DICOM Change Proposal

STATUS	Assigned
Date of Last Update	2025/05/18
Person Assigned	steven.nichols@gehealthcare.com
Submitter Name	Jeroen Medema, jeroen.medema@philips.com
Submission Date	2025/03/17

Change Number CP-2524
Log Summary: Update references to RFC 9110–9112 and related terminology
Name of Standard PS3.18
Rationale for Change: PS3.18 references several IETF RFCs in the HTTP/1.1 (RFC 7230–7235) series that have been obsoleted by RFC 9110 (HTTP Semantics), RFC 9111 (HTTP Caching), and RFC 9112 (HTTP/1.1). This CP updates references to the obsoleted RFCs. RFC 9110 is the normative specification for HTTP semantics and obsoletes RFC 7230–7235. RFC 9112 is normative for HTTP/1.1 message syntax and wire format. RFC 9111 defines HTTP caching semantics and is normative only where caching behavior is explicitly specified or relied upon. References to proactive content negotiation in this CP reflect terminology alignment with RFC 9110 and do not introduce new requirements, as such negotiation behavior was already implicit in DICOMweb through use of the Accept Header Field. RFC 9110 deprecates the Accept-Charset Header Field. However, PS3.18 currently defines Accept-Charset as one source of Acceptable Character Sets and uses it in the Selected Character Set algorithm (PS3.18, Sections 8.8.4 and 8.8.5). Removing support for Accept-Charset would therefore change existing behavior and may break user agents that rely on it. This CP retains Accept-Charset and adds a note identifying the RFC 9110 deprecation. An earlier version of this CP proposed updating RFC reference URLs in PS3.18 that use the legacy https://tools.ietf.org/ host to use canonical RFC Editor URLs at https://www.rfc-editor.org/. Since this URL update is editorial and applies across multiple Parts of the DICOM Standard, it will be handled separately as an editorial change.
Change Wording:

Modify Section 2.2 Internet Engineering Task Force (IETF) and Internet Assigned Names Authority (IANA) as indicated below:

…

[RFC7159] IETF. March 2014. *The JavaScript Object Notation (JSON) Data Interchange Format.* http://tools.ietf.org/html/rfc7159.

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

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

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

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

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

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

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

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

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

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

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

[RFC9110] IETF. HTTP Semantics. http://tools.ietf.org/html/rfc9110/.

[RFC9111] IETF. HTTP Caching. http://tools.ietf.org/html/rfc9111/.

[RFC9112] IETF. HTTP/1.1. http://tools.ietf.org/html/rfc9112/.

Modify Section 3.8 HyperText Transfer Protocol (HTTP/HTTPS) Definitions, as indicated below:

3.8 HyperText Transfer Protocol (HTTP/HTTPS) Definitions

This Part of the Standard makes use of the following terms defined in [RFC~~**7230**~~**9110**] ~~**Section 2.1 Client/Server Messaging**~~:

HTTP See [RFC~~**7230**~~**9110**].

HTTPS See [RFC~~**7230**~~**9110**].

origin server See [RFC~~**7230**~~**9110**].

user agent See [RFC~~**7230**~~**9110**].

Modify Section 5.1 Message Syntax, as indicated below:

5.1 Message Syntax

The syntax of the request and response messages for transactions are defined using the ABNF Grammar used in [RFC~~**7230**~~**9112**], which is based on the ABNF defined in [RFC5234]. This Part of the Standard also uses the ABNF extensions in [RFC7405], which defines '%s' prefix for denoting case sensitive strings.

…

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~~**7231**~~**9110**] Section **2.**1.~~**2**~~. For example:

…

5.1.3 List Rule('#')

The ABNF has been extended with the List Rule, which is used to define comma-separated lists. It does not allow empty lists, empty list elements, or the legacy list rules defined in [RFC~~7230~~9110] Section ~~7~~5.6.1.

…

Modify Section 5.3 Request and Response Header Field Tables, as indicated below:

…

The Name column contains the name of the HTTP Header Field as defined in [RFC9110~~7230, RFC7231~~].

…

Modify Section 8.1 Transactions, as indicated below:

8.1 Transactions

Each Transaction is composed of a request message and a response message, sometimes referred to as a request/response pair. When used in this Part of the Standard the term "request" means "request message", and "response" means "response message", unless clearly stated otherwise. Figure 8.1-1 is an interaction diagram that shows the message flow of a Transaction. **DICOMweb uses HTTP as defined in [RFC9110] and uses proactive content negotiation as defined in that specification. A user agent expresses representation preferences of the response payload in the request and the origin server selects the representation to be used in the response.** ~~**When it receives the request, the origin server processes it and returns a response.**~~

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

The response includes a status code, a reason phrase, Header Fields, and might also include a payload.

