Digital Imaging and Communications in Medicine (DICOM)

Supplement 193: RESTful Rendering

DICOM Standards Committee, Working Group 27: Web Technologies
1300 N. 17th Street Suite 900
Rosslyn, Virginia 22209 USA

VERSION: Final Text
Developed in accordance with work item 2013-08-B.

Table of Contents

1 Scope and Field of Application 4
2 Normative References 4
4 Terms and Definitions 4
5 Conventions 4
6 Data Communication Requirements 4
6.1.X Notification Sub-Service Abstract API 9
6.1.X.1 Overview 9
6.1.X1.1 Sub-Service 9
6.1.X.1.2 Subscriptions 9
6.1.X.1.3 Notifications and Event Reports 10
6.1.X.1.4 Unsubscribing 10
6..1.X.2 Resources 10
6.1.X.2.1 Typical Resource Templates 10
6.1.X.3 Service Notifications 11
6.1.X.4 Transactions 11
6.1.X.5.1 Open Notification Connection Transaction 11
6.1.X.5.2 Close Notification Connection Transaction 13
6.1.X.5.3 Subscribe Transaction 14
6.1.X.5.4 Unsubscribe Transaction 17
6.1.X.5.5 Notify 18
6.1.X.4 Media Types 20
6.1.X.4.1 "application/json" Media Type 20
6.9.7 UPS Notification Overview 20
6.9.8 Subscribe Transaction 21
6.9.8.1 Request 21
6.9.8.1.2 Query Parameters 21
6.9.8.1.3 Header Fields 21
6.9.8.1.4 Payload 21
6.9.8.2 Behavior 21
6.9.8.3 Response 21
6.9.8.3.1 Status Code 22
6.9.8.3.2 Response Header Fields 22
6.9.8.3.3 Response Message Body 22
6.9.9 Notify Transaction 22
6.9.9.1 Notification 22
6.9.9.1.1 UPS Event Reports 22
6.9.9.2 Behavior 25
6.9.9.3 Notification Acknowledgement 26
6.Y RS Studies Notifications 26
6.Y.1 Subscribe Transaction 26
6.Y.1.1 Request 26
6.Y.1.1.1 Target Resource 26
6.Y.1.1.2 Query Parameters 26
6.Y.1.1.3 Request Header Fields 27
6.Y.1.1.4 Request Payload 27
6.Y.1.2 Behavior 27
6.Y.1.3 Response 27
6.Y.1.3.1 Status Code 27
6.Y.1.3.2 Response Header Fields 27
6.Y.1.3.3 Response Message Body 27
6.Y.2 Notify Transaction 27

6.Y.2.1 Notification 27
6.Y.2.1.1 Instance Availability Event Report 27
6.Y.2.1.2 'application/json' Example Instance Notification Event Report 28
6.Y.2.1.3 Event Report Format 29
6.Y.2.2 Behavior 29
6.Y.2.3 Notification Acknowledgement 29

1 Scope and Field of Application

This supplement defines a generalized abstract API for a Notification Sub-system for Restful Services. The Abstract Notifications API defines Subscribers, Subscriptions, Events and Event Reports.

It also modifies the UPS API to conform to the Abstract API.

It add new Notifications API to the Studies Storage Service.

It is designed to allow a Restful Modality Worklist Service to be defined in parallel.

Security is beyond the scope of the RESTful services defined in this supplement. However generic Web security mechanisms are fully compatible.

2 Normative References

Insert the following in Section 2 at the appropriate place:

IETF RFC7405 Case Sensitive Strings in ABNF https://tools.ietf.org/html/rfc7405

4 Terms and Definitions

Insert the following in Section 4 at the appropriate place:

Claim

Claimer

Event Report

Notification

Subscriber

Subscription

5 Conventions

Insert the following in Section 5 at the appropriate place:

6 Conformance

Insert the following in Section 6 at the appropriate place:

7 RS Notifications

Use Cases

Studies Service Instance Availability Notification

Closed Issues

#	Description

1	Don't use the terms Collection or Set – just use resource
2	Distinguish between a Subscription and a Subscription Generator (auto-subscribe)
3	Each Notification Connection should be associated with a Subscriber UUID
4	Be careful to separate the notion of "Complete" vs "Ready to Do Something"

Open Issues

#	Description
1	Should a subscription be able to specify the events to which the subscribtion applies?
2	What should the queueing model for Notifications be: 1. In order delivery 2. If Subscriber is offline should the event queue be saved and for how long
3	Are subscriptions related to logging?
	Should we be supporting Notifications such as: Ready To Read Ready To ReRead Ready To Post-Process Ready to QC

Notes

#	Description
1	Look at modality worklist, MPPS, and print notifications
2	Add a section on use cases: For the abstract API For each concrete API Differentiate old use cases from new use cases
3	Difference between N-Create, N-Event….
4	Coordinate with IHE ITI RESTful DSUB whitepaper
5	
6	

Add a new Section 6.1.1:

Editor's Notes – Delete before Final Text:

Notes on the RS Abstract API Notifications and Subscriptions

1.
2.
3.

4.

I Major Changes from UPS Notifications

These are the major differences between the UPS definition and the Abstract Notification API. More details are contained in the sections below.

1. The Proxy interface is now the responsibility of the origin server.
2. The transactions have been simplified and modified to be easily uniform across services and to be more RESTful.

II Abstract API

Notifications and Subscriptions are defined and implemented in a standard way across all RS Services. The Abstract API also significantly reduces the documentation requirements of the Defining Service.

The following transactions are defined:

Transaction	Concrete API Defines
Open Notification Channel	Nothing
Close Notification Channel	
Subscribe	Resources, events, event reports
Create Subscription Generator	Resources, events, event reports, filters
Unsubscribe	Unsubscribe from a resource, i.e. deletes the Subscription
Notify	Notifies a Subscriber of an Event

III Concrete APIs

A Concrete API specifies the resources that may have subscriptions, and for each resource type it specifies:

- The types of subscriptions (instance or generative) that may be associated with them.
- The types of Events associated with that resource type, and their corresponding Event Reports.

