Digital Imaging and Communications in Medicine (DICOM)

Supplement 248: DICOMweb Send

Prepared by:

DICOM Standards Committee, Working Group 27

1812 N. Moore St, Suite 2200

Arlington, VA 22209, USA

Status: April 2026, Public Comment

Developed pursuant to DICOM Work Item 2025-05-B

Table of Contents

1

Document History

2025.06	Version 00	JM	Initial version with proposed approach, document structure and content.
2025.09	Version 01	JM	Reworked comments from WG06 2025.06's meeting. Elaborated Open and Closed Issues, added sections on Examples and Conformance.
2025.11	Version 02	JM	Reworked comments from WG06 2025.09's meeting. Accepted all changes, removed resolved comments, added section on C-MOVE, resolved open comments, added new comments, added some security considerations.
2026.01	Version 03	JM	Reworked comments from WG06 2025.11's meeting. Accepted all changes, removed resolved comments, resolved open comments, removed security considerations (as that is a separate CP now).
2026.03	Version 04	JM	Reworked comments from WG06 2026.01's meeting. Accepted all changes, removed resolved comments, resolved open comments. Added two new comments.
2026.04	Version 05 Public Comment	JM	Reworked comments from WG06 2026.03 meeting; also added several open issues, and incorporated security-related texts.

2

Open Issues

1	**Issue**: How to name the transactions and the supplement. What should be the colloquial name? **Context**: In DIMSE, the operation for which a DICOMweb equivalent is to be defined is called C-MOVE. While this is a misnomer, it has been suggested to seize the opportunity – also or perhaps even especially for newcomers to DICOM – and name it properly. Comment: Send is a residual alternative name for C-STORE in ACR-NEMA 1985, so that is perhaps not the best name. **Proposal**: There are several contenders: Move, Forward, Send, Provide, Deliver, Copy, …. It is currently proposed to name the supplement DICOMweb Send, and use the word Send in the transactions too. The colloquial name is proposed to be SEND-RS. **Decision**: [WG06-yyyymmdd]
2	**Issue**: How to deal with multiple responses from the origin server to the user agent? **Context**: DIMSE's C-MOVE allows for multiple (progress) responses by the SCP decided by the SCP. As DICOMweb is RESTful and hence transactional (request-response), how to deal with that aspect? There are several options: • **Combine them** – This follows the pattern used in the Search transaction. When a proxy of an origin server would convert a Search transaction to a C-FIND operation on an SCP, the pending results would need to be combined into one response; the proxy should also manage the limit and offset, and canceling the C-FIND when appropriate. Disadvantage: when applied to the forward transaction there would be no way to monitor progress. • **Enforce multipart responses** – This exploits a technical option in HTTP. When a proxy of an origin server would convert a Send transaction to a C-MOVE operation on an SCP, each pending result would need to be transformed into a part of the multi-part response, and a success/warning/failure result would yield the final part. Disadvantages:

	this a) does not follow existing DICOMweb interface patterns, b) does not solve the time-out issue, c) requires a somewhat more complex user agent, handling multiple HTTP parts separately instead of handling one HTTP payload. • **Poll for responses** – This follows the pattern used in the Storage Commitment service. When a proxy of an origin server would convert a Send transaction to a C-MOVE operation on an SCP, the last Pending result needs to be kept until the user agent asks for the response; at that time, a converted response can be returned. A difference with Storage Commitment is that there would be a payload (progress status) at every poll request, while Storage Commitment only provides a payload when done. Disadvantage: monitoring progress is by request of the user agent, while in DIMSE it is determined by the SCP. • **Use notifications** – The origin server can notify the user agent on progress, utilizing the notifications that are part of PS3.18. This follows a technical option that is currently available. When a proxy of an origin server would convert a Send Transaction to a C-MOVE, it would need to provide a subscription service for notifying the user agent, incl. extending the send transaction interface with a transaction UID. Disadvantage: requires subscription service by origin server, subscription by user agent, handling of multiple protocols by both parties. • **Long GET/long polling** – This follows a pattern described in RFC6202. When a proxy of an origin sever would convert a Send Transaction to a C-MOVE, it would keep the connection to the user agent open until the (first) status message that arrives is sent to the user agent, after which this – in case not all sub-operations are yet done – requests a new status. Disadvantages: yet another pattern in PS3.18 to deal with multiple messages from an SCP; seems to be old-fashioned; requires specific handling of time-outs (which are more likely). **Proposal**: Multiple responses are handled by polling for responses, as this best follows the existing DICOMweb interface patterns (see e.g. Storage Commitment in PS3.18, Section 13). Furthermore, this approach provides a solution to the HTTP time-out issue and avoids multi-part responses. **Decision**: [WG06-20250616] Agreed, yet reopened at WG06-20250901.
4	**Issue**: How should the user agent identify the destination to the origin server in this transaction? **Context**: In DIMSE, the destination of a C-MOVE is an AE Title, so in DICOMweb it would make most sense to use an HTTP endpoint (the URI of a DICOMweb Studies service that supports the Store Transaction) as destination. However, as an origin server can decide to use C-MOVE for its Store Transactions (see section 10.X), and a C-MOVE SCP can decide to use the Store Transaction for its sub-operations (see open issue 9 and updated section C.4.2 of PS3.4 below) the destination can, in principle, be either an AE Title or a base URI. In either case, there should be a mapping from one to another on such a server. Another option is to allow for either an endpoint or an AE Title as destination. That, however, gives the user agent the responsibility to choose, while the origin server is better positioned to do that. The chosen protocol of the sub-operation should in fact be irrelevant to the user agent. Using an HTTP endpoint as destination makes most sense from an API point of view: the web-world of DICOMweb is preferably mimicked into the parameters. **Proposal**: Use HTTP endpoint as destination, unless the security considerations imply to use the AE Title instead. **Decision**: [WG06-20250616] Agreed, yet reopened at WG06-20250901.
5	**Issue**: Can we learn from others w.r.t. the approach to third party push use cases? **Context**: DIMSE's C-MOVE operations allows for pushing instances to a third party: system A (the requester) asks system B (the data holder) to send certain instances to system C (the third-party destination). Do other use cases like this exist in the realm of HTTP and

	RESTful, and how have these been approached? That may provide input to the approach to be taken for the DICOMweb's C-MOVE equivalent. Some research provided us with examples like: • Cloud storage, where a client requests a cloud service to transfer data from one bucket or region to another, or even to another account. • Content Delivery Networks, where a system instructs a CDN to fetch and cache content from an origin server (and serve it to end users)[1]. • Genomics Data Pipeline, where a research portal requests a sequencing center to send genomics data to a bioinformatics platform. • Satellite Ground Station Networks, where a mission control system requests a ground station to forward telemetry data to a processing center. **Proposal**: The two-transaction model, where there is a transaction for requesting the transfer and one for checking the progress or status, is the most common approach in the RESTful world. There are other approaches too, for instance a) utilizing webhooks (with callback URLs) instead of having to poll or b) utilizing GraphQL mutations for initiating transfers and subscriptions for status updates. As the alternative approaches are beyond what is typically done in DICOMweb, the text proposes to use the two-transaction model. **Decision**: [WG06-yyyymmdd]
6	**Issue**: What should be the Group Number of the Attributes in the Send Request Response Module? **Context**: Within DIMSE's C-MOVE, (intermediate) results – like number of remaining operations and list of failed SOP instances – are largely conveyed as parameters at the message exchange / command level. How should these be presented in DICOMweb, esp. can we reuse the tags of the message fields, or should we introduce new ones? The attribute tags of the Status, Number of Remaining Sub-operations, Number of Completed Sub-operations, Number of Failed Sub-operations, and Number of Warning Sub-operations as currently available have 0000 as Group. Although not stated literally, this Group is meant to be used in commands only. Hence, other tags with the same semantics should be created. However, this would require a proxy to re-encode these values. An exception to the implied rule could be made here, as the Send Request Response Module is not to be sent via DIMSE. **Proposal**: It seems peculiar to introduce new tags for conveying the same, so it is proposed to use the tags that are already available. **Decision**: [WG06-yyyymmdd]
7	**Issue**: Is there any other applicable pattern for doing requests? **Context**: This question relates to issue 2, but this one is about a specific detail, namely that requests are to be dealt with as resources, as currently is done in Section 13, while mostly DICOMweb is about data resources. **Proposal**: As there are several uses of resources within DICOMweb beyond that of data resources, see also Sections 11.7 (Change Workitem State) and 11.8 (Request Cancellation), and the approach is in line with that of Section 13, it does not seem wise to use yet another pattern. **Decision**: [WG06-yyyymmdd]

[1] Actually, this is a reverse use case, as it concerns fetching data. Yet, there are three parties involved, and it can be considered a kind of 'pull from third party' instead of 'push to third party' use case that is currently not supported in DICOM.

8	**Issue**: Add colloquial naming for the transaction? **Context**: Currently we have STOW-RS, WADO-RS and QIDO-RS as colloquial names for the respective Studies service transactions. Should we have something like that for Send Transactions too? **Proposal**: Marketing terms make sense to ease communication, so it is proposed to add SEND-RS as colloquial name for this transaction. **Decision**: [WG06-yyyymmdd]
9	**Issue:** Do we want to allow C-MOVE to be able to use Store Transactions (STOW) for its sub-operations, as we allow DICOM Send to be able to use C-STORE operations? **Context**: Currently, C-MOVE must execute C-STORE sub-operations to get the request done. In hybrid eco-systems it may be needed to allow the SCP to use another means to handle the C-MOVE request. Use cases for this addition comprise: • A group of cooperating hospitals that allow DIMSE within a hospital, but do not allow DIMSE to cross the firewall to another hospital, while DICOMweb would be no issue. • For supporting easy access to studies for second opinions, a hospital wants to utilize cloud buckets from a medical cloud service. These services typically only support DICOMweb. **Proposal**: The current text of this supplement proposes to allow this; see PS3.4, Section C.4.2.3.1.1. Alternative Sub-operation Mechanism Option. Note that here a C-MOVE SCP is required to support C-STORE operations and may support DICOMweb Store Transactions. **Decision**: [WG-06-yyyymmdd]
10	**Issue**: How to document support of the Send Transaction in the conformance statement? **Context**: The Conformance Statements template needs additions for the Send Transaction. These sections should be N.5.3.2.X and N.5.3.4.X respectively. Currently, the content of these sections has been modelled after N.5.2.7.3 and N.5.2.7.4 (C-MOVE) respectively. Is this correct, or should it be modelled similar to N.5.3.2.4 (Search Transaction (QIDO-RS))? **Proposal**: Keep as is. **Decision**: [WG06-yyyymmdd]
11	**Issue**: Should we use URL or URI? **Context**: Is the identification of other parties in HTTP to be done using URLs or URIs? This question is broader than the scope of this supplement. **Proposal**: As this is broader than this supplement, this issue actually needs to be resolved elsewhere. **Decision**: [WG06-yyyymmdd]
12	**Issue**: Do we want to be able to create a DICOM conformant C-MOVE SCP proxy (proxying to DICOMweb Send), as we allow a DICOM conformant Send origin server proxy (proxying to DIMSE C-MOVE)? **Context**: In hybrid environments, one may want to utilize a proxy SCP that handles C-MOVE requests by transforming C-MOVE request content to DICOMweb content, and do a DICOM Send request using that content at an applicable origin server. Currently, such proxy cannot be DICOM conformant, as C-MOVE SCPs shall, for instance, perform C-STORE sub-operations and maintain counters. **Proposal**: The current text of this supplement proposes to allow this; see PS3.4, Section C.4.2.3.1.2. Proxy Option. Note that when claiming this option, the C-MOVE SCP is no longer required to support C-STORE sub-operations, as is does not execute sub-operations itself. **Decision**: [WG-06-yyyymmdd]

