td>120</td>
</tr>
<tr>
<td>12.4.1 Resources</td>
<td>120</td>
</tr>
<tr>
<td>12.4.2 Open Notification Channel Transaction</td>
<td>121</td>
</tr>
<tr>
<td>12.4.3 Subscribe Transaction</td>
<td>121</td>
</tr>
<tr>
<td>12.4.3.1 Request</td>
<td>121</td>
</tr>
<tr>
<td>12.4.3.1.1 Header Fields</td>
<td>121</td>
</tr>
<tr>
<td>12.4.3.1.2 Payload</td>
<td>121</td>
</tr>
<tr>
<td>12.4.3.2 Behavior</td>
<td>121</td>
</tr>
<tr>
<td>12.4.3.3 Response</td>
<td>122</td>
</tr>
<tr>
<td>12.4.3.3.1 Status Codes</td>
<td>122</td>
</tr>
<tr>
<td>12.4.3.3.2 Header Fields</td>
<td>122</td>
</tr>
<tr>
<td>12.4.3.3.3 Payload</td>
<td>122</td>
</tr>
<tr>
<td>12.4.3.4 Media Types</td>
<td>122</td>
</tr>
<tr>
<td>12.4.4 Unsubscribe Transaction</td>
<td>122</td>
</tr>
<tr>
<td>12.4.4.1 Request</td>
<td>122</td>
</tr>
<tr>
<td>12.4.4.1.1 Header Fields</td>
<td>123</td>
</tr>
<tr>
<td>12.4.4.1.2 Payload</td>
<td>123</td>
</tr>
<tr>
<td>12.4.4.2 Behavior</td>
<td>123</td>
</tr>
<tr>
<td>12.4.4.3</td>
<td>Response</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>12.4.4.3.1</td>
<td>Status Codes</td>
</tr>
<tr>
<td>12.4.4.3.2</td>
<td>Header Fields</td>
</tr>
<tr>
<td><strong>12.4.4.3.3</strong></td>
<td><strong>Payload</strong></td>
</tr>
<tr>
<td>12.4.5</td>
<td>Send Notification</td>
</tr>
<tr>
<td>12.4.5.1</td>
<td>Request</td>
</tr>
<tr>
<td>12.4.5.1.1</td>
<td>Payload</td>
</tr>
<tr>
<td>12.4.5.1.1.1</td>
<td>Event Report Format</td>
</tr>
<tr>
<td>12.4.5.2</td>
<td>Behavior</td>
</tr>
<tr>
<td>12.4.5.3</td>
<td>Response</td>
</tr>
<tr>
<td><strong>12.5</strong></td>
<td><strong>Conformance</strong></td>
</tr>
</tbody>
</table>
List of Figures

6-1. Interaction Diagram ................................................................. Error! Bookmark not defined.
6.5-1. Mapping between IOD and HTTP message parts .......... Error! Bookmark not defined.
# List of Tables

6.4-1. Error Codes ............................................................... Error! Bookmark not defined.

6.5-1. Media Type Mapping to Transfer Syntax ............................. Error! Bookmark not defined.

6.5-2. Error Codes ............................................................... Error! Bookmark not defined.

6.6.1-1. HTTP/1.1 Standard Response Code ............................... Error! Bookmark not defined.


6.7.1-1. QIDO-RS STUDY Search Query Keys .............................. Error! Bookmark not defined.

6.7.1-1a. QIDO-RS SERIES Search Query Keys ............................ Error! Bookmark not defined.

6.7.1-1b. QIDO-RS INSTANCE Search Query Keys ........................ Error! Bookmark not defined.

6.7.1-2. QIDO-RS STUDY Returned Attributes ............................ 93

6.7.1-2a. QIDO-RS SERIES Returned Attributes .......................... 93

6.7.1-2b. QIDO-RS INSTANCE Returned Attributes ....................... 94

6.7.1. QIDO-RS HTTP/1.1 Status Codes ................................... Error! Bookmark not defined.

6.8.1. Resources and Methods ................................................ Error! Bookmark not defined.

6.8-2. Server Options HTTP/1.1 Status Codes .............................. Error! Bookmark not defined.

6.9-1. UPS Interface Mapping ................................................ Error! Bookmark not defined.

6.9-1. Status Codes ............................................................. Error! Bookmark not defined.

6.9-2. Status Codes ............................................................. 107


6.9-4. Status Codes ............................................................. Error! Bookmark not defined.

6.9-5. Status Codes ............................................................. Error! Bookmark not defined.

6.9-6. Status Codes ............................................................. Error! Bookmark not defined.

6.9-7. Status Codes ............................................................. Error! Bookmark not defined.

6.9-7-2. Status Codes .......................................................... Error! Bookmark not defined.

6.9-8-1. Status Codes .......................................................... Error! Bookmark not defined.

6.9-8-1. Status Codes .......................................................... Error! Bookmark not defined.

6.9-7-1. Status Codes .......................................................... Error! Bookmark not defined.

6.9-10-1. Status Codes ......................................................... Error! Bookmark not defined.

6.9-10-1. Status Codes ......................................................... Error! Bookmark not defined.

D-1. IANA Mapping ............................................................ Error! Bookmark not defined.

F.2.3-1. DICOM VR to JSON Data Type Mapping ............................ Error! Bookmark not defined.

F.3.1-1. XML to JSON Mapping ................................................ Error! Bookmark not defined.
Notice and Disclaimer

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

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

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

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

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

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

The DICOM Standard is structured as a multi-part document using the guidelines established in [ISO/IEC Directives, Part 3].

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

This part of the standard specifies web services for managing and distributing DICOM (Digital Imaging and Communications in Medicine) Information Objects, such as medical images, annotations, reports, etc. It defines services and their APIs using the HTTP family of protocols. It is intended to be used for the distribution of medical imaging studies and related information to healthcare organizations, providers, and patients.

Security considerations, including access control, authorization, and auditing are beyond the scope of this Part of the Standard.
2 Normative References

[TODO: add all new references from any section]

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.

**ebRS**
- ebXML Registry Service

**HL7 CDA**
- Health Level Seven, Clinical Document Architecture (CDA)

**IEC 61966-2.1**

**IETF RFC 822**

**IETF RFC 2045**

**IETF RFC 2046**

**IETF RFC 2387**

**IETF RFC 2978**

**IETF RFC 3240**

**IETF RFC 3536**

**IETF RFC 3986**

**IETF RFC 4627**

**IETF RFC 5234**

**IETF RFC 6338**

**IETF RFC 6365**

**IETF RFC 6455**

**IETF RFC 6570**

**IETF RFC 6838**
- Media Type Specifications and Registration Procedures <https://tools.ietf.org/html/rfc6838>

**IETF RFC 7405**
<table>
<thead>
<tr>
<th>RFC Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7230</td>
<td>Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing</td>
</tr>
<tr>
<td>7231</td>
<td>Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content</td>
</tr>
<tr>
<td>7232</td>
<td>Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests</td>
</tr>
<tr>
<td>7233</td>
<td>Hypertext Transfer Protocol (HTTP/1.1): Range Requests</td>
</tr>
<tr>
<td>7234</td>
<td>Hypertext Transfer Protocol (HTTP/1.1): Caching</td>
</tr>
<tr>
<td>7235</td>
<td>Hypertext Transfer Protocol (HTTP/1.1): Authentication</td>
</tr>
<tr>
<td>7236</td>
<td>Initial Hypertext Transfer Protocol (HTTP) Authentication Scheme Registrations</td>
</tr>
<tr>
<td></td>
<td>IHE IT Infrastructure Technical Framework, Volume 2x, Appendix V (Web Services for IHE Transactions)</td>
</tr>
<tr>
<td>10918</td>
<td>JPEG Standard for digital compression and encoding of continuous-tone still images</td>
</tr>
<tr>
<td>2022:1994</td>
<td>Information technology -- Character code structure and extension</td>
</tr>
<tr>
<td></td>
<td><a href="http://www.ecma-international.org/publications/standards/Ecma-035.htm">http://www.ecma-international.org/publications/standards/Ecma-035.htm</a></td>
</tr>
<tr>
<td>SUBM-wadl-20090831</td>
<td>Web Application Description Language (WADL), W3C Member Submission 31 August 2009</td>
</tr>
<tr>
<td>WS-I Basic Profile</td>
<td>WS-I Basic Profile Version 2.0 <a href="http://ws-i.org/profiles/BasicProfile-2.0-2010-11-09.html">http://ws-i.org/profiles/BasicProfile-2.0-2010-11-09.html</a></td>
</tr>
</tbody>
</table>
3 Terms and Definitions

[TODO: Add any new terms and their definitions – merge tables]

3.1 DICOM Conformance Definitions
This part of the standard makes use of the following terms defined in PS3.2:

- Conformance Statement

3.2 DICOM Information Object
This Part of the Standard makes use of the following terms defined in PS3.3:

- Multi-frame image

3.3 DICOM Service Class Specifications
This Part of the Standard 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
This Part of the Standard makes use of the following terms defined in PS3.5:

- Data Element
- Data Element Tag
- Unique Identifier (UID)

3.5 DICOM Message Exchange Definitions
This Part of the Standard makes use of the following terms defined in PS3.7:
DICOM Message Service Element (DIMSE)

3.6 DICOM Web Services

This Part of the Standard defines the following terms and definitions.

ABNF

25 Augmented Backus Naur Form. See [RFC5234] and [RFC7405]

Accept Query Parameter

A query parameter that specifies one or more media types acceptable for the representation(s) contained in the response. See Section 6.1.1.5.

Acceptable Character Sets

One or more character sets acceptable to the user agent in the response. See Section 6.1.2.1.

Acceptable Media Types

One or more media type acceptable to the user agent in the response. See Section 6.1.1.4.

Character-Set Query Parameter

A query parameter that specifies one or more character sets for the representation(s) contained in the response. See Section 6.1.2.2.

DICOM Media Type

One of the media types defined by this Standard. See Section 6.1.1.1.

DICOM Persistent 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 Persistent Object is referred to as a Composite Service Object Pair (SOP) Instance.

DICOM Resource Categories

A set of categories for the content of DICOM SOP Instances. Examples include images, video, and text.

DICOMweb

45 Marketing term for PS3.18 Restful Web Services API

Event Report

Notification

Notification Connection

Selected Character Sets

The character sets selected by the origin server for the response payload. See Section 6.1.2.4.

Selected Media Type

The media type selected by the origin server for the response payload. See Section 6.1.1.7.

Subscriber

Subscription
DICOM PS3.18 2015b - Web Services

Page 24

55 Rendered Media Type

A non-DICOM media type into which DICOM objects may be transformed in order to display them using commonly available non-DICOM software, for example browsers. See Section 6.1.1.2.

RESTful Web Service

A web service is RESTful if it is implemented using the REST architecture and principles <http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf>.

sRGB


UTF-8

Unicode UTF-8 character set

65 Workitem

A Workitem is a unit of work that represents a single step in a procedure.

Worklist

A Worklist is a collection of Workitems.

3.7 HyperText Transfer Protocol

This Part of the Standard makes use of the following terms defined in IETF RFC 7230 Section 2.1 Client/Server Messaging:

origin server
user agent
# 4 Symbols and Abbreviated Terms

[TODO: All these terms should have links to their defining documents.]

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABNF</td>
<td>Augmented Backus-Naur Form. See [RFC5234]</td>
</tr>
<tr>
<td>DICOM</td>
<td>Digital Imaging and Communications in Medicine</td>
</tr>
<tr>
<td>HL7</td>
<td>Health Level Seven</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>HTTP/1.1</td>
<td>Version 1.1 of the HyperText Transfer Protocol</td>
</tr>
<tr>
<td>HTTP/2</td>
<td>Version 2 of the HyperText Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>HyperText Transfer Protocol Secure</td>
</tr>
<tr>
<td>HTTPS/1.1</td>
<td>Version 1.1 of the HyperText Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS/2</td>
<td>Version 2 of the HyperText Transfer Protocol</td>
</tr>
<tr>
<td>IHE</td>
<td>Integrating the Healthcare Enterprise</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>RESTful</td>
<td>A web service is RESTful if it is implemented using the REST architecture and principles.</td>
</tr>
<tr>
<td>RS, RS API</td>
<td>The RESTful web service API defined in PS3.18, Section 9, 10, and 11</td>
</tr>
<tr>
<td>SOP</td>
<td>Service Object Pair</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol (SOAP12 for SOAP version 1.2)</td>
</tr>
<tr>
<td>Studies Service</td>
<td>The Restful Storage Service for DICOM Studies</td>
</tr>
<tr>
<td>UID</td>
<td>Unique (DICOM) Identifier</td>
</tr>
<tr>
<td>UPS Service</td>
<td>The RESTful Unified Procedure Step Service</td>
</tr>
<tr>
<td>URI, URI API</td>
<td>The URI API defined in Section 7 of this document.</td>
</tr>
<tr>
<td>WADL</td>
<td>Web Application Description Language</td>
</tr>
<tr>
<td>WS, WS API</td>
<td>The WS API defined in Section 8 of this document</td>
</tr>
<tr>
<td>WSDL</td>
<td>Web Services Description Language</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>
5 Conventions

This section defines conventions used throughout the rest of this Part of the Standard.

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]. This Standard also supports 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 [RFC 7230, Section ???]. For example:
   
   ```
   Accept:
   
   “Accept:” media-type CRLF
   ```

   is equivalent to

   ```
   “Accept:” media-type CRLF
   ```


In this standard, as with HTTP in general, resources are referenced by URIs [RFC3986]. Each service defines the resources it manages and 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.

Annex X contains the Combined ABNF for DICOM Web Services.

5.2.1 Common Syntactic Rules for Data Types

Table 5.1-1 defines the syntax of some common rules used in defining data values in this Standard.

<table>
<thead>
<tr>
<th>Name</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>= [+ / -] 1*DIGIT ; An integer</td>
</tr>
<tr>
<td>uint</td>
<td>= 1*DIGIT ; An unsigned integer</td>
</tr>
<tr>
<td>pos-int</td>
<td>= NON-ZERO-DIGIT *DIGIT ; An integer greater than zero</td>
</tr>
<tr>
<td>decimal</td>
<td>= int [&quot;.&quot; uint] [&quot;E&quot; / &quot;e&quot;] int ; a fixed or floating point number</td>
</tr>
<tr>
<td>string</td>
<td>= DQ 1*QCHAR DQ ; A case sensitive string</td>
</tr>
<tr>
<td>ci-string</td>
<td>= &quot;%s&quot; DQ 1*QCHAR DQ ; A case insensitive string</td>
</tr>
<tr>
<td>base64</td>
<td>; Use base64URI defined in [RFC4648 <a href="https://tools.ietf.org/html/rfc4648#section-5">https://tools.ietf.org/html/rfc4648#section-5</a>]</td>
</tr>
<tr>
<td>uid</td>
<td>= uid-prefix &quot;.&quot; uid-part [&quot;.&quot; uid-part]</td>
</tr>
<tr>
<td></td>
<td>uid-prefix = &quot;0&quot; / &quot;1&quot; / &quot;2&quot;</td>
</tr>
<tr>
<td></td>
<td>uid-part = &quot;0&quot; / pos-int</td>
</tr>
</tbody>
</table>
5.2.2 URI Templates
All target resources in request messages are specified using URI Templates [RFC6570]; however, the 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:

\[\text{varchar} = \%x20-21 / \%x23-7E / \text{pct-encoded}\]

5.2.3 List Extension Rule (\('#\)\)
The list extension rule used in this standard is similar to the comma-separated list extension rule defined in [RFC7230, Section 7 <https://tools.ietf.org/html/rfc7230#section7>]. The list extension rule used in PS3.18 does not allow empty lists, empty list elements, or the legacy list rules defined in [RFC7230, Section 7].

\[\begin{align*}
1\#\text{element} &= \text{element} *(\text{OWS }"," \text{ OWS element}) \\
\#\text{element} &= 1\#\text{element} \\
\langle n\rangle\langle m\rangle\text{element} &= \text{element} <n-1>*<m-1>(\text{OWS }"," \text{ OWS element})
\end{align*}\]

Where
\[n \geq 1 \text{ and } m > 1\]

5.3 Standard Query Parameters
This section specifies the standard syntax for various query parameters.

5.3.1 Image Rendering Parameters
[TODO]

5.3.2 Search Query Parameters
This section defines conventions for query parameter names and values for Search requests. The URI Template for the query parameters is:

\[\{?\text{search-parameter}\} = "\?" \{&\text{match}\} \{&\text{fuzzy}\} \{&\text{include}\} \{&\text{offset}\} \{&\text{limit}\}\]

Where
\[\{&\text{match}\} = *("\&" \text{ attribute } "=" 1\#\text{value})\]

Zero or more DICOM attribute/values pairs that specify matching criteria for the search. See Section10.6.1.2.1 below for requirements. See Section 6.5 for the detailed syntax of attribute/values.

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.

\[\{&\text{fuzzy}\} = "\&" ("Fuzzy" / "fuzzymatching") "=" ("true" / "false")\]

A single parameter that specifies whether the origin server shall perform Fuzzy Matching. This parameter is optional. If the parameter is not present it defaults to “false”, and Fuzzy Matching shall not be performed. If the value is "true" and the origin server supports Fuzzy Matching, then additional fuzzy semantic matching of Person Name attributes is performed in the manner specified in the DICOM Conformance Statement for the origin server, and in the Retrieve Capabilities response. See PS3.X Section Y.Z.

\[\{&\text{include}\} = *("\&" ("include" / "includefield") "=" 1\#\text{attribute}) / ("\&" ("include" / "includefield") "=" "all")\]

A parameter that specifies the attributes that the origin server should include in the response. The value is either a comma-separated list of attributes, or the single keyword "all", which means that all attributes of the object should be included in the response.
The request may contain more than one "includefield" parameter; however, if an "&includefield=all" parameter is present, no other "includefield" parameters shall be present.

{&offset} = "&" "offset" "=" uint

A single parameter that specifies the number of results the origin server shall skip before the first returned result. The "offset" parameter value is an unsigned integer. If this query parameter is not present, it defaults to 0.

The user agent can use the "offset" and "limit" parameters to paginate the results. See Section 10.6.2.2.

{&limit} = "&" "limit" "=" uint

A single parameter that specifies the maximum number of results the origin server shall return in a single response. The "limit" parameter value is an unsigned integer. If this parameter is not present, the origin server determines the default value.

See Section 7.5 for the general syntax of query parameters.

5.4 Header Field Documentation

In each header field section there is a table specifying header fields and their usage requirements. Table 5.4-1 specifies the usage symbols, types, and definitions.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Mandatory</td>
<td>The header field shall be present</td>
</tr>
<tr>
<td>C</td>
<td>Conditional</td>
<td>The header field shall be present if the condition is true.</td>
</tr>
<tr>
<td>O</td>
<td>Optional</td>
<td>The header field is optional</td>
</tr>
</tbody>
</table>

5.5 Format for Web Service Definitions

Each Web Service defined in this standard is described using the following format:

X <service-name>

Describe the service and its purpose.

X.1 Resources

Describe the resources managed by this service or that can be the target of transactions. It should include the specification of Target URIs using URI-templates.

X.2 Transactions

Gives an overview of the transactions defined by the service, including their purpose in relationship to the resources managed by this service.

Note

This Part of the Standard uses the term Transaction to denote an HTTP request/response pair.

This section includes one or more subsections containing Transaction Descriptions in the following format:

X.2.1 <transaction-name> Transaction

Describe the purpose and behavior of this transaction.

X.2.1.1 Request

This section has the form:
The <transaction-name> transaction uses the <method> method and has the following syntax:

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

The syntax above is fully general and it is used here as an example. The request syntax definition should be as specific as possible and should include useful optional header fields.

**X.2.1.1 Target Resource and URIs**
Specify the target resources for this transaction. URI Templates should be used to specify the format of Target URIs.

**X.2.1.2 Query Parameters**
Specify the Query Parameters for this transaction, including their syntax, and whether each parameter is required or optional. Reference the standard Query Parameters defined in Section 5.3.

**X.2.1.3 Request Header Fields**
Specify the Mandatory, Conditional, and most common optional header fields contained in the request. If the request has a payload some or all of the relevant Content Negotiation, Content Representation and Payload Header Fields [see [ref]] should be present. The table should have the same format as the Table in Section 7.6.2.

**X.2.1.4 Request Payload**
A description of the payload, including the contents, using standard phrases such as:

- The success payload, if any.
- The failure payload, if any.

**X.2.1.2 Behavior**
Describes the behavior of the origin server on receiving the request, including how to process it, and how to create the response.

**X.2.1.3 Response**
This section has the form:

```
The response has the following syntax:
version SP status-code SP reason-pharse CRLF
*(header-field CRLF)
CRLF
[payload]
```

The syntax above is fully general and it is used here as an example. The response syntax definition should be as specific as possible and should include useful optional header fields.

**X.2.1.3.1 Status Code**
Specify the most common status codes, and reason phrases, for this transaction.

Standard Phrases:

- "A success response shall include a success status code from Table 7.7."
- "A failure response shall include failure status code from Table 7.7, if applicable; otherwise, an appropriate status code from Table 7.7 shall be used."
X.2.1.3.2 Response Header Fields
A table describing the Mandatory, Conditional, and most common optional header fields contained in the response. If the response has a payload some or all of the relevant Content Representation and Payload Header Fields (see Section 7.6) should be present. The table should have the same format as the Table in Section 7.6.2.

For example:

Table X.Y-Z: Response Header Fields

X.2.1.3.3 Response Payload
Specify the success and failure payload contents, if any; otherwise, a statement that the response has no payload.

X.2.1.4 Media Types
Not all Transactions include a conformance section, but if present it describes any special information about the media types used with this transaction.

X.2.1.5 Conformance
Not all Transactions include a conformance section, but if present it describes any special information about conformance to this transaction.

X.2.2 <Another> Transaction
Describe the rest of the transactions.

X.3 Media Types
A description of media types supported by this Service, along with references to the appropriate Annex.

X.4 Conformance
A statement describing the conformance requirements of this transaction.
6 Conformance

An implementation claiming conformance to this standard shall function in accordance with all its mandatory sections. 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 this Part. 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. [disclaimer?]
7 Overview of Web Services APIs

The Web Service APIs defined in PS3.18 are defined in terms of the HTTP protocols. 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.

7.1 Services, and Transactions

PS3.18 defines APIs for Web services. A Web Service is a software system designed to support interoperable machine-to-machine interaction over a network. There are three Web Service APIs: URI, WS, and RS. Each API is named for the type of web services protocol that it uses in its implementation. Each API defines one or more services. Each service operates on a set of resources, called a resource group, and each service defines a set of transactions that operate on its resources. All transactions are defined in terms of HTTP/1.1 request/response message pairs.

All URI transactions use the query component of the URI in the request message to specify the transaction. The URI API defines one service, also called URI, which retrieves representations of its resources, which are all individual DICOM SOP Instances. The URI service defines two transactions that retrieve SOP Instances in different media types.

All WS API transactions are based on the WS-I Basic Profile and use the Simple Object Access Protocol (SOAP). The WS API provides a DICOM definition that is a superset of the equivalent IHE XDS-I.b transactions. The WS API defines one WS service that retrieves representations of its resources, which are DICOM Studies and their components. The WS service defines four transactions that retrieve DICOM objects in different media types.

All RS API transactions are based on RESTful web services. The RS APIs defines a Studies Service and a Worklist Service. The Studies Service defines all the transactions necessary to manage a storage system for DICOM studies (see Section 11). The Work List Service defines all the transactions necessary to manage a Unified Procedure Step Worklist (see Section 12).

7.2 Resources, Target URIs and Representations

A resource is an abstract concept that is reified by a representation, which is a data object encoded in an octet-stream. Each representation has a media type, which defines the format or encoding of the representation. Most of the resources defined in PS3.18 are either web services or Information Objects defined by the DICOM Information Model (see PS3.3, Section 6). A resource may contain a collection of sub-resources.

An origin server’s resources are the collection of resources that it manages, either directly or indirectly as a proxy for a (DIMSE) service.

For example, the RS Studies Service manages a collection of DICOM studies, where a study contains one or more series, a series contains one or more instances, and an instance may contain one or more frames. An Instance resource might be a single or multi-frame image, a video, a presentation state, a structured report, etc.