…

8.1.1 Request Message Syntax

…

Note

The method, SP, version, CRLF, header-field, and payload are all HTTP productions from [RFC~~7230~~9110]~~,~~ and [RFC~~7231~~9112]. The definitions are reproduced here for convenience.

8.1.1.1 Method

The request method is one of the HTTP methods, such as CONNECT, DELETE, GET, HEAD, OPTIONS, POST, PUT. See [RFC~~7230~~9110] Section ~~4~~9.3.

…

8.1.2 Response Message Syntax

…

status-code = 3DIGIT	A three-digit code specifying the status of the response.
reason-phrase = *(HTAB / SP / VCHAR)	A human readable phrase that corresponds to the status. An implementation may define its own reason phrases. The reason-phrase syntax is slightly modified from that in

[RFC~~7230~~9112]; this Part of the Standard does not allow obsolete text (obs-text) in the reason-phrase.

Note

The status-code production is from [RFC~~7230~~9112].

Modify Section 8.4 Header Fields, as indicated below:

8.4.1 Content Negotiation Header Fields

…

Content Negotiation Header Fields in requests allow the user agent to specify acceptable representations for the response. Table 8.4.1-1 lists the Content Negotiation Header Fields. The values in these fields apply to any content in the response, including representations of the Target Resource, representations of error or processing status, and potentially even the miscellaneous text strings that might appear within the HTTP protocol. See [RFC~~7231~~9110] Section 12~~5.3~~.

Table 8.4.1-1. Content Negotiation Header Fields

Name	Value	Usage	Description
Accept	1#media-range	M	All requests that expect to receive a response with a payload shall contain an Accept Header Field. See Section 8.4.1.1.
Accept-Charset	1#charset	O	The Accept-Charset Header Field may be sent by a user agent to indicate what charsets are acceptable in response content. See [RFC7231] Section 5.3.3.
Accept-Encoding	1#encoding	O	The Accept-Encoding Header Field may be used to indicate the content-codings (see [RFC~~7231~~9110] Section 8.4~~3.1.2.1~~) acceptable in the response. See [RFC~~7231~~9110] Section 12.5.3~~5.3.4~~.
Accept-Language	1#language	O	The Accept-Language Header Field may be used by user agents to indicate the set of natural languages that are preferred in the response. See [RFC~~7231~~9110] Section 12.5.4~~5.3.5~~.

Note

The Accept-Charset Header Field is deprecated by [RFC9110] Section 12.5.2. Its use in this Part of the Standard is retained for historical reasons, primarily for rendered textual representations whose character set is not otherwise determined by the Selected Media Type.

8.4.1.1 Accept

…

Most requests have an Accept Header Field that contains a comma-separated list of one or more media ranges. A media-range extends media-type with wildcards (*/* or type/*) and parameters that are not defined for media-types. See [RFC~~7231~~9110] Section 12.5.1~~5.3.2~~ for details.

…

This weight is often referred to as "quality value" or "qvalue". See [RFC~~7231~~9110] Section 12.4.2~~5.3.1~~.

…

8.4.1.1.1 Charset Media Type Parameter

Many Media Types, especially text/* types, define a "charset" parameter that specifies the character set for the representation. See [RFC~~7231~~9110] Section **8.3.2~~3.1.1.2~~**.

…

8.4.2 Content Representation Header Fields

…

Table 8.4.2-1. Content Representation Header Fields

Name	Value	Usage	Requirement
Content-Type	media-type	C	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 [RFC~~7231~~9110] Section **8.**3.~~1.1.5~~.
Content-Encoding	encoding	C	Specifies any content encodings applied to the representation (beyond those inherent in the Media Type), and thus what decoding to apply to obtain a representation in the Media Type specified by the Content-Type. See [RFC~~7231~~9110] Section **8.4~~3.1.2.2~~**. Content-Encoding allows compression, encryption, and/or authentication of representations. Shall be present if a content encoding has been applied to the representation in the payload.
Content-Language	language	O	Specifies the natural language(s) of the intended audience used in representation. See [RFC~~7231~~9110] Section **8.5~~3.1.3.2~~**.
Content-Location	url	C	Contains a URL that references the specific resource corresponding to the representation in the payload. **See [RFC9110] Section 8.7**. Shall be present if the payload contains a representation of a resource.