13	**Issue**: Do we need to require HTTPS? **Context**: This came up during the discussion on the alternative sub-operation mechanism option in WG06. The text originally used HTTPS, and changing this triggered this issue. **Proposal**: Do not require HTTPS. Security is orthogonal to this supplement, and will be dealt with in a separate Supplement. **Decision**: [WG-06-yyyymmdd]
14	**Issue**: How to deal with cancellation in a C-MOVE SCP proxy? **Context**: While we have excluded cancelling DICOMweb Send (see issue 3), allowing a proxy that translates C-MOVE to DICOMweb Send will need to handle C-CANCEL-MOVE-RQ, and map that to DICOMweb behavior somehow. The behavior of the SCP when receiving a C-CANCEL-MOVE is described as: > The SCP may receive a C-CANCEL-MOVE request at any time during the processing of the C-MOVE. The SCP shall interrupt all C-STORE sub-operation processing and return a status of Cancel in the C-MOVE response. The C-MOVE response with a status of Cancel shall contain the number of Completed, Failed, and Warning C-STORE sub-operations. If present, the Remaining sub-operations count shall contain the number of C-STORE sub-operations that were not initiated due to the C-CANCEL-MOVE request. While this text raises questions too (why are C-STORE sub-operations that have been initiated but not yet finished excluded from the counters), it is clear on what to do: interrupt the execution of sub-operations and return with status Cancel. A few options have been identified to deal with this in a proxy: 1. Reopen closed issue 3 and do add cancellation to DICOMweb Send; this would require a new transaction, e.g. Cancel Send. The receipt of a Cancel Send request would require the DICOMweb Send origin server to interrupt all sub-operations as described above and return a proper Send Request Response Module with status Cancel. 2. Mostly ignore the C-CANCEL-MOVE-RQ and continue to work. Yet, do return the status Cancel. While this continues to Send all matching Instances (and hence does not interrupt the sub-operations at all), this does not require a new Transaction to be defined and implemented and does not reopen the generic cancellation discussion. One could argue that continuing the sub-operations is not that bad, as the C-MOVE SCU does not know the effect of the cancellation; the C-MOVE could just have been finished. 3. Disconnect from the DICOMweb Send origin server, returning the last known status regarding the counters. This is the sticking-your-head-in-the-sand approach, as the sub-operations will probably just continue, but no result reflecting the status can be returned, esp. regarding the counters. 4. Intentionally do not prescribe the proxy behavior in the standard and let implementations describe their behavior in the conformance statement. Another option would be to not allow for DICOM conformant C-MOVE proxies as described in the text of the supplement and in issue 12. Feedback proposing other options is appreciated, as none of the above options are ideal. **Proposal**: Select option 2 above, as that is the least invasive for both the standard and the implementations, still allowing DICOM conformant proxies, useful in hybrid eco-systems. **Decision**: [WG-06-yyyymmdd]
15	**Issue:** Should we combine Send and Check Send Result in PS3.2, Table N.1-9? **Context**: Table N.1-9 in PS3.2 gives an overview of the DICOMweb Services provided by <product>. While currently this table has no combined Transactions, it makes sense to do so as to make it easier for the reader and to prevent mistakes by the writer (allowing some resources to be available for the Send and not for the Check Send Result or vice versa). The same also holds for Table N.1-11. **Proposal**: Combine the two into one row. **Decision:** [WG-06-yyyymmdd]

Closed Issues

3	**Issue**: How to deal with cancelling DICOMweb Send? **Context**: Within DIMSE, it is possible to cancel C-MOVE, and the other basic operations C-FIND and C-GET using the C-operation-CANCEL-RQ. This has not been mimicked to DICOMweb: currently no Transaction can be cancelled. UPS-RS allows the cancellation of a workitem, but that is at application level and does not cancel a Transaction. **Proposal**: Canceling the send request transaction will, for now, not be supported. When the need arises, a New Work Item Proposal for cancelling Transactions can be created. This would need to include cancelling Searching, Committing Storage, Sending, … **Decision**: [WG06-20250616] Agreed.
16	**Issue**: How to register the tag numbers in the Send Request Response Module? **Context**: The tags of this module have been numbered according to the proposal of issue 6. Now these numbers are not present in PS3.6, and originally this supplement proposed to add them to Table 6-1. However, this may conflict with their presence in PS3.7, Annex E. Command Dictionary. **Proposal**: Do not add these tags to PS3.6. **Decision**: Proposed by Rob Horn.

Scope and Field of Application

This supplement adds Send Transactions to DICOMweb's Studies and Non-Patient Instances Services to mirror the C-MOVE operation that is already available in DIMSE. The Send Transactions have been designed with the intention of facilitating proxies from/to DIMSE. Furthermore, it defines a C-MOVE option to allow for STOW sub-operations and an option to allow for C-MOVE SCP proxies, delegating to a DICOMweb Send origin server.

Security plays an important role for this supplement, as this supplement defines the three party DICOMweb Send Transaction, and provides means to cross two DICOM protocols, i.e., between DICOMweb and DIMSE, at both the operation level and the sub-operation level. Yet, DICOM considers security orthogonal to functionality, and hence applicable security mechanisms and considerations are described in a separate supplement.

Changes to NEMA Standards Publications PS 3.18

Update first part of section 10 Studies Service and Resources as indicated below

10 Studies Service and Resources

10.1 Overview

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

Note It is not possible to request to send selected Frames.

The Retrieve transaction of this Service is also known as WADO-RS. The Store transaction of this Service is also known as STOW-RS. The Search transaction of this Service is also known as QIDO-RS. **The Send transactions of this Service are collectively known as SEND-RS.** See Section 10.3.

10.1.1 Resource Descriptions

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

There are three top level resources:

/studies references all Studies managed by the service.

/series references all Series managed by the service.

/instances references all Instances managed by the service.

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

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

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

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

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

{bulkdataURI} an opaque URI that references a Bulkdata Value.

The Studies Service defines the following resources:

Table 10.1-1. Resources and Descriptions

Resource	Description
Studies Service	The Base URI of the Studies Service.
All Studies	The All Studies resource references the entire collection of Studies contained in the Studies Service.

Resource	Description
Study	The Study resource references a single Study.
Study Metadata	The Study Metadata resource references the Metadata of a Study.
Study Bulkdata	The Study Bulkdata resource references the Bulkdata of a Study.
Study Pixel Data	The Study Pixel Data resource references the Pixel Data of a Study.
Rendered Study	The Rendered Study resource references an alternate Media Type rendering of a Study.
Rendered MPR Volume Study	The Rendered MPR Volume Study resource references a multiplanar reformat rendering of a Study.
Rendered 3D Volume Study	The Rendered 3D Volume Study resource references a volume rendering of a Study.
Study Thumbnail	The Study Thumbnail resource references a thumbnail image of a Study.
Study's Series	The Study's Series resource references the collection of all Series contained in a Study.
Study's Instances	The Study's Instances resource references the collection of all Instances in a single Study.
Study's Send Requests	**The Study Send Requests resource references the collection of send requests associated with a single study.**
All Series	The All Series resource references the collection of all Series in all Studies contained in the Studies Service.
Series	The Series resource references a single Series.
Series Metadata	The Series Metadata resource contains the Metadata of a Series in a Study.
Series Bulkdata	The Series Bulkdata resource references the Bulkdata of a Series.
Series Pixel Data	The Series Pixel Data resource references the Pixel Data of a Series.
Rendered Series	The Rendered Series resource references an alternate Media Type rendering of a Series.
Rendered MPR Volume Series	The Rendered MPR Volume Series resource references a multiplanar reformat rendering of a Series.
Rendered 3D Volume Series	The Rendered 3D Volume Series resource references a volume rendering of a Series.
Series Thumbnail	The Series Thumbnail resource references a thumbnail image of a Series.
Series' Instances	The Series' Instances resource references the collection of all Instances in a single Series.
Series' Send Requests	**The Series' Send Requests resource references the collection of send requests associated with a single Series.**
All Instances	The All Instances resource references the collection of all Instances in all Series in all Studies contained in the Studies Service.
Instance	The Instance resource references a single Instance.

Resource	Description
Instance Metadata	The Instance Metadata resource contains the Metadata of an Instance.
Instance Bulkdata	The Instance Bulkdata resource references the Bulkdata of a Instance.
Instance Pixel Data	The Instance Pixel Data resource references the Pixel Data of a Instance.
Rendered Instance	The Rendered Instance resource references an alternate Media Type rendering of an Instance.
Rendered MPR Volume Instance	The Rendered MPR Volume Instance resource references a multiplanar reformat rendering of an Instance.
Rendered 3D Volume Instance	The Rendered 3D Volume Instance resource references a volume rendering of an Instance.
Instance Thumbnail	The Instance Thumbnail resource references a thumbnail image of an Instance.
Instance's Send Requests	**The Instance's Send Requests resource references the collection of send requests associated with a single Instance.**
Frames	The Frames resource references an ordered collection of frames in a single multi-frame Instance.
Rendered Frames	The Rendered Frames resource references an alternate Media Type rendering of an ordered collection of frames of a single multi-frame Instance.
Rendered MPR Volume Frames	The Rendered MPR Volume Frames resource references a multiplanar reformat rendering of a collection of frames.
Rendered 3D Volume Frames	The Rendered 3D Volume Frames resource references a volume rendering of a collection of frames.
Frame Thumbnail	The Frame Thumbnail resource references a thumbnail image for frames within an Instance.
Bulkdata	The Bulkdata resource contains a Bulkdata Value.

10.1.2 Common Query Parameters

…

10.1.3 Common Media Types

…

10.2 Conformance

An origin server claiming conformance to **one or more of** the ~~**Retrieve**~~ Transaction**s** of the Studies Service:

- shall support the Retrieve Capabilities Transaction (see Section 8.9.1);
- shall support the**se** ~~**Retrieve**~~ Transaction**s** for ~~**all**~~**their respective** mandatory resources in Table 10.3-2.

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

- ~~**shall support the Retrieve Capabilities Transaction (see Section 8.9.1);**~~
- ~~**shall support the Store Transaction for all mandatory resources in Table 10.3-2.**~~

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

- ~~**shall support the Retrieve Capabilities Transaction (see Section 8.9.1);**~~
- ~~**shall support the Search Transaction for all mandatory resources in Table 10.3-2.**~~

An origin server shall claim conformance to either both or none of the Send Transaction and the Check Send Result Transaction.

The user agent may support any of the transactions for any of the corresponding resources in Table 10.3-2.

10.3 Transactions Overview

The Studies Service consists of the transactions listed in Table 10.3-1.

Table 10.3-1. Studies Service Transactions

Transaction Name	Method	Payload		Description
		Request	Success Response	
Retrieve	GET	N/A	Instance(s), Metadata, Renderings, Pixel Data, or Bulkdata	Retrieve one or more representations of DICOM Resources.
Store	POST	Instance(s)	Store Instances Response Module	Stores one or more representations of DICOM Resources, contained in the request payload, in the location referenced by the Target Resource.
Search	GET	N/A	Result(s)	Searches the Target Resource for DICOM objects that match the search parameters and returns a list of matches in an Acceptable Media Type.
Send	**POST**	**N/A**	**Send Request Response Module**	**Searches the Target Resource for DICOM objects that match the search parameters and sends or starts to send these to another server.**
Check Send Result	**GET**	**N/A**	**Send Request Response Module**	**Gets the status and results of a Send transaction.**

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

Table 10.3-2. Resources by Transaction

Resource	Retrieve	Store	Search	Send	Check Send Result
Studies Service					
All Studies		M	M		
Study	M	M			
Study Metadata	M				

Resource	Retrieve	Store	Search	**Send**	**Check Send Result**
Study Bulkdata	O				
Study Pixel Data	O				
Rendered Study	M				
Rendered MPR Volume Study	O				
Rendered 3D Volume Study	O				
Study Thumbnail	O				
Study's Series			M		
Study's Instances			M		
Study's Send Requests				**M**	**M**
All Series			M		
Series	M				
Series Metadata	M				
Series Bulkdata	O				
Series Pixel Data	O				
Series' Instances			M		
Series' Send Requests				**M**	**M**
Rendered Series	M				
Rendered MPR Volume Series	O				
Rendered 3D Volume Series	O				
Series Thumbnail	O				
All Instances			M		
Instance	M				
Instance Metadata	M				
Instance Bulkdata	O				
Instance Pixel Data	O				
Rendered Instance	M				

Resource	Retrieve	Store	Search	**Send**	**Check Send Result**
Rendered MPR Volume Instance	O				
Rendered 3D Volume Instance	O				
Instance Thumbnail	O				
Instance's Send Requests				**M**	**M**
Frames	M				
Rendered Frames	M				
Rendered MPR Volume Frames	O				
Rendered 3D Volume Frames	O				
Frame Thumbnail	O				
Bulkdata	M	M			

…

Add new section 10.X Send Transaction, immediately before new Section 10.Y given below

10.X Send Transaction

This Transaction uses the POST method to request the sending of Studies, Series, or Instances managed by the origin server to a destination server.

Note The user agent could be the destination server of this transaction.

Together with the Check Send Result Transaction, Section 10.Y, this Transaction corresponds to the DIMSE C-MOVE Operation (see PS3.4, Section C.4.2), but not supporting cancellation.

Due to the typically long-running nature of sending Studies, Series, or Instances to another server, the origin server may choose between two alternatives to communicate about the status and outcome of the request. When the outcome of the request is immediately available, the origin server typically returns the status and outcome in the response to the request. When the outcome of the request is not available immediately, the origin server indicates that to the client in its response, and the client uses the Check Send Result Transaction. This pattern of Transactions is similar to what has been illustrated in Figure 13.1-1.

The Store sub-operations that are necessary to fulfil the send request may be executed using DIMSE or DICOMweb, or even a combination of the two. The term sub-operations will henceforth be used for both DIMSE's sub-operations and DICOMweb's sub-transactions.

10.X.1 Request

The request shall have the following syntax:

```
POST SP "/" {resource} "/send-requests/" {transactionUID} "?" destination {&search*} SP version CRLF
Accept: 1#media-type
*(header-field CRLF)
CRLF
```

where

```
destination = "destination" "=" {URL}
```

and where `{URL}` is a percent-encoded absolute URL of the destination endpoint; see [RFC 3986].

10.X.1.1 Target Resources

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

An origin server that is a native implementation shall support all Mandatory (M) resources specified in the Native column in Table 10.X.1-1.

An origin server that is a DIMSE Proxy implementation shall support all Mandatory (M) resources specified in the Proxy column in Table 10.X.1-1.