In this standard, as with HTTP in general, resources are referenced by URIs. Each service defines the collection of resources that it manages as well as the format of the URIs used to access those resources.

While a resource is an abstract concept, a representation is a concrete data object, i.e., an octet-stream that encodes that concept. Representations can be transmitted over the web and can be stored by clients and servers. A media type describes the format or encoding of a representation of a resource. Examples of media types are 'application/dicom', 'application/json', 'image/jpeg', and 'text/html'. Media types are defined in [RFC6838](http://tools.ietf.org/html/rfc6838).

7.3 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. The user agent always initiates a transaction by sending a request to the origin server. When it receives the request, the origin server processes it and returns a response.

**Figure 7.3-1:** Interaction Diagram for Transactions

![Interaction Diagram for Transactions](image)

7.3.1 General Format of Request Messages

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
Accept: 1#media-type CRLF
*(header-field CRLF)
CRLF [payload]
```

Where,

- **method** = DELETE, GET, HEAD, OPTIONS, POST, PUT, etc.
- **target-uri** = "/" {/resource} {{?parameters*}

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 Sections 5.2.2 and 7.4.

- **version** = %x48.54.54.50 "/" ("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).

- **Accept** = "Accept:" 1#media-range
  - All requests shall have an Accept header field that contains a comma-separated list of one or more media ranges (see Section X.Y).

- ***(header-field CRLF)**

Zero or more header fields each followed by a CRLF, which indicates the end of the header field. See [RFC7230]

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

An optional payload containing zero or more 8-bit OCTETs.
7.3.1.1 Method
The request method is one of the HTTP methods, such as CONNECT, DELETE, GET, HEAD, OPTIONS, POST, PUT, etc. See [RFC7230, Section 4] <https://tools.ietf.org/html/rfc7231#section-4>.

7.3.1.2 Target Resource URI
The target resource of a request is specified by a URI contained in the request message. URI Templates are used to specify the format of the target resource URI.
The most general template for a target resource URI is:

```
target-uri = "/" {resource} {?parameter}
```
See Sections 5.2.2 and 7.4 for details.

7.3.1.3 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 7.5.

7.3.1.4 Request Header Fields
Request header fields are used to specify metadata for the request. Most request have one or more Content Negotiation (see Section 7.6.2) header fields. If a request has a payload it should have Representation (see Section 7.6.3) and Payload (see Section 7.6.4) header fields.

7.3.1.5 Request Payload
The payload is an octet-stream containing the content of the message. See Section 7.7. The presence of a payload in a request is signaled by a Content-Length or Transfer-Encoding header field.

7.3.2 General Format of Response Messages
The syntax of a response message is:

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

```
version = "HTTP/1.1" / "HTTP/2" ; The version of the HTTP protocol being used
SP = %x20 ; The Space character
CRLF = %13 %10 ; Carriage Return followed by Linefeed characters
status-code = 3DIGIT ; An HTTP status code
reason-phrase = *( HTAB / SP / VCHAR )
header-field = field-name ":" OWS field-value OWS
payload = *octet
```
The origin server shall always return a response.

7.3.2.1 Status Codes
The response shall always include a valid 3-digit status code. Section 7.7 defines the standard 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.
7.3.2.2  Response Header Fields

Response header fields are used to specify metadata for the response. If the response has a payload it should have one or more Content Representations Header Fields (see Section 7.6.3) and one or more Payload Header Fields (see Section 7.6.4).

7.3.2.3  Response Payload

The payload is an <octet-stream> containing one or more representations. See Section 7.7.

The response typically defines two types of payloads: a success payload, and a failure payload.

7.4  Target Resource URI

Each service defines the format of its target resource URIs (aka Target URIs) and Query Parameters using URI Templates; and each transaction specifies which target resources it can reference. The target resource is also known as the request target or the target URI.

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).

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

The most general template for a target resource URI is:

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

or if any of the Query Parameters are required

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

Where

```
"/
```

The slash character (’/’) us used to designate the Base URI.

```
{/resource*}
```

A URI template for the Target Resource Path, a relative path component that references the target resource on the service. The "/" in the template indicates that path expansion is used. See [RFC6570, Section 3.2.6 <https://tools.ietf.org/html/rfc6570#section-3.2.6>].

"{/resource}" indicates the absolute URI to the target resource on a host machines.

```
{?parameters}
```

A URI Template for parameters contained in the query component of the URI. See 6.4 for details.

```
{required*}
```

A URI Template for one or more required parameters. See 8.1.1 for an example.

```
{optional*}
```

A URI Template for zero or more optional parameters. See 8.3.1.4 for an example.

The URI API (see Section 7) and WS API (see Section 8) do not define resources {/resource}, which means that the target URI of each transaction defined by these APIs is a reference to the Base URI ("/"). In contrast, the RS APIs define many different resources that can be the target of its transactions.

7.4.1  The Base URI of a Service

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 this Standard start with a "/", which is short hand that designates the Base URI of the Service.

For example, the URI:

```
http://dicom.nema.org/uri/service/studies/2.52.123456789
```

The Base URI is:

```
http://dicom.nema.org/service
```
7.4.2 Target Resource Path

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, for the URI:

http://dicom.nema.org/service/studies/2.52.123456789/series/2.52.987654321

The Base URI is:

http://dicom.nema.org/service

The Target Resource Path is:

/studies/2.52.123456789/series/2.52.987654321

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

7.5 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>]). This standard defines the syntax of Query Parameters.

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 in order 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-Style Query Expansion {?var} and Continuation {&var} as defined in [RFC6570, Sections 3.2.8 <https://tools.ietf.org/html/rfc6570#section-3.2.8> and 3.2.9 <https://tools.ietf.org/html/rfc6570#section-3.2.9>].

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 Query Parameters, they will conform to the syntax defined here.

Any transaction that specifies a query parameter will specify whether it is required or optional for the user agent or origin server.

The services and transactions defined elsewhere in this standard may further refine the <name> and <qp-value> rules.

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

An origin server may define additional parameters. If additional parameters are defined, they shall conform to the syntax specified in this section and they shall be documented in the conformance statement.

Origin servers shall ignore any unknown parameters.

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 ="" 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:

\[
\text{name} = "\%s"\ DQ\ 1*(\text{ALPHA} / \_\_\_\_)\ *(\text{ALPHA} / \text{DIGIT} / \_\_\_)\ DQ
\]

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, which is the set of legal query component characters as defined by [RFC3986], minus the equal ("="), ampersand ("&"), and comma (",") characters.

\[
\text{qp-value} = "\%s"\ DQ\ 1*\text{qp-char}\ DQ
\]

qp-char = unreserved / pct-encoded / qp-special

qp-special = "!" / "$" / "'" / "(" / ")" / ":" / @" / "/" / "?"

Non-qp-char characters may be included in qp-values if they are percent encoded.

There are two kinds of attribute simple and sequence:

\[
\text{attribute} = \text{simple-attribute} / \text{sequence-attribute}
\]

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:

\[
\text{simple-attribute} = \text{keyword} / \text{tag}
\]

\[
\text{keyword} = "\%s"\ DQ\ 1*\text{ALPHA}\ *(\text{ALPHA} / \text{DIGIT})\ DQ
\]

\[
\text{tag} = 8\text{HEXDIG}
\]

DICOM keywords are case sensitive; they must 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 must have a VR of SQ, and the last attribute must not have a VR of SQ.

\[
\text{sequence-attribute} = \text{keyword} *(\".\"\ \text{attribute}) / \text{tag} *(\".\"\ \text{attribute})
\]

The following are examples of valid values for <attribute>:

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

Some Query Parameters are an attribute with 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

Each Service defined in PS3.18 should specify:

- Whether or not an origin server may define additional parameters. If additional parameters are defined, they shall conform to the syntax specified in this section and they shall be documented in the conformance statement.
- Whether an origin server ignores any unknown parameters, or returns error responses.

### 7.5.1 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 7.5-1 contains the collected syntax of Query Parameters. Specific HTTP transactions defined elsewhere in this standard may further refine the qp-name and/or qp-value rules.

<table>
<thead>
<tr>
<th>Name</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>query-parameters</td>
<td>&quot;?&quot; parameter *(&quot;&amp;&quot; parameter)</td>
</tr>
<tr>
<td>parameter</td>
<td>parameter = name / name &quot;=&quot; 1#qp-value / name &quot;=&quot; 1#attribute / attribute / attribute &quot;=&quot; 1#qp-value ; a name only / a name with one or more values / a name with one or more attributes / an attribute only / an attribute with one or more values</td>
</tr>
<tr>
<td>qp-name</td>
<td>&quot;%s&quot; DQ (ALPHA / &quot;_&quot;) *(ALPHA / DIGIT / &quot;)&quot;) DQ</td>
</tr>
<tr>
<td>qp-value</td>
<td>int / uint / pos-int / decimal / float /string / base64 / uid / uuid  / &quot;%s&quot; DQ 1*QCHAR DQ ; See Section 5.2.1</td>
</tr>
<tr>
<td>qp-char</td>
<td>= unreserved / pct-encoded / qp-special</td>
</tr>
<tr>
<td>qp-special</td>
<td>= &quot;!&quot; / &quot;;&quot; / &quot;;&quot; / &quot;(&quot; / &quot;)&quot; / &quot;=&quot; / &quot;;&quot; / &quot;;&quot; ; &quot;;&quot; / &quot;;&quot; / &quot;@&quot; / &quot;;&quot; / &quot;?&quot;</td>
</tr>
<tr>
<td>simple-attribute</td>
<td>= keyword / tag</td>
</tr>
<tr>
<td>sequence-attribute</td>
<td>= keyword *(&quot;.&quot; attribute) / tag *( &quot;.&quot; attribute )</td>
</tr>
<tr>
<td>keyword</td>
<td>= &quot;%s&quot; DQ ALPHA *( ALPHANUM / DIGIT ) DQ</td>
</tr>
<tr>
<td>tag</td>
<td>= 8HEXDIG</td>
</tr>
</tbody>
</table>

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 ";".

DICOM keywords start with an alphabetic character followed by zero or more alphanumeric characters. See PS3.6.

The Services and Transactions defined elsewhere in this standard may further refine the qp-name, attribute, and qp-value rules.

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

7.6.1 Header Field Usage
In each header field section there is a table specifying header fields and their usage requirements. Table 6.6-1 specifies the usage symbols, types, and definitions.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Mandatory</td>
<td>The header field shall be present</td>
</tr>
<tr>
<td>C</td>
<td>Conditional</td>
<td>The header field shall be present if the condition is true.</td>
</tr>
<tr>
<td>O</td>
<td>Optional</td>
<td>The header field is optional</td>
</tr>
</tbody>
</table>

7.6.2 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 7.6-1 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 5.3.3 <https://tools.ietf.org/html/rfc7231#section-5.3.3>]

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>1#media-range</td>
<td>M</td>
<td>All requests that may receive a response that includes a payload shall contain an Accept header field. See Section 6.6.1.1.</td>
</tr>
<tr>
<td>Accept-Charset</td>
<td>1#charset</td>
<td>O</td>
<td>The Accept-Charset header field can be sent by a user agent to indicate what charsets are acceptable in textual response content. See [RFC7231, Section 5.3.3 <a href="https://tools.ietf.org/html/rfc7231#section-5.3.3">https://tools.ietf.org/html/rfc7231#section-5.3.3</a>].</td>
</tr>
<tr>
<td>Accept-Encoding</td>
<td>1#encoding</td>
<td>O</td>
<td>The Accept-Encoding header field is used to indicate the (content-codings<a href="https://tools.ietf.org/html/rfc7231#section-3.1.2.1">https://tools.ietf.org/html/rfc7231#section-3.1.2.1</a>), acceptable in the response. See [RFC7231, Section 5.3.4 <a href="https://tools.ietf.org/html/rfc7231#section-5.3.4">https://tools.ietf.org/html/rfc7231#section-5.3.4</a>].</td>
</tr>
<tr>
<td>Accept-Language</td>
<td>1#language</td>
<td>O</td>
<td>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 <a href="https://tools.ietf.org/html/rfc7231#section-5.3.5">https://tools.ietf.org/html/rfc7231#section-5.3.5</a>].</td>
</tr>
</tbody>
</table>

7.6.2.1 Content Negotiation Header Field General Parameters

7.6.2.1.1 Quality Value Parameter

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

435

weight = OWS ";" OWS "q=" qvalue

440

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].

7.6.2.1.2 Charset Parameter

Many media types, especially text/* types, define a "charset" parameter that specifies the character set for the representation.

445

DICOM media types define a "charset" parameter. See Annex X

7.6.2.2 Accept

User agents use the Accept header field to specify a list of media types that are acceptable in the response payload. See [RFC7231, Section 5.3.2 <https://tools.ietf.org/html/rfc7231#section-5.3.2>].

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

450

The media types in the Accept header can be given a priority ordering by using Quality Values. See Section 6.6.1.1.1.

See Section 7.9.5 for Acceptable Media Types,
See Section 7.10.1 for Acceptable Character Sets.

7.6.3 Content Representation Header Fields

455

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 7.6-2, describes the Content Representation Header Fields, and the requirements (Mandatory, Conditional, or Optional) for when they shall be present.
Table 7.6-2: Content Representation Header Fields

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>Specifies the media type of the representation contained in the payload. 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].</td>
</tr>
<tr>
<td>Content-Encoding</td>
<td>encoding</td>
<td>C</td>
<td>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]. Mandatory if any content encoding have been applied to the representation in the payload. Content-Encoding allow compression, encryption, and/or authentication of representations.</td>
</tr>
<tr>
<td>Content-Language</td>
<td>language</td>
<td>O</td>
<td>Specifies the natural language(s) of the intended audience used in representation. See RFC7231, Section 3.1.3.2.</td>
</tr>
<tr>
<td>Content-Location</td>
<td>uri</td>
<td>C</td>
<td>Contains a URI that references the specific resource corresponding to the representation in the payload. It should be present in any message that has a payload containing a representation of a resource.</td>
</tr>
</tbody>
</table>

7.6.4 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 7.6-3 describes the payload header fields, and the requirements (Mandatory, Conditional, or Optional) for when they shall be present.

Table 7.6-3: Payload Header Fields

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Length</td>
<td>uint</td>
<td>C</td>
<td>Specifies the decimal number of octets in the payload. It shall not be present if the message has a Content-Encoding header field. It should be present otherwise, even if the size of the payload is zero. 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.</td>
</tr>
<tr>
<td>Content-Range</td>
<td>range</td>
<td>C</td>
<td>Specifies the range of a partial representation contained in a payload. See [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.</td>
</tr>
<tr>
<td>Trailer</td>
<td>?</td>
<td>C</td>
<td>See [RFC7230, Section 4.4].</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>?</td>
<td>C</td>
<td>See [RFC7230, Section 3.3.1].</td>
</tr>
</tbody>
</table>

7.6.5 Conditional Request Header Fields
### Table 7.6-4: Conditional Header Fields

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETag</td>
<td>O</td>
<td>Specifies an entity-tag for the representation contained in the response. Required if... [look at FHIR!]</td>
</tr>
<tr>
<td>If-Match</td>
<td>O</td>
<td>Specifies a precondition, which is true if the entity tag in this header field matches the entity tag associated with current representation of the target resource; otherwise, it is false.</td>
</tr>
<tr>
<td>If-Non-Match</td>
<td>O</td>
<td>Specifies a precondition, which is true if none of the entity tags in the header field match an entity tag associated with the target resource matches, it is false.</td>
</tr>
<tr>
<td>If-Modified-Since</td>
<td>O</td>
<td>[ToDo]</td>
</tr>
<tr>
<td>If-Unmodified-Since</td>
<td>O</td>
<td>[ToDo]</td>
</tr>
<tr>
<td>If-Range</td>
<td>O</td>
<td>[ToDo] ???[talk to jonathan]</td>
</tr>
</tbody>
</table>

### 7.6.6 Caching Header Fields

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last-Modified</td>
<td>O</td>
<td>[ToDo]</td>
</tr>
<tr>
<td>If-Modified-Since</td>
<td>O</td>
<td>[ToDo]</td>
</tr>
<tr>
<td>If-Unmodified-Since</td>
<td>O</td>
<td>[ToDo]</td>
</tr>
</tbody>
</table>

### 7.7 Status Codes

Every response message contains a status-code, which specifies the type of the response message.

The most common HTTP status codes used are listed in Table 6.7-1. 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 6.7-1: Response Status Codes [TODO: Add link to each status code]

<table>
<thead>
<tr>
<th>Status</th>
<th>Code &amp; Phrase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td></td>
<td>The 2xx (Successful) class of status code indicates that the client's request was successfully received, understood, and accepted.</td>
</tr>
<tr>
<td>200</td>
<td>Success</td>
<td>Indicates that all target resource representations are contained in the payload.</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
<td>Indicates that the request has been fulfilled and has resulted in one or more new resources being created.</td>
</tr>
<tr>
<td>202</td>
<td>Accepted</td>
<td>Indicates that the request has been accepted for processing, but the processing has not been completed. The payload of this response should contain a Status Details document. [RFC7231, Section 6.3.3 <a href="https://tools.ietf.org/html/rfc7231#section-6.3.3">https://tools.ietf.org/html/rfc7231#section-6.3.3</a>]. The user agent can use a Retrieve Resource Status request to determine the status at a later time, or if the user agent has a Notification Channel open with the origin server, the origin server might send an event report when the requested transaction is complete.</td>
</tr>
<tr>
<td>Status Code</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>Non-Authoritative Information. Indicates that 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 5.7.2.</td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>No-Content. Indicates that the server has successfully fulfilled the request and that 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 use in the response to a Conditional Retrieve request (see Section X.Y), when the target resource has not been modified.</td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>Reset Content. Indicates that the server has fulfilled the request and desires that the user agent reset the &quot;document view&quot;, which caused the request to be sent, to its original state as received from the origin server. This code could be returned in response to a Work List Service Create Work Item request.</td>
<td></td>
</tr>
<tr>
<td>206</td>
<td>Partial Content. Indicates that some, but not all, of the target resource representations are contained in the payload. This could be because the origin server only supports the media types or transfer syntaxes for some but not all requested content. The response should contain a Status Details document specifying success, warning, or failure information for each part of the request payload. The Warning header field might also contain relevant information.</td>
<td></td>
</tr>
<tr>
<td>207</td>
<td>Multi-Status. Indicates the server has fulfilled the request by returning a new URI or document which can be used for future requests in lieu of the original request. The action requested by the original request might or might not have been carried out.</td>
<td></td>
</tr>
<tr>
<td>208</td>
<td>Partial Content. Indicates that some, but not all, of the target resource representations are contained in the payload. This could be because the origin server only supports the media types or transfer syntaxes for some but not all requested content. The response should contain a Status Details document specifying success, warning, or failure information for each part of the request payload. The Warning header field might also contain relevant information.</td>
<td></td>
</tr>
<tr>
<td>210</td>
<td>Also. Indicates that the server has successfully fulfilled the request and that 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 use in the response to a Conditional Retrieve request (see Section X.Y), when the target resource has not been modified.</td>
<td></td>
</tr>
<tr>
<td>226</td>
<td>Redirected. Indicates that the server has fulfilled the request and desires that the user agent reset the &quot;document view&quot;, which caused the request to be sent, to its original state as received from the origin server. This code could be returned in response to a Work List Service Create Work Item request.</td>
<td></td>
</tr>
<tr>
<td>227</td>
<td>Accepted. Indicates that the server has accepted the request but has not processed it yet. The request is redelivered to one or more intermediaries so that it can be processed.</td>
<td></td>
</tr>
<tr>
<td>301</td>
<td>Moved Permanently. The origin server has assigned the target resource to a new permanent URI. Thus status should only be needed when the resource has been moved from one service to another, for example during a migration.</td>
<td></td>
</tr>
<tr>
<td>302</td>
<td>Found. The target resource is available at another URI, and the original request should be redelivered to that URI.</td>
<td></td>
</tr>
<tr>
<td>303</td>
<td>See Other. The origin server is redirecting the user agent to a different resource, as indicated by a URI in the Location header field, which is intended to provide an indirect response to the original request. For example, this status might be used when the request is for a de-identified study, but the target resource is an identified study.</td>
<td></td>
</tr>
<tr>
<td>304</td>
<td>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.</td>
<td></td>
</tr>
<tr>
<td>307</td>
<td>Multi-Status. Indicates the server has fulfilled the request by returning a new URI or document which can be used for future requests in lieu of the original request. The action requested by the original request might or might not have been carried out.</td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>Bad Request. Indicates that 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 ...). Also indicates that no instances were stored due to bad message syntax.</td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized. Indicates that 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.</td>
<td></td>
</tr>
<tr>
<td>402</td>
<td>Unprocessable Entity. Indicates that the request is well-formed but cannot be processed by the server. The reason for the failure can be either transient or permanent.</td>
<td></td>
</tr>
<tr>
<td>403</td>
<td>Forbidden. Indicates that 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.</td>
<td></td>
</tr>
<tr>
<td>404</td>
<td>Not Found. Indicates that 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.</td>
<td></td>
</tr>
<tr>
<td>405</td>
<td>Method Not Allowed. Indicates that the request method is not valid for the target resource. The status code 405 (Method Not Allowed) is typically used to indicate that the request method is not valid for the target resource. For example, if a POST request is sent to a URI that is meant to accept only a GET request, the server could return a 405 (Method Not Allowed) response.</td>
<td></td>
</tr>
<tr>
<td>406</td>
<td>Not Acceptable. Indicates that the origin server is unable to fulfill the request because the server is unable to produce a representation that satisfies the request header fields Accept. This code is used when the server is unable to produce a representation that satisfies the Accept headers because the server cannot produce a representation that satisfies the Accept headers.</td>
<td></td>
</tr>
<tr>
<td>407</td>
<td>Proxy Authentication Required. Indicates that the client must authenticate itself with the proxy before the request is sent further. This code is used when the client must authenticate itself with the proxy before the request can be sent further. This code is used when the client must authenticate itself with the proxy before the request can be sent further.</td>
<td></td>
</tr>
<tr>
<td>408</td>
<td>Request Header Fields Too Large. Indicates that the request message contains header fields that are too large. The server may return a list of headers to include in a future request message in theshanry-headers field.</td>
<td></td>
</tr>
<tr>
<td>409</td>
<td>Request Payload Too Large. Indicates that the request payload is too large. The server may return a list of header fields to exclude from future requests in theshanghai-headers field.</td>
<td></td>
</tr>
<tr>
<td>410</td>
<td>Gone. Indicates that 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. 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.</td>
<td></td>
</tr>
<tr>
<td>411</td>
<td>要求头部字段太大。指示请求消息中包含的头部字段太大。服务器可以返回一个包含将来请求消息中应包括的头部字段的列表的shanry-headers域。</td>
<td></td>
</tr>
<tr>
<td>412</td>
<td>Precondition Failed. Indicates that the request is valid with respect to the if-match, if-none-match, if-modified-since, and if-unmodified-since request header fields, but a precondition specified by another request header field has failed. The server may return a list of headers to include in a future request message in theshanry-headers field.</td>
<td></td>
</tr>
<tr>
<td>413</td>
<td>Request Entity Too Large. Indicates that the request entity is too large. The server may return a list of header fields to exclude from future requests in theshanry-headers field.</td>
<td></td>
</tr>
<tr>
<td>414</td>
<td>Request URI Too Long. Indicates that the request URI is too long. The server may return a list of header fields to exclude from future requests in theshanry-headers field.</td>
<td></td>
</tr>
<tr>
<td>415</td>
<td>Unsupported Media Type. Indicates that the server is unable to produce a representation that satisfies the request header fields Accept. This code is used when the server is unable to produce a representation that satisfies the Accept headers because the server cannot produce a representation that satisfies the Accept headers.</td>
<td></td>
</tr>
<tr>
<td>416</td>
<td>Request Range Not Satisfiable. Indicates that the server is unable to satisfy the range request specified in the request header field. The server may return a list of header fields to include in a future request message in theshanry-headers field.</td>
<td></td>
</tr>
<tr>
<td>417</td>
<td>Proxy Authentication Required. Indicates that the client must authenticate itself with the proxy before the request is sent further. This code is used when the client must authenticate itself with the proxy before the request can be sent further. This code is used when the client must authenticate itself with the proxy before the request can be sent further.</td>
<td></td>
</tr>
<tr>
<td>418</td>
<td>Request Header Field Not Negotiable. Indicates that the request header field is not negotiable, and the server is unable to produce a representation that satisfies the request header fields Accept. This code is used when the server is unable to produce a representation that satisfies the Accept headers because the server cannot produce a representation that satisfies the Accept headers.</td>
<td></td>
</tr>
<tr>
<td>419</td>
<td>Upgrade Required. Indicates that the server requires the client to use a different protocol. The server may return a list of acceptable protocols in theProtocol Features header field.</td>
<td></td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error. Indicates that the server has encountered an unexpected condition that prevents it from fulfilling the request.</td>
<td></td>
</tr>
<tr>
<td>501</td>
<td>Gateway Time-out. Indicates that the gateway failed to receive a timely response from the target resource.</td>
<td></td>
</tr>
<tr>
<td>502</td>
<td>Bad Gateway. Indicates that the gateway is unable to process the request because it is a gateway and the requested service is not available at another gateway.</td>
<td></td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable. Indicates that the server is currently unable to process the request due to maintenance.</td>
<td></td>
</tr>
<tr>
<td>504</td>
<td>Gateway Timeout. Indicates that the gateway failed to receive a timely response from the target resource.</td>
<td></td>
</tr>
<tr>
<td>505</td>
<td>HTTP Version Not Supported. Indicates that the server does not support the HTTP version number used in the request.</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td></td>
</tr>
</tbody>
</table>
| 405  | Method Not Allowed  
Indicates that the method received in the request-line is known by the origin server but not supported by the target service or resource. The origin server shall generate an Allow header field in a 405 response containing a list of the target service or resource's currently supported methods. | |
| 406  | Not Acceptable  
Indicates that the target resource does not have a representation that would be acceptable to the user agent, according to 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 containing a Status Details document that lists the available media types and corresponding resource identifiers. | |
| 409  | Conflict  
Indicates that 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 a Status Details document that includes enough information for the user agent to recognize the source of the conflict.  
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  
Indicates that access to the target resource is no longer available at the origin server and that this condition is likely to be permanent. If the origin server does not know, or has no facility to determine, whether or not the condition is permanent, the 404 (Not Found) status code should be used instead. | |
| 411  | Length Required  
Indicates that the origin server refuses to accept the request because the Content-Length header field was not specified. | |
| 412  | Payload Too Large  
Indicates that the server is refusing to process a request because the request payload is larger than the server is willing or able to process. | |
| 413  | URI Too Long  
Indicates that 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. | |
| 414  | Unsupported Media Type  
Indicates that 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 containing a Status Details document that lists the available media types and corresponding resource identifiers.  
Note  
This is different from 409 (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. Except when responding to a HEAD request, the server should send a Status Details document containing an explanation of the error situation, and whether it is a temporary or permanent condition. |
| 500  | Internal Server Error  
Indicates that the server encountered an unexpected condition that prevented it from fulfilling the request. | |
| 501  | Not Implemented  
Indicates that the server does not support the functionality required to fulfill the request. This status code shall be used for SOP Class Not Supported errors. | |
| 503  | Service Unavailable  
Indicates that 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  
Indicates that the origin server does not support, or refuses to support, the major version of HTTP that was used in the request message. | |
7.8 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 transfer coding has been applied, as described in [RFC7230, Section 3.3.1]. This standard uses the term ‘payload’ to denote the message body before any transfer coding has been applied to it.

A message may or may not have a payload. If a message has a payload, then if a transfer-coding has been applied to it, the message shall have a Transfer-Encoding header field; otherwise, it shall have a Content-Length header field. The Transfer-Encoding header field and Content-Length header field shall not be present in the same message. If a message has neither a Transfer-Encoding nor Content-Length header fields, then the message shall not have a payload.

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 7.9).

It is recommended that any message with a payload includes a Content-Location header field, and either a Content-Length or a Transfer-Encoding header field (see RFC7230, Section x y).

The rules for when a payload is allowed in a message differ for requests and responses. See [RFC7230, Section 3.3](https://tools.ietf.org/html/rfc7230#section-3.3).

- The presence of a payload in a request is signaled by a Content-Length or Transfer-Encoding header field.
- The presence of a message body in a response depends on both the request method to which it is responding and the response status code.
- A message may have a payload that is empty, i.e. its length is zero.

Note

A message with no payload is different from a message with an empty payload.


7.8.1 Payload Format

Payloads shall be in Single Part or Multipart format.

7.8.1.1 Single Part Payload

A single part payload contains one representation that is described by the Content Representation Header Fields (see Section 7.6.3) 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.

7.8.1.2 Multipart Payload

A message with a multipart payload contains one or more representations. The media type of the root representations (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. Other header fields may be included, such as Content-Location, and either one of Content-Length or Transfer-Encoding header field.

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