8.4.3 Payload Header Fields

…

Table 8.4.3-1. Payload Header Fields

Name	Value	Usage	Description
Content-Length	uint	C	Specifies the decimal number of octets in the payload. **See [RFC9110] Section 8.6**. If the response message has a payload and does not have a Transfer-Encoding Header Field, it shall have a Content-Length Header Field specifying the length in octets (bytes) of the payload. Shall not be present if the message has a Transfer-Encoding Header Field. Shall be present otherwise, even is the size of the payload is zero.
Content-Range	range	C	Specifies the range of a partial representation contained in a payload. See [RFC~~7233~~9110] Section **14.4~~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.

Name	Value	Usage	Description
Transfer-Encoding	encoding	C	See [RFC~~7230~~9112] Section **6.1~~3.3.1~~**. Shall be present if transfer-encodings have been applied to the payload.

Modify Section 8.5 Status Codes, as indicated below:

Each response message contains a status-code.

The most common HTTP status codes used are listed in Table 8.5-1 Most of these codes are described in detail in [RFC~~7231~~9110]. IANA maintains the HTTP Status Code Registry [IANA HTTP Status Code Registry], which contains a complete list of registered status codes.

Table 8.5-1. Status Code Meaning

Status	Code	Meaning
Success	The 2xx (Successful) class of status code indicates that the client's request was successfully received, understood, and accepted.	
	200 (OK)	All Target Resource representations are contained in the payload. See [RFC~~7231~~9110] Section **15~~6~~**.3.1.
	201 (Created)	The request has been fulfilled and has resulted in one or more new resources being created. See [RFC~~7231~~9110] Section **15~~6~~**.3.2.
	202 (Accepted)	The request has been accepted for processing, but the processing has not been completed. The payload of this response should contain a Status Report. [RFC~~7231~~9110] Section **15~~6~~**.3.3. The user agent may be able to inspect relevant resources to determine the status at some later time.
	203 (Non-Authoritative Information)	The request was successful, but the enclosed payload has been modified from that of the origin server's 200 (OK) response by a transforming proxy. See **~~[RFC7231] Section 5.7.2 and~~** [RFC~~7230~~9110] **~~[RFC7231]~~** Section **15~~6~~**.3.4.
	204 (No-Content)	The server has successfully fulfilled the request and there is no additional content to send in the response payload body. This should be the response when content is successfully uploaded, and the response has no payload. For example, this status code is used in the response to a Conditional Retrieve request), when the Target Resource has not been modified. See [RFC~~7231~~9110] Section **15~~6~~**.3.5.
	205 (Reset Content)	The server has fulfilled the request and desires that the user agent reset the "document view", which caused the request to be sent, to its original state as received from the origin server. **See [RFC9110] Section 15.3.6.**
Redirection	The 3xx (Redirection) class of status code indicates that further action needs to be taken by the user agent to fulfill the request.	
	301 (Moved Permanently)	The origin server has assigned the Target Resource to a new permanent URI, indicated in a Location Header Field. This status is typically needed when the resource has been moved from one service to another, for example during a migration. **See [RFC9110] Section 15.4.2.**