Table 10.X.1-1. Send Transaction Resources

Resource	URI Template	Native	Proxy	Query Type
All Studies Send Requests	`/studies/send-requests/{transactionUID}`	M	M	hierarchical
Study's Series Send Requests	`/studies/{study}/series/send-requests/{transactionUID}`	M	M	hierarchical

Resource	URI Template	Native	Proxy	Query Type
Study's Instances Send Requests	`/studies/{study}/instances/send-requests/{transactionUID}`	M	O	relational
All Series Send Requests	`/series/send-requests/{transactionUID}`	M	O	relational
Study Series' Instances Send Requests	`/studies/{study}/series/{series}/instances/send-requests/{transactionUID}`	M	M	hierarchical
All Instances Send Requests	`/instances/send-requests/{transactionUID}`	M	O	relational

For more information about Hierarchical Queries see Section C.4.1.3.1.1 “Hierarchical Search Method” in PS3.4. For more information about Relational Queries see Section C.4.1.2.2.1 “Relational-Queries” in PS3.4 and Section C.4.1.3.2.1 “Relational-Queries” in PS3.4.

Table 10.X.1-2 shows the resources supported by the Send transaction along with a description of the search performed and the results returned.

Table 10.X.1-2. Send Resource Descriptions

Resource	Description
All Studies’ Send Requests	Searches the entire service for Studies that match the search parameters, and sends these to the requested destination.
Study's Series’ Send Requests	Searches for all Series in the specified Study that match the search parameters, and sends these to the requested destination.
Study's Instances’ Send Requests	Searches for all Instances in the specified Study that match the search parameters, and sends these to the requested destination.
All Series’ Send Requests	Searches the entire service for Series that match the search parameters, and sends these to the requested destination.
Study Series' Instances’ Send Requests	Searches for all Instances in the specified Study and Series that match the search parameters, and sends these to the requested destination.
All Instances’ Send Requests	Searches the entire service for Instances that match the search parameters, and sends these to the requested destination.

10.X.1.2 Query Parameters

The origin server shall support Query Parameters as required in Table 8.3.4-1; however, the includefield parameter is ignored.

The user agent shall supply Query Parameters as required in Table 8.3.4-1; however, the includefield parameter is ignored.

The following sections further define details of the parameters of the search aspect of this transaction:

- Section 10.6.1.2.1 Attribute/Value Pair Requirements.

- Section 10.6.1.2.2 Search Key Types and Requirements.
- Section 10.6.1.2.3 Required Matching Attributes.
- Section 10.6.1.2.4 Optional Repository Query Attributes.

10.X.1.3 Request Header Fields

The origin server shall support request header fields as required in Table 10.X.1-3.

The user agent shall supply request header fields as required in Table 10.X.1-3.

Table 10.X.1-3. Request Header Fields

Name	Values	Usage		Description
		User Agent	Origin Server	
Accept	media-type	M	M	The Acceptable Media Types of the response payload.

See also Section 8.4.

10.X.1.4 Request Payload

The request shall have no payload.

10.X.2 Behavior

The origin server shall perform a search according to the requirements specified in Section 8.3.4.

The origin server shall initiate one or more sub-operations to send the matching instances to the requested destination. The sub-operations are permitted to be performed using DIMSE C-STORE or DICOMweb Store, or even a combination of the two, at the discretion of the origin server and depending on the capabilities of the destination.

Note To be able to handle the sub-operations using C-STORE operations, the origin server needs to have a function to translate the provided endpoint to the appropriate AE Title. The definition of this function is beyond the scope of DICOM.

Note It is conceivable that authentication is required to perform sub-operations. Credentials for authentication may be coming from the origin server or the user agent, depending on the security architecture at hand.

10.X.3 Response

The response shall have the following syntax:

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

10.X.3.1 Status Codes

Table 10.X.3-1 shows some common status codes corresponding to this Transaction. See also Section 8.5 for additional status codes.

Table 10.X.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The origin server finished processing the send request and performing the send. It is possible some Instances matching the request were not successfully sent, e.g. due SOP

		Class incompatibility for some sub-operations. The payload describes in detail what has been achieved.
	202 (Accepted)	The origin server successfully validated the request message but has not finished processing the send request; the payload describes in detail what has been achieved to this point. The user agent is expected to follow up with a Check Send Result Transaction, described in Section 10.Y, to get updated result information.
Failure	400 (Bad Request)	The origin server cannot handle the send request because of errors in the request headers or parameters.
	409 (Conflict)	The origin server cannot handle the send request because the provided Transaction UID is already in use.
	503 (Service Unavailable)	The origin server cannot handle the send request; this may be a temporary or permanent state.

Note A 200 (OK) success status code should only be understood to mean that the request was successfully parsed and a Send response was returned by the origin server. The Send response may indicate that sending failed for some or even all of the matching SOP Instances.

10.X.3.2 Response Header Fields

The origin server shall support header fields as required in Table 10.X.3-2. All success responses shall also contain the Content Representation (see Section 8.4.2) and Payload header fields (see Section 8.4.3) with appropriate values.

Table 10.X.3-2. Response Header Fields

Name	Values	Origin Server Usage	Description
Content-Type	media-type	C	See section 8.4.2.
Content-Encoding	encoding	C	See section 8.4.2.
Content-Length	uint	C	See section 8.4.3.
Retry-After	uint	O	The number of seconds the user agent is requested to wait until starting a Check Send Result Transaction.

Note The Retry-After header field may be useful in conjunction with a 202 and 503 response to allow the origin server to inform the user agent about effective polling intervals.

10.X.3.3 Response Payload

A success response payload shall contain a Send Request Response Module. See Annex @.

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

Add new section 10.Y Check Send Result Transaction, immediately after section 10.X as above

10.Y Check Send Result Transaction

This Transaction uses the GET method to allow a user agent to request an origin server to provide the current result of an earlier Send Transaction, see Section 10.X.

Together with the Send Transaction, this Transaction corresponds to the DIMSE C-MOVE Operation (see PS3.4, Section C.4.2), but not supporting cancellation.

10.Y.1 Request

The request shall have the following syntax:

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

10.Y.1.1 Target Resource

The Target Resource of this Transaction is a send request identified by its Transaction UID. The applicable resources are listed in Table 10.X.1-1.

10.Y.1.2 Query Parameters

The request has no Query Parameters.

10.Y.1.3 Request Header Fields

The origin server shall support Request Header Fields as required in Table 10.Y.1-1.

The user agent shall supply Request Header Fields as required in Table 10.Y.1-1.

Note The presence and values of the Check Send Result Header Fields should be the same as the Header Fields of the corresponding Send Request.

Table 10.Y.1-1. Check Send Result Header Fields

Name	Values	Usage		Description
		User Agent	Origin Server	
Accept	media-type	M	M	The Acceptable Media Types of the response payload.

See also Section 8.4.

10.Y.1.4 Request Payload

The request shall have no payload.

10.Y.2 Behavior

The origin server shall provide the current result status of the Send Request in an Acceptable Media Type (see section 8.7.4).

10.Y.3 Response

The response shall have the following syntax:

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

10.Y.3.1 Status Codes

Table 10.Y.3-1 shows some common status codes corresponding to this Transaction. See also Section 8.5 for additional status codes.

Table 10.Y.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The origin server finished processing the send request identified by the supplied Transaction UID (see Section 10.X) and performing the send; the payload describes in detail what has been achieved.
	202 (Accepted)	The origin server has not finished processing the send request; the payload describes in detail what has been achieved at this point. The user agent is expected to follow up with a Check Send Result Transaction to get to know the entire result of the send request.
Failure	404 (Not Found)	The origin server cannot find the send request result identified by the supplied Transaction UID.
	410 (Gone)	The origin server can no longer respond to the send request result identified by the supplied Transaction UID.
	503 (Service Unavailable)	The origin server cannot handle the Check Send Result request; this may be a temporary or permanent state.

Note

1. A 200 (OK) success status code should only be understood to mean that a Check Send Result response was returned by the origin server. The Check Send Result response may indicate that sending failed for some or even all of the matching SOP Instances.
2. The 404 (Not Found) status code may be caused by an incorrect Transaction UID that has been supplied by the user agent, or the origin server may have deleted the applicable result and not kept a record of it.
3. The 410 (Gone) status code may be caused by the origin server having a record of the Transaction UID but having deleted the request and/or applicable result.
4. When the 503 (Service Unavailable) status code is returned, the user agent might retry later with another Check Send Result Transaction.

10.Y.3.2 Response Header Fields

The Response Header Fields are the same as for the Send Transaction. See Section 10.X.3.2.

10.Y.3.3 Response Payload

The Response Payload is the same as for the Send Transaction. See Section 10.X.3.3.

Update first part of section 12 Non-Patient Instance Service and Resources as indicated below

12 Non-Patient Instance Service and Resources

12.1 Overview

The Non-Patient Instance (NPI) Storage Service enables a user agent to retrieve, store, and search an origin server for Instances that are not related to an individual patient. **It also enables a user agent to request a server to send such Instances to another server.**

Note

1. Non-Patient Instances adhere to a Composite Instance IOD Information Model that does not have at its root the Patient Information Entity representing an individual Patient.
2. "Non-patient" does not imply that there is no patient-related identifiable information in the Instances. E.g., the Inventory IOD does include Attributes of the patient, but it does not have a Patient IE at the root of its information model.

An NPI Storage Service manages a collection of resources belonging to the categories specified in Table 12.1.1-1.

All NPI Storage Service origin servers shall support the Retrieve Capabilities, Retrieve, and Search transactions. Support for the Store transaction is optional. All NPI Storage Service user agents support one or more of the Retrieve Capabilities, Retrieve, Store, or Search transactions.

12.1.1 Resource Descriptions

An NPI Service manages resources from the same NPI Category. Target URIs have the following templates:

```
/{npi-name}
/{npi-name}/{uid}
/{npi-name}/send-requests/{transaction-uid}
```

Where

```
npi-name        = "color-palettes"
                / "defined-procedure-protocols"
                / "hanging-protocols"
                / "implant-templates"
                / "inventories"
                / "protocol-approvals"
uid             ; is the Unique Identifier of an NPI Instance
transaction-uid ; is the Unique Identifier of a Send Request
```

Table 12.1.1-1 contains the templates for the NPI Resource Categories.

Table 12.1.1-1. Resource Categories, URI Templates and Descriptions

Resource Category	URI Template and Description	Corresponding IOD	Storage Class	Information Model
Color Palette	/color-palettes{/uid}	Section A.58 “Color Palette IOD” in PS3.3	Section GG “Non-Patient Object Storage Service Class” in PS3.4	Section X.1.3 “Query/Retrieve Information Model” in PS3.4

Resource Category	URI Template and Description	Corresponding IOD	Storage Class	Information Model
Defined Procedure Protocol	/defined-procedure-protocols{/uid}	Section A.82 “Procedure Protocol IODs” in PS3.3	Section GG “Non-Patient Object Storage Service Class” in PS3.4	Section HH.1.3 “Query/Retrieve Information Model” in PS3.4
Hanging Protocol	/hanging-protocols{/uid}	Section A.44 “Hanging Protocol IOD” in PS3.3	Section GG “Non-Patient Object Storage Service Class” in PS3.4	Section U.1.3 “Query/Retrieve Information Model” in PS3.4
Implant Template	/implant-templates{/uid}	Section A.61 “Generic Implant Template IOD” in PS3.3	Section GG “Non-Patient Object Storage Service Class” in PS3.4	Section BB.1.3 “Query/Retrieve Information Model” in PS3.4
Inventory	/inventories{/uid}	Section A.88 “Inventory IOD” in PS3.3	Section GG “Non-Patient Object Storage Service Class” in PS3.4	Section JJ.2 “Inventory Q/R Information Model” in PS3.4

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

12.1.2 Common Query Parameters

…

12.1.3 Common Media Types

…

12.2 Conformance

An origin server conforming to the NPI Service shall implement the Retrieve Capabilities Transaction (see Section 8.9.1).

The origin server shall support the transactions listed as Required in Table 12.2-1.

Table 12.2-1. Required and Optional Transactions

Transaction	Support	Section
Retrieve Capabilities	Required	Section 8.9
Retrieve	Required	Section 12.4
Store	Optional	Section 12.5
Search	Required	Section 12.6
Send	**Optional**	**Section 12.X**
Check Send Result	**Optional**	**Section 12.Y**

An origin server shall claim conformance to either both or none of the Send Transaction and the Check Send Result Transaction.

Implementations shall specify in their Conformance Statement (see PS3.2) and the Retrieve Capabilities Transaction (see Section 8.9 and Annex H):

- The implementation's role: origin server, user agent, or both.
- The supported resources (IODs) for each role.

In addition, for each supported transaction they shall specify:

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

12.3 Transactions Overview

The NPI Service consists of the transactions listed in Table 12.3-1.

Table 12.3-1. NPI Service Transactions

Transaction	Method	Resource	Payload		Description
			Request	Success Response	
Retrieve Capabilities	OPTIONS	/	N/A	Capabilities Description	Retrieves a description of the capabilities of the NPI Service, including transactions, resources, query parameters, etc.
Retrieve	GET	/{npi-name}/{uid}	N/A	Instance and/or Status Report	Retrieves an Instance, specified by the Target Resource in an Acceptable DICOM Media Type.
Store	POST	/{npi-name}{/uid}	Instance(s)	Status Report	Stores one or more DICOM Instances contained in the request payload, in the location referenced by the Target URL.
Search	GET	/{npi-name} ?{params*}	N/A	Result(s) and/or Status Report	Searches the Target Resource for Instances that match the search parameters and returns a list of matches in an Acceptable DICOM Media Type.
Send	**POST**	**/{npi-name}/send-requests?{transaction-uid}**	**N/A**	**Send Request Response Module**	**Searches the Target Resource for Instances that match the search parameters and sends or starts to send these to another server.**
Check Send Result	**GET**	**/{npi-name}/send-requests?{transaction-uid}**	**N/A**	**Send Request Response Module**	**Gets the result of a Send Transaction.**

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

Table 12.3-2 summarizes the Target Resources permitted for each transaction.