```
request-line / response-line
Content-Type: multipart-media-type CRLF
Content-Location: /{resource} CRLF
```
The Content-Type header field shall have a multipart media-type such as:

```
Content-Type: multipart/related; type={root-media-type}; 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 Transfer-Encoding header field, optionally followed by other header fields.

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

### 7.8.1.2.1 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-ID: uid
```

```c
Content-Header-Fields
*(header-field CRLF)
CRLF
part-payload
```

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-Length: uint
[Content-Location: uri CRLF]
[Content-id: uid CRLF]
[Content-Description: {description} CRLF]
CRLF
payload CRLF
-++++CRLF
Content-Type: media-type CRLF
...
payload CRLF
-++++CRLF
Content-Type: media-type CRLF
...
payload CRLF
-+++++
```

--- Draft ---
7.8.2 DICOM Representations

All DICOM Information Objects (DIOs) are defined by Information Object Definitions (IODs). See PS3.3. DICOM representations 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 ValueMultiplicity, 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 Annex X [TODO]. Other media types used in PS3.18 are defined in Annex Y [TODO].

7.8.2.1 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.

7.8.2.2 Web Service Constraints

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

7.8.2.3 Metadata and Bulkdata

In a Composite SOP Instance Dataset each Data Element has a Value Field. However, some Data Elements such as Pixel Data (7FE0,0010) can be very large, containing kilobytes to gigabytes of data. In order to improve performance of applications that need to quickly retrieve enough Study data to make decisions about such things as hanging protocols, relevant priors, post processing, etc., some DICOM Web Services (WS and RS) allowDatasets to be separated into Metadata and Bulkdata.

**Metadata Dataset** A Metadata Dataset, aka Metadata, is a Dataset that contains one or more Bulkdata References.

**Bulkdata Reference** A Bulkdata Reference is a Data Element whose Value Field contains a URI that references a Bulkdata Value Field.

Bulkdata References are opaque. Their structure and format are not defined by this Standard, but rather by the origin server that implements them. Bulkdata References are only contained in Metadata resources, and thus a Bulkdata Value can only be accessed by retrieving a Metadata object that contains a Bulkdata Element referencing the Value.

**Bulkdata Element** A Bulkdata Element is a Data Element with one of the following Value Representations: DS, FL, FD, IS, LT, OB, OD, OF, OW, SL, SS, ST, UC, UL, UN, US, and UT.

**Bulkdata** A Bulkdata resource contains one or more Bulkdata values. Bulkdata resources and their representations are opaque. Their structure and format are not defined by this Standard, but rather by the origin server that implements them. Bulkdata Values are accessed with Bulkdata References.

**Bulkdata Value Field** A Bulkdata Value Field is a Value Field contained in a Bulkdata resource. A Bulkdata Value Field can be accessed using a Bulkdata Reference.

Large Data Elements having their Value Fields moved to separate objects, called Bulkdata objects, and replaced with URIs that reference them. Such Data Elements are called Bulkdata Elements.

The Metadata of a Dataset contains the same Data Elements as the Dataset, but with some large Data Elements having their Value Fields moved to separate objects, called Bulkdata objects, and replaced with URIs that reference them. Such Data Elements are called Bulkdata Elements.

The Metadata equivalent of a Dataset is typically 10 to 100s of times smaller and can thus be retrieved much more quickly.

The format and structure of Metadata representation depends on its media type. **See Annex X.**

The format and structure of Bulkdata objects and Bulkdata References are opaque. They are not defined by this Standard, but rather by the origin server implementing them. User agents do not need to understand their structure to use them.
Figure 7.8-2: Data Element and Equivalent Bulkdata Element

Figure 7.8-1 shows the format of an abstract Data Element and its corresponding Bulkdata Element. The length of the Value Field is contained in the Value Field.

7.8.2.3.1 Metadata and Bulkdata Definitions
The following definitions define various concepts related to Metadata and Bulkdata. The structure of Datasets and Metadata are defined by the media type. The structure and format of Bulkdata and Bulkdata References are defined by the origin server implementing them.

Metadata
- **Metadata** is defined to be a DICOM Dataset that has one or more Bulkdata References.
- **Study Metadata** is defined to be all Study, Series, and Instance data elements in a specified Study.
- **Series Metadata** is defined to be all Study, Series, and Instance data elements in a specified Series.
- **Instance Metadata** is defined to be all Study, Series, and Instance data elements in a specified Instance.

Bulkdata Reference
A Bulkdata Reference is a Data Element that contains a URI that references a Bulkdata Value.

7.8.2.3.2 Converting SOP Instances to/from Metadata and Bulkdata
[TODO: Converting Composite SOP Instance to/from Metadata and Bulkdata.]
An origin server may convert any Bulkdata Element into a Bulkdata Reference. Typically, Bulkdata Elements are only converted to Bulkdata References when their Value Field is larger than the Bulkdata Threshold, which is defined by the origin server. This threshold is typically in the range: 128 <= threshold <= 4096 bytes.

7.9 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 RS API. See Section 7.8.1.2 and Section 10.4.4.2.
Media types are identifiers used to define the data format of a representation. HTTP uses media types in the Content-Type and Accept header fields in order to provide open and extensible data typing and type negotiation. The syntax of media types is:

\[
\text{media-type} = \text{type} "/" \text{subtype} *(\text{OWS }";" \text{OWS mt-parameter})
\]

Where

- \text{type} = \text{token}
- \text{subtype} = \text{token}
- \text{mt-parameter} = \text{mtp-name} "=" \text{mtp-value}
- \text{mtp-name} = \text{token}
- \text{mtp-value} = (\text{token} / \text{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 "charset" 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].

### 7.9.1 Multipart Media Types

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

\[
\text{multipart-media-type} = "multipart" "/" \text{subtype} *(\text{OWS }";" \text{OWS parameter})
\]

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

\[
\text{multipart-related} = "multipart/related"
\]

Where

- \text{boundary} ; See Section 7.8.1.2.1
- \text{bchar} = \text{bchar-nospace} / \text{SP}
- \text{bchar-nospace} = \text{DIGIT} / \text{ALPHA} / "-" / ";" / ";" / ";" / ";" / _-
  / ";" / /[" / [" / ";" / [" / ";" / ";"
- \text{related-parameters} = [";" "start" =" cid]
- \text{cid-list} = \text{cid} \text{cid-list}
- \text{cid} = \text{token} / \text{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.
7.9.2 DICOM Resource Categories

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

<table>
<thead>
<tr>
<th>Resource Category</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Frame Image</td>
<td>This category includes all resources that:</td>
</tr>
<tr>
<td></td>
<td>1) are instances of a single frame SOP Class, or</td>
</tr>
<tr>
<td></td>
<td>2) are instances of a multi-frame SOP Class that contain only one frame, or</td>
</tr>
<tr>
<td></td>
<td>3) are a single frame selected from an instance of a multi-frame SOP Class.</td>
</tr>
<tr>
<td>Multi-Frame Image</td>
<td>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.</td>
</tr>
<tr>
<td>Video</td>
<td>This category includes all resources that contain more than one frame and:</td>
</tr>
<tr>
<td></td>
<td>1) are instances encoded in the MPEG family of transfer syntaxes (which includes MP4 and H265), or</td>
</tr>
<tr>
<td></td>
<td>2) are time based (motion) multi-frame images that the origin server is capable of encoding in the MPEG family.</td>
</tr>
<tr>
<td>Text</td>
<td>This category includes all resources that:</td>
</tr>
<tr>
<td></td>
<td>1) contain the SR Document Content Module (see PS3.4, Section C.17.3), such as narrative text, structured reports,</td>
</tr>
<tr>
<td></td>
<td>CAD, measurement reports, and key object selection documents, or</td>
</tr>
<tr>
<td></td>
<td>2) contain the Encapsulated Document Module (see PS3.4, Section C.24.2).</td>
</tr>
<tr>
<td>Other</td>
<td>This category includes all resources that are not included above.</td>
</tr>
</tbody>
</table>

7.9.3 DICOM Media Types

[TODO: Insert text from CP1509 when final]

7.9.4 Rendered Media Types

DICOM resources may be converted into non-DICOM media types in order to render them using commonly available non-DICOM software, such as browsers.

For example:

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

Note: Rendered Media Types are usually consumer format media types.

Table 6.1.1-2 specifies the meaning of media type requirements in Table 6.1.1-3.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>The origin server shall support all default media types.</td>
</tr>
<tr>
<td>Required</td>
<td>The origin server shall support these media types.</td>
</tr>
<tr>
<td>Optional</td>
<td>The origin server may support these media types.</td>
</tr>
</tbody>
</table>

Table 6.1.1-3 defines the Rendered Media Types by their Resource Category for the URI, WS, and RS modes.
Table 6.1.1-3: Rendered Media Types by Resource Category

<table>
<thead>
<tr>
<th>Category</th>
<th>Media Type</th>
<th>URI</th>
<th>WS</th>
<th>RS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Frame Image</td>
<td>image/jpeg</td>
<td>default</td>
<td>default</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>image/gif</td>
<td>optional</td>
<td>optional</td>
<td>required</td>
</tr>
<tr>
<td></td>
<td>image/png</td>
<td>optional</td>
<td>optional</td>
<td>required</td>
</tr>
<tr>
<td></td>
<td>image/jp2</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>Multi-Frame Image</td>
<td>image/gif</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>Video</td>
<td>video/mpeg</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td></td>
<td>video/mp4</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td></td>
<td>video/H265</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>Text</td>
<td>text/html</td>
<td>default</td>
<td>default</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>text/plain</td>
<td>required</td>
<td>required</td>
<td>required</td>
</tr>
<tr>
<td></td>
<td>text/xml</td>
<td>optional</td>
<td>optional</td>
<td>required</td>
</tr>
<tr>
<td></td>
<td>text/rtf</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td></td>
<td>application/pdf</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
</tbody>
</table>

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

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

The origin server may support additional rendered media types.

7.9.5 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 default media type for the target resource, if any.

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

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

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

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

The <accept> query parameter has the following syntax:

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

Note
The "%s" that prefixes the accept-name specifies that it is a case sensitive token. See [RFC7405]. Its value is a comma-separated list of one or more media-types, possibly including parameters. It shall be supported by the origin server. It is optional for the user agent.

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

<table>
<thead>
<tr>
<th>Service</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>URI</td>
<td>accept-name = &quot;contentType&quot;</td>
</tr>
<tr>
<td>WS</td>
<td>not applicable</td>
</tr>
<tr>
<td>RS</td>
<td>accept-name = &quot;accept&quot;</td>
</tr>
</tbody>
</table>

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

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

Note:
For example, the presence of image/jpeg in the <accept> query parameter will require the Accept header field to include one of the following values: image/jpeg, image/*, or */*. Typically, the angle brackets are not used when syntactic variables appear in text, but in the case of <accept> they are used to avoid any confusion.

7.9.7 Accept Header Field

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

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

Where,

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

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

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

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

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

7.9.8 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].

The Selected Media Type is chosen as follows:

1. Select the target’s Resource Category
2. Select the representation with the highest priority supported media type for that category in the <accept> query parameter, which is compatible with the Accept header field.
3. If no media type in the <accept> query parameter is supported, select the highest priority supported media type for that category in the Accept header field, if any.

4. Otherwise, select the default media type for the category if the Accept header field contains a wildcard media range matching the category, if any.

5. Otherwise, return a 406 (Not Acceptable).

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

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

For example, consider an origin server 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>
<thead>
<tr>
<th>Media Type</th>
<th>qvalue</th>
<th>Determining Media Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>text/html; level=1</td>
<td>1.0</td>
<td>text/html; level=1</td>
</tr>
<tr>
<td>text/html; level=2</td>
<td>0.7</td>
<td>text/html; level=2</td>
</tr>
<tr>
<td>text/plain</td>
<td>0.5</td>
<td>text/*</td>
</tr>
<tr>
<td>text/rtf</td>
<td>0.5</td>
<td>text/*</td>
</tr>
<tr>
<td>text/html</td>
<td>0.4</td>
<td>text/html</td>
</tr>
<tr>
<td>text/x-latex</td>
<td>0.4</td>
<td><em>/</em></td>
</tr>
</tbody>
</table>

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.

7.10 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 be identified by using the DICOM Defined Terms for the character set. See PS3.3, Section C.12.1.1.2, and PS3.5, Section 6.1.2.3.

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

The syntax is:

```
charset = token / defined-term / 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.12.1.2.

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 RS Services shall also document all supported character sets.

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

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

7.10.1 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.

7.10.2 Character Set Query Parameter

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

The character-set query parameter has the following syntax:

\[
\text{character-set} = \text{charset} "\text{=} 1#(\text{charset [weight]})
\]

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

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

The qp-name of the character-set query parameter is defined by the Service. **Table 7.10-1** contains the names of the character-set query parameter for some services.

<table>
<thead>
<tr>
<th>Service</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>URI</td>
<td>name = &quot;charset&quot;</td>
</tr>
<tr>
<td>WS</td>
<td>not applicable</td>
</tr>
<tr>
<td>RS Studies</td>
<td>name = &quot;charset&quot;</td>
</tr>
</tbody>
</table>

7.10.3 Character Set Media Type Parameters

[TODO]
7.10.4 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.

7.10.5 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 Unicode UTF-8 character set for RS Rendered Retrieve.

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

1. The user agent may omit the Accept-Charset header field or send the "*" wildcard.
2. The user agent may transcode the character set replacing all unknown characters with a suitable replacement. For example:

- A question mark ("?"), or other similar character indicating an unknown character.
- The corresponding Unicode Code Point for the character, represented as "U+xxxx".
- The four characters "\nnn", where "nnn" is the 3-digit octal representation of each byte (see PS3.5, 6.1.2.3).

7.11 Conformance

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

This standard does not introduce any security-related requirements. However, it is very likely that DICOM objects contain Individually Identifiable Information. Many privacy regulations, such as HIPAA in the United States and (Europe or elsewhere) require that all Individually Identifiable Information be kept private. It is the responsibility of implementers of the DICOM standard to ensure that all required governmental regulations are satisfied.

The HTTPS protocol can be used to protect information contained in request and/or response messages.

Each origin server decides whether or not to grant a user agent access to a particular DICOM object based on the security policy and mechanism(s) it implements. An origin server is unlikely to fulfill a request from an unknown user agent (e.g., accessed via the HTTP protocol) unless it is certain that the target resource of a request contains no Individually Identifying Information and has been approved for public access.
8 **WADO-URI Service API**

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.

### Note
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.

#### 8.1 General Query Parameters

The Query Parameters defined in this section may be used with either the Retrieve DICOM Instance or Retrieve Rendered Instance transactions.

#### 8.1.1 Required Query Parameters

The Query Parameters defined in this section shall be present in all URI requests. Both the origin server and user agent shall support them.

##### 8.1.1.1 Request Type

```
requestType = %"requestType=WADO"
```

This query parameter identifies this as a URI request message.

If this parameter has a value that is not "WADO", the origin server shall return a 409 (Conflict) response, and should include a payload containing an appropriate error message.

##### 8.1.1.2 Study UID

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

The value of this parameter is a Study Instance UID as defined in PS3.3. The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length.

If this parameter has a value that is not a Study Instance UID, the origin server shall return a 409 (Conflict) response, and should include a payload containing an appropriate error message.

##### 8.1.1.3 Series UID

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

The value of this parameter is a Series Instance UID as defined in PS3.3. The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length.

If this parameter has a value that is not a Series Instance UID, the origin server shall return a 409 (Conflict) response, and should include a payload containing an appropriate error message.

##### 8.1.1.4 Instance UID

```
instance = "&objectUID=" uid
```

The value of this parameter is a SOP Instance UID as defined in PS3.3. The value shall be encoded as a Unique Identifier (UID) string, as specified in PS3.5, except that it shall not be padded to an even length.
If this parameter has a value that is not a SOP Instance UID, the origin server shall return a 409 (Conflict) response, and should include a payload containing an appropriate error message.

### 8.2 WADO-URI Retrieve DICOM Instance Transaction

This transaction retrieves a single DICOM SOP Instance in the 'application/dicom' media type. See Section 7.X.Y.

#### 8.2.1 Request Message

The Retrieve DICOM Instance request uses the GET method and has the following syntax:

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

Where

- `{requestType}`; See Section 8.1.1.1
- `{study}`; See Section 8.1.1.2
- `{series}`; See Section 8.1.1.3
- `{instance}`; See Section 8.1.1.4
- `{&accept}`; See Section 8.1.2.1
- `{&charset}`; See Section 8.1.2.2
- `{&anonymize}`; See Section 8.2.1.2.1
- `{&transferSyntax}`; See Section 8.2.1.2.2

#### 8.2.1.1 Target Resource

The target resource shall be an Instance of a Composite SOP class defined in PS3.3.

#### 8.2.1.2 Query Parameters

All parameters defined in this section are optional for the origin server and user agent.

##### 8.2.1.2.1 Acceptable Media Type

```
accept = "contentType=" media-type
```

Note

The name of this production is "accept"; although the query parameter name is "contentType" for historical reasons.

The value of this parameter if present shall be 'application/dicom'. If 'application/dicom' is present no other media types shall be present. If other media types are present, the response shall be 409 (Conflict), and should include a payload containing an appropriate error message.

See Section X.Y.

##### 8.2.1.2.2 Character Set

```
charset = "charset=" *#charset
```

The value of this parameter is a comma-separated list of one or more character set identifiers.
If this parameter specifies an invalid character set, the response shall be 409 (Conflict), and should include a payload containing an appropriate error message.

**8.2.1.3 Anonymize**

anonymize = %s"anonymize=""yes" / "no"

This parameter specifies that the returned representations shall remove [Individually Identifiable Information (III)](https://example.com) if any, removed, as defined in [PS3.15 Annex E Basic Profile with Clean Pixel Data Option](https://example.com). Its name is "anonymize" and its value is "yes" or "no". If this parameter is not present it defaults to "no".

If the target resource has not already been de-identified, the returned instance shall have a new SOP Instance UID.

Anonymize

8.2.1.2.3

995

If this parameter specifies an invalid value, the response shall be 409 (Conflict), and should include a payload containing an appropriate error message.

The origin server may return a failure response if it is either unable or refuses to anonymize the target resource.

**8.2.1.4 Transfer Syntax**

transfer-syntax = %s"transferSyntax" "=" 1#transfer-syntax-uid

This parameter specifies a Transfer Syntax UID, as defined in PS3.6. Its name is "transferSyntax" and its value is a comma-separated list of one or more Transfer Syntax UIDs. It is optional for the user agent and origin server.

Implicit VR Little Endian and Explicit VR Big Endian transfer syntaxes shall not be used. If this parameter specifies an invalid Transfer Syntax, the response shall be 409 (Conflict), and should include a payload containing an appropriate error message.

**8.2.1.3 Request Header Fields**

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>media-range</td>
<td>M</td>
<td>Shall result in a Selected Media Type of 'application/dicom'. <a href="https://example.com">See Section 7.9.7</a>.</td>
</tr>
<tr>
<td>Accept-Charset</td>
<td>1#charset</td>
<td>O</td>
<td><a href="https://example.com">See Section 7.10</a>.</td>
</tr>
</tbody>
</table>

**8.2.4 Request Payload**

The request has no payload.

**8.2.2 Behavior**

The response shall contain the requested Composite SOP Instance.

**8.2.3 Response Message**

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]

**8.2.3.1 Status Codes**

See Section 7.7.
8.2.3.2 Response Header Fields

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>C</td>
<td>Required if response contains a payload</td>
</tr>
<tr>
<td>Content-Encoding</td>
<td>encoding</td>
<td>C</td>
<td>Required if response payload has a content coding</td>
</tr>
<tr>
<td>Content-Length</td>
<td>int</td>
<td>C</td>
<td>Required if response payload does not have a content coding</td>
</tr>
<tr>
<td>Content-Location</td>
<td>uri</td>
<td>O</td>
<td>Optional if response has a payload</td>
</tr>
</tbody>
</table>

See Section 7.1.3.2.

1030 8.2.3.3 Response Payload

A successful response shall have a payload containing the target resource as a PS3.10 octet-stream.

A failure response may include a payload describing the error.

8.2.4 Acceptable Media Types

The Acceptable Media Types shall contain only 'application/dicom'. See Section 7.X.Y.

1035 8.3 WADO-URI Retrieve Rendered Instance Transaction

This transaction returns a single DICOM SOP Instance in a Rendered Media Type. See Section 7.9.3.

8.3.1 Request Message

The Request Rendered Instance request uses the GET method and has the following syntax:

```
GET SP /??{requestType}&{study}&{series}&{instance}&{&frames}
```

Where

- `{requestType}`; See Section 8.1.1.1
- `{study}`; See Section 8.1.1.2
- `{series}`; See Section 8.1.1.3
- `{instance}`; See Section 8.1.1.4
- `{accept}`; See Section 8.3.1.3.1
- `{&charset}`; See Section 8.3.1.3.2
- `{&frame}`; See Section 8.3.1.4.1
- `{&annotation}`; See Section 8.3.1.4.2
- `{&rows}`; See Section 8.3.1.4.3.1
- `{&columns}`; See Section 8.3.1.4.3.2

--- Draft ---
8.3.1.1 Target Resource

The target resource shall be a Composite SOP Instance.

8.3.1.2 Query Parameters

All parameters defined in this section are optional for the origin server and user agent.

8.3.1.3 Common Query Parameters

8.3.1.3.1 Acceptable Media Type

\[
\text{accept} = \text{"contentType=rendered-media-type"}
\]

Note
The name of this production is "accept"; although the query parameter name is "contentType" for historical reasons.

The value of this parameter if present shall be a Rendered Media Type. See Section X.Y.

If this parameter specifies both DICOM and Rendered Media Types, the response shall be 409 (Conflict), and should include a payload containing an appropriate error message.

8.3.1.3.2 Character Set

\[
\text{charset} = \text{"charset=1#charset"}
\]

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

If this parameter specifies an invalid character set, the response shall be 409 (Conflict), and should include a payload containing an appropriate error message.

8.3.1.4 Image Related Query Parameters

The Query Parameters in this section shall only be included if the DICOM Category of the target resource is Single Frame, Multi-Frame, or Video as defined in Section 6.10.1.3.4.

8.3.1.4.1 Frame Number

\[
\text{frame-number} = \%s\text{"frameNumber"} = \text{"} \text{uint}
\]

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).

If the target resource is not a multi-frame image and the frame number is not 1; or if the frame number is greater than the number of frames in the instance, then the response shall be a 409 (Conflict), and should include a payload containing an appropriate error message.

8.3.1.4.2 Image Annotation

\[
\text{annotation} = \%s\text{"annotation"} \%s\text{"patient"} / \%s\text{"technique"}
\]

This parameter specifies that the returned images shall have corner annotations. Its name is "annotation" and its value is a comma-separated list of one or more of the following keywords:

"patient" indicates patient information annotations, such as patient name, birth date, etc.
"technique" indicates procedure technique annotations, such as image number, study date, image position, etc.

The exact nature and presentation of the annotations is determined by the origin server. The annotations shall be burned into the returned image pixels.

If this parameter is not present, no additional annotation may be burnt in.

Note

The DICOM image that is the target resource may already contain burned in text in the Pixel Data that is beyond the control of this parameter.

When used in conjunction with a presentation state instance, the annotations shall be applied after the presentation state has been rendered. When used in conjunction with the region parameter, the annotations shall be applied after the selection of the region.

The origin server may define additional values for this parameter.

The origin server shall ignore any values it does not support.

8.3.1.4.3 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.

If this parameter specifies viewport dimensions that are either ill-defined or not supported, then the response shall be a 409 (Conflict), and should include a payload containing an appropriate error message.

8.3.1.4.3.1 Viewport Rows

rows = %s"rows" "=" uint

This parameter specifies the number of pixel rows in returned images. 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.

8.3.1.4.3.2 Viewport Columns

columns = %s"columns" "=" uint

This parameter specifies the number of pixel columns in returned images. 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.

8.3.1.4.4 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 top row of the region
- ymin the left column of the region
- xmax the bottom row of the region
- ymax the right column 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 corresponds to the top row and left column of the image, and
- 1.0 corresponds to the bottom row and right column of the image.
and

\[ 0.0 \leq \text{xmin} < \text{xmax} \leq 1.0 \]
\[ 0.0 \leq \text{ymin} < \text{ymax} \leq 1.0 \]

Note

This means that the bottom right hand corner values cannot be 0.0.

If this parameter is supported, an image corresponding to the specified normalized coordinates shall be returned in the response.

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.

If this parameter specifies an ill-defined region, the response shall be a 409 (Conflict), and should include a payload containing an appropriate error message.

8.3.1.4.5 Windowing

The windowing parameters are optional; however, if either is present, both shall be present. If one is present and not the origin server shall return a 409 (Conflict) response, and should include a payload containing an appropriate error message.

The Windowing and Presentation State parameters shall not be present in the same request. If both are present the origin server shall return a 409 (Conflict) response, and should include a payload containing an appropriate error message.

8.3.1.4.5.1 Window Center

\[ \text{window-center} = \%s"\text{windowCenter}" \"=\" \text{decimal} \]

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.

8.3.1.4.5.2 Window Width

\[ \text{window-width} = \%s"\text{windowWidth}" \"=\" \text{decimal} \]

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.

8.3.1.4.6 Image Quality

\[ \text{image-quality} = \%s"\text{imageQuality}" \"=\" \text{uint} \]

This parameter specifies the quality factor for lossy image media types. Its name is "imageQuality" and its value is an integer between 1 and 100 inclusive, with 100 indicating highest quality.

Note

Decompression and re-compression may degrade the image quality if the original image was already irreversibly (lossy) compressed. If the image is already encoded in an acceptable lossy media type at an acceptable quality, it may be returned without decompressing and re-compressing it.

If the value of this parameter is less than 1 or greater than 100, the response shall be a 409 (Conflict), and should include a payload containing an appropriate error message.

The specific interpretation of the meaning of this parameter is left to the implementers, but shall be documented in the conformance statement.

8.3.1.4.7 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. If any of the other parameters are present the response shall be 409 (Conflict), and should include a payload containing an appropriate error message.
If the target resource is not a Presentation State then the response will be 409 (Conflict), and should include a payload containing an appropriate error message.

8.3.1.4.7.1 Presentation Series UID

\texttt{presentation-series = \%s"presentationSeriesUID" \"=" \texttt{uid}}

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

8.3.1.4.7.2 Presentation UID

\texttt{presentation-instance = \%s"presentationUID" \"=" \texttt{uid}}

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

### Request Header Fields

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>1#media-range</td>
<td>M</td>
<td>Shall result in a Selected Media Type that is a Rendered Media Type. See Section 7.9.7.</td>
</tr>
<tr>
<td>Accept-Charset</td>
<td>1#charset</td>
<td>O</td>
<td>See Section 7.10.</td>
</tr>
</tbody>
</table>

8.3.1.6 Request Payload

The request message has no payload.

8.3.2 Behavior

1205 The target resource is rendered according to the Query Parameters, by applying the transformations according to the appropriate rendering pipeline specified in PS3.4, Section N.2.

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

8.3.2.1 Without Presentation State

The origin server shall scale the rendered images, maintaining their original aspect ratio, until either the image width is the same as the viewport columns or the image height is the same as the viewport rows, whichever comes first. In other words, the scaling makes the image(s) as large as possible, within the viewport, without overflowing the viewport area and without distorting the image. If the viewport parameter is not present, the returned image shall contain the same pixel matrix size as the source DICOM image.

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 images is not defined by this standard.

8.3.2.2 With Presentation State

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

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

1225 If the Presentation Size Mode is MAGNIFY, then the referenced images shall be scaled to the Specified Displayed Area in the Presentation State, and then they shall be cropped to the size specified by the "viewport" parameter. If the request does not contain a "viewport" parameter, then the referenced images shall not be cropped.
Any Specified Displayed Area relative annotations in the Presentation State shall be rendered relative to the Specified Displayed Area within the Presentation State, not the size of the viewport.

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

8.3.3 Response Message

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

8.3.3.1 Status Codes
See Section 7.7.

8.3.3.2 Response Header Fields

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>C</td>
<td>Required if response contains a payload. See Section X.Y.</td>
</tr>
<tr>
<td>Content-Encoding</td>
<td>encoding</td>
<td>C</td>
<td>Required if response payload has a content coding. See Section X.Y.</td>
</tr>
<tr>
<td>Content-Length</td>
<td>uint</td>
<td>C</td>
<td>Required if response payload does not have a content coding. See Section X.Y.</td>
</tr>
<tr>
<td>Content-Location</td>
<td>uri</td>
<td>O</td>
<td>Optional if response has a payload. See Section X.Y.</td>
</tr>
</tbody>
</table>

8.3.3.3 Response Payload

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

A failure response may include a payload describing the error.

8.3.4 Response Payload

A successful response shall have a payload containing the rendered target resource as an octet-stream.

A failure response may include a payload describing the error.

8.3.4 Acceptable Media Types

The Acceptable Media Types shall contain only Rendered Media Types. See Section 7.9.4.

8.4 Media Types

8.5 Conformance

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

[URI First Read Completed on 2015-11-12]
9 WADO-WS Services API

[TODO: insert CP_WADO_WS when final]

9.1 Conformance

An implementation conforming to the WADO-WS service shall support retrieval of one or more of the Information Objects specified for the DIMSE C-STORE based Storage Service Class, as specified in PS3.4 Annex B.4.

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

[To be inserted]
10 RESTful Services APIs

The RESTful Services (RS) APIs are defined using...

The RS transaction definitions defined using the HTTP/1.1 ABNF definitions in [RFC7230]; however, some of the RS definitions, eliminate the obsolete ("obs-*) terms in the grammar. All RS ABNF definitions are normative, but they shall be consistent with HTTP except for obsolete terms.

10.1 Services

Table 10.1-1 shows the Restful services that are currently defined.

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studies</td>
<td>Returns one or more DICOM objects specified by the target resource in an Acceptable Media Type.</td>
</tr>
<tr>
<td>Work List</td>
<td>Returns one or more rendered images specified by the target resource in an Acceptable Media Type.</td>
</tr>
</tbody>
</table>

10.2 Target Resources

The target resources of the RS APIs are typically DICOM Information Entities; but they may be other entities. The general format of resource URI Templates is:

```
/resource-type/{resource-id}/resource-subtype/{sub-resource-id}/...{?parameter*}
```

Where,
- `resource-type` is a literal string, for example "studies", and
- `{resource-id}` is a variable, typically a UID that identifies a specific resource
- `resource-subtype` is a literal string, for example "series"
- `{sub-resource-id}` is a variable, typically a UID, that identifies a specific sub-resource
- `{?parameters*}` is a URI Template for one or more Query Parameters

10.3 Transaction Overview

Each RS transaction is composed of a request message and a response message, sometimes referred to as a request/response pair. The messages have the general format specified in Section 7.3.

Table X.Y-Z below contain the names of the most common transactions in RS services. For each transaction in include the method, the resource template, the type of payload, if any, and a description of the transaction. These transaction names should be use whenever a transaction has a request and response format similar to those in Table 10.3-1 and 10.3-2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Method</th>
<th>Resource</th>
<th>Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRUD Transactions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create</td>
<td>POST</td>
<td>/{+r}</td>
<td>new resource</td>
<td>Create a resource</td>
</tr>
<tr>
<td>Retrieve</td>
<td>GET</td>
<td>/{+r}{?p*}</td>
<td>empty</td>
<td>Retrieve a resource</td>
</tr>
<tr>
<td>Headers</td>
<td>HEAD</td>
<td>/{+r}{?p*}</td>
<td>empty</td>
<td>HEAD for retrieve</td>
</tr>
<tr>
<td>Append</td>
<td>POST</td>
<td>/{+r}</td>
<td>sub-resource</td>
<td>Appends a new sub-resource to a resource</td>
</tr>
<tr>
<td>Update</td>
<td>PATCH</td>
<td>/{+r}</td>
<td>modifications</td>
<td>Updates a resource</td>
</tr>
<tr>
<td>Replace</td>
<td>PUT</td>
<td>/{+r}</td>
<td>resource</td>
<td>Replaces a resource</td>
</tr>
</tbody>
</table>
### Supplement 183

**Delete** | **DELETE** | /{+r}/(?p*) | empty | Deletes a resource
---|---|---|---|---
**Search** | **POST** | /{+r}/search(?p*) | search spec | Searches a resource for contained resources
**Lock** | **GET** | /{+r}/lock | empty | Locks a resource

**Capabilities Sub-Service**

**Retrieve Capabilities** | **GET** | /capabilities | empty | Retrieves the service’s capabilities document

**Notification Sub-Service**

**Open Notification Connection** | **GET** | / | empty | Opens a Notification Connection
**Subscribe** | **POST** | /subscriptions/{r} | empty | Creates a subscription to a resource
**Unsubscribe** | **DELETE** | /{+s} | empty | Sends an Event Report to a user agent

**State Transitions**

{(state change)} | **PUT** | /{+r}/(new-state) | modifications | Similar to Update, but for state. The state transition must be from a valid predecessor state.

---

### 1300

{+r}, {r} Any resource managed by the service.
(+s) A subscription
?{?parameters} one or more required parameters
{?parameters} Zero or more optional parameters

#### Table 10.3-2: Standard Response Templates

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CRUD+ Transactions</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create</td>
<td>201 (Created)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Retrieve</td>
<td>200 (OK)</td>
<td>resource</td>
<td>Success w/ resource, failure w/ Status Details</td>
</tr>
<tr>
<td>Headers</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Append</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Update</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Replace</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Delete</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Search</td>
<td>200 (OK), 206 (Partial)</td>
<td>search results</td>
<td>success w/ search results, failure w/ Status Details</td>
</tr>
<tr>
<td>Lock</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td><strong>Capabilities Sub-Service</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Retrieve Capabilities</td>
<td>/capabilities</td>
<td>cap-doc</td>
<td>success w/ service’s Capabilities document, failure w/ Status Details</td>
</tr>
<tr>
<td><strong>Notification Sub-Service</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Notification Connection</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Subscribe</td>
<td>201 (created)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Unsubscribe</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
<tr>
<td>Send Notification</td>
<td>NA</td>
<td>ack-nak</td>
<td>Success w/ ack message, failure w/ Nak message</td>
</tr>
<tr>
<td><strong>State Transitions</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(state change)</td>
<td>200 (OK)</td>
<td>e / sd</td>
<td>success w/o payload, failure w/ Status Details</td>
</tr>
</tbody>
</table>
1305 Where

e / sd = success with empty payload, failure with Status Details document
cap-doc = Capabilities document
ack-nak = Notification Status document

10.4 Generic Service Transactions

1310 The transactions defined in this section are either required or optional for each RS Service. They are generic transactions that should be uniform across the different services.

10.4.1 Retrieve Information Transaction

The Retrieve Information transaction retrieves information about the origin server. All Restful services defined by this standard shall implement this transaction.

1315 The target resource for this transaction shall be the origin server. The response shall contain a machine readable Capabilities Description document. The Capabilities Description document describes the transactions, resources, representations, etc. that are supported by the service(s). The Capabilities Description document shall describe the service in as much detail as possible.

Table 9.4-1: Retrieve Capabilities Transaction

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Method</th>
<th>URI Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve Capabilities</td>
<td>OPTIONS</td>
<td>/capabilities</td>
<td>Retrieve a Capabilities Description document from the service in an Acceptable Media Type.</td>
</tr>
<tr>
<td>Retrieve Capabilities</td>
<td>GET</td>
<td>/</td>
<td></td>
</tr>
<tr>
<td>Retrieve Conformance</td>
<td>GET</td>
<td>/conformance</td>
<td>Retrieves the service’s conformance statement in an acceptable media type.</td>
</tr>
<tr>
<td>Retrieve Log</td>
<td>GET</td>
<td>/log[?start][&amp;end]</td>
<td>Retrieves the transaction log of the service.</td>
</tr>
</tbody>
</table>

1320 10.4.1.1 Request

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

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

However, since some web servers do not support the OPTIONS method, the Service capabilities may also be retrieved using the GET method and the Service Capabilities resource, as follows:

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

Where

```
/capabilities is a literal string specifying the capabilities resource.
```

1330 10.4.1.1 Resource

The target resource for this transaction is the Base URI (“/”) or the Capabilities resource (“/capabilities”). **See Section 6.4.1 for details.**

Table 10.4-X: Server Information Resources

<table>
<thead>
<tr>
<th>URI Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/capabilities</td>
<td>Capabilities Description document</td>
</tr>
</tbody>
</table>
10.4.1.2 Query Parameters
This transaction has no Query Parameters.

10.4.1.3 Request Header Fields
Required: Accept
Recommended: Content Negotiation Header Fields (see Section 7.3.1.3)

10.4.1.4 Request Payload
The request has no payload.

10.4.1.2 Behavior
The origin server will return a machine readable description of its capabilities in an Acceptable Media Type.

10.4.1.3 Response
The format of the response is as follows:

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

payload

A success response shall have a payload containing a Capabilities Description document (see Annex X) encoded in the Selected Media Type.

10.4.1.3.1 Status Codes
A success response will have a status code of 200 (OK).
A failure response shall have a status code from Table X.Y-Z, and should include a Status Details document.

10.4.1.3.2 Response Header Fields
Required: Content-Type
See Section 7.3.1.3.2.

10.4.1.3.3 Response Payload
A success payload contains a Capabilities Description document in the Selected Media Type.
A failure response should have a payload containing a Status Details document. See Section X.Y.

10.4.1.4 Media Types
The media types supported by the Retrieve Capabilities service shall be defined by the implementing service.
See Annex X, Y-z.

10.4.2 Retrieve Resource Status
[TODO: finish]
Returns an Instance Availability Notification document for the target resource.

--- Draft ---
10.5 Notification Sub-Service (or Subsystem)

[TODO: Update with CP_rs_notification_subsystems when final]

10.6 Conditional Requests

[TODO: Update with CP_rs Conditional_requests when final]

All RS Protocol transactions shall support conditional requests with strong entity tags. There are two kinds of conditional request:

**Conditional Retrieve**

If a user agent has already retrieved a representation of a resource, a subsequent retrieve of the same resource, shall include If-None-Match header field containing the entity tag from the most recent retrieval of that resource. If the resource has not changed, the origin server can return of 304 (Not Modified) response with no payload. This lowers the load on the origin server and lowers the latency and bandwidth of the response.

**Conditional Update**

Typically a user agent must retrieve a resource before it can append, modify, or delete it. The user agent must include the If-Match header field containing the most recent entity tag it has for the resource to be updated. If the current entity tag of the resource is equal to the entity tag in the If-Match header, then the update may proceed. If they are not equal the update will fail with a 412 (Precondition Failed) response.

Performing modifications this way ensures that the content of the resource remain consistent, and avoids the "lost update" problem.

### 10.6.1 Origin Server Requirements

All origin servers may support conditional requests, by including the following header fields in responses to transactions using the GET or HEAD methods or for transactions that create a new resource.

- **ETag**
  - Its value is an entity-tag. How the entity-tag is generated is determined by the origin server, but it must support Strong Validation, i.e. it cannot have the weak prefix "W/". Each entity-tag must identify a unique version of the corresponding resource.

- **Last-Modified**
  - Its value is the date and time of the most recent modification to the target resource in HTTP-date format.

### 10.6.2 User Agent Requirements

The user agent shall use conditional requests whenever possible. In particular:

- The user agent shall include the If-None-Match header with the most recent entity tag for a resource that has previously been retrieved, or which it created.
- The user agent shall include the If-Match header for any requests using the POST, PUT, PATCH, or DELETE methods that modify the state of a resource.

### 10.6.3 Conditional Retrieve

A Conditional Retrieve transaction allows the origin server to return a response with a status code of 304 (Not Modified) and no payload, if the target resource has not been modified since the user agent last retrieved it.

Given a target resource R, the user agent shall not include an If-None-Match header field in its first GET or HEAD request that targets R; however, all subsequent GET or Head requests that target R shall include an If-None-Match header field with an entity-tag that is the most recent entity-tag received for R.

If the entity-tag in the request is equal to that of the current R, then a success response shall contain a status code of 304 (Not Modified) with an empty payload; otherwise, the entity-tag is not equal to that of the current R, and a success response shall contain a status code of 200 (OK) with a payload corresponding to current version of R in an Acceptable Media Type.
An origin server generating a 304 response MUST generate any of the following header fields that would have been
sent in a 200 (OK) response to the same request: Cache-Control, Content-Location, Date, ETag, Expires, and Vary.

[Give examples]
- A radiologist dictating a report
- A cardiologist that decides to create a presentation state while reading an exam.

10.6.4 Conditional Modifications (POST, PUT, PATCH, DELETE)

In order to modify (Append, Update, or Delete) a resource, the user agent must have the most recent entity tag for
the target resource. If the user agent does not have an entity tag for the target resource, it shall first retrieve the
current entity-tag of that resource using either the Retrieve (GET) or Retrieve Headers (HEAD) transaction.

A modification request shall contain an If-Match header field with the most recent entity-tag for the target resource.

On receiving an Update request, the origin server shall compare the If-Match entity tag with the current target
resource entity-tag, and return a status code and payload as defined in Table X.Y-Z.

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Payload</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 (OK)</td>
<td>empty</td>
<td>If the entity tags are equal, then the origin server should perform the update.</td>
</tr>
<tr>
<td>204 (No Content)</td>
<td>empty</td>
<td>If the entity tags are not equal, and the request was to change the state of the target resource, and the state is the same as it would be if the update were applied, the origin server should do nothing.</td>
</tr>
<tr>
<td>412 (Precondition Failed)</td>
<td>Status Details</td>
<td>If the entity tags are not equal, and the request was to modify data contained in the target resource, the origin server should do nothing.</td>
</tr>
<tr>
<td>4XX</td>
<td>Status Details</td>
<td>Some other error. <a href="#">See Table 6.7-1.</a></td>
</tr>
</tbody>
</table>

10.7 Caching and Proxies
11  RESTful Studies Service

The RS Studies Service defines a set of Restful transactions that enable a user agent to store, retrieve, update, and search an origin server for DICOM Studies, Series, and Instances – along with their Metadata variants; as well as Frames and Bulkdata.

11.1  Resources

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 of the studies managed by the service.
- `/series` references all of the series managed by the service.
- `/instances` references all of the instances managed by the service.

The following URI Template variables are used in the definitions of the resources throughout Section 10.

- `{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.
- `{frame_list}` a comma-separated list of frame numbers in ascending order contained within an instance.
- `{+bulkdata}` a URI that references a Bulkdata Value.

The Studies Service defines the following resources:

<table>
<thead>
<tr>
<th>Resource</th>
<th>URI Template and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studies Service</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>The Base URI of the Studies Service.</td>
</tr>
<tr>
<td>All Studies</td>
<td>/studies</td>
</tr>
<tr>
<td></td>
<td>The All Studies resource references the entire collection of studies contained in the Studies Service. All Studies Service resources begin with this resource.</td>
</tr>
<tr>
<td>Study</td>
<td>/studies/{study}</td>
</tr>
<tr>
<td></td>
<td>The Study resource references a single study.</td>
</tr>
<tr>
<td>Study Metadata</td>
<td>/studies/{study}/metadata</td>
</tr>
<tr>
<td></td>
<td>The Study Metadata resource references the Metadata of a single study.</td>
</tr>
<tr>
<td></td>
<td>See Section 6.5.1.2</td>
</tr>
<tr>
<td>Study Bulkdata</td>
<td>/studies/{study}/bulkdata</td>
</tr>
<tr>
<td></td>
<td>The Study Bulkdata resource references the Bulkdata of a single study.</td>
</tr>
<tr>
<td></td>
<td>See Section X.Y</td>
</tr>
<tr>
<td>Study's Series</td>
<td>/studies/{study}/series</td>
</tr>
<tr>
<td></td>
<td>The Study's Series resource references the collection of all series contained in a study.</td>
</tr>
</tbody>
</table>
The Study's Instances resource references the collection of all instances in a single study.

The All Series resource references the collection of all series in all studies contained in the Studies Service.

The Series resource references a single series in a study.

The Series Metadata resource contains the Metadata of a single series in a study. See Section 6.5.1.2.

The Series Bulkdata resource contains the Bulkdata of a single series in a study. See Section 6.5.1.2.

The Series' Instances resource references the collection of all series in a single study.

The All Instances resource references the collection of all instances in all series in all studies contained in the Studies Service.

The Instance resource references a single instance contained in a series.

The Instance Metadata resource contains the Metadata of a single instance in a series. See Section 6.8.1.2.

The Instance Bulkdata resource contains the Bulkdata of a single instance in a series. See Section 6.8.1.2.

The Frames resource references an ordered collection of frames in a single multi-frame instance.

The Bulkdata resource contains one or more Bulkdata Values. See Section 6.5.1.2.

### 11.2 Transactions

The Studies Service defines the transactions specified in the following table:

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Method</th>
<th>Success Payload Request</th>
<th>Success Payload Response</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve Capabilities</td>
<td>OPTIONS</td>
<td>N/A</td>
<td>Capabilities Description</td>
<td>Retrieves a description of the capabilities of the Studies Service, including transactions, resources, Query Parameters, etc.</td>
</tr>
<tr>
<td>Retrieve DICOM</td>
<td>GET</td>
<td>N/A</td>
<td>Instance(s)</td>
<td>Retrieves one or more representations, specified by the target resource in an Acceptable DICOM Media Type.</td>
</tr>
<tr>
<td>Retrieve Rendered</td>
<td>GET</td>
<td>N/A</td>
<td>Rendered instance(s)</td>
<td>Retrieves one or more representations, specified by the target resource, which shall not be a presentation state, in an</td>
</tr>
</tbody>
</table>
Table 10.2-2 shows the target resources permitted for each transaction.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Retrieve DICOM</th>
<th>Retrieve Rendered</th>
<th>Retrieve PS</th>
<th>Store</th>
<th>Search</th>
<th>Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studies Service</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>All Studies</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X X</td>
</tr>
<tr>
<td>Study</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Study Metadata</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Study’s Series</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Study’s Instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>All Series</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Series</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Series Metadata</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Series’ Instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>All Instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Instance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X X</td>
</tr>
<tr>
<td>Instance Metadata</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X X</td>
</tr>
<tr>
<td>Frame List</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bulkdata</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X X</td>
</tr>
</tbody>
</table>

### 11.3 Retrieve Capabilities Transaction

#### 11.3.1 Request

The Retrieve Capabilities request uses the GET method and has the following syntax:

```
GET SP /{resource} SP version CRLF
Accept: dicom-media-type CRLF
[If-None-Match: entity-tag CRLF]
```

#### 11.3.2 Behavior

[TODO]
11.3.3 Response

The Retrieve Capabilities response has the following syntax:

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

11.3.4 Conformance

An origin server that implements the Studies service shall implement the Capabilities transaction and shall include it in its Conformance Statement. A user agent that implements the Capabilities transaction shall include it in its Conformance Statement.

11.4 Retrieve DICOM Transaction

The Retrieve DICOM transaction retrieves the target resource in a DICOM media type. If non-DICOM media types are present in the Acceptable Media Types of the request, the origin server shall return an appropriate failure response.

The Retrieve DICOM Transaction supports Conditional Retrieve requests. See Section 10.X.Y.

11.4.1 Request

The Retrieve DICOM services has the following request message format:

```
GET SP /{resource} SP version CRLF
Accept: dicom-media-type CRLF
[If-None-Match: entity-tag CRLF]
*(header-field CRLF)
CRLF
```

Where

- `dicom-media-type`: One of the media types from Table X.Y-2, and Annex X.
- `entity-tag`: if present, is the most recent entity-tag value received from the origin server for the target resource. See Section 9.X.Y.

All other syntactic variables are as defined in Section 6.3.2.

11.4.1.1 Resources

The target resources for the Retrieve DICOM transaction are listed in Table 10.2-2 under the Retrieve DICOM column. If an origin server supports this transaction it shall support all defined resources.

Table 10.3-1 shows the resources, along with their URI Templates, supported by the Retrieve DICOM transaction.

<table>
<thead>
<tr>
<th>Resource</th>
<th>URI Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Study</td>
<td>/studies/{study}</td>
</tr>
<tr>
<td>Study Metadata</td>
<td>/studies/{study}/metadata</td>
</tr>
<tr>
<td>Study Bulkdata</td>
<td>/studies/{study}/bulkdata</td>
</tr>
<tr>
<td>Series</td>
<td>/studies/{study}/series/{series}</td>
</tr>
<tr>
<td>Series Metadata</td>
<td>/studies/{study}/series/{series}/metadata</td>
</tr>
<tr>
<td>Series Bulkdata</td>
<td>/studies/{study}/series/{series}/bulkdata</td>
</tr>
<tr>
<td>Instance</td>
<td>/studies/{study}/series/{series}/instances/{instance}</td>
</tr>
<tr>
<td>Instance Metadata</td>
<td>/studies/{study}/series/{series}/instances/{instance}/metadata</td>
</tr>
</tbody>
</table>
### 11.4.1.2 Query Parameters

There are no Query Parameters.

[TODO: accept and charset]

### 11.4.1.3 Request Header Fields

Table 10.3-1 list the most common Mandatory, Conditional, and Optional header fields for this transaction. See Section 6.5.2.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>media-type</td>
<td>M</td>
<td>List of one or more media types</td>
</tr>
<tr>
<td>Accept-Charset</td>
<td>charset</td>
<td>O</td>
<td>List of one or more character sets</td>
</tr>
</tbody>
</table>

### 11.4.1.4 Request Payload

The request shall have no payload.

### 11.4.2 Behavior

The origin server locates the target resource(s) and returns it (them) in an Acceptable DICOM Media Type. If the resource cannot be located an appropriate error response shall be returned.

### 11.4.3 Response

The response has the following format:

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

Where

- `dicom-media-type` is one of the media types from Table X.Y-z, and Annex X
- `entity-tag` is the most recent entity-tag value received from the origin server for the target resource.
- `http-date` The current time in HTTP-date format. See X.Y.

All other syntactic variables are as defined in Section 1.3.3.

### 11.4.3.1 Status Codes

The following status codes are the most common for the Retrieve Transaction. See Section 6.X.Y for other status codes.

<table>
<thead>
<tr>
<th>Status</th>
<th>Code</th>
<th>Phrase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td>200</td>
<td>OK</td>
<td>Indicates that all instances were successfully retrieved.</td>
</tr>
</tbody>
</table>
### 11.4.3.2 Response Header Fields

Table 10.3-3 lists the most common Mandatory, Conditional, and Optional header fields for the response. See Section 6.5.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>The media type of the payload</td>
</tr>
<tr>
<td>ETag</td>
<td>entity-tag</td>
<td>R</td>
<td>If the response status code is 200 or 206, the entity tag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>associated with the representation contained in the payload.</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>HTTP-date</td>
<td>R</td>
<td>The current time in HTTP-date format.</td>
</tr>
</tbody>
</table>

### 11.4.3.3 Response Payload

A success response shall have a payload containing one or more representations of the DICOM instances specified by the target resource in an Acceptable DICOM Media Type. The payload may be single part or multipart depending on the media type.

The response shall contain a Status Details document that describes any warnings or errors encountered by the origin server.

### 11.4.4 Media Types

A success response payload shall contain representation in a DICOM media type as specified in Section X.Y and Annex X.

The Studies Service shall support uncompressed Bulkdata.

A failure response should contain an error message in the payload.

### 11.4.5 Conformance

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

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

- Transactions supported (DICOM, rendered)
- Media types supported
- Rendered PS – referenced SOP classes
- Character sets (if other than UTF-8)

### 11.5 RS Retrieve Rendered Transaction

The media type of the response payload shall be a rendered media type. See Section 6.5.Y and Annex X.

The Retrieve Rendered transaction retrieves DICOM objects rendered as: images, text-based documents, or other appropriate representations depending on the target resource. Its primary use case is to provide user agents with a simple interface for displaying medical images and related documents, without requiring deep knowledge of DICOM data structures and encodings. It is similar to the Retrieve DICOM service in that it uses the same method, resources, header fields and status codes. The primarily difference is the Query Parameters and media types supported.
The origin server shall be able to render all valid instances of the Composite SOP classes for which conformance is claimed, e.g., all photometric interpretations that are defined in the IOD for the SOP class.

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

### 11.5.1 Request

The Retrieve Rendered service has the following request message syntax:

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

Where

- `/{resource}` references a non-Presentation State resource.
- `{?parameter*}` zero or more Query Parameters as defined in Section 6.5.8.1.2.
- `version` HTTP version = 'HTTP/1.1'
- `1#rendered-media-type` one or more Rendered Media Types See Section 6.1.1.3.

### 11.5.1.1 Target Resources

Table 6.5-1 shows the resources supported by the Retrieve Rendered transaction along with their associated URI templates.

<table>
<thead>
<tr>
<th>Target Resource</th>
<th>Resource URI Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Study</td>
<td>/studies/{study_uid}</td>
</tr>
<tr>
<td></td>
<td>Retrieves a study in acceptable Rendered Media Types.</td>
</tr>
<tr>
<td>Series</td>
<td>/studies/{study_uid}/series/{series_uid}</td>
</tr>
<tr>
<td></td>
<td>Retrieves a series in an acceptable Rendered Media Type.</td>
</tr>
<tr>
<td>Instance</td>
<td>/studies/{study_uid}/series/{series_uid}/instances/{instance_uid}</td>
</tr>
<tr>
<td></td>
<td>Retrieves an instance in an acceptable Rendered Media Type.</td>
</tr>
<tr>
<td>Frames</td>
<td>/studies/{study_uid}/series/{series_uid}/instances/{instance_uid}/frames/{frame_list}</td>
</tr>
<tr>
<td></td>
<td>Retrieves one or more frames in an acceptable rendered media type.</td>
</tr>
</tbody>
</table>

### 11.5.1.2 Query Parameters

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

The origin server shall support all of the Query Parameters defined in this section. An implementer of the origin server may define additional parameters. If additional parameters are defined, they shall be documented in the Conformance Statement (and for RS services in the Retrieve Capabilities response). Origin servers shall ignore any unknown parameters.

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

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

<table>
<thead>
<tr>
<th>Key</th>
<th>Values</th>
<th>Target Resource</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>annotation</td>
<td>&quot;patient&quot; and/or &quot;procedure&quot;</td>
<td>All</td>
<td>6.5.8.1.1.1</td>
</tr>
<tr>
<td>charset</td>
<td>token</td>
<td>All</td>
<td>6.1.2.2</td>
</tr>
<tr>
<td>quality</td>
<td>integer</td>
<td>All</td>
<td>6.5.8.1.1.2</td>
</tr>
<tr>
<td>viewport</td>
<td>vw, vh, [sx, sy, sw, sh]</td>
<td>Non-Presentation States</td>
<td>6.5.8.1.1.3</td>
</tr>
<tr>
<td>window</td>
<td>center, width, shape</td>
<td>Non-Presentation States</td>
<td>6.5.8.2.1.2.2</td>
</tr>
</tbody>
</table>

11.5.1.2.1 Image Annotation

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

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

Where

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

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

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

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

Note

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

11.5.1.2.2 Image Quality

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

```
"quality=" integer
```

Where

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

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

Note:

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

11.5.1.2.3 Scaling a Region of a Source Image(s) to a Viewport

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

If the target resource is a Presentation State Instance, the syntax for this parameter is:
"viewport=", vh

Otherwise it is:

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

Where

vw and vh are positive integers specifying the width and height, in pixels, of the rendered image. Both parameters values are required.

sx and sy are decimal numbers non-negative integers specifying, 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. These parameters shall not be used when rendering a Presentation State.

sw and sh are decimal numbers non-negative integers specifying, 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. These parameters shall not be used when a rendering Presentation State.

The origin server shall crop the source images to the region specified by <sx>, <sy>, <sw>, and <sh>. It shall then scale the target sources images, maintaining their original aspect ratio, until either the rendered image width or height is the same as the viewport width or height, whichever comes first. 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) shall not be scaled, i.e., the rendered image(s) shall contain the same sized pixel matrix as the source DICOM image.

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.

11.5.1.2.4 Windowing

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

window = "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.
If the target resource is a Presentation State, this parameter shall not be used. If this parameter is present when the target resources is a Presentation state, the origin server shall return a 409 (Conflict) response.

11.5.1.3 Request Header Fields

Required: Accept

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

11.5.1.4 Request Payload

This request has no payload.

11.5.2 Behavior

The target resource(s) are rendered according to the Query Parameters, by applying the transformations according to the appropriate rendering pipeline specified in PS3.4, Section N.2.

Any Presentation State Instances contained in that resource contains shall not be rendered.

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

11.5.2.1 Presentation State Instance

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

The origin server shall render the Presentation State instance in an Acceptable Media Type. Rendering a Presentation State instance requires rendering all of the images that the Presentation State references using the rendering pipeline specified in PS3.4.

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

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

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

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

If the Presentation State instance contains a Blending Sequence, then the images contained in the response shall be those specified by the Blending Sequence. See PS3.3, Section C.11.14.1.1.

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

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

11.5.3 Response

The Retrieve Rendered service has the following response message syntax:

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

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

**Table 6.5.8-3: Common Status Codes**

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 Success</td>
<td>The origin server successfully rendered and is returning representations for the resource.</td>
</tr>
<tr>
<td>206 Partial Content</td>
<td>The origin server successfully rendered and is returning representations for part, but not all, of the resource.</td>
</tr>
<tr>
<td>413 Payload Too Large</td>
<td>The target resource is too large to be rendered by the origin server.</td>
</tr>
<tr>
<td>406 Not Acceptable</td>
<td>The origin server does not support any of the requested media type(s).</td>
</tr>
</tbody>
</table>

11.5.3.2 Response Header Fields

**Required:** Content-Type

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

11.5.3.3 Response Payload

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

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

11.5.4 Media Types

The origin server shall be capable of returning representations in default and required DICOM and Rendered Media Types for supported SOP Classes. **See Section 6.1.1.**

The media type of the response payload shall be a rendered media type. **See Section 6.X.Y and Annex X.**

11.5.5 Conformance

An implementation of the Studies service supporting the Retrieve Rendered transaction and shall declare document in its Conformance Statement the Information Objects (resources) supported for this transaction, and whether it plays the role of origin server or user agent, or both.

An implementation playing the role of origin server shall declare in its Conformance Statement

**Media types supported**

- Rendered PS – referenced SOP classes
- Character sets (if other than UTF-8)
11.6 Store Transaction

This transaction requests that the origin server stores the representations of Study, Series, and Instance resources contained in the request payload so that they may be retrieved in the future using the Study, Series, and Instance UIDs.

The Store Service shall support only DICOM media types.

[TODo: discuss conditional updates and atomic transactions]

11.6.1 Request

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

```
POST SP/studies/[/{study}] SP version CRLF
Content-Type: dicom-media-type CRLF
*(header-field CRLF)
CRLF
```

Where:

- `/{study}` An optional Study Instance UID of a DICOM Study. If the `{study}` is specified all representations contained in the payload shall share that Study Instance UID. Any representations in the payload with Study Instance UIDs that differ from `{study}` shall not be stored.
- `dicom-media-type` A DICOM media type. See Section 6.X.Y and Annex X.
- `payload` A payload containing the representations to be stored.

11.6.1.1 Resources

The target URI shall reference either the Studies or Study resource. The Store transaction and their associated resources and templates are specified in Table 10.6.1.

<table>
<thead>
<tr>
<th>Resource</th>
<th>URI Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studies</td>
<td>/studies</td>
<td>Stores a set of representations, which may have different Study Instance UIDs.</td>
</tr>
<tr>
<td>Study</td>
<td>/studies/{study}</td>
<td>Stores a set of representations for the same study, each of which shall have the same Study Instance UID.</td>
</tr>
</tbody>
</table>

11.6.1.2 Query Parameters

The Store Service has no Query Parameters.

11.6.1.3 Request Header Fields

Table 10.6-2 lists the Mandatory, Conditional, and most common Optional header fields for this transaction. See Section 6.5.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>The DICOM media type of the request payload</td>
</tr>
<tr>
<td>Accept</td>
<td>media-type</td>
<td>M</td>
<td>One or more media types acceptable in the response</td>
</tr>
<tr>
<td>Content-Length</td>
<td>uint</td>
<td>C</td>
<td>If no transfer coding has been applied to the payload</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>encoding</td>
<td>C</td>
<td>If a transfer coding has been applied to the payload</td>
</tr>
</tbody>
</table>

Request See Section 7.Y.
11.6.1.4 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.

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

Before storing 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 element is coerced or corrected, the Original Attribute Sequence (0400,0561) shall be included in the stored DICOM objects, and shall be included in the Status Details document returned in the response. See Section X.Y.

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

11.6.3 Response
The response shall have the following format:

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

The response shall contain an appropriate status code and a payload containing a Status Details document, as defined in Annex X.Y, in an Acceptable Media Type. The Status Details document shall describe the success, warning, or failure status for each Instance contained in the request payload.

11.6.3.1 Status Codes
The origin server shall return an appropriate status code. The most common status codes are:

<table>
<thead>
<tr>
<th>Status</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td>200 (OK)</td>
<td>Indicates that 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 Status Details document detailing the disposition of all representations contained in the request.</td>
</tr>
</tbody>
</table>
| 202 (Accepted)  |          | Indicates that the origin server successfully received and validated the request message, but may not have validated, processed, or stored the representations in the request payload.  
The response shall include a Status Details document detailing the disposition of all representations contained in the request.  
The user agent can use a Query or Retrieve transaction at a later time to determine if the request has completed.  
Alternatively, if the Service supports notifications, and the user agent has a subscription for newly created resources, it will receive an event notification when the transaction has completed. |
Error

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>409 (Conflict)</td>
<td>Indicates that the origin server did not store any of the representations contained in the request payload because of representation specific errors. For example, unsupported SOP Class, or Study Instance UID mismatch.</td>
</tr>
<tr>
<td>415 (Unsupported Media Type)</td>
<td>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</td>
</tr>
</tbody>
</table>

For additional information on status codes see Section 6.6. The response shall not contain the 201 (Created) or 204 (No Content) status codes. The reason for this is that the request payload shall always contain a Status Details document describing the status of each instance contained in the request.

11.6.3.2 Response Header Fields

Table 10.6-4 lists the Mandatory, Conditional, and most common Optional header fields for this transaction. See Section 6.5.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>The DICOM media type of the request payload</td>
</tr>
<tr>
<td>Content-Length</td>
<td>uint</td>
<td>C</td>
<td>If no transfer coding has been applied to the payload</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>encoding</td>
<td>C</td>
<td>If a transfer coding has been applied to the payload</td>
</tr>
</tbody>
</table>

All success responses shall contain the Representation Header Fields (see [RFC7231, Section 3.1]) and Payload Header Fields with appropriate values. See Section 2.3.3.2.

It is recommended that the text returned in the Warning header field (see [RFC7234, Section 5.5]) contain a DICOM Status Code and descriptive reason as defined in Section 10.6.4.2.1. For example,

Warning: A700: Out of memory

See Section 6.X.Y.

11.6.3.3 Response Payload

The response payload shall contain a Status Details document describing any additions, modifications, or deletions to the stored representations. See Annex X.

11.6.4 Media Types

The request payload shall be encoded in a <dicom-media-type> as specified in Section X.Y and Annex X.

The Store Transaction shall support uncompressed Bulksdata.

11.6.5 Conformance

An implementation conforming to the Store transaction shall support transfer of one or more of the Information Objects specified for the DIMSE C-STORE based Storage Service Class, as specified in PS3.4 Annex B.4.

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.

11.7 Search Transaction

The Search transaction specifies a search of the Studies resource on the origin server. A successful response shall contain details of resources that match the search criteria specified in the Query Parameters of the request.
11.7.1 Request

The Search service uses the GET method and has the following message format:

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

Where

- `{/resource}`: One of the resources defined in Section 10.6.1.1 below
- `{?parameter*}`: Zero or more of the parameters defined in Section 10.6.1.2 below.

11.7.1.1 Resources

The path component of the target URI specifies the collection of resources that is the target of the search.

Table 10.6-1 shows the resources supported by the Search transaction along with the URI Templates for their target URIs. Any origin server that supports the Search transaction shall support all required resources specified in Table 10.6-1.

<table>
<thead>
<tr>
<th>Resource</th>
<th>URI Template</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Studies</td>
<td>/studies{?parameter*}</td>
<td>Required</td>
</tr>
<tr>
<td>Study's Series</td>
<td>/studies/{study}/series{?parameter*}</td>
<td>Required</td>
</tr>
<tr>
<td>Study's Instances</td>
<td>/studies/{study}/instances{?parameter*}</td>
<td>Optional</td>
</tr>
<tr>
<td>All Series</td>
<td>/series{?parameter*}</td>
<td>Optional</td>
</tr>
<tr>
<td>Study Series' Instances</td>
<td>/studies/{study}/series/{series}/instances{?parameter*}</td>
<td>Required</td>
</tr>
<tr>
<td>Series' Instances</td>
<td>/series/{series}/instances{?parameter*}</td>
<td>Optional</td>
</tr>
<tr>
<td>All Instances</td>
<td>/instances{?parameter*}</td>
<td>Optional</td>
</tr>
</tbody>
</table>

Table 10.6-2 shows the resources supported by the Search transaction along with a description of the search performed and the results returned.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Studies</td>
<td>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.</td>
</tr>
<tr>
<td>Study's Series</td>
<td>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.</td>
</tr>
<tr>
<td>Study's Instances</td>
<td>Searches for all Instances in the specified Study that match the search parameters, and returns a list of matching Instances, including the default and requested attributes that are supported for each Instance.</td>
</tr>
<tr>
<td>All Series</td>
<td>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.</td>
</tr>
<tr>
<td>Study Series Instances</td>
<td>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.</td>
</tr>
<tr>
<td>Series' Instances</td>
<td>Searches for all Instances in the specified Series that match the search parameters, and returns a list of matching Instances, including the default and requested attributes that are supported for each Instance.</td>
</tr>
<tr>
<td>All Instances</td>
<td>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.</td>
</tr>
</tbody>
</table>
Supplement 183

11.7.1.2 Search Query Parameters

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

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

Where

```text
{&match*} = "(" "attribute "=" 1#value
```

Zero or more DICOM attribute/values pairs that specify the matching criteria for the search. See Section 10.6.1.2.1 below for requirements. See Section 6.5 for the detailed syntax of attribute/values.

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.

```text
{&fuzzy} = "&" "fuzzymatching" "=" ("true" / "false")
```

A single parameter that specifies whether Fuzzy Matching (see PS3.xxx) should be used. If this parameter is not present it defaults to "false".

The "fuzzymatching" parameter specifies whether Fuzzy Matching of Person Names shall be performed by the origin server. It is optional. If the parameter is not present then its value is "false", and Fuzzy Matching shall not be performed. If the value is "true" and the origin server supports Fuzzy Matching, then additional fuzzy semantic matching of person names shall be performed in the manner specified in the DICOM Conformance Statement for the origin server, and in the Retrieve Capabilities response. See PS3.X Section Y.Z.

```text
{&include*} = "(" "includefield" "=" 1#attribute) / ("&" "includefield" "=" "all")
```

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

The request may contain more than one "includefield" parameters; however, if an "&includefield=all" parameter is present, no other "includefield" parameters shall be present.

In general, it is a best practice to have only one "includefield" parameter with multiple attributes, rather than having multiple "includefields".