IV Resource Types

There are two types of resources:

1. A resource that contains sub-resources, to which new resources may be added, and existing resources may be modified or deleted.
2. A resource that does not have sub-resources.

V Subscriptions

There are two types of Subscriptions:

- A Simple Subscription, which generates a Notification when an Event happens to the corresponding resource.
- A generative subscription, which generates Simple Subscriptions to new sub-resources when they are created.

V.1 Simple Subscriptions

A simple subscription specifies the resource and the events related to the resource that it wants to be notified about.

V.2 Generative Subscriptions

Generative Subscriptions combine UPS's notion of Global and Filtered Subscriptions.

- They may have Retention (Deletion) Locks

- They may have Filters.
- They no longer use the Well Known UIDs or AETitles in the URL.

VI Transactions

VI.1 Open Notification Connection Transaction

- Its definition is the same across all services.
- The Abstract API defines a concrete implementation of this transaction
- It is recommended that user agents use the same port for both the HTTP and WebSocket connections.
- Add Subscriber UUID

Notes

- This transaction was renamed from "OpenEventChannel", because it is a "Web Socket Connection" not "channel" and it is sending notifications about events not events.
- It eliminates the use of AETitles, as they are not needed. Proxy servers can implement a table from user agent credentials (which are much stronger than AETitles) to AETitles.
-

VI.2 Close Notification Connection Transaction

This is a new transaction that allows either the user agent of origin server to close the channel with a reason message.

This is important in order to allow for:

- Orderly shutdowns
- Garbage collection of resources especially by the origin server when a user agent closes a connection without intention of reopening it.
- Automatic reestablishment of lost connections by either user agent or origin server, for a better user experience

VI Subscribe Transaction

The Subscribe Transaction is the same across all services except for the resources that may be subscribed to.

- The name has been changed from "Create Subscription". It is a stronger verb.
- The target resource is now "/{resource}{/subscription}. If the <uid> is not present in the origin server will create a new unique <uid> for the new workitem. This adheres more closely to RESTful best practices.
- Global and Filtered Global Subscriptions are now just Collection subscriptions.
- The names of some terms have been changed to better reflect their meaning:
  - "Deletion Lock" -> "Retention Lock"
- Defines to types of Subscription resources: Collections and Instances
- Standard URI Templates for Collection resources and Instance Resources are defined.
  - They eliminate the use of Well Known UIDs. A Proxy server can still use them on the DIMSE side.
- Query parameter names have been changed to be simpler and better reflect their meaning:
  - "deletionlock" -> "retain"
  - {query} –> "filter=" 1#(attribute "=" value)
- The response header field "Content-Location" is replaced by "Location" as specified in HTTP.
- The Subscription reference returned in the Location header field is opaque; and must be used to Unsubscribe from the Subscription resource created by this transaction.

VII Unsubscribe Transaction

The Unsubscribe transaction is the same across all RS Services. Its target resource is the Subscription (either Collection or Instance) referenced by the URL returned in the Location header field of the Subscribe transaction.

It uses the "existing" query parameter, which shall only be present if the target resource is a Collection subscription, to specify whether to "keep" or "delete" existing Instance Subscriptions that where created by the target Collection Subscription.

- The transaction merges the Suspend Global Subscription and Delete Subscription transactions.
- Changes the target resource from a known URI Template to an opaque URL, and thereby eliminates the need for Well Known UIDs and AETitles.
- Uses DELETE method

VIII Notify Transaction

Sends a Notification message containing an Event Report from the origin server to the user agent (could be generalized to another origin server) over the Notification Channel. The Notification message has a Content Type header field specifying the media type of the Event Report or Status Details.

- When a Notification is received the receiver must send a success or failure acknowledgement message in response.
- Notifications and Acknowledgements have Content-Type header fields
- Failure Acknowledgements contain Status Details documents.
- Event Reports are specified by the Defining Service
- Status Reports are defined by the Abstract API and may be extended by the Defining Service.

Sequence Diagram

```
title Notifications


User Agent->+Origin Server: Open Connection 1\n(Subscriber UUID)
Origin Server->-Subscription DB: Subscriber UUID is on Connection 1
User Agent->+Origin Server: Subscribe Resource A\n(Subscriber UUID)
Origin Server->Subscription DB: Subscriber UUID interested in Resource A\n(Subscription X)
Origin Server->-User Agent: (URL Subscription X)
Origin Server->+Origin Server: <something happens to Resource A>
Origin Server->+Subscription DB: Who is interested in Resource A
Subscription DB->Origin Server: Subscriber UUID
Subscription DB->-Origin Server: Subscriber UUID is on Connection 1
Origin Server->Origin Server: <Oh good, it's open>
Origin Server->-User Agent: Notify A1
User Agent->+Origin Server: Unsubscribe \n(Subscription X)
Origin Server->Subscription DB:Delete Subscription X\n(Subscriber UUID no longer interested in Resource A)
Origin Server->-User Agent: OK
User Agent->Origin Server: Close Connection 1
```

End of Editors Notes

Insert the following in PS3.18, after Section 6.1.W

6.1.X Notification and Subscription Abstract API

This section defines the Abstract API for Notifications and Subscriptions that may be implemented by any RS service.

6.1.X.1 Overview

A Notification Sub-Service allows user agents to subscribe to be notified of events associated with resources managed by an origin server. The term Sub-Service is used because the API is always defined within another RS Service specification, referred to as the Defining Service.

6.1.X1.1 Sub-Service

The Defining Service specifies:

- The Sub-Service defines the resources of the Defining Service that may have Subscriptions
- The matching criteria for a Search transaction, if any
- The matching criteria for Subscription Filters
- The Event Name, Event Trigger, and Event Payload
- The Status Details documents

6.1.X.1.1 Notification Connection

If an RS service defines a Notification Sub-Service, and an origin server implements that service, then a user agent may open a Notification Connection with the origin server. It does that by using the Open Notification Connection Transaction.

Notification Connections are implemented using the WebSocket protocol.

In order for a user agent to create Subscriptions to the resources managed by an origin server, it must open a Notification Connection with it.