296 **Table 12.3-2. Resources by Transaction**

Resource	URI	Retrieve	Store	Search	**Send**	**Check Send Result**	Capabilities
NPI Service	/						X
All Instances	/{npi-name}		X	X			
Instance	/{npi-name}/{uid}	X	X				
Send Requests	**/{npi-name}/send-requests?{transaction-uid}**				**X**	**X**	

297
298 …

Add new section 12.X Send Transaction, immediately before new Section 12.Y below

12.X Send Transaction

This Transaction uses the POST method to request the sending of Non-Patient Instances managed by the origin server to a destination server.

Note The user agent could be the destination server of this Transaction.

Together with the Check Send Result Transaction, Section 12.Y, this Transaction corresponds to the DIMSE C-MOVE Operation (see PS3.4, Sections X.4.2, BB.4.2, HH.4.2, II.4.2, and JJ.3.2 for each kind of NPI respectively), but not supporting cancellation.

Due to the typically long-running nature of sending Instances to another server, the origin server may choose between two alternatives to communicate about the status and outcome of the request When the outcome of the request is immediately available, the origin returns the status and outcome in the response to the request. When the outcome of the request is not available immediately, the origin server indicates that to the client in its response, and the client uses the Check Send Result Transaction. This pattern of Transactions is similar to what has been illustrated in Figure 13.1-1.

The Store sub-operations that are necessary to fulfil the send request may be executed using DIMSE or DICOMweb, or even a combination of the two. The term sub-operations will henceforth be used for both DIMSE's sub-operations and DICOMweb's sub-transactions.

12.X.1 Request

The request shall have the following syntax:

```
POST SP /{npi-name}/send-requests?{transactionUID}?" destination {&search*} SP version CRLF
Accept: 1#media-type
*(header-field CRLF)
CRLF
```

where

```
destination = "destination" "=" {URL}
```

and where `{URL}` is a percent-encoded absolute URL of the destination endpoint; see [RFC 3986].

12.X.1.1 Target Resources

The Target URI shall reference one of the resources shown in Table 12.X.1-1.

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

Table 12.X.1-1. Send Transaction Resources

Resource	URI Template	Description
All Send Requests	/{npi-name}/send-requests/{transactionUID}	Searches a collection of NPI Instances that match the search parameters and sends these to the requested destination.

12.X.1.2 Query Parameters

The user agent shall supply, and the origin server shall support, the Common Query Parameters in Section 12.1.2.

The origin server shall support Query Parameters as required in Table 8.3.4-1; however, the includefield parameter is ignored.

The user agent shall supply in the request Query Parameters as required in Table 8.3.4-1; however, the includefield parameter is ignored.

For each Resource Category the origin server supports, it shall support the behaviors and matching key Attributes specified in the corresponding sections in Table 12.X.1-2.

Table 12.X.1-2. NPI Resource Send Attributes

Resource Category	Behaviors and Matching Key Attributes
Color Palette	Section X.6.1.2 “Color Palette Attributes” in PS3.4.
Defined Procedure Protocol	Section HH.6.1.2 “Defined Procedure Protocol Attributes” in PS3.4.
Hanging Protocol	Section U.6.1.2 “Hanging Protocol Attributes” in PS3.4.
Implant Template	Section BB.6.1.2 “Implant Template Attributes” in PS3.4.
Inventory	Section JJ.2.2 “Inventory Q/R Information Model Attributes” in PS3.4.
Protocol Approval	Section II.6.1.2 “Protocol Approval Attributes” in PS3.4

12.X.1.3 Request Header Fields

The origin server shall support request header fields as required in Table 12.X.1-3.

The user agent shall supply request header fields as required in Table 12.X.1-3.

Table 12.X.1-3. Request Header Fields

Name	Values	Usage		Description
		User Agent	Origin Server	
Accept	1#-dicom-media-type	M	M	The Acceptable Media Types of the response payload

See also Section 8.4.

12.X.1.4 Request Payload

The request has no payload.

12.X.2 Behavior

The origin server shall perform the search indicated by the request, using the matching behavior specified in Section 8.3.4.1.1 and in the corresponding sections in Table 8.3.4-1.

The rules for search results are specified in Section 8.3.4.

The origin server shall initiate one or more sub-operations to send the matching instances to the requested destination. The sub-operations are permitted to be performed using DIMSE C-STORE or DICOMweb Store, or even a combination of the two, at the discretion of the origin server and depending on the capabilities of the destination.

Note To be able to handle the sub-operations using C-STORE operations, the origin server needs to have a function to translate the provided endpoint to the appropriate AE Title. The definition of this function is beyond the scope of DICOM.

Note It is conceivable that authentication is required to perform sub-operations. Credentials for authentication may be coming from the origin server or the user agent, depending on the security architecture at hand.

12.X.3 Response

The response shall have the following syntax:

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

12.X.3.1 Status Codes

Table 12.X.3-1 shows some common status codes corresponding to this Transaction. See also Section 8.5 for additional status codes.

Table 12.X.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The origin server finished processing the send request and performing the send. It is possible some Instances matching the request were not successfully sent, e.g. due SOP Class incompatibility for some sub-operations. The payload describes in detail what has been achieved.
	202 (Accepted)	The origin server successfully validated the request message but has not finished processing the send request; the payload describes in detail what has been achieved to this point. The user agent is expected to follow up with a Check Send Result Transaction, described in Section 12.Y, to get updated result information.
Failure	400 (Bad Request)	The origin server cannot handle the send request because of errors in the request headers or parameters.
	409 (Conflict)	The origin server cannot handle the send request because the provided Transaction UID is already in use.
	503 (Service Unavailable)	The origin server cannot handle the send request; this may be a temporary or permanent state.

Note A 200 (OK) success status code should only be understood to mean that the request was successfully parsed and a Send response was returned by the origin server. The Send response may indicate that sending failed for some or even all of the matching SOP Instances.

12.X.3.2 Response Header Fields

The origin server shall support header fields as required in Table 12.X.3-2. All success responses shall also contain the Content Representation (see Section 8.4.2) and Payload header fields (see Section 8.4.3) with appropriate values.

Table 12.X.3-2. Response Header Fields

Name	Values	Origin Server Usage	Description
Content-Type	media-type	C	See section 8.4.2.
Content-Encoding	encoding	C	See section 8.4.2.
Content-Length	uint	C	See section 8.4.3.

Retry-After	uint	O	The number of seconds the user agent is requested to wait until starting a Check Send Result Transaction.

Note The Retry-After header field may be useful in conjunction with a 202 and 503 response to allow the origin server to inform the user agent about effective polling intervals.

12.X.3.3 Response Payload

A success response shall contain a Send Request Response Module. See Annex @.

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

Add new section 12.Y Check Send Result Transaction, immediately after section 12.X above

12.Y Check Send Result Transaction

This Transaction uses the GET method to allow a user agent to request an origin server to provide the current result of an earlier Send Transaction, see Section 12.X.

Together with that Send Transaction, this Transaction corresponds to the DIMSE C-MOVE Operation (see PS3.4, Sections X.4.2, BB.4.2, HH.4.2, II.4.2, and JJ.3.2 for each kind of NPI respectively) but not supporting cancellation.

12.Y.1 Request

The request shall have the following syntax:

```
GET SP /{npi-name}/send-requests/{transactionUID} SP version CRLF
Accept: 1#media-type CRLF
*(header-field CRLF)
CRLF
```

12.Y.1.1 Target Resource

The Target Resource of this Transaction is a send request identified by its Transaction UID. The applicable resources are listed in Table 12.X.1-1.

12.Y.1.2 Query Parameters

The request has no Query Parameters.

12.Y.1.3 Request Header Fields

The origin server shall support Request Header Fields as required in Table 12.Y.1-1.

The user agent shall supply Request Header Fields as required in Table 12.Y.1-1.

Note The presence and values of the Check Send Result Header Fields should be the same as those of the Header Fields of the corresponding Send Request.

Table 12.Y.1-1. Check Send Result Header Fields

Name	Values	Usage		Description
		User Agent	Origin Server	
Accept	media-type	M	M	The Acceptable Media Types of the response payload.

See also Section 8.4.

12.Y.1.4 Request Payload

The request shall have no payload.

12.Y.2 Behavior

The origin server shall provide the current result status of the Send Request in an Acceptable Media Type (see section 8.7.4).

12.Y.3 Response

The response shall have the following syntax:

```
version SP status-code SP reason-phrase CRLF
```

```
Content-Type: media-type CRLF
[retry-after CRLF]
*(header-field CRLF)
CRLF
[payload]
```

12.Y.3.1 Status Codes

Table 12.Y.3-1 shows some common status codes corresponding to this Transaction. See also Section 8.5 for additional status codes.

Table 12.Y.3-1. Status Code Meaning

Status	Code	Meaning
Success	200 (OK)	The origin server finished processing the send request identified by the supplied Transaction UID (see Section 12.X); the payload describes in detail what has been achieved.
	202 (Accepted)	The origin server has not finished processing the send request; the payload describes in detail what has been achieved until now. The user agent is expected to follow-up with a Check Send Result Transaction to get to know the entire result of the send request.
Failure	404 (Not Found)	The origin server cannot find the send request result identified by the supplied Transaction UID.
	410 (Gone)	The origin server can no longer provide the send request result identified by the supplied Transaction UID.
	503 (Service Unavailable)	The origin server cannot handle the Check Send Result request; this may be a temporary or permanent state.

Note

1. A 200 (OK) success status code should only be understood to mean that a Check Send Result response was returned by the origin server. The Check Send Result response may indicate that sending failed for some or even all of the matching SOP Instances.
2. The 404 (Not Found) status code may be caused by an incorrect Transaction UID that has been supplied by the user agent, or the origin server may have deleted the applicable result and not kept a record of it.
3. The 410 (Gone) status code may be caused by the origin server having a record of the Transaction UID but having deleted the request and/or applicable result.
4. When the 503 (Service Unavailable) status code is returned, the user agent might retry later with another Check Send Result Transaction.

12.Y.3.2 Response Header Fields

The Response Header Fields are the same as for the Send Transaction. See Section 12.X.3.2.

12.Y.3.3 Response Payload

The Response Payload is the same as for the Send Transaction. See Section 12.X.3.3.

Update Section 4 Symbols and Abbreviated Terms

4 Symbols and Abbreviated Terms

…

RESTful	A service implemented using the REST architecture.
SEND-RS	**Server-Enabled Network Delivery by RESTful Services**
SOP	Service Object Pair

…

Update Section B Examples: add new examples for the respective Send Transactions

B Examples (Informative)

…

B.X1 Request to Send all Studies of a Patient to Another Server

This example shows the flow of messages between the user agent and the origin server for the scenario in which 1) the user agent requests all studies of a certain patient be sent to another server, 2) the origin server tells the user agent to check for the result of this request later, returning the current status of the request, 3) the user agent checks for the result, and 4) the result provided by the origin server shows that it has sent all studies without any exceptions.

The patient in this example has three studies, an X-ray study with two Instances in one Series, an enhanced CT study with two series (each with one Instance), and an ultrasound study with one series with one enhanced multi-frame Instance. The origin server in this example typically sends a study using study level STOW requests, resulting in three requests: one per study. This scenario does, however, result in five sub-operations – being the number of Instances.

Step 1: the user agent sends a POST request with Transaction UID 1.1.99999.20250901 for patientID 11235813, the images to be sent to the host 'another.server' whose endpoint is https://another.server/stow:

```
POST /radiology/studies/send-requests/1.1.99999.20250901?destination=https%3A%2F%2Fanother.server/stow&PatientID=11235813 HTTP/1.1
Host: www.hospital-stmarco
Accept: application/dicom+json
```

Step 2: the origin server returns its response to the request, including a Send Request Response Module. In this scenario, there is no immediate result (return code 202 Accepted), and the server also notifies the user agent that it ought to wait at least 300 seconds before making a follow-up request for the result. In case where there is an immediate result, the response would be as shown below in step 4, effectively skipping steps 2 and 3.

```
HTTP/1.1 202 Accepted
Content-Length: 247
Content-Type: application/dicom+json; charset=utf-8
Retry-After: 300

[ { "00000900": { "vr": "US", "Value": [ 65280 ] }
  , "00001020": { "vr": "US", "Value": [ 5 ] }
  , "00001021": { "vr": "US", "Value": [ 0 ] }
  , "00001022": { "vr": "US", "Value": [ 0 ] }
  , "00001023": { "vr": "US", "Value": [ 0 ] }
} ]
```

According to Table @.1-1, which defines the Send Request Response Module, this shows that:

- the request is pending – value 65280 for tag (0000,0900) is value 0xFF00 for the Status, which is, according to PS3.4, Table 4.2-1, the Pending Service Status;
- the Number of Remaining Sub-operations, tag (0000,1020), is 5;
- the Number of Completed Sub-operations, tag (0000,1021), is 0;
- the Number of Failed Sub-operations, tag (0000,1022), is 0;
- the Number of Warning Sub-operations, tag (0000,1023), is 0.