```text
{&offset} = "&" "offset" "=" uint
```

A single parameter that specifies the number of results the origin server should skip before the first returned result. The "offset" parameter value is an unsigned integer. If this query parameter is not present, its value shall be 0.

```text
{&limit} = "&" "limit" "=" uint
```

A single parameter that specifies the maximum number of results 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 shall be 50.

The "limit" parameter value is an unsigned integer, which specifies the maximum number of results the origin server should return. If the "limit" parameter is not present, its value is 50.

The user agent can use the "offset" and "limit" parameters to paginate the results. See Section 10.6.2.2.

See Section 6.5 for the general syntax of Query Parameters.

11.7.1.2.1 Attribute / Values Pair Requirements

DICOM Attribute/Values pairs shall satisfy the following requirements:

1. Each attribute shall be a Data Element Tag or Keyword.

2. Each attribute must refer to one of:
   - Patient IE attributes
   - Study IE attributes
1865• Series IE attributes (only allowed in Search for Series or Search for Instances requests)
• Composite Instance IE attributes (only allowed in Search for Instances requests)
• Additional Query/Retrieve Attributes (see PS3.4 Section C.3.4)
• Timezone Offset From UTC (0008.0201)

18703. Each attribute shall be unique and 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 for <value>s 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 QCHARs shall be percent-encoded. In particular, all non US-ASCII character shall be percent encoded. See [RFC3986] for details.

   Note
   The only US-ASCII characters disallowed in the query component by [RFC3986] are “#”, [“,” “]”.
   This standard further disallows “&”, “=”, and “.”.

5. 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.

The following are examples of valid Search URIs:

/studies?PatientID=11235813
/studies?PatientID=11235813&StudyDate=20130509
/studies?00100010=SMITH*&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

11.7.1.2.2 Required Attributes for Query Parameters

The origin server shall support the attributes specified in Table 10.6-3 for matching.

<table>
<thead>
<tr>
<th>IE Level</th>
<th>Keyword</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Study</td>
<td>StudyDate</td>
<td>00080020</td>
</tr>
<tr>
<td></td>
<td>StudyTime</td>
<td>00080030</td>
</tr>
<tr>
<td></td>
<td>AccessionNumber</td>
<td>00080050</td>
</tr>
<tr>
<td></td>
<td>ModalitiesInStudy</td>
<td>00080061</td>
</tr>
<tr>
<td></td>
<td>ReferringPhysicianName</td>
<td>00080090</td>
</tr>
<tr>
<td></td>
<td>PatientName</td>
<td>00100010</td>
</tr>
<tr>
<td></td>
<td>PatientID</td>
<td>00100020</td>
</tr>
<tr>
<td></td>
<td>StudyInstanceUID</td>
<td>0020000D</td>
</tr>
<tr>
<td></td>
<td>StudyID</td>
<td>00200010</td>
</tr>
<tr>
<td>Series</td>
<td>Modality</td>
<td>00080060</td>
</tr>
<tr>
<td></td>
<td>SeriesInstanceUID</td>
<td>0020000E</td>
</tr>
<tr>
<td></td>
<td>SeriesNumber</td>
<td>00200011</td>
</tr>
<tr>
<td></td>
<td>PerformedProcedureStepStartDate</td>
<td>00400244</td>
</tr>
<tr>
<td></td>
<td>PerformedProcedureStepStartTime</td>
<td>00400245</td>
</tr>
</tbody>
</table>
If study is not specified in the resource-path and Series-Level Relational Search (see X,Y) is supported, all Study level attributes specified in Table 10.6-X shall also be supported.

If <series> is not specified in the URI and Instance-Level Relational Search is supported (see X,Y), all Series-level attributes specified in Table 10.6-X shall also be supported.

### 11.7.1.3 Request Header Fields

Table 10.6-4 lists the Mandatory, Conditional, and most common Optional header fields for this transaction. See Section 6.5.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>media-type</td>
<td>M</td>
<td>One or more media types acceptable in the response</td>
</tr>
</tbody>
</table>

### 11.7.1.4 Request Payload

The request has no payload.

### 11.7.2 Behavior

The origin server shall perform the search indicated by the request, using the matching rules below, and return a response containing the specified number of search results, or an appropriate failure response.

[TODO: JW to add a note about queries being too broad.]

#### 11.7.2.1 Matching Rules

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

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

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

Note

If an origin server is acting as a proxy for a C-FIND SCP that does not support Combined DateTime matching, it will need to perform a C-FIND request using Date only and filter results outside the time range before returning a response.

If the TimezoneOffsetFromUTC (0008,0201) attribute is included in the request, dates and times in the request are to be interpreted in the specified time zone.

If the "fuzzymatching" query parameter is included in the request and its value is "true", and it is supported, then additional fuzzy semantic matching of person names shall be performed in the manner specified in the DICOM Conformance Statement for the origin server. If Fuzzy Matching is not supported, the response shall include the following Warning header field:

Warning: 299 <service>: "Fuzzy Matching is not supported. Only literal matching has been performed."
Where \(<\text{service}>\) is the base URI for the service. See RFC 7230 Section X.Y for more information about the Warning header field.

### 11.7.2.2 Number of Results and Paging

The origin server shall ensure that search requests are idempotent, that is, two separate search requests with the same target resource and Query Parameters, shall return the same ordered list of results, if the set of matching results on the origin server has not changed.

Given the following values:

- \(\text{offset}\) = the value of the "offset" query parameter.
- \(\text{limit}\) = the value of the "limit" query parameter.
- \(\text{max}\) = the maximum number of results allowed by the origin server in a single response.
- \(\text{total}\) = the total number of matching results.

The set of returned results is determined as follows:

Let \(p = (\text{total} - \text{offset})\) be the number of potential results.

1. If \((p <= 0)\) then no results will be returned; otherwise, the results will start at \((\text{offset} + 1)\).
2. If \((p <= \text{max})\) then \(p\) is the number of results; otherwise \(\text{max}\) is the number of results, in which case the response shall include the following Warning header field:

   Warning: 299: "The number of results exceeded the maximum supported by the origin server. Additional results can be requested."

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

### 11.7.3 Response

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

A failure response should contain a payload describing the error(s) encountered.

#### 11.7.3.1 Status Codes

A successful transaction will have a 200 (OK) status code. Unsuccessful transactions will have one of the failure status codes listed in Section 6.6-1.

#### 11.7.3.2 Response Header Fields

Table 10.6-4 lists the Mandatory, Conditional, and most common Optional header fields for this transaction. See Section 6.5.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>The DICOM media type of the response payload</td>
</tr>
<tr>
<td>Content-Length</td>
<td>uint</td>
<td>C</td>
<td>If no transfer coding has been applied to the payload</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>encoding</td>
<td>C</td>
<td>If a transfer coding has been applied to the payload</td>
</tr>
</tbody>
</table>

All success responses shall contain the Representation Header Fields (see [RFC7231, Section 3.1]) and Payload Header Fields (See ...) with appropriate values, See Section 2.3.3.2.

#### 11.7.3.3 Response Payload

The response payload will contain the following information in the Selected Media Type.
11.7.3.3.1 Search for Study Payload

For each matching Study, the origin server shall return all attributes in accordance with Table 6.7.1-2:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific Character Set</td>
<td>(0008,0005)</td>
<td>If necessary for encoding any returned attributes</td>
</tr>
<tr>
<td>Study Date</td>
<td>(0008,0020)</td>
<td></td>
</tr>
<tr>
<td>Study Time</td>
<td>(0008,0030)</td>
<td></td>
</tr>
<tr>
<td>Accession Number</td>
<td>(0008,0050)</td>
<td></td>
</tr>
<tr>
<td>Instance Availability</td>
<td>(0008,0056)</td>
<td></td>
</tr>
<tr>
<td>Modalities in Study</td>
<td>(0008,0061)</td>
<td></td>
</tr>
<tr>
<td>Referring Physician's Name</td>
<td>(0008,0090)</td>
<td></td>
</tr>
<tr>
<td>Timezone Offset From UTC</td>
<td>(0008,0201)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Retrieve URI</td>
<td>(0008,1190)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Patient's Name</td>
<td>(0010,0010)</td>
<td></td>
</tr>
<tr>
<td>Patient ID</td>
<td>(0010,0020)</td>
<td></td>
</tr>
<tr>
<td>Patient's Birth Date</td>
<td>(0010,0030)</td>
<td></td>
</tr>
<tr>
<td>Patient's Sex</td>
<td>(0010,0040)</td>
<td></td>
</tr>
<tr>
<td>Study Instance UID</td>
<td>(0020,000D)</td>
<td></td>
</tr>
<tr>
<td>Study ID</td>
<td>(0020,0010)</td>
<td></td>
</tr>
<tr>
<td>Number of Study Related Series</td>
<td>(0020,1206)</td>
<td></td>
</tr>
<tr>
<td>Number of Study Related Instances</td>
<td>(0020,1208)</td>
<td></td>
</tr>
</tbody>
</table>

All other Study Level attributes passed as <match> parameters that are supported by the origin server as matching or "includefield" parameters.

The origin server shall ignore all Series and Instance Level attributes contained in "includefield" parameters, and they shall not be returned.

Note

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

11.7.3.3.2 Search for Series Payload

For each matching Series, the origin server shall return all attributes listed in Table 6.7.1-2a:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific Character Set</td>
<td>(0008,0005)</td>
<td>If necessary for encoding any returned attributes</td>
</tr>
<tr>
<td>Modality</td>
<td>(0008,0056)</td>
<td></td>
</tr>
<tr>
<td>Timezone Offset From UTC</td>
<td>(0008,0201)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Series Description</td>
<td>(0008,103E)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Retrieve URI</td>
<td>(0008,1190)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Tag</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Series Instance UID</td>
<td>(0020,000E)</td>
<td></td>
</tr>
<tr>
<td>Series Number</td>
<td>(0020,0011)</td>
<td></td>
</tr>
<tr>
<td>Number of Series Related Instances</td>
<td>(0020,1209)</td>
<td></td>
</tr>
<tr>
<td>Performed Procedure Step Start Date</td>
<td>(0040,0244)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Performed Procedure Step Start Time</td>
<td>(0040,0245)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Request Attribute Sequence</td>
<td>(0040,0275)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>&gt;Scheduled Procedure Step ID</td>
<td>(0040,0009)</td>
<td></td>
</tr>
<tr>
<td>&gt;Requested Procedure ID</td>
<td>(0040,1001)</td>
<td></td>
</tr>
</tbody>
</table>

All other Series Level attributes passed as "match" parameters or "includefield" parameters that are supported by the origin server.

All other Study or Series Level attributes "includefield" parameters that are supported by the origin server, or all available Instance Level attributes, if the "includefield" parameter value is "all"

If study is not specified, all study-level attributes specified in Table 6.7.1-2

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific Character Set</td>
<td>(0008,0005)</td>
<td>If necessary for encoding any returned attributes</td>
</tr>
<tr>
<td>SOP Class UID</td>
<td>(0008,0016)</td>
<td></td>
</tr>
<tr>
<td>SOP Instance UID</td>
<td>(0008,0018)</td>
<td></td>
</tr>
<tr>
<td>Instance Availability</td>
<td>(0008,0056)</td>
<td></td>
</tr>
<tr>
<td>Timezone Offset From UTC</td>
<td>(0008,0201)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Retrieve URI</td>
<td>(0008,1190)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Instance Number</td>
<td>(0020,0013)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Rows</td>
<td>(0028,0010)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Columns</td>
<td>(0028,0011)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Bits Allocated</td>
<td>(0028,0100)</td>
<td>May be absent if no value is available</td>
</tr>
<tr>
<td>Number of Frames</td>
<td>(0028,0008)</td>
<td>May be absent if no value is available</td>
</tr>
</tbody>
</table>

All other Instance Level Attributes passed as "match" parameters that are supported by the origin server.

All other Study, Series or Instance Level attributes "includefield" values that are supported by the origin server, or all available Instance Level Attributes if the "includefield" value is "all"
### Attribute Name

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>If study is not specified, all Study-level attributes specified in</td>
<td>Table 6.7.1-2</td>
<td></td>
</tr>
<tr>
<td>If &lt;series&gt; is not specified, all Series-level attributes specified in</td>
<td>Table 6.7.1-2a</td>
<td></td>
</tr>
</tbody>
</table>

**Note**

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

#### 11.7.4 Media Types

The origin server shall support the following media types:

**Table 2.7-1: Default, Required, and Optional Media Types**

<table>
<thead>
<tr>
<th>Media Type</th>
<th>Requirement Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>application/json</td>
<td>Required</td>
</tr>
<tr>
<td>multipart/related; type=application/dicom+xml</td>
<td>Required</td>
</tr>
</tbody>
</table>

#### 11.7.5 Conformance

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

**An implementation playing the role of origin server shall declare in its Conformance Statement...**

- **Fuzzy-matching**
- **Paging limit / offset**

**Optional resources supported**

**Optional Attributes supported**

#### 11.8 Retrieve Capabilities Transaction

The Retrieve Capabilities Transaction retrieves a Capabilities Description document for the Studies Service. The Capabilities Description document describes the transactions, resources, representations, etc. that are supported by this service. Details of the Capabilities Description document are defined in Annex X. See Section 9.x and Annex X for details.

[TODO: finish]

#### 11.9 Studies Service Notification Transactions

[TODO: Update this section when Supplement193: Web Service Notifications is final]

**The Studies service defines a Notification Sub-Service.** See Section 9.7. The Worklist Service specifies Subscriptions for individual Studies and Collection Subscriptions for the Studies Service itself.

The user agent shall open a notification channel before it subscribes to any resources. Once that channel is opened, the user agent may create subscriptions to either 1) the entire Studies resource, or 2) individual Studies.

The Studies Service supports **Deletion Locks** for both the /studies and its /studies/{study} resources. The origin server shall not delete a resource until all **Deletion Locks** for that resource have been released. A user agent can release a Deletion Lock either by creating a new subscription to the same resource without a Deletion Lock, or by Unsubscribing from the resource.

The Studies resource is a Collection resource. So, when a user agent subscribes to /Studies, a Collection Subscription is created and a filter may be supplied.

When an event related to a subscription occurs, the origin server shall send a Notification containing an Event Report to the Subscriber over the Notification Channel.
If a user agent creates a subscription to /Studies then it will receive notifications of events for all Studies in the collection. When a user agent subscribes to /Studies, it can specify a filter for the event notifications. The origin server will only send notifications for events that satisfy the filter.

Upon receipt of the Subscribe or Unsubscribe request, the origin server shall update the appropriate subscription state as described in Table CC.2.3-2 in PS3.4.

### 11.9.1 Resources

The Worklist Service supports the resources in Table X.Y-Z.

<table>
<thead>
<tr>
<th>Resource</th>
<th>URI Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Studies</td>
<td>/studies</td>
<td>Receive event notifications for all, or a filtered subset, of the Studies in the Studies resource.</td>
</tr>
<tr>
<td>Study</td>
<td>/studies/(study}</td>
<td>Receive event notifications for a single Study.</td>
</tr>
</tbody>
</table>

### 11.9.2 Open Notification Channel Transaction

This transaction opens a Notification Channel between the user agent and the origin server. The Notification Channel must be opened before the user agent can create any subscriptions. This transaction is described in detail in Section 9.Y.

### 11.9.3 Subscribe Transaction

This transaction creates a subscription to a single Study or to all Studies in the /Studies resource. All subscribers to a resource will receive notifications containing event reports for any future events associated with that resource.

The /studies resource is a collection of Studies. When a user agent subscribes to /studies the origin server creates subscriptions to the, possibly filtered, set of current and future Studies contained in the Studies Collection. If the user agent provides a filter parameter, then the target resource shall be the Studies Collection.

### Deletion Locks

#### 11.9.3.1 Request

The request shall be formatted as follows:

```
POST SP /studies/subscriptions{/study} {?retain} {&filter} SP version CRLF
*(header-field CRLF)
CRLF
```

Where,

{-study} = "/" study-uid

Specifies an optional Study, if it is not present then the subscription will be global, that is, the user agent shall have subscriptions to all current and future Studies in the collection.

{-retain} = "retain" "true" / "false"

If present, the subscription shall be created with a Deletion Lock.

{-filter} = "filter" 1#(attributes "=" values)

If present, specifies the key/value pairs describing the filter parameters. Each {attribute} shall refer to an attribute of the Unified Procedure Step IOD (see PS3.3, Section B.26.2). See Section 6.7.1.1 for {attribute} and {value} encoding rules.
11.9.3.1 Request Header Fields

[TODO: add table]
See Section 6.3.1.3

11.9.3.2 Payload
The request shall have no payload.

11.9.3.2 Behavior
The origin server shall create a subscription to the target resource for the user agent.

The origin server shall support the management of subscriptions as specified by the SCP behavior in PS3.4, Section CC.2.3.3.

Upon receipt of the Subscribe request, the origin server shall create a Subscriptions for the user agent to all existing Studies that pass through the provided Filter, if any, and shall ensure that any new Studies created by the origin server, which pass the Filter, if any, will have Subscriptions for the user agent.

It will also attempt to update the Global Subscription State, Filtered Global Subscription and/or Study Subscription State 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 the appropriate response.

11.9.3.3 Response
The response shall have the following format:

```
version SP status-code SP reason-phrase CRLF
Location: subscription CRLF
*(header-field CRLF)
CRLF
```

Where,

- `subscription` is the URI for the subscription resource. This shall include the WebSocket protocol (either WS or WSS) and may include a combination of authority and path.
- `[status-details]` TODO

11.9.3.3.1 Status Codes
A success response shall contain a status code of 201 (Created).

A failure response shall contain a status code from Table 6.X.Y.

11.9.3.3.2 Header Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>R/O</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>uri</td>
<td>R</td>
<td>An opaque URI-reference to the created subscription</td>
</tr>
</tbody>
</table>

Required 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.
```

Required if the request was rejected with a status code of 403, because Filtered Global Subscriptions are not supported, the response message shall include the following Warning header field:

```
Warning: 299 {+SERVICE}: The origin server does not support Global Subscription Filtering.
```
11.9.3.3 Payload

A success response payload shall be empty.

A failure response payload shall contain a Status Details document.

11.9.3.4 Media Types

An origin server shall support all media types allowed in the request.

The media types supported by this transaction are:

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

11.9.4 Unsubscribe Transaction

The target subscription shall be deleted, and no future Event Notifications related to that subscription will be sent to the subscription holder.

If the target resource is a Study subscription, it shall be deleted.

If the target resource is the Studies, the "suspend" query parameter may be included in the request. If the value of the "suspend" parameter is "true" then all existing Workitems subscriptions that resulted from this subscription shall not be deleted. If the "suspend" query parameter is either "false" or not present, all current and future subscriptions shall be deleted.

11.9.4.1 Request

The request shall have the following syntax:

```
DELETE SP /subscribers{/workitem}{?existing} SP version CRLF
*(header-field CRLF)
CRLF
```

```
DELETE SP /{subscription}{?suspend} SP version CRLF
*(header-field CRLF)
CRLF
```

Where,

- `{subscription}` = uri
  A URI returned by the Subscribe transaction that references a Subscription.
- `{?suspend}` = "suspend=" "true" / "false"
  If present, indicates that the subscription should be suspended, i.e. current subscriptions should not be deleted, but no future subscriptions should be created.

11.9.4.1.1 Request Header Fields

See Section 6.X.Y.

11.9.4.1.2 Request Payload

The request payload shall be empty.

11.9.4.2 Behavior

If the "suspend" query parameter is "true" the origin server shall no longer automatically subscribe the user agent to newly-created Studies; however, it shall not delete existing subscriptions to Studies. If the "suspend" query parameter is "false" or not present the origin server shall remove any of the user agent's existing subscriptions to the target resource.
The origin server shall conform to the behavior described in Section 12.1.8.4, “Behavior”. [TODO: rewrite for clarity?]

11.9.4.3 Response

The response shall have the following format:

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

11.9.4.3.1 Status Codes

A success response shall contain a status code of 204 (OK), indicating that the target subscription(s) have been suspended or deleted, and no payload.

A failure response will contain an appropriate status code from Table 12.2-A.

11.9.4.3.2 Response Header Fields

[TODO: add table]

See Section 6.5.

11.9.4.3.3 Response Payload

A success response payload shall be empty.

A failure response payload shall contain a Status Details document.

11.9.5 Notify

This transaction sends Event Reports from the origin server to a user agent over an established Notification Connection. See Section 9.X.Y.

11.9.5.1 Request

The request shall use the WebSocket Data Frame transmission protocol. See 9.X.Y.

11.9.5.1.1 Payload

The Event Report shall contain all mandatory attributes described by the Conformance Statement and Capabilities document of the Defining Service for the event type.

11.9.5.1.1.1 Event Report Format

Events Reports are encoded as WebSocket Data Frames. The Data Frame Type is specified by the media type of the Event Report.

The Event Report shall contain the Instance Availability Notification Contained on Table 10.8-X below, which is based on the IAN described in PS 3.4 Section R.1.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Tag</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific Character Set</td>
<td>(0008,0005)</td>
<td>1C/1C</td>
</tr>
<tr>
<td>All other Attributes of the SOP Common Module</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Referenced Performed Procedure Step Sequence</td>
<td>(0008,1111)</td>
<td>2/2</td>
</tr>
<tr>
<td>&gt;Referenced SOP Class UID</td>
<td>(0008,1150)</td>
<td>1/1</td>
</tr>
<tr>
<td>&gt;Referenced SOP Instance UID</td>
<td>(0008,1155)</td>
<td>1/1</td>
</tr>
</tbody>
</table>

--- Draft ---
WebSocket Events are encoded as WebSocket data frames with an opcode of "%x1" (text). The frame payload data shall be a DICOM JSON dataset containing the attributes of the Event Report. The following is an example Instance Availability Notification payload:

```json
{
00000002": ["1.2.840.10008.5.1.4.33" ],
00000100": [320 ],
00000110": [23 ],
00008000": [0 ],
00001000": ["1.2.840.10008.5.1.4.34.6.4.2.34.2223.334.1" ],
00081115": {
0020000E": ["1.2.840.10008.5.1.4.34.6.4.2.34.2223.334.1.1" ],
...
0020000D": ["1.2.840.10008.5.1.4.34.6.4.2.34.2223.334.1" ]
}
```

Note

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

--- Draft ---
However, the Defining Service may extend the Event Report with additional attributes. The following is an example "application/json" WebSocket payload:

```
{
    "00000002": [ "1.2.840.10008.5.1.4.34.6.4" ],
    "00000100": [ 256 ],
    "00000110": [ 23 ],
    "00001000": [ "1.2.840.10008.5.1.4.34.6.4.2.3.44.22231" ],
    "00001001": [ 1 ],
    "00741238": [ "SCHEDULED" ],
    "00744041": [ "READY" ]
} CRLF
```

### 11.9.5.3 Behavior

Each service shall define the scenarios in which an origin server sends Event Reports to a subscriber, as well as the content of the Event Report messages.

### 11.9.5.4 Response

The user agent shall send a response containing a success or failure acknowledgement. See 9.X.Y. An acknowledgement is encoded as a WebSocket data frame with an opcode of "%x1" (text).

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

### 11.10 Conformance

An implementation conforming to the RS Studies service shall implement the Retrieve Capabilities and Retrieve DICOM transactions, and may implement the other transactions defined by the service. It shall also document whether it plays the role of origin server or user agent, or both.

For each transaction supported, an implementation shall document in its Conformance Statement the Information Objects (resources) supported, and any other conformance requirements defined by the transaction.
12 RS Unified Worklist Service (UPS)

The Worklist Service defines a RESTful interface to the Unified Procedure Step Service SOP Classes defined in PS3.3, Section XX and PS3.4, Section XX.

The Worklist Service manages a single Worklist containing a sequence of Workitems. User agents can create, retrieve, update, claim, complete, cancel, and search for Workitems in the Worklist.

12.1.1 Workitems

Each Workitem unifies the details of a single requested procedure step together with the details of the corresponding performed procedure step. The Workitem embodies a one to one relationship between the procedure step requested and the procedure step performed. See PS3.3, Annex CC.

Workitems may be used to represent a variety of scheduled tasks such as: Image Processing, Quality Control, Computer Aided Detection, Interpretation, Transcription, Report Verification, or Printing.