Status	Code	Meaning
	303 (See Other)	The origin the server is redirecting the user agent to a different resource, as indicated by a URI in the Location Header Field, which will provide a response to the original request. **See [RFC9110] Section 15.4.4.**
	304 (Not Modified)	The origin server has received a conditional GET or HEAD request that would have resulted in a 200 (OK) response if it were not for the fact that the condition evaluated to false. **See [RFC9110] Section 15.4.5.**
Client Error	The 4xx (Client Error) class of status code indicates that the user agent has erred. For all these error codes,the origin server should return a payload containing an explanation of the error situation, and whether it is a temporary or permanent condition, except when responding to a HEAD request.	
	400 (Bad Request)	The server cannot or will not process the request due to something that is perceived to be a client error (e.g., malformed request syntax, invalid request â€¦). **See [RFC9110] Section 15.5.1.**
	401 (Unauthorized)	The request has not been fulfilled because it lacks valid authentication credentials for the service or Target Resource. The server generating a 401 response shall send a WWW-Authenticate Header Field ([RFC~~7235~~**9110**] Section **11.6.1**~~4.1~~) containing at least one challenge applicable to the server or Target Resource. **See [RFC9110] Section 15.5.2.**
	403 (Forbidden)	The origin server understood the request, but refused to authorize it (e.g., an authorized user with insufficient privileges). If authentication credentials were provided in the request, the server considers them insufficient to grant access. The origin server may respond with a 404 (Not Found) if not permitted to use this status code. **See [RFC9110] Section 15.5.4.**
	404 (Not Found)	The origin server did not find a representation for the Target Resource or is not willing to disclose that one exists. This might be a temporary condition. If the origin server knows that the resource has been deleted, the 410 (Gone) status code shall be returned rather than 404. **See [RFC9110] Section 15.5.5.**
	405 (Method Not Allowed)	The method in the request is known by the origin server but not supported by the target service or resource. The origin server shall include an Allow Header Field in a 405 response containing a list of the target service or resource's currently supported methods. **See [RFC9110] Section 15.5.6.**
	406 (Not Acceptable)	The Target Resource does not have a representation that would be acceptable to the user agent, per the Content Negotiation Header Fields in the request, and the server is unwilling to supply a default representation. The origin server should return a payload that lists the available Media Types and corresponding resource identifiers. **See [RFC9110] Section 15.5.7.**
	409 (Conflict)	The request could not be completed due to a conflict with the current state of the Target Resource. This code is used in situations where the user agent might be able to resolve the conflict and resubmit the request. The origin server should return a payload containing enough information for the user agent to recognize the source of the conflict. In the DICOM context, this code might indicate that the origin server was unable to store any Instances due to a conflict in the request (e.g., unsupported SOP Class or Instance mismatch). **See [RFC9110] Section 15.5.10.**
	410 (Gone)	Access to the Target Resource is no longer available at the origin server and this condition is likely to be permanent. If the origin server does not know, or has no facility to determine, whether the condition is permanent, the 404 (Not Found) status code should be used instead. **See [RFC9110] Section 15.5.11.**

Status	Code	Meaning
	411 (Length Required)	The origin server refuses to accept the request because the Content-Length Header Field was not specified. **See [RFC9110] Section 15.5.12.**
	413 (Payload Too Large)	The server is refusing to process the request because the request payload is larger than the server is willing or able to process. **See [RFC9110] Section 15.5.14.**
	414 (URI Too Long)	The server is refusing to service the request because the ~~**request-**~~target **URI ~~([RFC7230] Section 5.3)~~** is longer than the server is willing to interpret. **See [RFC9110] Section 15.5.15.**
	415 (Unsupported Media Type)	The origin server does not support the Content-Type in the request payload. This error typically occurs when the user agent is trying to create or update a resource. The origin server should return a payload that lists the available Media Types and corresponding resource identifiers. **See [RFC9110] Section 15.5.16.** Note This is different from 406 (Not Acceptable).
Server Error	The 5xx (Server Error) class of status code indicates that the server is aware that it has erred or is incapable of performing the requested method. For all these error codes, the server should send an explanation of the error situation, and whether it is a temporary or permanent condition, except when responding to a HEAD request.	
	500 (Internal Server Error)	The server encountered an unexpected condition that prevented it from fulfilling the request. **See [RFC9110] Section 15.6.1.**
	501 (Not Implemented)	The server does not support the functionality required to fulfill the request. In the DICOM context, this status code shall be used for SOP Class Not Supported errors. **See [RFC9110] Section 15.6.2.**
	503 (Service Unavailable)	The origin server is currently unable to handle the request due to a temporary overload or scheduled maintenance, which will likely be alleviated after some delay. **See [RFC9110] Section 15.6.4.**
	505 (HTTP Version Not Supported)	The origin server does not support, or refuses to support, the major version of HTTP that was used in the request message. **See [RFC9110] Section 15.6.6.**

Modify Section 8.6 Payloads, as indicated below:

8.6 Payloads

Both request and response messages may have message bodies. The message body (if any) of an HTTP message is used to carry the payload of the message. The message body is identical to the payload unless a content coding has been applied, as described in [RFC~~7230~~**9112**] Section **~~6~~3.~~3~~**.1. This Part of the Standard uses the term "payload" to denote the message body before any content coding has been applied to it.

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

Any message containing a payload shall have appropriate ~~**Content**~~ Representation [RFC~~7231~~**9110**] Section 3.**2**~~1~~ and Payload Header Fields [RFC~~7231~~**9110**] Section **6.4.1**~~3.3~~.

Note

[RFC9110] replaces the terms "payload" and "payload body" with "content". This Part of the Standard retains the term "payload" for historical reasons.