Step 3: after waiting 300 seconds, the suggested period of time to retry, the user agent GETs the status of the request using the same Transaction UID as the original request:

```
GET /radiology/studies/send-requests/1.1.99999.20250901 HTTP/1.1
Host: www.hospital-stmarco
Accept: application/dicom+json
```

Step 4: the origin server returns the response of the Check Send Result request, which contains the result of the original Send Request. Note that in case the server initially responds to the POST request of step 1 with the HTTP response status code 200 (the synchronous case) the same result would be returned:

```
HTTP/1.1 200 OK
Content-Length: 195
Content-Type: application/dicom+json; charset=utf-8
Retry-After: 300

[ { "00000900": { "vr": "US", "Value": [ 0 ] }
  , "00001021": { "vr": "US", "Value": [ 5 ] }
  , "00001022": { "vr": "US", "Value": [ 0 ] }
  , "00001023": { "vr": "US", "Value": [ 0 ] }
} ]
```

This response shows – see for details Table @.1-1 or earlier in this scenario – that the send request has been handled successfully, and that it resulted in 5 completed sub-operations, with no warnings and failures. The three studies were sent, although that number is not reflected in the response.

B.X2 Flow: Handling of a Send Request

The flow in Figure B.X2-1 shows how a Send Request can be handled by an origin server. In this case, the example of Annex B.X1 has been used; however, here the internals of the Origin Server are elaborated. It is also shown that the origin server may choose between DIMSE and DICOMweb for its sub-operations and it may choose to respond with one or more results. Note that multiple responses are recommended for long-running Transactions.

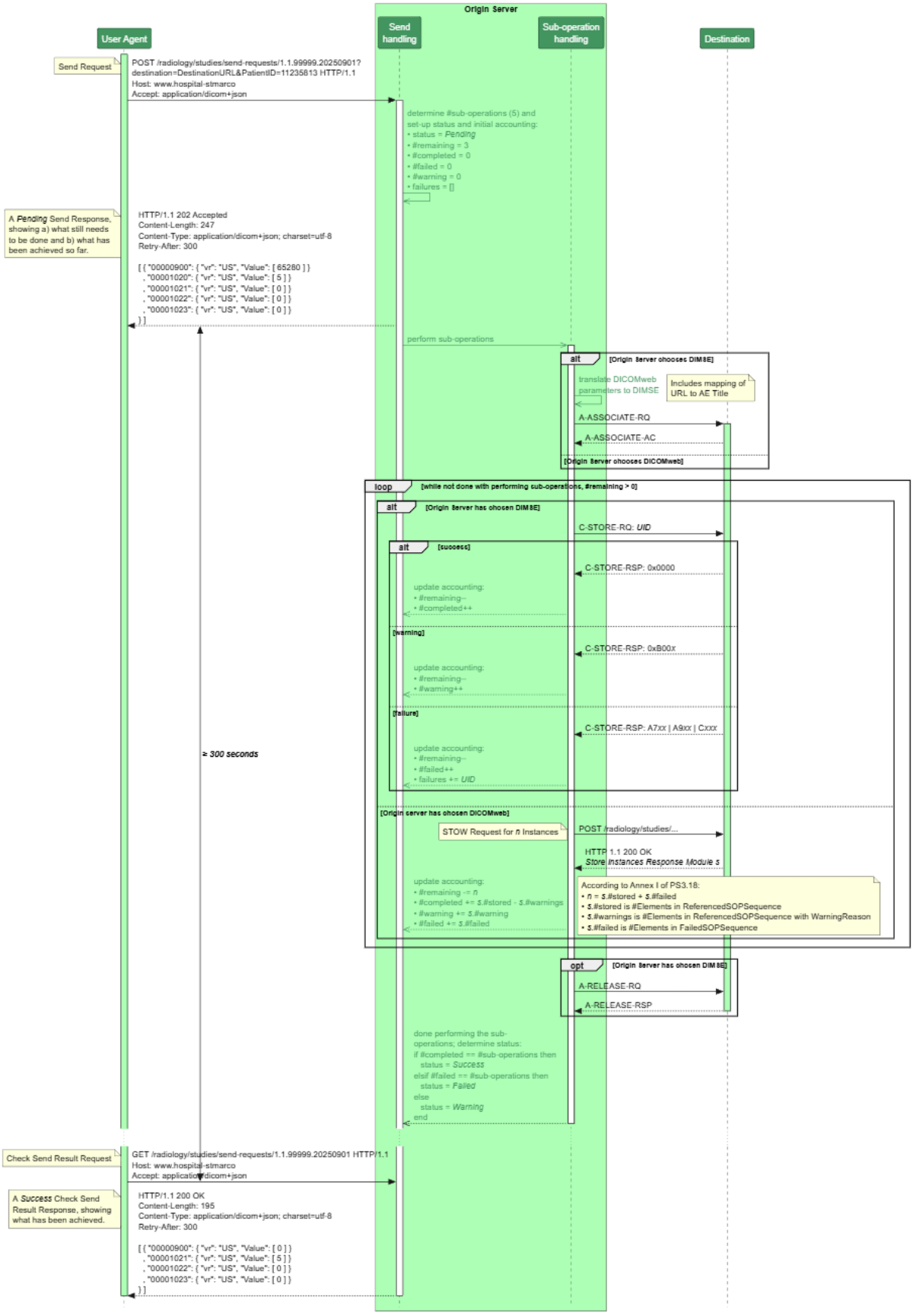

Origin Server
User Agent
Send handling
Sub-operation handling
Destination
Send Request
POST /radiology/studies/send-requests/1.1.99999.20250901?destination=DestinationURL&PatientID=11235813 HTTP/1.1
Host: www.hospital-stmarco
Accept: application/dicom+json
determine #sub-operations (5) and set-up status and initial accounting:
• status = Pending
• #remaining = 3
• #completed = 0
• #failed = 0
• #warning = 0
• failures = []
A Pending Send Response, showing a) what still needs to be done and b) what has been achieved so far.
HTTP/1.1 202 Accepted
Content-Length: 247
Content-Type: application/dicom+json; charset=utf-8
Retry-After: 300
[{ "00000900": { "vr": "US", "Value": [65280] }
, "00001020": { "vr": "US", "Value": [5] }
, "00001021": { "vr": "US", "Value": [0] }
, "00001022": { "vr": "US", "Value": [0] }
, "00001023": { "vr": "US", "Value": [0] }
}]
perform sub-operations
alt
[Origin Server chooses DIMSE]
translate DICOMweb parameters to DIMSE
Includes mapping of URL to AE Title
A-ASSOCIATE-RQ
A-ASSOCIATE-AC
[Origin Server chooses DICOMweb]
loop
[while not done with performing sub-operations, #remaining > 0]
alt
[Origin Server has chosen DIMSE]
C-STORE-RQ: UID
alt
[success]
C-STORE-RSP: 0x0000
update accounting:
• #remaining--
• #completed++
[warning]
C-STORE-RSP: 0xB00x
update accounting:
• #remaining--
• #warning++
[failure]
C-STORE-RSP: A7xx | A9xx | Cxxx
update accounting:
• #remaining--
• #failed++
• failures += UID
[Origin server has chosen DICOMweb]
STOW Request for n Instances
POST /radiology/studies/...
HTTP 1.1 200 OK
Store Instances Response Module s
update accounting:
• #remaining -= n
• #completed += s.#stored - s.#warnings
• #warning += s.#warning
• #failed += s.#failed
According to Annex I of PS3.18:
• n = s.#stored + s.#failed
• s.#stored is #Elements in ReferencedSOPSequence
• s.#warnings is #Elements in ReferencedSOPSequence with WarningReason
• s.#failed is #Elements in FailedSOPSequence
opt
[Origin Server has chosen DIMSE]
A-RELEASE-RQ
A-RELEASE-RSP
done performing the sub-operations; determine status:
if #completed == #sub-operations then
status = Success
elsif #failed == #sub-operations then
status = Failed
else
status = Warning
end
≥ 300 seconds
Check Send Result Request
GET /radiology/studies/send-requests/1.1.99999.20250901 HTTP/1.1
Host: www.hospital-stmarco
Accept: application/dicom+json
A Success Check Send Result Response, showing what has been achieved.
HTTP/1.1 200 OK
Content-Length: 195
Content-Type: application/dicom+json; charset=utf-8
Retry-After: 300
[{ "00000900": { "vr": "US", "Value": [0] }
, "00001021": { "vr": "US", "Value": [5] }
, "00001022": { "vr": "US", "Value": [0] }
, "00001023": { "vr": "US", "Value": [0] }
}]

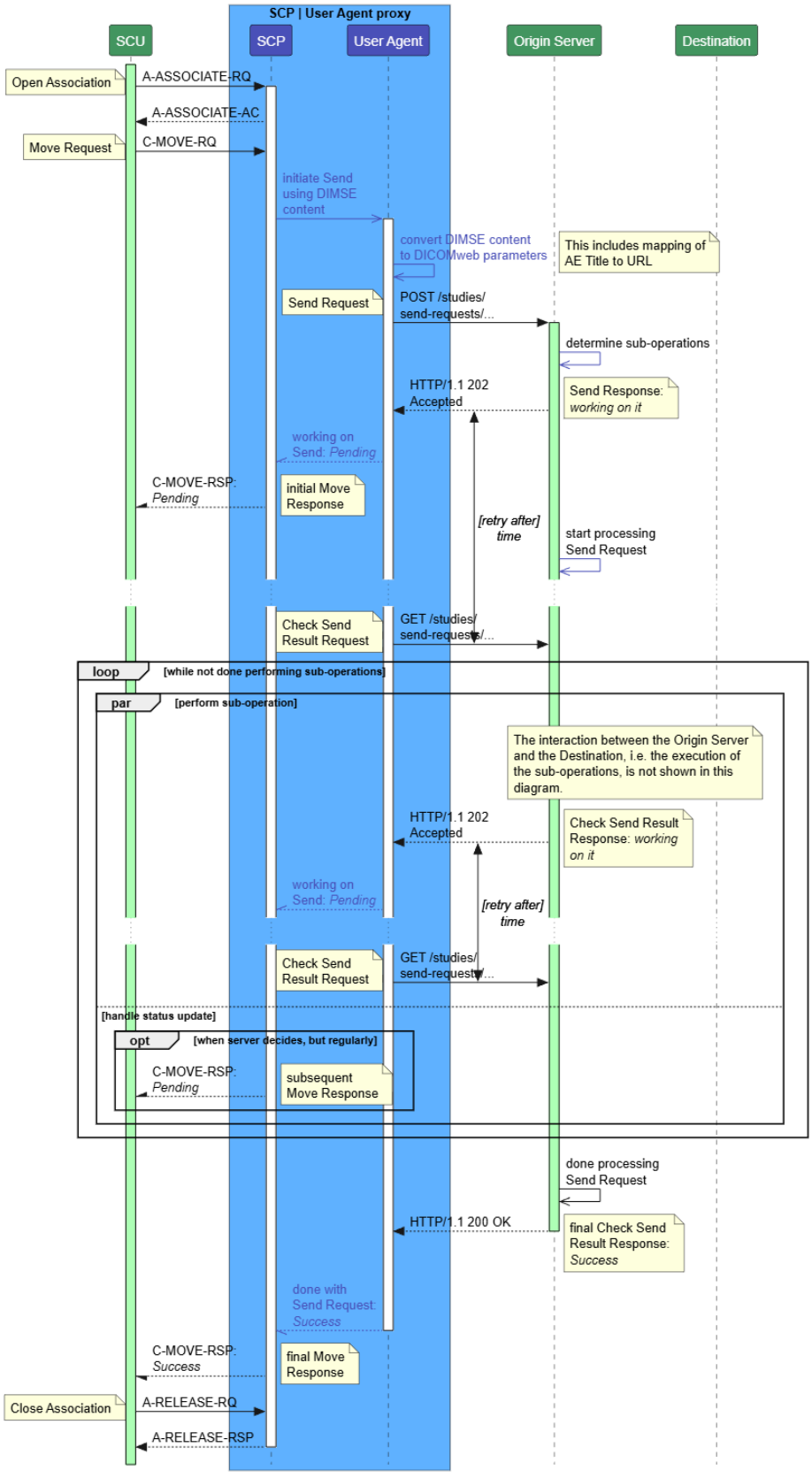
Figure B.X2-1 Handling a Send Request

B.X3 Bi-Directional Proxy for Send Transaction

The DICOMweb Send Transaction may be deployed in a hybrid environment, i.e., an environment in which both DICOMweb and DIMSE are used. In such a hybrid environment, a proxy can broker Transactions from one service to the other, allowing a DICOMweb Send origin server or a DIMSE C-MOVE SCP to support a mixed set of DICOMweb user agents and DIMSE SCUs.

DICOM does not mandate implementation of proxies; however, since they would be very useful in a hybrid environment, the examples in this section show how this could be done. It is the implementer's responsibility to match the possibly asynchronous DIMSE behavior with the polling DICOMweb behavior, for example management of Transaction UIDs.

Figure B.X3-1 shows how a proxy could facilitate a C-MOVE request from a DIMSE SCU to a DICOMweb origin server; the diagram does not show the sub-operations, as the emphasis is on proxying, not on the entire server behavior.



552

553 **Figure B.X3-1. Send DIMSE Proxy for a DICOMweb Origin Server**

Figure B.X3-2 shows how a proxy could facilitate a Send Request from a DICOMweb user agent to a DIMSE SCP; the diagram does not show the Sub-transactions, as the emphasis is on proxying, not on the entire server behavior. When proxying in this direction, the proxy will receive information from the SCP that it is not able to dispatch immediately to the user agent.