Each Workitem resource can be referenced by a URI. A Workitem URI has the following syntax:

/worklist/{workitem}

A Workitem URI is either canonical or opaque. A canonical Workitem has the following syntax:

workitem = uid

An opaque workitem has the following syntax:

workitem = opaque

Where <opaque> is a Relative URI created by the origin server, and used by the user agent to identify the Workitem as the target resource of a request.

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 knows that the Workitem did exist, but has been deleted, then a 410 (Gone) response should be sent; otherwise the response should be a 404 (Not Found). See Section CC.2.1.3 in PS3.4.

12.1.1.1 Deletion Locks

A Workitem that has moved to the Completed or Cancelled state, and has no deletion locks may be deleted by the origin server. Once a Workitem has been deleted by the origin server, a request for that Workitem shall receive a 404 – Not Found response. See Section CC.2.1.3 in PS3.4.

12.1.2 Subscriptions and Notifications

A user agent can be notified of certain events related to the Worklist or Workitems. In order to receive notifications the user agent shall first create a notification channel between itself and the origin server using the Open Notification Channel transaction.

Once the Notification Channel has been opened the user agent can create subscriptions to either

User agents can also create Subscriptions to the Worklist or to individual Workitems. A subscription provides the subscribing user agent with event notifications related to the resource. Each notification contains an Event Report

--- Draft ---
related to a Workitem. In order to receive notifications, the user agent must first open a notification channel between itself and the origin server.

12.1.2.1 Workitem Subscriptions

12.1.2.2 Worklist Subscriptions

12.1.2.2.1 Subscription Filters

[Explain filters]

12.2 Transactions Overview

An origin server shall support all transactions defined by the Worklist Service, and shall comply with all requirements placed on it as described in PS3.4 Annex CC and where else?

12.2.1 Service Transactions

Table 11.2-1 shows the transactions, along with a brief description, of the service level transactions supported by this service. The service itself, i.e. ‘/’, is the target resource of these transactions.

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Method</th>
<th>Resource</th>
<th>Request Payload</th>
<th>Response Payload</th>
<th>DIMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve Capabilities</td>
<td>OPTIONS</td>
<td>/</td>
<td>-</td>
<td>capabilities</td>
<td></td>
</tr>
<tr>
<td>Open Notification Connection</td>
<td>GET</td>
<td>/</td>
<td>-</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>Send Notification</td>
<td>N/A</td>
<td>channel</td>
<td>Event Report</td>
<td>Ack/Nak</td>
<td>N-EVENT-REPORT</td>
</tr>
</tbody>
</table>

[TODO: Add links to table above to the corresponding DIMSE spec]

12.2.2 Worklist Transactions

Table 11.2-2 shows the method, resource, request and response payloads and the corresponding DIMSE action for the transactions defined by this service. The DIMSE column contains links to the corresponding DIMSE service specifications in PS3.3.

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Method</th>
<th>Resource</th>
<th>Success Payload</th>
<th>DIMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Workitem</td>
<td>POST / PUT</td>
<td>/worklist/{workitem}</td>
<td>workitem</td>
<td>N-CREATE</td>
</tr>
<tr>
<td>Retrieve Workitem</td>
<td>GET</td>
<td>/worklist/{workitem}</td>
<td>N/A workitem</td>
<td>N-GET</td>
</tr>
<tr>
<td>Update Workitem</td>
<td>POST / PUT</td>
<td>/worklist/{workitem}?transactionID</td>
<td>dataset / workitem</td>
<td>N-SET</td>
</tr>
<tr>
<td>Claim Workitem</td>
<td>GET</td>
<td>/worklist/{workitem}/claim</td>
<td>N/A N/A workitem</td>
<td>N-ACTION</td>
</tr>
<tr>
<td>Complete Workitem</td>
<td>POST / PUT</td>
<td>/worklist/{claim}/complete</td>
<td>dataset / workitem</td>
<td>N-ACTION</td>
</tr>
<tr>
<td>Cancel Workitem</td>
<td>POST / PUT</td>
<td>/worklist/{claim}/cancel</td>
<td>dataset / workitem</td>
<td>N-ACTION</td>
</tr>
<tr>
<td>Request Workitem Cancellation</td>
<td>POST</td>
<td>/worklist/{workitem}/cancel</td>
<td>dataset / workitem</td>
<td>N-ACTION</td>
</tr>
<tr>
<td>Search Worklist</td>
<td>GET</td>
<td>/worklist/{?parameters}</td>
<td>parameters results</td>
<td>C-FIND</td>
</tr>
</tbody>
</table>

Notification Sub-Service
12.2.3 Workitem States and Transitions

Figure 11.2-1 shows the four states of a Workitem, along with the legal state transitions. All state transitions are atomic, that the transition from one state to another happens instantaneously from the point of view of any observer.

Once created a Workitem is always in some state. It is never between one state and another.

The following four transaction cause Workitem state transitions:

1. Create Workitem

A user agent uses this transaction to insert a new Workitem into the Worklist. Once inserted in the Worklist, the Workitem shall be in the Scheduled state. A Workitem shall only be inserted into a Worklist once.

2. Claim Workitem

A user agent uses this transaction to claim a Workitem to perform. The Workitem shall be in the Scheduled state when the origin server begins processing the request, and shall transition to the In Progress state before the response is returned. This transaction returns a Claim on the Workitem. A Claim is an opaque URI that can be used in a Complete or Cancel transaction. If the Workitem is in any state than Scheduled the origin server shall return a failure response.

3. Complete Workitem

A user agent uses this transaction to complete a Workitem that it has claimed. The Workitem shall be in the In Progress state when the request is received, and shall transition to the Completed state before the response is returned. If the user agent has not claimed the target Workitem, or if the Workitem is not in the Claimed state the origin server shall return a failure response.

4. Cancel Workitem

A user agent uses this transaction to cancel a Workitem that it has claimed, or to cancel a Workitem that is in the Scheduled state, or to request the cancellation of a Workitem that has been claimed by another user agent.

If the user agent has claimed the Workitem, it shall be in the In Progress state when the request is received, and shall transition to the Cancelled state before the response is returned.
If the Workitem is in the Scheduled state when the request is processed, it shall be in the Cancelled state when the response is returned.

If the Workitem has been claimed by another user agent, and that user agent has a notification channel open, then the origin server shall send a Request to Cancel Notification to the claiming user agent. The claiming user agent may honor or ignore the Request to Cancel Notification. If the claiming user agent does not have an open notification channel, the origin server shall return a failure response.

Once the Workitem is in the Cancelled state, and all subscribers have been notified (See Section 11.X.Y), the origin server may remove (delete) the Workitem from the Worklist.

Table 11.2-3: State Changing Transactions

<table>
<thead>
<tr>
<th>Transaction</th>
<th>State</th>
<th>Status</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Workitem</td>
<td>none</td>
<td>Scheduled</td>
<td>204 Creates new Workitem in the Scheduled State.</td>
</tr>
<tr>
<td>Claim Workitem</td>
<td>Scheduled</td>
<td>In Progress</td>
<td>204 Returns a Claim (URI) in Location header field</td>
</tr>
<tr>
<td>Complete Workitem</td>
<td>In Progress</td>
<td>Completed</td>
<td>204 Uses Claim as target resource</td>
</tr>
<tr>
<td></td>
<td>Completed</td>
<td>Completed</td>
<td>304 Error?</td>
</tr>
<tr>
<td></td>
<td>Cancelled</td>
<td>Completed</td>
<td>304 Error?</td>
</tr>
<tr>
<td>Cancel Workitem</td>
<td>Scheduled</td>
<td>Cancelled</td>
<td>204 Must use Conditional Update</td>
</tr>
<tr>
<td></td>
<td>In Progress</td>
<td>Cancelled</td>
<td>204 If user agent is Claim holder then cancel immediately; otherwise, the origin server should send a cancellation Notification sent to the claim holder requesting cancellation. If override value present Workitem moved to cancelled and claim invalidated</td>
</tr>
<tr>
<td></td>
<td>Cancelled</td>
<td>Cancelled</td>
<td>304 Error?</td>
</tr>
</tbody>
</table>

Table 11.2-3 shows the legal state transitions for Workitems.

When a Workitem is created, it is inserted into the Worklist in the Scheduled state. When a Workitem is claimed, it shall be in the Scheduled state and transitions to the In Progress state before the response is returned.

When a Workitem is completed, it shall be in the In Progress state and transitions to the Completed state. When a Workitem is cancelled, by the user agent that holds the claim, it shall be in the In Progress state and moves to the Cancelled state.

When a Workitem is cancelled, by the user agent that does not hold the claim, it shall be in the In Progress state. The origin server will send a Request to Cancel Notification to the user agent that holds the claim requesting that it cancel the Workitem.

All transactions not listed in Table 11.2.-3 do not cause state transactions. When the response is returned the Workitem shall be in the same state as when the request was received.

When a state changing request is received, if the Workitem is in any other state than those listed in table 11.2-3, then the request is invalid, and the origin server shall generate a failure response including a Status Details document.

12.2.4 Updating Workitems

12.2.4.1 Optimistic Concurrency

HTTP/1.1 provides mechanisms for ensuring consistent updates using conditional requests. See [RFC7232].

- All origin servers shall support ETag and Last Modified header fields.
- All update requests shall have If-Match header field.
12.2.5 Request Overview

12.2.5.1 Resources

There are two resources defined by this service:

- **Worklist**: A sequence of Workitems managed by the origin server.
- **Workitem**: A SOP Instance with a dataset containing the Unified Procedure Step attributes specified in [TODO: ref].

The resources defined by this service are specified in Table 11.2-4.

<table>
<thead>
<tr>
<th>Resource</th>
<th>URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worklist</td>
<td>/worklist</td>
<td>An alternative URI for the root path of the Worklist Service.</td>
</tr>
<tr>
<td>Workitem</td>
<td>/worklist/{workitem}</td>
<td>An alternative URI for a Workitem being managed by this service.</td>
</tr>
</tbody>
</table>

12.2.5.2 Query Parameters

Each transaction specifies its Query Parameters, if any.

12.2.5.3 Header Fields

Each transaction defines required and optional header fields. See Section 6.3.1.3 [TODO: add table]

12.2.5.4 Payload

The request payload is specified by the specific transaction. Table 11.2-5 gives an overview of request and response payloads.

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Request Payload</th>
<th>Response Payload*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve Capabilities</td>
<td></td>
<td>capabilities</td>
</tr>
<tr>
<td>Create Workitem</td>
<td>workitem</td>
<td></td>
</tr>
<tr>
<td>Retrieve Workitem</td>
<td></td>
<td>workitem</td>
</tr>
<tr>
<td>Update Workitem</td>
<td>dataset</td>
<td></td>
</tr>
<tr>
<td>Cancel Workitem</td>
<td>dataset</td>
<td></td>
</tr>
<tr>
<td>Request Workitem Cancellation</td>
<td>dataset</td>
<td></td>
</tr>
<tr>
<td>Search Workitem</td>
<td>Parameters?</td>
<td>results</td>
</tr>
<tr>
<td>Change Workitem State</td>
<td>dataset</td>
<td></td>
</tr>
<tr>
<td><strong>Notification Sub-Service</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Notification Connection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subscribe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unsubscribe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Send Notification</td>
<td>Event Report</td>
<td>ACK or NAK</td>
</tr>
</tbody>
</table>

*Note: The payload for a failure response for any transaction should be a Status Details document. See [ref].

12.2.6 Behavior Overview

Each transaction specifies its behavior.
12.2.7 Response Overview

12.2.7.1 Status Codes

The status codes defined in Table 11.2-6 below are the primary status codes used by this service; however, any of the status codes in Table 6.X-Z might be used by the transactions defined by this service.

Table 11.2-6: Status Codes

<table>
<thead>
<tr>
<th>Status</th>
<th>Code</th>
<th>Reason Phrase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td>200</td>
<td>OK</td>
<td>The transaction was successful</td>
</tr>
<tr>
<td></td>
<td>201</td>
<td>Created</td>
<td>The Workitem was created</td>
</tr>
<tr>
<td></td>
<td>204</td>
<td>Not Content</td>
<td>Indicates that the transaction succeeded and the response has no payload.</td>
</tr>
<tr>
<td></td>
<td>304</td>
<td>Not Modified</td>
<td>Indicates that the resource has not been modified. See Conditional Retrieve &amp; Update</td>
</tr>
<tr>
<td>Failure</td>
<td>400</td>
<td>Bad Request</td>
<td>The UPS-RS origin server was unable to understand the request</td>
</tr>
<tr>
<td></td>
<td>404</td>
<td>Not found</td>
<td>The specified Workitem does not exist or is not managed by this origin server.</td>
</tr>
<tr>
<td></td>
<td>410</td>
<td>Gone</td>
<td>The target resource is no longer available at the origin server and that this condition is likely to be permanent.</td>
</tr>
<tr>
<td></td>
<td>409</td>
<td>Conflict</td>
<td>The request cannot be performed for one of the following reasons: 1. the submitted request is inconsistent with the current state of the UPS Instance, 2. the Transaction UID is missing, or 3. the Transaction UID is incorrect</td>
</tr>
</tbody>
</table>

For additional status codes see Section 6.6.

12.2.7.2 Header Fields

See Section 6.X.
[TODO: add table]

12.2.7.3 Payload

Success responses may or may not have a payload depending on the transaction. Failure Responses shall contain a Status Details document. See Table X.Y-Z above for specific payload information by transaction.

12.2.8 Media Types

The media types supported by the Worklist Service are:

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

The Worklist Service shall not support Metadata or Bulkdata media types.

12.2.9 Web Services and DIMSE Terminology

Add a table of language mappings from PS3.4 to this section

<table>
<thead>
<tr>
<th>RS Term</th>
<th>DIMSE Term</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worklist</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Workitem</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

--- Draft ---
12.3 Worklist Transactions

[TODO: This will be updated when CP_jfp19_ups_fixes is final]

12.3.1 Create Workitem Transaction

This transaction requests the creation of a Scheduled Workitem resource on the Worklist resource.

12.3.1.1 Request

This transaction uses the POST method and has the following format:

```
POST /worklist{/uid} SP version CRLF
Accept: media-type CRLF
Content-Type: media-type CRLF
(Content-Length uint / Content-Encoding: encoding) CRLF
[if-None-Match: "*" CRLF]
*(header-field CRLF)
CRLF
workitem
```

Where

- `uid` An optional Affected SOP Instance UID.
- `workitem` A UPS Workitem

12.3.1.2 Query Parameters

This transaction has no Query Parameters.

12.3.1.3 Header Fields

<table>
<thead>
<tr>
<th>Header</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>The media-type of the payload</td>
</tr>
<tr>
<td>Content-Encoding</td>
<td>encoding</td>
<td>C</td>
<td>Required if payload content is encoded</td>
</tr>
</tbody>
</table>
Content-Length | uint | C | Required if payload content is not encoded

See Also Section 6.5.

12.3.1.4 Payload

The payload shall have a single part, which shall contain a single Workitem encoded in the media type specified in the Content-Type header field. The payload shall contain all data elements to be stored, with the possible exception of the Affected SOP Instance UID (0000,1000). If no Affected SOP Instance UID is specified the origin server will create and assign a new UID to the created Workitem. See [TODO: Insert the ref to creating 2.52.uuid UIDs].

The Workitem shall comply with all instance requirements in the Req. Type N-CREATE column of Table CC.2.5-3 in PS3.4.

12.3.1.2 Behavior

If no Affected SOP Instance UID is contained in the Workitem in the request payload, the origin server creates a new UID identifier for the Workitem and inserts it into the Affected SOP Instance UID of the Workitem. It then inserts the Workitem into the Worklist in the Scheduled state and returns a URI that references the newly created Workitem in the Location header field of the response.

The origin server shall create and maintain the Workitem as specified in the request, and as specified by the SCP behavior defined in PS3.4 Section CC.2.5.3.

12.3.1.3 Response

The response has the following format:

```
version SP status-code SP reason-phrase CRLF
ETag: entity-tag CRLF
Location: reference CRLF
Last-Modified: HTTP-date CRLF
.*(header-field CRLF)
CRLF
```

Where

```
reference is a URI-reference to the resource corresponding to the newly created Workitem.
```

Note:

Typically the <reference> is an opaque URI defined by the origin server; however, if the origin server is a proxy service for a DIMSE service, then the <reference> URI should have the format:

```
/worklist/{uid}
```

Where the <uid> is the Affected SOP Instance UID of the <workitem>.

12.3.1.3.1 Status Codes

A successful response shall contain a status code of 201 (Created).

A failure response shall contain an appropriate failure status code from Table 6.X.Y.

12.3.1.3.2 Header Fields

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Length</td>
<td>uri</td>
<td>O</td>
<td>The value is a URI-reference to the specific representation of the created Workitem. See [RFC7231] Section 3.1.4.2.</td>
</tr>
<tr>
<td>ETag</td>
<td>entity-tag</td>
<td>M</td>
<td>Entity tag for &lt;workitem&gt; in payload</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>HTTP-date</td>
<td>M</td>
<td>Date and Time of most recent modification to &lt;workitem&gt;</td>
</tr>
<tr>
<td>Location</td>
<td>uri</td>
<td>M</td>
<td>The value is a URI-reference to the created Workitem. The format of the URI is up to the origin server. The URI shall correspond to the created resource, not to a particular</td>
</tr>
</tbody>
</table>
Warning: see below

If <workitem> was modified by origin server include warning below

See also Section 6.5.

If the 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.
```

12.3.1.3.3 Payload

The response shall have no payload.

12.3.2 Retrieve Workitem Transaction

This transaction retrieves a Workitem from the target resource, which shall be a Worklist.

12.3.2.1 Request

This transaction uses the GET method and has the following format:

```
GET SP /{workitem} SP version CRLF
Accept media-types CRLF
[If-None-Match: entity-tag CRLF]
```

```
*(header-field CRLF)
```

Note

The If-None-Match header field should be used for conditional retrieves. See [RFC7232].

12.3.2.1.1 Resource

The URI of the target Workitem.

12.3.2.1.2 Query Parameters

This request shall have no Query Parameters.

12.3.2.1.3 Header Fields

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>1#media-type</td>
<td>M</td>
<td>List of one or more media types</td>
</tr>
<tr>
<td>If-None-Match</td>
<td>#entity-tag</td>
<td>C</td>
<td>Required for Conditional Retrieve</td>
</tr>
<tr>
<td>Cache-Control</td>
<td>&quot;no-cache&quot;</td>
<td>O</td>
<td>Used to avoid ‘stale’ cache results. If present, specifies that search results returned should be current and not cached. See Section 6.3.1.3</td>
</tr>
</tbody>
</table>

12.3.2.1.4 Payload

The request shall have no payload.

12.3.2.2 Behavior

The origin server shall return the specified Workitem in an Acceptable Media Type. However, The returned Workitem shall not contain the Transaction UID (0008,1195) attribute of the Workitem, since that is an access lock that should only be known to the user agent that requested the lock.
12.3.2.3 Response

The response shall have the following format:

```
version SP status-code SP reason-phrase CRLF
Content-Type: media-type CRLF
2465 [Content-Location: uri CRLF]
ETag: entity-tag CRLF
Last-Modified: http-date
*(header-field CRLF)
CRLF
```

A successful response will have a payload containing the target Workitem in the Selected Media Type; however, the Workitem shall not contain the Transaction UID (0008,1195) attribute.

A failure response will have a failure status code (see Section X.Y), and a payload containing a Status Details document in the Selected Media Type.

12.3.2.3.1 Status Codes

The response shall have an appropriate status code. Table 11.3 lists the most common status codes for this transaction. For other Status Codes see Section 6.X.Y.

<table>
<thead>
<tr>
<th>Code</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 (OK)</td>
<td>The normal success response</td>
</tr>
<tr>
<td>304 (Not Modified)</td>
<td>This code is returned if the request is a Conditional Get request and the ETag value in the request corresponds to the current version of the resource</td>
</tr>
<tr>
<td>404 (Not Found)</td>
<td>This code is returned if the origin server has no knowledge of the Workitem. See Section CC.2.1.3 in PS3.4.</td>
</tr>
<tr>
<td>410 (Gone)</td>
<td>This code is returned if the origin server knows that the Workitem did exist, but has been deleted,</td>
</tr>
</tbody>
</table>

12.3.2.3.2 Header Fields

Table 11.3 lists the most common header fields for this response. See Also Section 6.X.Y.

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>media type of &lt;workitem&gt; representation in payload</td>
</tr>
<tr>
<td>Content-Location</td>
<td>uri</td>
<td>O</td>
<td>A URI referencing the &lt;workitem&gt; on the origin server</td>
</tr>
<tr>
<td>ETag</td>
<td>entity-tag</td>
<td>M</td>
<td>Entity tag for &lt;workitem&gt; in payload</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>HTTP-date</td>
<td>M</td>
<td>Date and Time of most recent modification to &lt;workitem&gt;</td>
</tr>
</tbody>
</table>

12.3.2.3.3 Payload

A success response has a single part payload containing the requested Workitem in the Selected Media Type.

A failure response payload shall contain a Status Details document in the Selected Media Type.

12.3.3 Update Workitem Transaction

The transaction modifies an existing Workitem, which shall be in the Scheduled or In Progress state. If the Workitem is in the Scheduled state then the target URI is the normal URI of the Workitem; otherwise, the Workitem shall be in the In Progress state and the target URI shall be the Claimed Workitem URI that was returned in the response to the Claim Transaction.

This transaction uses the Conditional Update mechanism described in Section X.Y.

There are two forms of this transaction:
1. The first uses the POST method and the payload contains a <dataset> with the data elements to be updated.
2. The second uses the PUT method and the payload contains a <workitem> that updates the existing Workitem in the Worklist.

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

The request processing shall be atomic (indivisible) and idempotent (repeating the same request has no additional effect), all modifications contained in the request shall leave the Workitem in an internally consistent state.

12.3.3.1 Request

CP: No longer takes a transaction UID; however, it must use the conditional update mechanism or the Lock mechanism.

The request has the following format:

```
POST /{workitem} {workitem} version CRLF
Content-Type: media-type CRLF
If-Match: entity-tag CRLF
[Cache-Control: no-cache, private CRLF]
*(header-field CRLF)
CRLF
```

Where, method

- POST if the <payload> contains a <dataset>, or
- PUT if the <payload> contains a <workitem>

12.3.3.1.1 Resource

The target resource for this transaction is a Workitem, if in the Scheduled state, or a Claimed Workitem, if in the In Progress state.

12.3.3.1.2 Query Parameters

This transaction has no Query Parameters.

12.3.3.1.3 Header Fields

Table 11.3-Z lists the most common header fields for this request.

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>&lt;media-types&gt;</td>
<td>M</td>
<td>List of one or more media types</td>
</tr>
<tr>
<td>If-Match</td>
<td>entity-tag</td>
<td>M</td>
<td>The entity-tag of target Workitem</td>
</tr>
<tr>
<td>Cache-Control</td>
<td>no-cache, private</td>
<td>O</td>
<td>[TODO: Explain directives]</td>
</tr>
</tbody>
</table>

See also Section 6.5

12.3.3.1.4 Payload

The request payload shall have a single part, and shall contain either:

1. a <dataset> with only new data elements or data elements whose values are to be modified, or
2. a <workitem> that was retrieved from the origin server using the Retrieve Workitem transaction, which the user agent has updated with new or modified data elements.
All modifications to the Workitem shall comply with all requirements described in PS3.4 Section CC.2.6.2.

12.3.3.2 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.

12.3.3.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
[Content-Location: workitem CRLF]
[ETag: entity-tag CRLF]
[Last-Modified: http-date CRLF]
*(header-field CRLF)
CRLF
[status-details]
```

Where

- `Workitem` is a URI reference to the updated Workitem
- `entity-tag` is the (new) entity tag of the updated Workitem
- `status-details` is a Status Details document that shall be present if...

12.3.3.3.1 Status Codes

The response shall have an appropriate status code. Table 11.3 lists the most common status codes for this transaction. For other Status Codes see Section 6.X.Y.