6.1.X.1.2 Subscriptions

Once the notification connection has been opened, the user agent may subscribe to a resource managed by an origin server, thus creating a Subscription.

A Subscription is a resource that references another resource. It represents a contract between the origin server creating the Subscription and the user agent requesting it. The contract states that when an event occurs on the subscribed to resource, the origin server will send a Notification to the Subscriber that owns the Subscription.

The Defining Service specifies the types of resources to which a user agent may subscribe.

A user agent that owns a Subscription to a resource is said to be a Subscriber to that resource. A resource may have zero of more subscriptions associated with it.

There are two types of Subscriptions: Collection Subscriptions and Instance Subscriptions.

~~It is not permitted to Subscribe to Subscriptions?.~~

6.1.X.1.2.1 Subscribing to Resources

Each Defining Service specified the types of resources that may have Subscriptions. A user agent subscribes to a resource using the Subscribe Transaction. See Section X.Y.

6.1.X.1.2.1.1 Retention Locks

Each Subscription may have a Retention Lock, which prevents the origin server from deleting the corresponding resource until all locks on that resource are released. Since a resource may have more

than one Subscription associated with it, it may also have more than one Retention Lock. The origin server should not delete the resource until all Retention Locks on that resource have been released.

6.1.X.1.2.2 Instance Subscriptions

An Instance Subscription is a Subscription to a single Instance. Instance Subscriptions cause Notifications to be sent to the Subscriber whenever the Instance has its state or contents modified.

6.1.X.1.2.2.1 Subscriptions Filters

[TODO]

6.1.X.1.2.3 Collection Subscription

A Collection Subscription can only be applied to a resource that contains a collection of Instances. A collection resource specifies that the user agent should receive notifications for all events which occur to any Instance of the collection.

When a Collection Subscription is created, it may have a filter associated with it. If a Collection Subscription has a filter, then when an event occurs to any instance in the collection, the filter is applied to the event, and if the event passes the filter a notification is sent to the user agent.

6.1.X.1.3 Notifications and Event Reports

A Notification is message containing an Event Report about a Subscription that is sent from an origin server to the Subscriber over the Subscriber's Notification Connection.

A Notification is triggered by an Event.

An Event is a change to the state or content of a resource.

An Event Report is a Dataset containing Attributes that describe the associated Event.

When an event related a resource occurs, the origin server sends a notification containing an Event Report to all Subscribers to that resource.

Each Subscriber may receive a stream of Notifications associated with each subscription. Each Notification message contains an Event Report.

6.1.X.1.3.1 Acknowledgements

An acknowledgement is a success or failure response to a Notification.

6.1.X.1.4 Unsubscribing

6.1.X.1.4.1 New versus Existing Subscriptions

6..1.X.2 Resources

Each service defines the resources to which user agents may subscribe.

There are two principal resource types Collections and Instances.

A collection resource contains a collection of sub-resources. Often collection resources contain trees of sub-resources. For example, a Study resource is a collection resource that contains Series resources, and each Series resource contains Instance resources.

An Instance resource contains no sub-resources. It is usually a leaf in a resource tree.

6.1.X.2.1 Typical Resource Templates

The resource used to create Subscriptions to a root collection is:

```
/{collection}/subscriptions
```

The resource used to create Subscriptions to an instance of a collection is:

```
/{collection}/subscriptions/{instance}
```

6.1.X.3 Service Notifications

Any RS service is itself a resource, and there are some standard event types for which notifications are defined for all services.

All services that define a Notification Sub-Service shall provide Subscriptions to the Defining Service itself. This Subscription will provide Notifications of events related to the origin server itself, rather than the resources it manages. The Defining Service shall specify notifications for the following server related events.

- Service shutting down
- Service load increasing?
- Service low on resources?
- Service problem?

The Defining Service or an implementation of the Defining Service may extend the set of available service related notifications. Any extensions should be documented in the Conformance Statement and in the response to the Retrieve Capabilities request.

6.1.X.4 Transactions

Any service that supports the Notification Subservice must support the following transactions:

Table 12.2-1: Notification Sub-System Transactions

Name	Method	Description
Open Notification Connection	GET	The user agent requests that the origin server create a Notification Connection between them.
Close Notification Connection	N/A	Either user agent or origin server can close a Notification Connection
Subscribe	POST	Creates a Subscription
Unsubscribe	DETETE	Deletes a Subscription
Notify	N/A	The origin server sends a Notification containing an Event Report to a user agent.

Note

'/' is the root path of the service.

6.1.X.5.1 Open Notification Connection Transaction

The Open Event Connection transaction is the same for all services. It opens a WebSocket connection to the origin server that the origin server will use to send a Notification to the user agent. The user agent shall open a Notification Connection before executing any of the other transactions specified by this sub-service.

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

See [RFC-6455] for details of the WebSocket protocol.

6.1.X.5.1.1 Request

The request shall have the following Syntax:

```
GET SP / SP version CRLF
Host: server CRLF
Upgrade: "websocket" CRLF
Connection: "Upgrade" CRLF
Origin: origin CRLF
```

```
Sec-WebSocket-Key: key CRLF
Sec-WebSocket-Protocol: protocol CRLF
Sec-WebSocket-Version: ws-version CRLF
*(<header-field> CRLF)
CRLF
```

Where,

/	is the Root Path of the origin server
server	the domain name or IP address of origin server
origin	see [RFC6454]
key	a 16 byte <octet-string> encoded in base64. It is recommended that a binary UUID be used.
protocols	one or more of “dicom”, “dicom+json”, “dicom+xml”.
ws-version	The version of the WebSocket protocol desired.

6.1.X.5.1.1.1 Taget Resource

The Target Resource is an origin server implementing a Defining Service, i.e., a service that has defined this sub-service.

6.1.X.5.1.1.2 Query Parameters

This transaction has no query parameters.