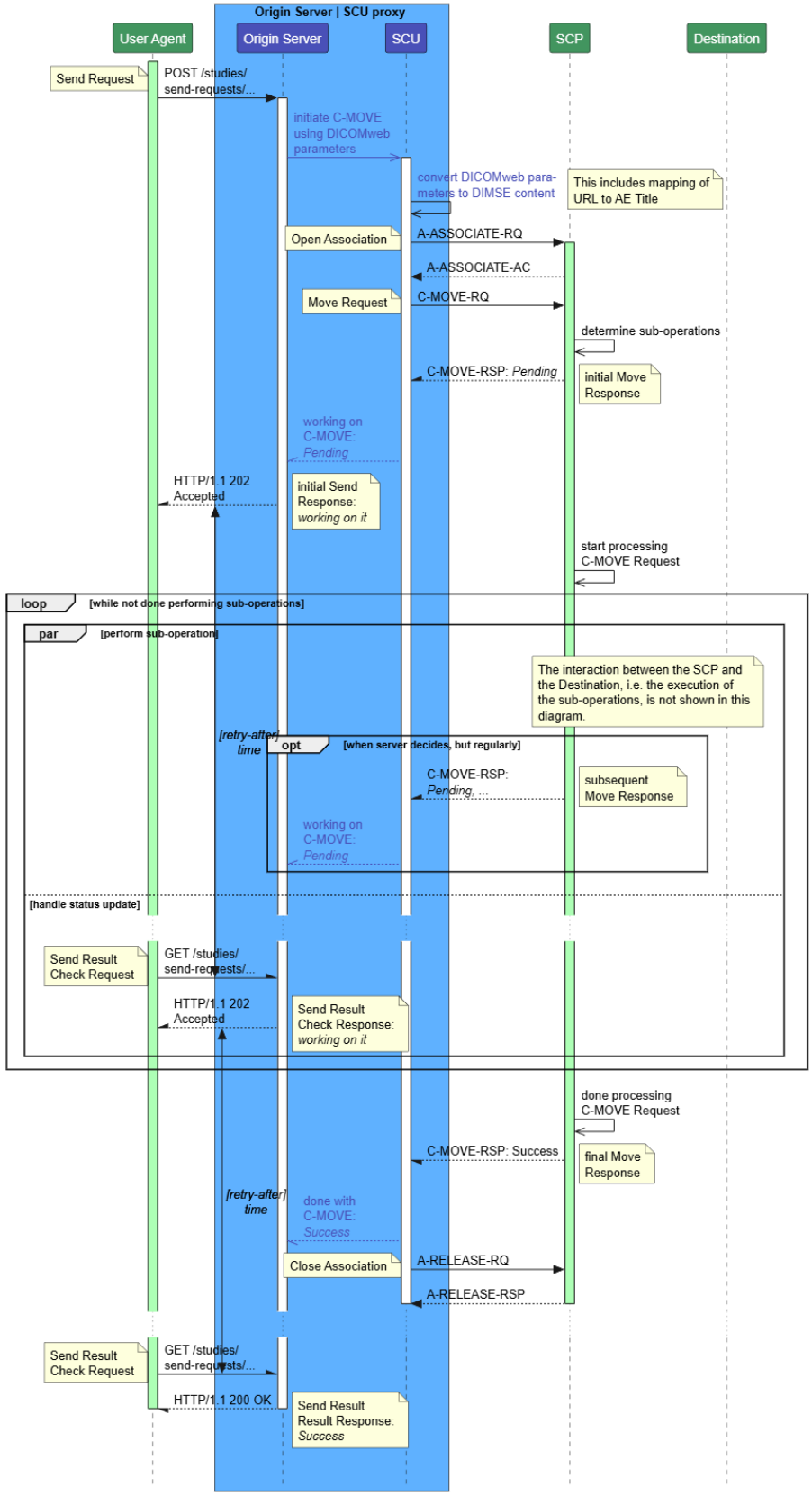


Figure B.X3-2. Send DICOMweb Proxy for a DIMSE SCP

Update Table H-1 Resources and Methods: add new resources and methods for Send Transactions

H Capabilities Description

Service	Resource	Transactions	Reference
Studies (see Section 10.1.1)			
	studies	Search for Studies	Section 10.6
		Store Instances	Section 10.5
		Send Studies	**Section 10.X**
		Check Send Result for Studies	**Section 10.Y**
	{StudyInstance}	Retrieve Study	Section 10.4
		Store Study Instances	Section 10.5
		Send Study	**Section 10.X**
		Check Send Result for a Study	**Section 10.Y**
	metadata	Retrieve Study Metadata	Section 10.4
	rendered	Retrieve Rendered Study	Section 10.4
	renderedmpr	Retrieve Rendered MPR Volume Study	Section 10.4
	rendered3d	Retrieve Rendered 3D Volume Study	Section 10.4
	thumbnail	Retrieve Study Thumbnail	Section 10.4
	bulkdata	Retrieve Study Bulkdata	Section 10.4
	pixeldata	Retrieve Study Pixel Data	Section 10.4
	series	Search for Study Series	Section 10.6
		Send Series	**Section 10.X**
		Check Send Result for Series	**Section 10.Y**
	{SeriesInstance}	Retrieve Series	Section 10.4
		Send Series	**Section 10.X**
		Check Send Result for a Series	**Section 10.Y**
	metadata	Retrieve Series Metadata	Section 10.4
	rendered	Retrieve Rendered Series	Section 10.4

Service	Resource	Transactions	Reference
	renderedmpr	Retrieve Rendered MPR Volume Series	Section 10.4
	rendered3d	Retrieve Rendered 3D Volume Series	Section 10.4
	thumbnail	Retrieve Series Thumbnail	Section 10.4
	bulkdata	Retrieve Series Bulkdata	Section 10.4
	pixeldata	Retrieve Series Pixel Data	Section 10.4
	instances	Search for Study Series Instances **Send Instances** **Check Send Result for Instances**	Section 10.4 **Section 10.X** **Section 10.Y**
	{SOPInstance}	Retrieve Instance **Send Instance** **Check Send Result for an Instance**	Section 10.4 **Section 10.X** **Section 10.Y**
	metadata	Retrieve Instance Metadata	Section 10.4
	rendered	Retrieve Rendered Instance	Section 10.4
	renderedmpr	Retrieve Rendered MPR Volume Instance	Section 10.4
	rendered3d	Retrieve Rendered 3D Volume Instance	Section 10.4
	thumbnail	Retrieve Instance Thumbnail	Section 10.4
	bulkdata	Retrieve Instance Bulkdata	Section 10.4
	pixeldata	Retrieve Instance Pixel Data	Section 10.4
	frames	N/A	N/A
	{framelist}	Retrieve Frames	Section 10.4
	rendered	Retrieve Rendered Frames	Section 10.4
	renderedmpr	Retrieve Rendered MPR Volume Frames	Section 10.4
	rendered3d	Retrieve Rendered 3D Volume Frames	Section 10.4
	thumbnail	Retrieve Frame Thumbnail	Section 10.4
	pixeldata	Retrieve Frame Pixel Data	Section 10.4
	instances	Search for Study Instances	Section 10.6
	series	Search for Series	Section 10.6

Service	Resource	Transactions	Reference
	{SeriesInstance}	N/A	N/A
	{instances}	Search for Instances	Section 10.6
	instances	Search for Instances	Section 10.6
	{BulkDataReference}	Retrieve Bulkdata	Section 10.4
Worklist (see Section 11.1.1)			
	workitems	Search for Workitem Create Workitem	Section 11.9 Section 11.4
	{Workitem}	Retrieve Workitem Update Workitem	Section 11.4 Section 11.6
	state	Change Workitem State	Section 11.7
	cancelrequest	Request Workitem Cancellation	Section 11.8
	subscribers	N/A	N/A
	{AETitle}	Subscribe Unsubscribe	Section 11.10 Section 11.11
	1.2.840.10008.5.1.4.34.5	N/A	N/A
	subscribers	N/A	N/A
	{AETitle}	Subscribe Unsubscribe	Section 11.10 Section 11.11
	suspend	Unsubscribe	Section 11.11
	1.2.840.10008.5.1.4.34.5.1	N/A	N/A
	subscribers	N/A	N/A
	{AETitle}	Subscribe Unsubscribe	Section 11.10 Section 11.11
	suspend	Suspend Worklist Subscription	Section 11.11
Non-Patient Instance (see Section 12.1.1)			
	color-palettes	N/A	N/A
	{uid}	Retrieve	Section 12.4

Service	Resource	Transactions	Reference
		Store	Section 12.5
		Search	Section 12.6
		Send	**Section 12.X**
		Check Send Result	**Section 12.Y**
	defined-procedure-protocols	N/A	N/A
	{uid}	Retrieve	Section 12.4
		Store	Section 12.5
		Search	Section 12.6
		Send	**Section 12.X**
		Check Send Result	**Section 12.Y**
	hanging-protocols	N/A	N/A
	{uid}	Retrieve	Section 12.4
		Store	Section 12.5
		Search	Section 12.6
		Send	**Section 12.X**
		Check Send Result	**Section 12.Y**
	implant-templates	N/A	N/A
	{uid}	Retrieve	Section 12.4
		Store	Section 12.5
		Search	Section 12.6
		Send	**Section 12.X**
		Check Send Result	**Section 12.Y**
	inventories	N/A	N/A
	{uid}	Retrieve	Section 12.4
		Store	Section 12.5
		Search	Section 12.6
		Send	**Section 12.X**
		Check Send Result	**Section 12.Y**
	protocol-approvals	N/A	N/A

Service	Resource	Transactions	Reference
		Retrieve	Section 12.4
		Store	Section 12.5
		Search	Section 12.6
		Send	**Section 12.X**
		Check Send Result	**Section 12.Y**
Storage Commitment Requests (see Section 13.1.1)			
	commitment-requests	Request	Section 13.4
		Result Check	Section 13.5
…			

563

564

Add new Annex @ – Send Request Response Module

Annex @ Send Request Response Module

@.1 Response Message Body

Table @.1-1 specifies the Attributes for of the Send Request Response Module; this is similar to what has been described for DIMSE's C-MOVE in PS3.4, Section C.4.2.1.

Table @.1-1. Send Request Response Module Attributes

Attribute Name	Tag	Type	Attribute Description
Status	(0000,0900)	1	The status of the associated sub-operations performed to this point. The status value shall be populated as described in PS3.4, Section C.4.2.3.1, using values as defined in PS3.4, Section C.4.2.1.5. Note: DICOMweb Send does not support Cancel.
Number of Remaining Sub-operations	(0000,1020)	1C	The number of store sub-operations associated with this Send Request that are remaining. Required when the Status (0000,0900) is Pending.
Number of Completed Sub-operations	(0000,1021)	1	The number of store sub-operations associated with this Send Request that yielded Success.
Number of Failed Sub-operations	(0000,1022)	1	The number of store sub-operations associated with this Send Request that yielded Failure.
Number of Warning Sub-operations	(0000,1023)	1	The number of store sub-operations associated with this Send Request that yielded Warning.
Failed SOP Instance UID List	(0008,0058)	1C	The list of UIDs of SOP Instances for which the associated sub-operation yielded Failure. Required when Status (0000,0900) is not Pending and Number of Failed Sub-operations (0000,1022) is not zero.

Note Group 0000 are used in DIMSE commands and not permitted in DICOM Instances. The above Tags are used in this Send request response to match the semantics of DIMSE's C-MOVE response message.

Changes to NEMA Standards Publications PS 3.4

Adapt Section C.4.2 such that DIMSE's C-MOVE operation can be used in combination with STOW-RS and SEND-RS. The entire Section is given here for reference, while only the Alternative Sub-operation Mechanism Option and Proxy Option have been added for the SCP

C.4.2 C-MOVE Operation

SCUs of some SOP Classes of the Query/Retrieve Service Class may generate retrievals using the C-MOVE operation as described in PS3.7. The C-MOVE operation allows an application entity to instruct another application entity to transfer stored SOP Instances to another application entity using the C-STORE operation. Support for the C-MOVE service shall be agreed upon at Association establishment time by both the SCU and SCP of the C-MOVE in order for a C-MOVE operation to occur over the Association. The C-STORE sub-operations shall always be accomplished over an Association different from the Association that accomplishes the C-MOVE operation. Hence, the SCP of the Query/Retrieve Service Class serves as the SCU of the Storage Service Class.

Note

The application entity that receives the stored SOP Instances may or may not be the originator of the C-MOVE operation.

A C-MOVE request may be performed to any level of the Query/Retrieve Information Model. However, the transfer of stored SOP Instances may not be performed at this level. The level at which the transfer is performed depends upon the SOP Class (see Section C.6).

C.4.2.1 C-MOVE Service Parameters

C.4.2.1.1 SOP Class UID

The SOP Class UID identifies the Query/Retrieve Information Model against which the C-MOVE is to be performed. Support for the SOP Class UID is implied by the Abstract Syntax UID of the Presentation Context used by this C-MOVE operation.

C.4.2.1.2 Priority

The Priority Attribute defines the requested priority of the C-MOVE operation and corresponding C-STORE sub-operations with respect to other DIMSE operations being performed by the same SCP.