<table>
<thead>
<tr>
<th>Code</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 (OK)</td>
<td>The update was successful and the payload contains a Status Details document.</td>
</tr>
<tr>
<td>204 (No Content)</td>
<td>The update was successful and the response has no payload.</td>
</tr>
<tr>
<td>409 (Conflict)</td>
<td>[TODO: finish]</td>
</tr>
<tr>
<td>412</td>
<td></td>
</tr>
<tr>
<td>428</td>
<td></td>
</tr>
</tbody>
</table>

If the target Workitem was in the In Progress state and the target URI did not reference a Claimed Workitem, or the `<dataset>` or `<workitem>` was inconsistent with the current state of the `<workitem>`; then the response shall contain a 409 (Conflict) status code.

12.3.3.3.2 Header Fields

Table 11.3-Z lists the most common header fields for this response. See Also Section 6.X.Y.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Location</td>
<td>uri</td>
<td>O</td>
<td>The value is a URI-reference to the specific representation of the updated Workitem. See [RFC7231] Section 3.1.4.2.</td>
</tr>
<tr>
<td>ETag</td>
<td>entity-tag</td>
<td>M</td>
<td>Entity tag for <code>&lt;workitem&gt;</code> in payload</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>HTTP-date</td>
<td>M</td>
<td>Date and Time of most recent modification to <code>&lt;workitem&gt;</code></td>
</tr>
<tr>
<td>Warning</td>
<td>see below</td>
<td>O</td>
<td>If <code>&lt;workitem&gt;</code> was modified by origin server include warning below</td>
</tr>
</tbody>
</table>
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:

\[
\text{Warning: 299 \{+service\}: The Workitem was updated with modifications.}
\]

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

\[
\text{Warning: 299 \{+service\}: Requested optional attributes are not supported.}
\]

If the request was rejected with a 409 status code, the response shall include a Warning header field with one of the following messages that best describes the nature of the conflict:

\[
\text{Warning: 299 \{+service\}: The Target URI did not reference a claimed Workitem.}
\]

\[
\text{Warning: 299 \{+service\}: The submitted request is inconsistent with the current state of the Workitem.}
\]

### 12.3.3.3 Payload

A success response shall have no payload. A failure response shall have a payload containing a Status Details document in the Selected Media Type.

### 12.3.3.4 Claim Workitem Transaction

This transaction enables a user agent to "claim" a Workitem in order to perform the procedure step it embodies. The target Workitem shall be in the Scheduled state. A successful Claim changes the Workitem's state from Scheduled to In Progress, and gives the user agent exclusive access to the Workitem through the Claimed Workitem URI contained in the Location header field of the response. The Claimed Workitem URI is opaque. It is defined by the origin server.

If the origin server is a proxy to a DIMSE service, it shall also create a Transaction UID (0008,1195) for the Workitem before transmitting the state change to the DIMSE SCP.

#### 12.3.4.1 Request

The request shall have the following format:

\[
\text{POST SP }{/\text{workitem}}/{\text{claim}} \text{ SP version CRLF}
\]

\[
*(header-field \text{ CRLF})
\]

\[
\text{CRLF}
\]

#### 12.3.4.1.1 Header Fields

[TODO: add table]

There are no required header fields for this transaction.

#### 12.3.4.1.2 Payload

The request shall have no payload.

#### 12.3.4.2 Behavior

The origin server shall modify the target Workitem's state from Scheduled to In Progress. If the Workitem is in a state other than Scheduled the origin server will return a failure response.

--- Draft ---
12.3.4.3  Response

The response shall have the following format:

```
version SP status-code SP reason-phrase CRLF
Accept: media-type CRLF
Location: claimed-workitem CRLF
*(header-field CRLF)
```

Where

- `claimed-workitem` A new unique and opaque URI to which the requesting user agent has access.
- `workitem` A representation of the `<workitem>` in an Acceptable Media Type

12.3.4.3.1  Status Codes

The response shall have an appropriate status code. Table 11.3 lists the most common status codes for this transaction. For other Status Codes see Section 6.X.Y.

<table>
<thead>
<tr>
<th>Code</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 (OK)</td>
<td>The Workitem was successfully Claimed and the payload contains the Workitem.</td>
</tr>
<tr>
<td>409 (Conflict)</td>
<td>The target Workitem was not in the Scheduled state.</td>
</tr>
</tbody>
</table>

12.3.4.3.2  Header Fields

Table 11.3-Z lists the most common header fields for this response. See Also Section 6.X.Y.

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>uri</td>
<td>M</td>
<td>A new unique URI-reference to the claimed Workitem</td>
</tr>
<tr>
<td>Content-Location</td>
<td>uri</td>
<td>O</td>
<td>The value is a URI-reference to the specific representation of the claimed Workitem. See [RFC7231] Section 3.1.4.2.</td>
</tr>
<tr>
<td>ETag</td>
<td>entity-tag</td>
<td>M</td>
<td>Entity tag for <code>&lt;workitem&gt;</code> in payload</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>HTTP-date</td>
<td>M</td>
<td>Date and Time of most recent modification to <code>&lt;workitem&gt;</code></td>
</tr>
<tr>
<td>Warning</td>
<td>see below</td>
<td>O</td>
<td>If <code>&lt;workitem&gt;</code> was modified by origin server include warning below</td>
</tr>
</tbody>
</table>

12.3.4.3.3  Payload

A success response shall have a payload containing the claimed Workitem.

A failure response shall have a payload containing a Status Details document in an Appropriate Media Type.

12.3.5  Complete Workitem Transaction

This transaction enables a user agent to complete a Claimed Workitem. The target URI shall be the `<claimed-workitem>` URI returned in the response to the Claim transaction. A successful Complete transaction changes the Workitem’s Procedure Step State (0074,1000) from IN-PROGRESS to COMPLETED, and makes the Workitem generally accessible again, i.e. the claiming user agent no longer has exclusive access to the Workitem.

Once the Workitem has moved to the Completed state, it shall not modified.

[TODO: Discuss Atomicity]
12.3.5.1 Request
The request shall have the following format:

```
PUT SP /{claimed-workitem}/complete SP version CRLF
*(header-field CRLF)
CRLF
```

Where

```
claimed-workitem
```

is the URI return in the Location header field of the Claim transaction response.

12.3.5.1.1 Header Fields

[TODO: Add table]
This request has no required header fields.

12.3.5.1.2 Payload

This request has no payload.

12.3.5.2 Behavior

The origin server shall modify the target Workitem’s state from IN-PROGRESS to COMPLETED. If this request is made by any user agent other than the claiming user agent the origin server shall return a failure response.

The origin server shall refuse to perform this transaction, unless the Workitem meets the final state requirements specified in Table CC.2.5-3.

[TODO: discuss Atomicity]

12.3.5.3 Response

The response shall have the following format:

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

12.3.5.3.1 Status Codes

A success response will contain a status code of 200 (OK).

If the Workitem is already in the COMPLETED state, and it meets the final state requirements specified in Table CC.2.5-3, then the response status code will be 304 (Not Modified).

Otherwise, a failure status code shall be returned. A failure response will contain an appropriate status code from Table 6.X.Y, and an appropriate Status Details message.

12.3.5.3.2 Header Fields

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>uri</td>
<td>M</td>
<td>A URI-reference to the completed Workitem resource</td>
</tr>
<tr>
<td>ETag</td>
<td>entity-tag</td>
<td>M</td>
<td>Entity tag for completed &lt;workitem&gt;</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>HTTP-date</td>
<td>M</td>
<td>Date and Time of last modification to &lt;workitem&gt;</td>
</tr>
<tr>
<td>Warning</td>
<td>see below</td>
<td>O</td>
<td>See below</td>
</tr>
</tbody>
</table>

If the Workitem is already in the COMPLETED state, the response message shall include the following Warning header field:
Warning: 299 {+service}: The UPS is already in the requested state of COMPLETED.

12.3.5.3.3 Payload

A success response shall have an empty payload.

A failure response shall have a payload containing a Status Details document in an Appropriate Media Type.

12.3.6 Cancel Workitem Transaction

This transaction either:

1. Changes the Workitem state to CANCELLED, if the request is from the claiming user agent, but only if the Workitem meets all the Final State requirements specified in PS3.4 Table CC.2.5-3.

or

2. If the request is from a non-claiming user agent, the origin server will send a Cancel Event Report to the claiming user agent, requesting that the user agent cancel the Workitem.

Once the Workitem is in the Cancelled state, it shall not modified.

12.3.6.1 Request

This transaction uses the PUT method and has the following format:

PUT SP /{workitem}/cancel SP version CRLF
*(header-field CRLF)
CRLF

12.3.6.1.1 Header Fields

This request has no required header fields.

12.3.6.1.2 Payload

A payload is not required, but the payload may contain a Status Details document with the following attributes:

1. Reason for Cancellation and/or


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

12.3.6.2 Behavior

If the user agent has claimed the target Workitem, the origin server shall change the state of the target Workitem from In-Progress to CANCELED as shown in Figure CC.1.1-1 in PS3.4, and shall process the request as described by the SCP behavior in Section CC.2.2.3 in PS3.4, but only if the Workitem meets the Final State requirements specified in PS3.4 Table CC.2.5-3.

If the user agent does not hold the claim for the target Workitem... [TODO: finish]

12.3.6.3 Response

version SP status-code SP reason-phrase CRLF
*(header-field CRLF)
CRLF

12.3.6.3.1 Status Codes

If the user agent performing this transaction has claimed the Workitem, and the target URI of the request is a <claimed-workitem> URI, then a success response, where the Workitem is in the CANCELLED state, shall have a status code of 200 (OK).

If the user agent performing this transaction has NOT claimed the Workitem, and the target URI is not a <claimed-workitem> URI, then a success response will contain a status code of 202 (Accepted). This status code means that
the Request was accepted, not that the Workitem has been canceled. 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.

If the Workitem is already in the COMPLETED state the response status code will be 304 (Not Modified), and shall
have no payload.

Otherwise, a failure response will contain an appropriate status code from Table 6.X.Y.

12.3.6.3.2 Header Fields

Table X.Y-Z: Required and Optional Header Fields

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warning</td>
<td>see below</td>
<td>O</td>
<td>See below</td>
</tr>
</tbody>
</table>

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

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

12.3.6.4 Payload

The response has no payload.

12.3.7 Search Worklist Transaction

This transaction searches a Worklist for Workitems that match the specified Query Parameters, and returns the
specified list of matching Workitems. Each Workitem in the returned list includes any return attributes specified in the
request.

12.3.7.1 Request

The request has the following format:

```
GET SP /?$match* {&include} {&fuzzy} {&offset} {&limit} SP version CRLF
Accept: media-types CRLF
[Cache-Control: "no-cache" CRLF]
*(header-field CRLF)
```

12.3.7.1.1 Query Parameters

Each Query Parameters has one of the formats in the following table:

Table X.Y-Z: Search Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>match</td>
<td>; An ampersand separated list of &lt;attribute-name&gt; = &lt;values&gt; ; See PS3.3 Section B.26.2</td>
</tr>
<tr>
<td>include</td>
<td>= #attribute / &quot;all&quot;</td>
</tr>
<tr>
<td>fuzzy</td>
<td>= true / false</td>
</tr>
<tr>
<td>limit</td>
<td>= uint ; Maximum number of results</td>
</tr>
<tr>
<td>offset</td>
<td>= uint ; Number of skipped results</td>
</tr>
</tbody>
</table>

The grammar for Query Parameters is defined in Section 6.X.Y and Annex X.

12.3.7.1.2 Header Fields

Table X.Y-Z: Request Header Fields

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>&lt;media-types&gt;</td>
<td>M</td>
<td>List of one or more media types</td>
</tr>
<tr>
<td>Cache-Control</td>
<td>&quot;no-cache&quot;</td>
<td>O</td>
<td>If included, specifies that search results</td>
</tr>
</tbody>
</table>
See also Section 6.5.

12.3.7.1.3 Payload
The request payload is empty.

12.3.7.2 Behavior
The origin server shall perform a search according the requirements for the RS Storage Service Search transaction. See Section 10.6.3.

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

See Section 10.6.3.1 for matching behavior.

For each matching Workitem, the origin server shall return:

- All Unified Procedure Step Instance Attributes in PS3.4 Table CC.2.5-3 with a Return Key value of 1 and 2.
- All Unified Procedure Step Instance Attributes in PS3.4 Table CC.2.5-3 with a Return Key value of 1C for which the conditional requirements are met.
- All other Unified Procedure Step Instance Attributes passed as {attribute} parameter names, which are supported by the origin server as matching or return attributes.
- All other Unified Procedure Step Instance Attributes passed as "include" parameter values that are supported by the origin server as return attributes.

12.3.7.3 Response
The response shall have the following format:

```
version SP status-code SP reason-phrase CRLF
Content-Type: media-type CRLF
[ETag: entity-tag CRLF]
[Last-Modified: http-date CRLF]
*(header-field CRLF)
CRLF
search-results / status-details
```

Where

- `search-results` is a Search Results document.

12.3.7.3.1 Status Codes
A success response will contain a status code of 200 (OK).

A failure response will contain an appropriate status code from table X.Y.A.

12.3.7.3.2 Header Fields

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Value</th>
<th>Usage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Type</td>
<td>media-type</td>
<td>M</td>
<td>The media-type of the payload</td>
</tr>
<tr>
<td>ETag</td>
<td>entity-tag</td>
<td>O</td>
<td>The entity-tag of search-results</td>
</tr>
<tr>
<td>Last-Modified</td>
<td>HTTP-date</td>
<td>O</td>
<td>Date and time of last modification to search results</td>
</tr>
<tr>
<td>Warning</td>
<td>text</td>
<td>O</td>
<td>A warning message</td>
</tr>
</tbody>
</table>
12.3.7.3.3 Payload
The response payload contains the search results in the Selected Media Type. If there are no matching results the payload will be empty.

12.3.7.4 Media Types
The origin server shall support all media types allowed in the request. The media types supported by this transaction are:

- multipart/related; type="application/dicom+xml"; boundary={boundary}
  Specifies that the payload is a multipart message body where each part is a DICOM Native Model element containing the attributes for one matching Workitem. See PS3.19 Section A.1.

- application/dicom+json, application/json
  Specifies that the payload is a JSON array containing one property for each matching Workitem, which in turn contains sub-properties describing the matching attributes for that Workitem. See Section F.2.

12.4 Worklist Notification Sub-Service
[TODO: This will be updated when CP_jfp18_generalized_nado is final]

The user agent shall open a notification channel before it subscribes to any resources. Once that channel is opened, the user agent may create subscriptions to either 1) the entire Worklist or 2) individual Workitems.

The Worklist Service supports Deletion Locks for both the Worklist and its Workitem resources. An origin server shall not delete a resource until all Deletion Locks for that resource have been released. A user agent can release a Deletion Lock either by creating a new subscription to the same resource without a Deletion Lock, or by Unsubscribing from the resource.

The Worklist is a Collection resource. So, when a user agent subscribes to the Worklist, a Collection Subscription is created and a filter may be supplied.

When an event related to a subscription occurs, the origin server shall send a Notification containing an Event Report to the Subscriber over the Notification Channel.

If a user agent creates a subscription to the Worklist then it will receive notifications of events for all Workitems in the Worklist. When a user agent subscribes to a Worklist, it can specify a filter for the event notifications. The origin server will only send notifications for events that satisfy the filter.

Upon receipt of the Subscribe or Unsubscribe request, the origin server shall update the appropriate subscription state as described in Table CC.2.3-2 in PS3.4.

12.4.1 Resources
The Worklist Service supports the resources in Table X.Y-Z.

<table>
<thead>
<tr>
<th>Resource</th>
<th>URI Template</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worklist</td>
<td>/worklist</td>
<td>Receive event notifications for all, or a filtered subset, of the Workitems in the Worklist.</td>
</tr>
<tr>
<td>Workitem</td>
<td>/worklist/(workitem)</td>
<td>Receive event notifications for a single Workitem.</td>
</tr>
</tbody>
</table>

--- Draft ---
12.4.2 Open Notification Channel Transaction

This transaction opens a Notification Channel between the user agent and the origin server. The Notification Channel must be opened before the user agent can create any subscriptions.

This transaction is described in detail in Section X.Y.

12.4.3 Subscribe Transaction

This transaction creates a subscription to a Worklist or Workitem resource. All subscribers to a resource will receive notifications containing event reports for any future events associated with the resource.

The Worklist resource is a collection of Workitems. When a user agent subscribes to a Worklist the origin server creates subscriptions to the, possibly filtered, set of current and future Workitems contained in the Worklist.

If the user agent provides a filter parameter, then only Workitems

Deletion Locks

12.4.3.1 Request

The request shall be formatted as follows:

```
POST SP /worklist/subscriptions{/workitem} {?retain} {&filter} SP version CRLF
*(header-field CRLF)
CRLF
```

Where,

```
workitem = uid
```

Specifies an optional Workitem, if it is not present then the subscription will be global, that is, the user agent shall have subscriptions to all current and future items in the Worklist.

```
retain = true / false
```

If present, the subscription shall be created with a Deletion Lock.

```
filter = 1#(attribute "=" value)
```

If present, specifies the key/value pairs describing the filter parameters. Each (attribute) shall refer to an attribute of the Unified Procedure Step IOD (see PS3.3, Section B.26.2). See Section 6.7.1.1 for (attribute) and (value) encoding rules.

12.4.3.1.1 Header Fields

[TODO: add table]

See Section 6.3.1.3

12.4.3.1.2 Payload

The request shall have no payload.

12.4.3.2 Behavior

The origin server shall create a subscription to the target resource for the user agent.

The origin server shall support the management of subscriptions as specified by the SCP behavior in PS3.4, Section CC.2.3.3.

Upon receipt of the Subscribe request, the origin server shall create Subscriptions for the user agent to all existing Workitems that pass through the provided Filter, if any, and shall ensure that any new Workitems created by the origin server, which pass the Filter, if any, will have Subscriptions for the user agent.

It will also attempt to update the Global Subscription State, Filtered Global Subscription and/or Workitem Subscription State 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 the appropriate response.
12.4.3.3  **Response**

The response shall have the following format:

```
version SP status-code SP reason-phrase CRLF
Location: subscription CRLF
*(header-field CRLF)
CRLF
[status-details]
```

Where,

- `subscription` is the URI for the subscription resource. This shall include the WebSocket protocol (either WS or WSS) and may include a combination of authority and path.

12.4.3.3.1  **Status Codes**

A success response shall contain a status code of 201 (Created).

A failure response shall contain a status code from Table 6.X.Y.

12.4.3.3.2  **Header Fields**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>R/O</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>uri</td>
<td>R</td>
<td>An opaque URI-reference to the created subscription</td>
</tr>
</tbody>
</table>

Required 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.
```

Required if the request was rejected with a status code of 403, because Filtered Global Subscriptions are not supported, the response message shall include the following Warning header field:

```
Warning: 299 {+SERVICE}: The origin server does not support Global Subscription Filtering.
```

12.4.3.3.3  **Payload**

A success response payload shall be empty.

A failure response payload shall contain a Status Details document.

12.4.3.4  **Media Types**

An origin server shall support all media types allowed in the request.

The media types supported by this transaction are:

```
application/dicom+json
application/dicom+xml
application/json
```

12.4.4  **Unsubscribe Transaction**

The target subscription shall be deleted, and no future Event Notifications related to the Workitem will be sent to the subscription holder.

If the target resource is a Workitem subscription it shall be deleted. Once the subscription is deleted, no Event Notification related to the Workitem will be sent to the subscription holder.

If the target resource is the Worklist, the "suspend" parameter may be supplied in the target URI. If the value of the "suspend" parameter is "true" then all existing Workitems subscriptions that resulted from this subscription shall not
be deleted. If the "suspend" query parameter is either "false" or not present, the all current and future subscriptions shall be deleted.

### 12.4.4.1 Request

The request shall have the following syntax:

```
DELETE SP /subscribers{/workitem}{{?existing}} SP version CRLF
*(header-field CRLF)
```

```
DELETE SP /{subscription}{{?suspend}} SP version CRLF
*(header-field CRLF)
```

Where,

- `subscription = uri` - A URI reference to a subscription returned in a response to a Subscribe transaction.
- `suspend = "true" / "false"` - If present, indicates that the subscription should be suspended, i.e. current subscriptions should not be deleted, but no future subscriptions should be created.
- `existing = "keep" / "delete"` - If present, indicates that the subscription should be suspended, i.e. current subscriptions should not be deleted, but no future subscriptions should be created.

#### 12.4.4.1.1 Header Fields

See Section 6.X.Y.

#### 12.4.4.1.2 Payload

The request payload shall be empty.

#### 12.4.4.2 Behavior

If the "suspend" query parameter is "true" the origin server shall no longer automatically subscribe the user agent to newly-created Workitems; however, this does not delete existing subscriptions to Workitems. If the "suspend" query parameter is "false" or not present the origin server shall remove any of the user agent's existing subscriptions to the target resource.

The origin server shall conform to the behavior described in Section 12.1.8.4, "Behavior".

[TODO: rewrite for clarity?]

#### 12.4.4.3 Response

The response shall have the following format:

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

#### 12.4.4.3.1 Status Codes

A success response shall contain a status code of 200 (OK), indicating that the target subscription(s) have been suspended or deleted.
A failure response will contain an appropriate status code from Table 12.2-A.

12.4.3.2 Header Fields

[TODO: add table]
See Section 6.5.

12.4.3.3 Payload

A success response payload shall be empty.
A failure response payload shall contain a Status Details document.

12.4.5 Send Notification

This transaction sends Event Reports from the origin server to a user agent over an established Notification Connection. See Section 9.X.Y.

12.4.5.1 Request

The request shall use the WebSocket Data Frame transmission protocol. See 9.X.Y.

12.4.5.1.1 Payload

The Event Report shall contain all mandatory attributes described by the Conformance Statement and Capabilities document of the Defining Service for the event type.

12.4.5.1.1.1 Event Report Format

Events Reports are encoded as WebSocket Data Frames. The Data Frame Type is specified by the media type of the Event Report.

The Event Report shall contain the attributes in Table 11.4-1.

Table 11.4-1: Event Report Attributes

<table>
<thead>
<tr>
<th>Tag</th>
<th>Attribute Name</th>
<th>VR</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0000,0002)</td>
<td>Affected SOP Class UID</td>
<td>UI</td>
</tr>
<tr>
<td>(0000,0100)</td>
<td>Command Field</td>
<td>US</td>
</tr>
<tr>
<td>(0000,0110)</td>
<td>Message ID</td>
<td>US</td>
</tr>
<tr>
<td>(0000,1000)</td>
<td>Affected SOP Instance UID</td>
<td>UI</td>
</tr>
<tr>
<td>(0000,1001)</td>
<td>Requested SOP Instance UID</td>
<td>UI</td>
</tr>
<tr>
<td>(xxxx,yyyy)</td>
<td>Document Type</td>
<td>UR</td>
</tr>
<tr>
<td>(0074,1238)</td>
<td>Reason for Cancellation</td>
<td>??</td>
</tr>
<tr>
<td>(0074,4041)</td>
<td>Input Readiness State</td>
<td>??</td>
</tr>
</tbody>
</table>

However, the Defining Service may extend the Event Report with additional attributes.

The following is an example "application/json" WebSocket payload:

```json
{
"00000002": [ "1.2.840.10008.5.1.4.34.6.4" ],
"00000100": [ 256 ],
"00000110": [ 23 ],
"00001000": [ "1.2.840.10008.5.1.4.34.6.4.2.3.44.22231" ],
"00001001": [ 1 ],
"00741238": [ "SCHEDULED" ],
"00744041": [ "READY" ]
}
```
12.4.5.2  Behavior
Each service shall define the scenarios in which an origin server sends Event Reports to a subscriber, as well as the content of the Event Report messages.

12.4.5.3  Response
The user agent shall send a response containing a success or failure acknowledgement. See 9.X.Y.
An acknowledgement is encoded as a WebSocket data frame with an opcode of "%x1" (text).
The frame payload data shall be a US-ASCII string "success" for a success acknowledgement (ACK), or "failure" for a failure acknowledgement (NAK).

12.5  Conformance
An implementation supporting the RS UPS service shall declare it in its Conformance Statement, and whether it plays the role of origin server or user agent, or both.
An implementation playing the role of origin server shall declare in its Conformance Statement...

Subscriptions / global / filtered

notifications