6.1.X.5.1.1.3 Request Header Fields

Table 6.1.X-Y: Request Header Fields

Name	Value	Usage	Notes
Host	<server>	R	
Upgrade	“websocket”	R	
Connection	“Upgrade”	R	
Origin	<origin>	R	
Sec-WebSocket-Key	<key>	R	
Sec-WebSocket- Protocol	"http"	R	
Sec-WebSocket-Version	<ws-version>	R	

6.1.X.5.1.1.4 Request Payload

Typically the request has no payload, but a payload may be specified by the Defining Service.

6.1.X.5.1.2 Behavior

The origin server opens and maintains a Notification Connection, using the WebSocket protocol, between the user agent and itself, and records that association. . The connection shall remain open, and shall be used by the origin server to send Notifications the user agent.

If the Notification Connection is lost at any point the user agent or origin server can re-establish it by repeating this transaction.

The state of a Notification Connection does not affect subscriptions. An origin server should queue notifications when the Notification Connection is down, but is not required to do so.

6.1.X.5.1.3 Response

The response shall have the following format:

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

Where

```
response-key [TODO: finish]
protocol
```

6.1.X.5.1.3.1 Status Codes

A success response will contain a status code of 101 (Switching Protocols), which indicates that the origin server has opened the connection.

A failure response will contain an appropriate failure status code.

Table X.Y contains the most common status codes for this transaction.

Table X.Y: Common Status Codes

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

6.1.X.5.1.3.2 Response Header Fields

Table 6.1.X-Y: Response Header Fields

Name	Value	Usage	Notes
Upgrade	"websocket"	R	
Connection	"Upgrade"	R	
Sec-WebSocket-Accept	<encode-key>	R	
Sec-WebSocket-Protocol	<protocol>	R	

6.1.X.5.1.3.3 Response Payload

A success response shall have no payload.

A failure response payload shall contain a Status Details document.

6.1.X.5.2 Close Notification Connection Transaction

This transaction shall use the WebSocket Data Frame transmission protocol, and may be initiated by either the user agent or origin server.

The Close Notification Connection Transaction shall be performed as specified in [RFC6455 < https://tools.ietf.org/html/rfc6455>].

Table 6.1.X-Y: Common Status Codes and Reason Messages for Close Notification

Status Code	Reason Message	
1000	Normal Close	
1001		
1002		
1003		

6.1.X.5.3 Subscribe Transaction

This transaction creates a Subscription resource to a target resource.

The origin server shall report future events associated with that resource to the user agent using the user agent's Notification Connection.

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

The user agent shall create a Notification Connection with the origin server prior to creating any Subscriptions. If the origin server does not have a Notification with the requesting user agent, it shall return a 400 (Bad Request), with a Status Details payload.

6.1.X.5.3.1 Request

This request uses the POST method and has the following syntax:

```
POST SP /{/resource}{?accept}{&retain}{&filter} SP version CRLF
Accept: media-types CRLF
*(header-field CRLF)
CRLF
```

6.1.X.5.3.1.1 Target Resource

The target resource has the following URI template:

```
/{/resource}
```

Where

/	The Service Root Path. See Section X.Y
{/resource}	The path from the Service Root to the target resource.

The target resource has the following syntax:

```
1*"/" resource "/subscriptions"
```

Where <resource> is a path to a resource in the Defining Service.

[if auto-subscribe the correct default?, or should it be a parameter to the Subscribe request?]

6.1.X.5.3.1.2 Query Parameters

The query parameters specified below are optional, but if any are supported, they should have the similar defined here.

```
{?accept} = "accept" "=" 1#media-type
```

The Accept query parameter specifies one or more media types that are acceptable in the Notifications stream associated with the subscription created by this transaction. If the "accept" query parameter is not present, the default media type for Notifications shall be used.

```
{?retain} = "retain" "=" "true" / "false"
```

If present with a value of "true", it indicates that the user agent is requesting a Retention Lock, which if granted, prevents the resource from being deleted. If the “retain” parameter is not present, its default value is "false", and no Retention Lock will be created.

The Retention Lock may be removed either by deleting the subscription or by creating a new subscription to the resource without the “retain” parameter.

More than one user agent can request a Retention Lock on the same resource. All Retention Locks associated with a resource must be released before the origin server is free to delete the resource.

For each resource, which the user agent has acquired a Retention Lock, the user agent shall remove the Retention Lock as soon as it has obtained any needed final state information for the instance.

```
{&filter} = "filter" "=" 1#(attribute "=" values)
```

If present, this parameter specifies the attribute/value pairs used to filter the notifications.

The “filter” parameter may only be present if the target resource is a collection resource. It specifies a comma-separated list of matching attributes/value pairs that are used as a filter to determine if a new instance should have a subscription. See Section 6.X.Y for attribute/value pair syntax.

If the new instance has the specified attribute/value pairs then the Filter is satisfied and the instance will have a Subscription.

The permissible attributes and values are specified by the Defining Service.

6.1.X.5.3.1.3 Request Header Fields

Table 6.1.X-Y: Common Header Fields for Subscribe Request

Name	Value	Usage	Notes
Accept	<media-type>	CM	The Acceptable media types of any payload that might be contained in the response.

Other required and optional header fields may be specified by the Defining Service.

6.1.X.5.3.1.4 Request Payload

Typically the request has no payload, but a payload may be specified by the Defining Service.

6.1.X.5.3.2 Behavior

The origin server shall create a subscription to the target resource for the user agent, and it shall support the management of subscriptions as specified by the Defining Service.

The origin server shall send Notifications for this Subscription over the Notification Connection create by the user agent. The exact Notifications sent shall be determined by the query parameters to this transaction and the events defined by the Defining Service.

The origin server shall honor any Retention Locks provided to subscriptions.

Upon receipt of a valid Subscribe request, the Origin-Server shall create the subscription(s) specified in the request and then return an appropriate response.

6.1.X.5.3.3 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,

505 `subscription` is an opaque URL that references the created subscription resource.

6.1.X.5.3.3.1 Status Codes

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

A failure response shall contain a status code from Table X.Y. Table CC.2.1.4.