Processing of priority requests is not required of SCPs. Whether or not an SCP supports priority processing, and the meaning of the different priority levels shall be stated in the Conformance Statement of the SCP. The same priority shall be used for all C-STORE sub-operations.

C.4.2.1.3 Move Destination

Move Destination specifies the Application Entity Title of the receiver of the C-STORE sub-operations.

C.4.2.1.4 Identifier

The C-MOVE request shall contain an Identifier. The C-MOVE response shall conditionally contain an Identifier as required in Section C.4.2.1.4.2.

Note

The Identifier is specified as U in the definition of the C-MOVE primitive in PS3.7 but is specialized for use with this service.

C.4.2.1.4.1 Request Identifier Structure

An Identifier in a C-MOVE request shall contain:

- Query/Retrieve Level (0008,0052), which defines the level of the retrieval
- Unique Key Attributes, which may include Patient ID (0010,0020), Study Instance UIDs (0020,000D), Series Instance UIDs (0020,000E), and the SOP Instance UIDs (0008,0018)

- Conditionally, the Attribute Query/Retrieve View (0008,0053). This Attribute may be included if Enhanced Multi-Frame Image Conversion has been accepted during Association Extended Negotiation. It shall not be included otherwise.

Specific Character Set (0008,0005) shall be present if Patient ID (0010,0020) is using a character set other than the Default Character Repertoire.

The Unique Keys at each level of the hierarchy and the values allowable for the level of the retrieval shall be defined in the SOP Class definition for the Query/Retrieve Information Model.

Note

1. In the non-Relational behavior, more than one entity may be retrieved if the Query/Retrieve Level is IMAGE, SERIES or STUDY, using List of UID matching, but only Single Value Matching value may be specified for Patient ID (0010,0020).

2. The issuer of the Patient ID (0010,0020) is implicit; there is no provision to send the Issuer of Patient ID (0010,0021). When there is a possibility of ambiguity of the Patient ID (0010,0020) value, a STUDY level retrieval should be used instead of a PATIENT level retrieval.

C.4.2.1.4.2 Response Identifier Structure

The Failed SOP Instance UID List (0008,0058) specifies a list of UIDs of the C-STORE sub-operation SOP Instances for which this C-MOVE operation has failed. An Identifier in a C-MOVE response shall conditionally contain the Failed SOP Instance UID List (0008,0058) based on the C-MOVE response status value. If no C-STORE sub-operation failed, Failed SOP Instance UID List (0008,0058) is absent and therefore no Data Set shall be sent in the C-MOVE response.

Specific Character Set (0008,0005) shall not be present.

The Identifier in a C-MOVE response with a status of:

- Cancel, Failure, or Warning shall contain the Failed SOP Instance UID List Attribute

- Pending shall not contain the Failed SOP Instance UID List Attribute (no Data Set)

C.4.2.1.5 Status

Table C.4-2 defines the specific Status Code values that might be returned in a C-MOVE response. General Status Code values and fields related to Status Code values are defined for C-MOVE DIMSE Service in PS3.7.

Table C.4-2. C-MOVE Response Status Values

Service Status	Further Meaning	Status Codes	Related Fields
Failure	Refused: Out of resources - Unable to calculate number of matches	A701	(0000,0902)
	Refused: Out of resources - Unable to perform sub-operations	A702	(0000,1021) (0000,1022) (0000,1023)
	Refused: Move Destination unknown	A801	(0000,0902)
	Error: Data Set does not match SOP Class	A900	(0000,0901) (0000,0902)
	Failed: Unable to process	Cxxx	(0000,0901) (0000,0902)
Cancel	Sub-operations terminated due to Cancel Indication	FE00	(0000,1020)

Service Status	Further Meaning	Status Codes	Related Fields
			(0000,1021) (0000,1022) (0000,1023)
Warning	Sub-operations Complete - One or more Failures	B000	(0000,1021) (0000,1022) (0000,1023)
Success	Sub-operations Complete - No Failures	0000	(0000,1021) (0000,1022) (0000,1023)
Pending	Sub-operations are continuing	FF00	(0000,1020) (0000,1021) (0000,1022) (0000,1023)

Some Failure Status Codes are implementation specific.

An SCP implementation shall assign specific Failure Status Codes by replacing each 'x' symbol with a hexadecimal digit in the range from 0 to F. An SCP implementation wishing to differentiate between causes of "Failed: Unable to process" Failure Meaning shall assign those causes specific Status Code Values within valid range specified in Table C.4-2.

An SCU implementation shall recognize any Failure Status Code within the value range specified in Table C.4-2 as an indicator of the Failure Meaning stated in the table. There is no requirement for an SCU implementation to differentiate between specific Status Codes within the valid range.

C.4.2.1.6 Number of Remaining Sub-Operations

Inclusion of the Number of Remaining Sub-operations is conditional based upon the status in the C-MOVE response. The Number of Remaining Sub-operations specifies the number of Remaining C-STORE sub-operations necessary to complete the C-MOVE operation.

A C-MOVE response with a status of:

- Pending shall contain the Number of Remaining Sub-operations Attribute
- Cancel may contain the Number of Remaining Sub-operations Attribute
- Warning, Failure, or Success shall not contain the Number of Remaining Sub-operations Attribute

C.4.2.1.7 Number of Completed Sub-Operations

Inclusion of the Number of Completed Sub-operations is conditional based upon the status in the C-MOVE response. The Number of Completed sub-operations specifies the number of C-STORE sub-operations generated by the requested transfer that have completed successfully.

A C-MOVE response with a status of:

- Pending shall contain the Number of Completed Sub-operations Attribute
- Cancel, Warning, Failure, or Success may contain the Number of Completed Sub-operations Attribute

C.4.2.1.8 Number of Failed Sub-Operations

Inclusion of the Number of Failed Sub-operations is conditional based upon the status in the C-MOVE response. The Number of Failed sub-operations specifies the number of C-STORE sub-operations generated by the requested transfer that have failed.

A C-MOVE response with a status of:

- Pending shall contain the Number of Failed Sub-operations Attribute
- Cancel, Warning, Failure, or Success may contain the Number of Failed Sub-operations Attribute

C.4.2.1.9 Number of Warning Sub-Operations

Inclusion of the Number of Warning Sub-operations is conditional based upon the status in the C-MOVE response. The Number of Warning sub-operations specifies the number of C-STORE sub-operations generated by the requested transfer that had a status of warning.

A C-MOVE response with a status of:

- Pending shall contain the Number of Warnings Sub-operations Attribute
- Cancel, Warning, Failure, or Success may contain the Number of Warning Sub-operations Attribute

C.4.2.2 C-MOVE SCU Behavior

This Section discusses both the baseline and extended behavior of the C-MOVE SCU.

C.4.2.2.1 Baseline Behavior of SCU

An SCU conveys the following semantics with a C-MOVE request:

- The SCU shall supply a single value in the Unique Key Attribute for each level above the Query/Retrieve level. For the level of retrieve, the SCU shall supply a single value for one unique key if the level of retrieve is above the STUDY level and shall supply one UID, or a list of UIDs if a retrieval of several items is desired and the retrieve level is STUDY, SERIES or IMAGE. The SCU shall also supply a move destination. The move destination shall be the DICOM Application Entity Title of a DICOM Application Entity capable of serving as the SCP of the Storage Service Class.
- The SCU shall interpret responses to the C-MOVE with status equal to Pending during the processing of the C-STORE sub-operations. These responses shall indicate the number of Remaining, Completed, Failed, and Warning C-STORE sub-operations.
- The SCU shall interpret responses with a status equal to Success, Warning or Failure as final responses. The final response shall indicate the number of Successful C-STORE sub-operations and the number of Failed C-STORE sub-operations resulting from the C-MOVE operation. The SCU shall interpret a status of:
  - Success to indicate that all sub-operations were successfully completed
  - Warning to indicate one or more sub-operations were successfully completed and one or more sub-operations were unsuccessful or had a status of warning, or all sub-operations had a status of warning
  - Failure to indicate all sub-operations were unsuccessful.
- The SCU may cancel the C-MOVE service by issuing a C-CANCEL-MOVE request at any time during the processing of the C-MOVE. The SCU shall interpret a C-MOVE response with a status of Cancel to indicate the transfer was canceled. The C-MOVE response with a status of Cancel shall contain the number of Completed, Failed, and Warning C-STORE sub-operations. If present, the Remaining sub-operations count shall contain the number of C-STORE sub-operations that were not initiated due to the C-CANCEL-MOVE request.

C.4.2.2.2 Extended Behavior of SCU

Extended SCU behavior shall be negotiated at Association establishment time. If an option within the extended behavior is not agreed upon in the negotiation, then only baseline SCU behavior shall be performed with respect to that option. Extended SCU behavior includes all baseline behavior with the following option:

- Relational-retrieve

- Enhanced Multi-Frame Image Conversion

More than one option may be agreed upon.

C.4.2.2.2.1 Relational-Retrieve

The C-MOVE Service with relational-retrieve removes the restriction that the SCU supply Unique Key values for levels above the Query/Retrieve level to identify an entity at the level of the retrieval. Hence, the Identifier of a C-MOVE request may transfer:

- all Composite Object Instances related to a study by only providing a Study Instance UID (0020,000D)
- all Composite Object Instances related to a series by only providing a Series Instance UID (0020,000E)
- individual Composite Object Instances by only providing a list of SOP Instance UIDs (0008,0018)

C.4.2.2.2.2 Enhanced Multi-Frame Image Conversion

The C-MOVE Service with Enhanced Multi-Frame Image Conversion allows for selection of the default or an alternative view of the instances represented by the Information Model, and hence the retrieval of either the legacy or the converted images, together with any unconverted instances, all of which are required to be processed to maintain referential integrity within the scope of the Patient.

Support for Enhanced Multi-Frame Image Conversion allows the SCU to specify the Attribute Query/Retrieve View (0008,0053) in the request Identifier with a value of either "CLASSIC" or "ENHANCED".

If Query/Retrieve View (0008,0053) is not present in the request Identifier, then the SCU requests that the SCP provide all the requested instances it possesses, as received.

If Query/Retrieve View (0008,0053) is present with a value of "CLASSIC", then the SCU requests that the SCP provide all the Classic single frame Instances (converted from Enhanced multi-frame Instances if required), as well as any instances that were converted to preserve referential integrity, and any that did not need to be converted.

If Query/Retrieve View (0008,0053) is present with a value of "ENHANCED", then the SCU requests that the SCP provide all the Enhanced multi-frame Instances (converted from Classic single frame Instances if required), as well as any instances that were converted to preserve referential integrity, and any that did not need to be converted.

Note

1. The SCU may assume that no duplicate information will be provided. For example, if an entire series of single frame instances can be converted to a separate series of converted instances, a STUDY level C-MOVE will not provide both series.

2. The Query Information Model is unchanged, and the same unique keys are equally applicable to both views, except that the values for the SERIES and IMAGE level queries will be different and will depend on the converted instance content.

3. The Query/Retrieve View is still required in an IMAGE or SERIES level request Identifier, even though the requested unique key(s) are unambiguous, and the view is in a sense "redundant", because the conversion that created the requested instances may not have been executed yet. It is not permitted to specify a view that is inconsistent with the requested unique key(s).

C.4.2.3 C-MOVE SCP Behavior

This section discusses both the baseline and extended behavior of the C-MOVE SCP.

C.4.2.3.1 Baseline Behavior of SCP

An SCP conveys the following semantics with a C-MOVE response:

- The SCP shall identify a set of Entities at the level of the transfer based upon the values in the Unique Keys in the Identifier of the C-MOVE request. The SCP shall initiate C-STORE sub-operations for the corresponding storage SOP Instances. These C-STORE sub-operations shall occur on a different Association (that may already exist) from the C-MOVE operation. The SCP of the Query/Retrieve Service Class shall serve as an SCU of the Storage Service Class.

- The SCP shall either reuse an established and compatible Association or establish a new Association for the C-STORE sub-operations. The SCP shall initiate C-STORE sub-operations over that Association for all stored SOP

Instances related to the Patient ID, List of Study Instance UIDs, List of Series Instance UIDs, or List of SOP Instance UIDs depending on the Query/Retrieve level specified in the C-MOVE request. A sub-operation is considered a Failure if the SCP is unable to negotiate an appropriate presentation context for a given stored SOP Instance.

- Optionally, the SCP may generate responses to the C-MOVE with status equal to Pending during the processing of the C-STORE sub-operations. These responses shall indicate the Remaining, Completed, Failed, and Warning C-STORE sub-operations.

- When the number of Remaining sub-operations reaches zero, the SCP shall generate a final response with a status equal to Success, Warning or Failure. This response shall indicate the number of Completed sub-operations, the number of Failed sub-operations, and the number of sub-operations with Warning Status. The status contained in the C-MOVE response shall contain:

  - Success if all sub-operations were successfully completed

  - Warning if one or more sub-operations were successfully completed and one or more sub-operations were unsuccessful or had a warning status

  - Warning if all sub-operations had a warning status

  - Failure if all sub-operations were unsuccessful

- The SCP may receive a C-CANCEL-MOVE request at any time during the processing of the C-MOVE. The SCP shall interrupt all C-STORE sub-operation processing and return a status of Cancel in the C-MOVE response. The C-MOVE response with a status of Cancel shall contain the number of Completed, Failed, and Warning C-STORE sub-operations. If present, the Remaining sub-operations count shall contain the number of C-STORE sub-operations that were not initiated due to the C-CANCEL-MOVE request.