…

8.6.1.1 Single Part Payload

…

Single part payloads may be requested with an HTTP Range **Request** [RFC~~7233~~**9110**] **Section 14**~~**request**~~. If supported by the server, the response shall be according to the standard HTTP Range **response, also specified in** [RFC~~7233~~**9110**] **Section 14.4**~~**response**~~.

…

8.6.1.2 Multipart Payload

…

2. An HTTP Range ~~**[RFC7233] r**~~**R**equest **[RFC9110]** may be used with Multipart payloads, but the range applies to the entire response, including the multipart markers. In order for the response to be valid across requests, the ordering of items and the choice of multipart separator must remain the same.

…

Table 8.6.1-1. Multipart Header Fields

Name	Value	Usage	Description
Content-Type	media-type	M	
Content-Length	uint	C	Shall be present if the response payload does not have a transfer encoding
Content-Location	url	C	Shall be present if the response payload contains a representation of a resource. See [RFC~~7231~~**9110**] Section **8.7**~~3.1.4.2~~.
Location	url	C	See [RFC~~7231~~**9110**] Section **10.2.2**~~7.1.2~~.

See Section 8.7.1 ~~**and [RFC7231]**~~.

…

8.6.3 Status Report

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

Note

A Status Report might be returned using a Media Type that is not an Acceptable Media Type in the request. See [RFC9110], Section 12.5.1.

Modify Section 8.7 Media Types, as indicated below:

8.7 Media Types

…

Media Types are defined in [RFC~~7231~~9110] Section 8.3.1~~.1.1~~.

…

8.7.3.3.2 Compressed Bulkdata Media Types

…

Note

1. The resource on the origin server may have been encoded in the Deflated Explicit VR Little Endian (1.2.840.10008.1.2.1.99) Transfer Syntax. If so, the origin server may inflate it, and then convert it into an Acceptable Transfer Syntax. A Content-Encoding header field of 'deflate' cannot be used to transfer the deflated bytes unaltered since the required PS3.10 File Meta Information is not included in the deflated bytes, and the Content-Encoding applies to the entire multipart stream, not each part within it individually.

   The resource on the origin server may have been encoded in the Deflated Image Frame Compression (1.2.840.10008.1.2.8.1) Transfer Syntax. If so, the origin server may return the compressed bit stream if it is an Acceptable Transfer Syntax, or the origin server may inflate it, and then convert it into an Acceptable Transfer Syntax. Alternatively, if the user agent allowed a Content-Encoding header field of 'deflate', then the deflated bytes for a single part response may be transferred after adding a zlib container per [RFC~~7230~~9110] Section 8.4.1.2~~4.2.2~~, but the Transfer Syntax parameter in the response should be the Explicit VR Little Endian Transfer Syntax.

…

8.7.5 Acceptable Media Types

…

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 [RFC~~7231~~9110] Section 12.4.2~~5.3.1~~.

…

8.7.7 Accept Header Field

…

The Accept Header Field value shall be a comma-separated list of one or more media ranges acceptable in the response. See [RFC~~7231~~9110] Section 12.5.1~~5.3.2~~.

…

8.7.8.1 Selected Media Type

The Selected Media Type is the Media Type selected by the origin server for the representation in 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 [RFC~~7231~~9110] Section 12.4.2~~5.3.1~~.

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

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

The Selected Media Type is chosen as follows:

1. Identify the target's Resource Category

2. Select the representation with the highest priority supported Media Type for that category in the Accept Query Parameter.

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).

Note

1. If the Selected Media Type is the Explicit VR Little Endian and the pixel data is compressed and when uncompressed is of such length that it cannot contained in a Value Field, then the origin server will respond with a 406 (Not Acceptable), and the user agent may try again with a different set of Acceptable Media Types.
2. If transcoding to the Explicit VR Little Endian Transfer Syntax, a VR of UN may be needed for the encoding of Data Elements with explicit VR whose value length exceeds 65534 (2^{16}-2) (FFFEH, the largest even length unsigned 16-bit number) but which are defined to have a 16-bit explicit VR length field. See Section 6.2.2 “Unknown (UN) Value Representation” in PS3.5.
3. **[RFC9110] permits an origin server to disregard an Accept Header Field rather than return a 406 (Not Acceptable) response. This Part of the Standard specifies more specific media type selection requirements for resource representations.**

Add the following examples to B Examples (Informative), as indicated below:

B Examples (Informative)