Table X.Y: Common Status Codes for the Subscribe Transaction

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

510

Table CC.2.3.4: Status Codes for Subscribe

Status	Meaning	Code	
Success	The requested change of subscription state was performed	0000	201
Warning	Retention Lock not granted	B301	207?
Failure	Specified SOP Instance UID does not exist or is not a UPS Instance managed by this SCP	C307	404
	Receiving AE-TITLE is Unknown to this SCP	C308	401
	Refused: Specified action not appropriate for specified instance	C314	409
	Refused: SCP does not support Event Reports	C315	403

6.1.X.5.3.3.2 Response Header Fields

Table 6.1.X-Y: Common Header Fields for Subscribe Request

Name	Value	Usage	Notes
Location	<subscription>	M	An unique and opaque URL, defined by the origin server, that references the Subscription. Required If the request is successful,
Warning	<string>	CM	See below

515 If a Subscription was created but a requested Retention Lock was not granted. The Warning header field shall contain the following string:

```
Warning: 299 {+service}: Retention Lock not granted.
```

If the request contained a "filter" query parameter and the origin server does not support them, a 403 (Forbidden) response shall be returned and the Warning header field shall contain the following string:

```
Warning: 299 {+service}: Subscription Filters are not supported.
```

Other required and optional header fields may be specified by the Defining Service.

6.1.X.5.3.3.3 Response Payload

The success payload, if any, shall be specified by the Defining Service.

A warning or failure payload shall contain a Status Details document.

6.1.X.5.4 Unsubscribe Transaction

This transaction removes (deletes) a subscription from the target resource.

When a user agent unsubscribes from a resource, its Retention Locks on any resources associated with that subscription are released. The origin server shall not delete a resource until all Retention Locks on the resource have been released.

Upon receipt of the Unsubscribe request, the origin server shall update the appropriate Subscription state as described by the Defining Service.

6.1.X.5.4.1 Request

The request uses the DELETE method and has the following syntax:

```
DELETE SP {/subscription}{?existing} SP version CRLF
Accept: media-type CRLF
*(header-field CRLF)
CRLF
```

6.1.X.5.4.1.1 Target Resource

The target resource is a subscription:

```
{/subscription} = url
```

A <subscription> is an opaque URL returned in the Location header field of a successful Subscribe response.

6.1.X.5.4.1.2 Query Parameters

The Defining Service may implement the following parameters for this transaction:

```
{?existing} = "existing" "=: “keep” / “delete”
```

The “existing” parameter, is optional. It shall only be present if the target resource is a Collection Subscription. It shall not be present otherwise. The default value is "delete".

6.1.X.5.4.1.3 Request Header Fields

The Defining Service may specify additional Mandatory, Conditionally Mandatory or Optional header fields

Table 6.1.X-Y: Common Header Fields for Subscribe Request

Name	Value	Usage	Notes
Accept	1#media-type	CM	The Acceptable Media Types for a response

6.1.X.5.4.1.4 Request Payload

The request has no payload.

6.1.X.5.4.2 Behavior

The target Subscription shall be deleted and its Retention Lock shall be removed from the resource that it was subscribed to.

If the target Subscription is a Collection Subscription, and the value of the "existing" query parameter is “keep”, all existing Instance Subscriptions created by the target Collection Subscription are left in place, but no more shall be created; otherwise, is the "existing" query parameter is not present or its value is "delete", all existing Instance Subscriptions created by the target resource are deleted and no more shall be created.

6.1.X.5.4.3 Response

The response shall have the following format:

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

6.1.X.5.4.3.1 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 6.1.X-Z.

Table 6.1.X-Z: Common Status Codes for Unsubscribe

Status	Meaning	Code	
Success	The requested change of subscription state was performed	0000	200
~~Warning~~	~~Retention Lock not granted~~	~~B301~~	
Failure	Specified SOP Instance UID does not exist or is not a UPS Instance managed by this SCP	C307	404
	Receiving AE-TITLE is Unknown to this SCP	C308	401
	Refused: Specified action not appropriate for specified instance	C314	409
	~~Refused: SCP does not support Event Reports~~	~~C315~~	

6.1.X.5.4.3.2 Response Header Fields

The Defining Service may specify additional Mandatory, Conditionally Mandatory or Optional header fields

Table 6.1.X-Y: Common Header Fields for Subscribe Request

Name	Value	Usage	Notes
Accept	1#media-type	CM	The Acceptable Media Types for a failure response

6.1.X.5.4.3.3 Response Payload

The success payload, if any, shall be specified by the Defining Service.

A warning or failure payload shall contain a Status Details document.

6.1.X.5.5 Notify

This transaction is unlike most other Web Service transactions defined by PS3.18. It uses a Notification Connection, which is based on the WebSocket protocol, rather than HTTP. It is initiated by the origin server, which sends a Notification, containing an Event Report, to a user agent. The user agent responds with a success or failure acknowledgement message.

The format of the Notify transaction is the same for all Defining Services; however, the Event Report contained in the payload of the request shall be defined by the Defining Service.

6.1.X.5.5.1 Notification

Notification messages shall use the WebSocket Data Frame transmission protocol.

There are two types of Data Frames specified by the Opcode field:

- %x1 denotes a text frame
- %x2 denotes a binary frame

Notification messages have the following syntax:

```
frame-header SP media-type CRLF
event-report
```

Where

```
frame-header          = text-frame-header / binary-frame-header
text-frame-header     ; is a Data Frame with an Opcode of Text (%x1)
binary-frame-header   ; is a Data Frame with an Opcode of Binary (%x2)
media-type            ; is an HTTP media type
ack         = "success"
nack        = "failure"
event-report          ; is a required Event Report specified by the defining service (see Section X.Y)
```

The Data Frame Type shall be specified by the media type of the Event Report.

The Event Report content, and available media types shall be specified by the Defining Service.

6.1.X.5.5.1.1 Event Report Format

Events Reports are encoded as WebSocket Data Frames. There are two types of Data Frames specified by the Opcode field:

- %x1 denotes a text frame
- %x2 denotes a binary frame

The Data Frame Type is specified by the media type definition of the Event Report.

The Event Report shall contain the attributes in Table X.Y-Z:

Table X.Y-Z: Event Report Attributes

Tag	Attribute Name	VR
(0000,0002)	Affected SOP Class UID	UI
(0000,0100)	Command Field	US
(0000,0110)	Message ID	US
(0000,1000)	Affected SOP Instance UID	UI
(0000,1001)	Requested SOP Instance UID	UI
(xxxx,yyyy)	Document Type	UR
(0074,1238)	Reason for Cancellation	
(0074,4041)	Input Readiness State	

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
```

6.1.X.5.5.2 Behavior

Each Defining Service shall specify the resources and the related events or conditions that determine when a notification is sent to a subscriber.

6.1.X.5.5.3 Notification Acknowledgement

The user agent shall send a Data Frame Protocol response, using the Notification Connection containing a success (ACK) or failure (NACK) acknowledgement.

The response message will have the following format:

```
frame-header SP ack / nack SP media-type CRLF
[status-details]
```

Where

```
Frame-header          = text-frame-header / binary-frame-header
text-frame-header     ; is a Data Frame of type Text (%x1)
binary-frame-header   ; is a Data Frame of type Text (%x2)
media-type         ; is an HTTP media type
ack         = "success"
nack        = "failure"
[status-details]   ; is an optional Status Details document (see Section X.Y)
```

A success response may contain a Status Details document in the media type defined in the message

A failure response shall contain a Status Details document in the media type specified in the message.

6.1.X.4 Media Types

All implementations of this sub-service must support the “application/json” media type. Other required or recommended media types may be specified by the Defining Service.

6.1.X.4.1 "application/json" Media Type

"application/json" notification messages use Data Frames of type Text.

Replace PS3.18, Section 6.9 with the following:

6.9.7 UPS Notification Overview

The UPS Worklist Notification Sub-System conforms to the API described in Section 6.1.X.

The Worklist resource supports Collection Subscriptions. See Section 6.1.X.1.2.2.

The Workitem resource supports Instance Subscriptions. See Section 6.1.X.1.2.3.

The UPS Notifications system defines the transactions in Table X.Y, to conform to the definitions in the corresponding Section in the Table.

Table 6.9-X: Standard Notification Transactions

Transaction	Definition

Open Notification Connection	Section 6.1.X.5.1
Close Notification Connection	Section 6.1.X.5.2
Unsubscribe	Section 6.1.X.5.4
Notify	Section 6.1.X.5.5

6.9.8 Subscribe Transaction

This transaction creates a Subscription to a Worklist or Workitem. See Section 6.1.X.5.3.

6.9.8.1 Request

The request uses the POST method and has the following syntax:

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

6.9.8.1.1 Target Resource

The UPS Worklist Service defines two subscription resources:

```
/worklist/subscriptions
```

Which is used to create Collection Subscription for the Worklist, and

```
/studies/subscriptions {/workitem}
```

Which is used to create Instance Subscriptions to Workitems.

6.9.8.1.2 Query Parameters

The standard "retain" and "filter" query parameters are supported. See Section 6.1.X.5.3.1.2

6.9.8.1.3 Header Fields

See Section 6.1.X.5.3.1.3.

6.9.8.1.4 Payload

The request shall have no payload.

6.9.8.2 Behavior

The origin server shall create a subscription to the target resource, and return a URL reference to the Subscription in the Location header field of the response. See Section 6.1.X.5.3.2 for details.

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

Upon receipt of a valid Subscribe request, the Origin-Server shall create the subscription(s) specified in the request and then return an appropriate response.

If the target resource is "/worklist/subscriptions" the origin server shall create a Collection Subscription, as described in Section 6.1X.1.2.2 and PS3.4, Table CC.2.3-2. The Collection Subscription shall create an Instance Subscription for any new Workitems that passes the Filter.

6.9.8.3 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]
```

See Section 6.1.X.5.3.3.

6.9.8.3.1 Status Code

A success response shall have a status code of 201 (Created), and shall contain a "Location" header field with a URL referencing the created Subscription.

A failure response shall have an appropriate failure status code. See Section 6.1.X.5.3.3.1.

6.9.8.3.2 Response Header Fields

See Section 6.1.X.5.3.3.2.

6.9.8.3.3 Response Message Body

The response shall have no message body.

6.9.9 Notify Transaction

This transaction sends a UPS Event Report from the origin server to the user agent over a Notification Connection. See Section 6.1.X.5.5.

PS3.4, Section CC.2.4.3 describes the scenarios in which an origin server sends Event Reports to a subscriber and the content of the Event Report messages.

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

6.9.9.1 Notification

The Notification is a Web Socket Data Frame, sent over the Notification Connection. See Section 6.1.X.5.5.

6.9.9.1.1 UPS Event Reports

The Notification shall contain an Event Report in one of the formats defined in Tables 6.9-X through 6.9.Y.

Table 6.9-X: Event Report Types

Name	Event Type ID	Description
Workitem State Change	1	A user agent may receive State Change Report either due to a state change to the Workitem, or in response to initial subscription to the Workitem (possibly when the Workitem is initially created). See Section CC.2.3.3.
Cancel Workitem Request	2	If user agent with a Claimed Workitem (In-Progress) and an open Notification Connection may receive a Cancel Request Report if another user agent performs a Request Cancellation transaction. The claiming user agent may, at its own discretion, choose to cancel the Workitem as described above. See Section CC.2.4.3.
Workitem Progress	3	Each time the origin server updates the Procedure Step Progress (0074,1004), the Procedure Step Progress

		Description (0074,1006), or the contents of the Procedure Step Communications URI Sequence (0074,1008) for a UPS instance, it shall send a UPS Progress Report, with the current contents of the Progress Information Sequence (0074,1002), to subscribed user agents. See Section CC.2.4.3.
Service Status Change	4	Each time the origin server is restarted, it shall send a Status Change Event. The SCP, if it knows it is going down, may send an additional SCP Status Change Event before it is shut down. See Section CC.2.4.3.