…

B.x Retrieve an Instance with Any Acceptable Media Type

This example illustrates a request to retrieve an Instance in which the user agent indicates that any Media Type for the Instance Resource Category is acceptable by using the wildcard media range */*. Since the request does not express a preference for a more specific Media Type, the origin server returns the default Media Type for the Instance Resource Category, which is application/dicom.

```
GET /radiology/studies/1.2.250.1.59.40211.12345678.678910
  /series/1.2.250.1.59.40211.789001276.14556172.67789
  /instances/1.2.250.1.59.40211.2678810.87991027.899772.2 HTTP/1.1
Host: www.hospital-stmarco
Accept: */*

HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: multipart/related; type="application/dicom";
  boundary="5e3c723b-ad24-42f9-88e0-10d29d87115c"

--5e3c723b-ad24-42f9-88e0-10d29d87115c
Content-Type: application/dicom; transfer-syntax=1.2.840.10008.1.2.1
Content-Location: /radiology/studies/1.2.250.1.59.40211.12345678.678910
                  /series/1.2.250.1.59.40211.789001276.14556172.67789
                  /instances/1.2.250.1.59.40211.2678810.87991027.899772.2

<BINARY DICOM DATA in multipart boundaries>

--5e3c723b-ad24-42f9-88e0-10d29d87115c--
```

B.y Retrieve an Instance Preferring a Compressed Transfer Syntax

This example illustrates a request to retrieve an Instance in which the user agent prefers a JPEG Baseline Transfer Syntax representation, but also indicates that Explicit VR Little Endian is acceptable with lower priority. The origin server returns one of the explicitly acceptable Transfer Syntaxes identified in

the Accept Header Field. In this example, the preferred compressed Transfer Syntax is available and is therefore selected for the response.

```
GET /radiology/studies/1.2.250.1.59.40211.12345678.678910
  /series/1.2.250.1.59.40211.789001276.14556172.67789
  /instances/1.2.250.1.59.40211.2678810.87991027.899772.2 HTTP/1.1
Host: www.hospital-stmarco
Accept: multipart/related; type="application/dicom";
  transfer-syntax=1.2.840.10008.1.2.4.50; q=1.0,
  multipart/related; type="application/dicom";
  transfer-syntax=1.2.840.10008.1.2.1; q=0.1

HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: multipart/related; type="application/dicom";
boundary="b7e7f9d6-2d4b-4b8c-9e73-0eaa5b8f5d1a"

--b7e7f9d6-2d4b-4b8c-9e73-0eaa5b8f5d1a
Content-Type: application/dicom;
transfer-syntax=1.2.840.10008.1.2.4.50
Content-Location: /radiology/studies/1.2.250.1.59.40211.12345678.678910
                  /series/1.2.250.1.59.40211.789001276.14556172.67789
                  /instances/1.2.250.1.59.40211.2678810.87991027.899772.2

<BINARY DICOM DATA in multipart boundaries>

--b7e7f9d6-2d4b-4b8c-9e73-0eaa5b8f5d1a--
```

B.z Retrieve Frames Preferring a Compressed Bulk Data Media Type

This example illustrates a request to retrieve Frames in which the user agent indicates that any compressed image Media Type is acceptable. The wildcard media range image/* indicates that any supported compressed image Bulk Data Media Type is acceptable, but does not express a preference among the available compressed image encodings. The origin server returns JPEG compressed pixel data. If the origin server could not provide a supported compressed image representation, it might return another explicitly acceptable representation or return 406 (Not Acceptable).

```
GET /radiology/studies/1.2.250.1.59.40211.12345678.678910
  /series/1.2.250.1.59.40211.789001276.14556172.67789
  /instances/1.2.250.1.59.40211.2678810.87991027.899772.2
  /frames/1 HTTP/1.1
Host: www.hospital-stmarco
Accept: multipart/related; type="image/*"; q=1.0,
  multipart/related; type="application/octet-stream"; q=0.1

HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: multipart/related; type="image/jpeg";
boundary="a3f1c2d5-8e4b-47a9-b6c3-1f2e3d4c5b6a"

--a3f1c2d5-8e4b-47a9-b6c3-1f2e3d4c5b6a

Content-Type: image/jpeg;
transfer-syntax=1.2.840.10008.1.2.4.50
Content-Location: /radiology/studies/1.2.250.1.59.40211.12345678.678910
                  /series/1.2.250.1.59.40211.789001276.14556172.67789
                  /instances/1.2.250.1.59.40211.2678810.87991027.899772.2
                  /frames/1

<COMPRESSED PIXEL DATA in multipart boundaries>

--a3f1c2d5-8e4b-47a9-b6c3-1f2e3d4c5b6a--
```