720

Table 10.3-1. N-EVENT-REPORT-RQ Message Fields

Message Field	Tag	VR	VM	Description of Field
Command Group Length	(0000,0000)	UL	1	The even number of bytes from the end of the value field to the beginning of the next group.
Affected SOP Class UID	(0000,0002)	UI	1	SOP Class UID of the SOP Instance for which this event occurred.
Command Field	(0000,0100)	US	1	This field distinguishes the DIMSE-N notification conveyed by this Message. The value of this field shall be set to 0100H for the N-EVENT-REPORT-RQ Message.
Message ID	(0000,0110)	US	1	Implementation-specific value that distinguishes this Message from other Messages.
Command Data Set Type	(0000,0800)	US	1	This field indicates if a Data Set is present in the Message. This field shall be set to the value of 0101H if no Data Set is present; any other value indicates a Data Set is included in the Message.
Affected SOP Instance UID	(0000,1000)	UI	1	Contains the UID of the SOP Instance for which this event occurred.
Event Type ID	(0000,1002)	US	1	Values for this field are application-specific.
Event Information	(no tag)	-	-	Application-specific Data Set containing additional information related to the operation.

Note

1. Service Class Specifications contained in PS3.4 defines the values needed for the Event Type ID parameter.

725 2. Service Class Specifications contained in PS3.4 defines the Data Set needed for the Event Information parameter.

Table 6.9-X: UPS Event Report Message Header

Message Field	Tag	VR	VM	Value
Command Group Length	(0000,0000)	UL	1	The length of the message (an even number of bytes)
Affected SOP Class UID	(0000,0002)	UI	1	**"1.2.840.10008.5.1.4.34.6.4"**
Command Field	(0000,0100)	US	1	256 (0100H)

Message Field	Tag	VR	VM	Value
Message ID	(0000,0110)	US	1	Origin server controlled number
Command Data Set Type	(0000,0800)	US	1	257 (0101H)
Affected SOP Instance UID	(0000,1000)	UI	1	UID of the Workitem this report is abou.t
Event Type ID	(0000,1002)	US	1	1, 2, 3, or 4. See Table 6.9-X.
Event Information	(no tag)	-	-	Application-specific Data Set containing additional information related to the operation.

730

The Workitem State Change Event Report shall contain the attributes in Table 6.9.9-

Table 6.9-X: Workitem State Change Event Report (Event Type ID = 1)

Attribute Name	Tag	Usage
Affected SOP Instance UID	(0000,1000)	-/1
Input Readiness State	(0040,4041)	-/1
Procedure Step State	(0074,1000)	-/1
Procedure Step Discontinuation Reason Code Sequence	(0074,100e)	-/3
>Code Value	(0008,0100)	-/1
>Coding Scheme Designator	(0008,0102)	-/1
>Coding Scheme Version	(0008,0103)	-/1C Required if the value of Coding Scheme Designator (0008,0102) is not sufficient to identify the Code Value (0008,0100) unambiguously
>Code Meaning	(0008,0104)	-/1
Reason For Cancellation	(0074,1238)	-/3

The Cancel Requested Event Report shall contain the attributes in Table 6.9.9-

735

Table 6.9-X: Cancel Requested Event Report (Event Type ID = 2)

Attribute Name	Tag	Usage
Affected SOP Instance UID	(0000,1000)	-/1
Contact URI	(0074,100a)	-/1C Required if provided in the triggering N-ACTION
Contact Display Name	(0074,100c)	-/1C Required if provided in the triggering N-ACTION
Procedure Step Discontinuation Reason Code Sequence	(0074,100e)	-/1C Required if provided in the triggering N-ACTION
>Code Value	(0008,0100)	-/1
>Coding Scheme Designator	(0008,0102)	-/1

>Coding Scheme Version	(0008,0103)	-/1C Required if the value of Coding Scheme Designator (0008,0102) is not sufficient to identify the Code Value (0008,0100) unambiguously
>Code Meaning	(0008,0104)	-/1
Requesting AE	(0074,1236)	-/1
Reason For Cancellation	(0074,1238)	-/1C Required if provided in the triggering N-ACTION

The Workitem Progress Event Report shall contain the attributes in Table 6.9.9-

Table 6.9-X: Workitem Progress Event Report (Event Type ID = 3)

Attribute Name	Tag	Usage
Affected SOP Instance UID	(0000,1000)	-/1
Progress Information Sequence	(0074,1002)	-/1
>Procedure Step Progress	(0074,1004)	-/3
>Procedure Step Progress Description	(0074,1006)	-/3
>Procedure Step Communications URI Sequence	(0074,1008)	-/3
>>Contact URI	(0074,100a)	-/1
>>Contact Display Name	(0074,100c)	-/3

The Origin Server Status Change Event Report shall contain the attributes in Table 6.9.9-

Table 6.9-X: Origin Server Status Change Report (Event Type ID = 4)

Attribute Name	Tag	Usage
Affected SOP Instance UID	(0000,1000)	-/1
SCP Status	(0074,1242)	-/1
Subscription List Status	(0074,1244)	-/1
Unified Procedure Step List Status	(0074,1246)	-/1

However, a UPS 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": [ 257 ],
"00001000": [ "1.2.840.10008.5.1.4.34.6.4.2.3.44.22231" ],
"00001001": [ 1 ],
"00741238": [ "SCHEDULED" ],
"00744041": [ "READY" ]
} CRLF
```

6.9.9.2 Behavior

The user agent should take appropriate action when a Notification is received.

6.9.9.3 Notification Acknowledgement

The user agent shall send a success or failure Acknowledgement as specified in Section 6.1.X.5.5.3

Add PS3.18, Section 6.Y as follows:

6.Y RS Studies Notifications

The Studies Notification Sub-Service is used to send Instance Availability Event Reports from the origin server to the user agent. This sub-service corresponds to the Instance Availability Notification Service Class (see PS3.4, Annex R).

The Studies Notification System conforms to the API described in Section 6.1.X.

The Study resource supports Collection Subscriptions. See Section 6.1.X.1.2.2.

The Study Instnace resource supports Instance Subscriptions. See Section 6.1.X.1.2.3.

The Studies Notifications system defines the transactions in Table X.Y, to conform to the definitions in the corresponding Sections.

Table 6.9-X: Standard Notification Transactions

Transaction	Definition
Open Notification Connection	Section 6.1.X.5.1
Close Notification Connection	Section 6.1.X.5.2
Unsubscribe	Section 6.1.X.5.4
Notify	Section 6.1.X.5.5

6.Y.1 Subscribe Transaction

This transaction creates a Subscription to a Study resource.

6.Y.1.1 Request

This request uses the POST method and has the following syntax:

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

6.Y.1.1.1 Target Resource

There are two different resources that can be subscribed to: the Studies Subscription resource, which creates a Collection Subscription:

```
/studies/subscriptions
```

And the Study Subscription resource, which creates an Instance Subscription:

```
/studies/subscriptions{/study}
```

6.Y.1.1.2 Query Parameters

The standard "retain" and "filter" query parameters are supported. See Section 6.1.X.5.3.1.2.

6.Y.1.1.3 Request Header Fields

See Section 6.1.X.5.3.1.3.

6.Y.1.1.4 Request Payload

The request shall have no payload.

6.Y.1.2 Behavior

The origin server shall create a subscription to the target resource, and return a URL reference to the Subscription in the Location header field of the response. See Section 6.1.X.5.3.2 for details.

6.Y.1.3 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 an opaque URL for the created subscription resource.

6.Y.1.3.1 Status Code

A success response shall have a 200 (OK) status code.

A failure response shall have an appropriate failure status code. See Section 6.1.X.5.3.3.1.

6.Y.1.3.2 Response Header Fields

See Section 6.1.X.5.3.3.2.

6.Y.1.3.3 Response Message Body

The response shall have no message body.

6.Y.2 Notify Transaction

This transaction sends an Instance Availability Event Report from the origin server to the user agent over a Notification Connection. See Section 6.1.X.5.5 and PS3.4, Annex R.

6.Y.2.1 Notification

See Section 6.1.X.5.5.

6.Y.2.1.1 Instance Availability Event Report

The Instance Availability Event Report Attributes are specified in Table 6.Y..

Table 6.Y.1-X: Instance Availability Event Report Attributes

Attribute Name	Tag	VR	Type (UA/OS)
Specific Character Set	(0008,0005)	CS	1C/1C*
All other Attributes of the SOP Common Module			3/3
Referenced Performed Procedure Step Sequence	(0008,1111)	SQ	2/2

Attribute Name	Tag	VR	Type (UA/OS)
>Referenced SOP Class UID	(0008,1150)	UI	1/1
>Referenced SOP Instance UID	(0008,1155)	UI	1/1
>Performed Workitem Code Sequence	(0040,4019)	CS	2/2
>>Code Value	(0008,0100)	SH	1/1
>>Coding Scheme Designator	(0008,0102)	SH	1/1
>>Code Meaning	(0008,0104)	LO	1/1
>>All other Attributes of the Performed Workitem Code Sequence			3/3
Study Instance UID	(0020,000D)	UI	1/1
Referenced Series Sequence	(0008,1115)	SQ	1/1
>Series Instance UID	(0020,000E)	UI	1/1
>Referenced SOP Sequence	(0008,1199)	SQ	1/1
>>Referenced SOP Class UID	(0008,1150)	UI	1/1
>>Reference SOP Instance UID	(0008,1155)	UI	1/1
>>Instance Availability	(0008,0056)	CS	1/1
~~>>Retrieve AE Title~~	~~(0008,0054)~~	~~AE~~	~~1/1~~
~~>>Retrieve Location UID~~	~~(0040,E011)~~	~~UI~~	~~3/3~~
~~>>Retrieve URI~~	~~(0040,E010)~~	~~UR~~	~~3/3~~
>>Retrieve URL	(0008,1190)	UR	~~3/3~~ 1/1
~~>>Storage Media File-Set ID~~	~~(0088,0130)~~		~~3/3~~
~~>>Storage Media File-Set UID~~	~~(0088,0140)~~		~~3/3~~

* Required if an extended or replacement character set is used.

6.Y.2.1.2 'application/json' Example Instance Notification Event Report

```
{
  {"00081111": {"vr": "SQ",
                "Value": [{"00081150": {"vr": "UI", "Value": [ uid ],
                           "00081155": {"vr": "UI", "Value": [ uid ],
                           "00404019": {"vr": "CS", "Value": [...],
                           "00080100": {"vr": "SH", "Value": [...],
                           "00080102": {"vr": "SH", "Value": [...],
                             "00080104": {"vr": "LO", "Value": [...]
                           ...  ; Other Attributes of Performed Workitem Code Sequence
                          }
                          ...
                         ],
   "0020000D": {"vr": "UI", "Value": [ uid ],
   "00081115": {"vr": "SQ",
                "Value": [{"0020000E": {"vr": "UI", "Value": [ uid ],
                           "00081199": {"vr": "SQ",
                                        "Value": [{"00081150": {"vr": "UI",
                                                                "Value": [ uid ],
                                                   "00081155": {"vr": "UI",
                                                                "Value": [ uid ],
                                                   "00080056": {"vr": "CS",
                                                                "Value": [...],
                                                   "00081190": {"vr": "UR",
```

```
                                                                "Value": [ url ]
                                                            }
                                                           ]
                                    }
                                    ...
                                   ]
        }
    }
```

6.Y.2.1.3 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

Tag	Attribute Name	VR
(0000,0002)	Affected SOP Class UID	UI
(0000,0100)	Command Field	US
(0000,0110)	Message ID	US
(0000,1000)	Affected SOP Instance UID	UI
(0000,1001)	Requested SOP Instance UID	UI
(xxxx,yyyy)	Document Type	UR
(0074,1238)	Reason for Cancellation	??
(0074,4041)	Input Readiness State	??

6.Y.2.2 Behavior

The user agent should take appropriate action when a Notification is received.

6.Y.2.3 Notification Acknowledgement

The user agent shall send a success or failure Acknowledgement as specified in Section 6.1.X.5.5.3