- If the SCP manages images in multiple alternate encodings (see Section C.6.1.1.5.1), only one of the alternate encodings of an image shall be included in the set of object instances retrieved by a C-MOVE request at the Patient, Study, or Series level.

  Note

  For retrieval of images with alternate encodings using a C-MOVE request at the Patient, Study, or Series level, the SCP may select the appropriately encoded Instance for the retrieval based on identity of the SCU, transfer syntaxes accepted in the C-STORE Association Negotiation, or other factors.

Note

If the association on which the C-MOVE operation was issued is abnormally terminated, then it will not be possible to issue any further pending responses nor a final response, nor will C-CANCEL-MOVE requests be received. The behavior of the C-MOVE SCP acting as a C-STORE SCU is undefined in this condition. Specifically, whether or not any uncompleted C-STORE sub-operations continue is undefined.

C.4.2.3.1.1 Alternative Sub-operation Mechanism Option

When this Option is supported, a C-MOVE SCP may, at its discretion, satisfy the storage sub-operations of a C-MOVE by acting as a user agent in a Store Transaction, issuing one or more Store requests to the specific DICOMweb endpoint determined by a local policy (e.g., by mapping the provided AE Title to its endpoint equivalent). See Sections 10.5 and 12.5 in PS3.18 for more information about the Store Transaction.

C.4.2.3.1.1.1 Scope

This Option applies to all Query/Retrieve Information Model – MOVE SOP Classes.

C.4.2.3.1.1.2 Interface

This option does not change the interface between the C-MOVE SCU and C-MOVE SCP. Regarding the destination, the C-MOVE SCU continues to supply only a Move Destination AE Title in the C-MOVE request; it does not supply URLs, credentials, or delivery preferences. The C-MOVE SCP selects the delivery mechanism, taking the capabilities of the destination into account.

C.4.2.3.1.1.3 Behavior

When the C-MOVE SCP chooses to utilize Store transactions under this Option, storage is performed using HTTP messages.

The C-MOVE SCP shall comply with section C.4.2.3 changing C-STORE operation to DICOMweb STOW while maintaining the same semantics and per-instance sub-operation bookkeeping.

Note Good idea to provide user identification [Rob].

C.4.2.3.1.1.4 Sub-operation Counters

The C-MOVE SCP shall maintain four counters and treat them as mutually exclusive categories for per-instance outcomes:

a) **Completed = stored with no warnings;**
b) **Warning = stored with warnings;**
c) **Failed = not stored;**
d) **Remaining = not yet attempted or not yet known.**

The sum of Completed, Warning, Failed, and Remaining shall always equal the number of Composite SOP Instances matching the Identifier of the C-MOVE request.

When using Store transactions, the C-MOVE SCP shall count the per-instance outcome from the Store Instance Response Module (see PS3.18, Annex I) as follows:

- **SOP Instance UIDs in the Referenced SOP Sequence (0008,1199) that have no associated Warning Reason (0008,1196) shall increment the Completed counter;**
- **SOP Instance UIDs in the Referenced SOP Sequence (0008,1199) having an associated Warning Reason (0008,1196) shall increment the Warning counter;**
- **SOP Instance UIDs in the Failed SOP Sequence (0008,1198) shall increment the Failed counter.**

All SOP Instance UIDs in the Failed SOP Sequence (0008,1198) of the Store Instance Response Module shall be added to the Failed SOP Instance UID List (0008,0058) of the C-MOVE Response Identifier.

C.4.2.3.1.1.5 Mixed-mechanism Delivery

The C-MOVE SCP may perform some sub-operations via Store transactions and others via C-STORE. It shall maintain a single set of counters for the entire C-MOVE.

C.4.2.3.1.1.6 Pending Responses

A C-MOVE SCP may send zero or more Pending responses during processing. When it sends a Pending response, it shall report the current values of the Remaining, Completed, Warning, and Failed counters.

C.4.2.3.1.1.7 Mixed-mechanism Failure Retry

In the event a C-STORE sub-operation yields Failure, it might be retried with a DICOMweb Store Transaction. Likewise, when a DICOMweb Store Transaction sub-operation yields Failure, it might be retried with a DIMSE C-STORE operation. Implementations may be configured to such mixed-mechanism failure retries.

Warnings shall not be retried. This is because Warnings are a partial Success.

C.4.2.3.1.1.8 Backward Compatibility

A C-MOVE SCP claiming this Option shall still support C-STORE sub-operations to the Move Destination AE Title.

C.4.2.3.1.1.9 Conformance

A C-MOVE SCP claiming this option shall describe in its Conformance Statement:

- **decision logic for when it uses a C-STORE operation or a Store Transaction**
- **rules for mapping AE Titles to endpoints**
- **security controls (use of HTTPS, credential sourcing, allow-lists)**

C.4.2.3.1.2 Proxy Option

When this option is supported, a C-MOVE SCP may, at its discretion, proxy the C-MOVE request using a Send Transaction Request.

This involves:

- **converting the C-MOVE request content into equivalent Send Transaction content**
- **mapping the C-MOVE destination AE Title to the DICOMweb endpoint, determined by a local policy**
- **issuing a Send Transaction request as a user agent**
- **reformating the results into the DIMSE format**
- **sending the DIMSE results in the C-MOVE responses**

See Sections 10.X and 12.X in PS3.18 for more information about the Send transaction.

C.4.2.3.1.2.1 Scope

This Option applies to all Query/Retrieve Information Model – MOVE SOP Classes.

C.4.2.3.1.2.2 Interface

This option does not change the interface between the C-MOVE SCU and C-MOVE SCP. The C-MOVE SCU supplies a Move Destination AE Title in the C-MOVE request. It may supply credentials in the association negotiation. It will not supply URLs or delivery preferences. The C-MOVE SCP selects the delivery mechanism, taking the capabilities of the destination into account.

C.4.2.3.1.2.3 Behavior

When the C-MOVE SCP selects to utilize a Send transaction under this Option, communication with the Send origin server is performed using HTTP messages.

The C-MOVE SCP proxy delegates all responsibilities of the C-MOVE behavior to the Send origin server, including maintaining sub-operation counters (see Section C.4.2.3.1.1.4). The C-MOVE SCP proxy shall transform received messages into corresponding messages in the other protocol and forward them.

C.4.2.3.1.2.3 Conformance

A C-MOVE SCP claiming this option shall describe in its Conformance Statement:

- **rules for mapping AE Titles to endpoints**
- **security controls (use of HTTPS, credential sourcing, allow-lists)**

C.4.2.3.2 Extended Behavior of SCP

Extended SCP behavior shall be negotiated at Association establishment time. If an option within the extended behavior is not agreed upon in the negotiation, then only baseline SCP behavior shall be performed with respect to that option. Extended SCP behavior includes all baseline behavior with the following option:

- Relational-retrieve
- Enhanced Multi-Frame Image Conversion

More than one option may be agreed upon.

C.4.2.3.2.1 Relational-Retrieve

The C-MOVE Service with relational-retrieve removes the restriction that the SCU supplies Unique Key values for levels above the Query/Retrieve level to help identify an entity at the level of the retrieval. Hence, the Identifier of a C-MOVE request may specify the transfer of:

- all Composite Object Instances related to a study by only providing a Study Instance UID (0020,000D)
- all Composite Object Instances related to a series by only providing a Series Instance UID (0020,000E)
- individual Composite Object Instances by only providing a list of SOP Instance UIDs (0008,0018)

C.4.2.3.2.2 Enhanced Multi-Frame Image Conversion

If Query/Retrieve View (0008,0053) is not present in the request Identifier, then the SCP shall identify a set of Entities at the level of the transfer based upon the values in the Unique Keys in the Identifier of the C-MOVE request that correspond to the instances it possesses, as received, and shall initiate C-STORE sub-operations for all the corresponding storage SOP Instances.

If Query/Retrieve View (0008,0053) is present with a value of "CLASSIC", then the SCP shall identify a set of Entities at the level of the transfer based upon the values in the Unique Keys in the Identifier of the C-MOVE request that correspond to the Classic single frame Instances (converted from Enhanced multi-frame Instances if required), as well as any instances that were converted to preserve referential integrity, and any that did not need to be converted, and shall initiate C-STORE sub-operations for all the corresponding storage SOP Instances.

If Query/Retrieve View (0008,0053) is present with a value of "ENHANCED", then the SCP shall identify a set of Entities at the level of the transfer based upon the values in the Unique Keys in the Identifier of the C-MOVE request that correspond to the Enhanced multi-frame Instances (converted from Classic single frame Instances if required), as well as any instances that were converted to preserve referential integrity, and any that did not need to be converted, and shall initiate C-STORE sub-operations for all the corresponding storage SOP Instances.

Note

1. The SCP will not send information that is duplicated to the C-STORE SCP. For example, if an entire series of single frame instances can be converted to a separate series of converted instances, a STUDY level C-MOVE will not send both series.
2. The C-STORE SCP will need to support the necessary SOP Classes for converted instances, otherwise the C-STORE sub-operations will fail in the normal manner and this will be reflected in the C-MOVE responses.
3. The Query Information Model is unchanged, and the same unique, required and optional keys are equally applicable to both views, except that the values for the SERIES and IMAGE level queries will be different and will depend on the converted instance content.
4. The Query/Retrieve View is still required in an IMAGE or SERIES level request Identifier, even though the requested unique key(s) are unambiguous.

Changes to NEMA Standards Publications PS 3.2

Add to N.1.3 for the Send transactions

N.1 Overview

…

N.1.3 DICOM Web Services

…

N.1.3.2 Studies Service

Table N.1-9 lists details on the support of the Studies Service.

[Complete Table N.1-9 to indicate support for the Studies Web Service]

Table N.1-9. Study Service

Service	Transaction	Resource	User Agent	Origin Server
Studies Web Service	Retrieve Capabilities			
	Retrieve (WADO-RS)	Study		
		Study Metadata		
		Study Bulkdata		
		Study Pixel Data		
		Rendered Study		
		Rendered MPR Volume Study		
		Rendered 3D Volume Study		
		Study Thumbnail		
		Series		
		Series Metadata		
		Series Bulkdata		
		Series Pixel Data		
		Rendered Series		
		Rendered MPR Volume Series		
		Rendered 3D Volume Series		
		Series Thumbnail		

Service	Transaction	Resource	User Agent	Origin Server
		Instance		
		Instance Metadata		
		Instance Bulkdata		
		Instance Pixel Data		
		Rendered Instance		
		Rendered MPR Volume Instance		
		Rendered 3D Volume Instance		
		Instance Thumbnail		
		Frames		
		Rendered Frames		
		Rendered MPR Volume Frames		
		Rendered 3D Volume Frames		
		Frame Thumbnail		
		Bulkdata		
	Search (QIDO-RS)	All Studies		
		Study's Series		
		Study's Instances		
		All Series		
		Series Instances		
		All Instances		
	Store (STOW-RS)	All Studies		
		Study		
	Send and Check Send Result (SEND-RS)	**All Studies Send Requests**		
		Study's Series Send Requests		
		Study's Instances Send Requests		
		All Series Send Requests		

Service	Transaction	Resource	User Agent	Origin Server
		Study Series' Instances Send Requests		
		All Instances Send Requests		

…

N.1.3.4 Non-Patient Instance Service

Table N.1-11 lists details on the support of Non-Patient Instances Service.

For details on the supported resource categories (e.g., Color Palette, Defined Procedure Protocol, Hanging Protocol**.** ~~**or**~~ Implant Templates**, or Protocol Approvals**), see Table N.1-1.

[Complete Table N.1-11 to indicate support for the Non-Patient Instance Web Service.]

Table N.1-11. Non-Patient Instance Service

Service	Transaction	Resource	User Agent	Origin Server
Non-Patient Instances Web Service	Retrieve Capabilities			
	Retrieve			
	Store			
	Search ~~***(Note)***~~			
	Send and Check Send Result			

Add to section N.5.2 Supported DIMSE Services for C-MOVE Options

N.5 Service and Interoperability Description

…

N.5.2 DIMSE Services

…

N.5.2.7 Query/Retrieve Service Class

…

N.5.2.7.7 SCP of the Patient Root Q/R Information Model - MOVE SOP Class

As the SCP of the Study Root Q/R - Information Model - MOVE, the <Product> receives the C-MOVE-RQ and in turn uses the C- STORE-RQ sub operation to send matching SOP Instances to the Move Destination AE included in the C-MOVE-RQ.

[Provide a list of Storage SOP Classes supported or reference Storage Table in Overview e.g.]

As the SCP of the Storage Service Class, all Storage SOP Classes listed in Section N.1.1 are supported.

[Describe the relationship between the incoming C-MOVE Request and the C-STORE Sub-operation, e.g., is each instance sent on one Association or is the same Association used for all instances, is this behavior configurable.]

[Describe your product behavior if a C-MOVE-CANCEL Request is received.]

[Describe whether <product> supports the Alternative Sub-operation Mechanism Option, as described in PS3.4, C.4.2.3.1.1. If so, provide decision logic for when it uses a C-STORE operation or a Store Transaction, rules for mapping AE Titles to endpoints, and security controls (use of HTTPS, credential sourcing, allow-lists).]

[Describe whether <product> supports the Proxy Option, as described in PS3.4, C.4.2.3.1.2. If so, provide rules for mapping AE Titles to endpoints, and security controls (use of HTTPS, credential sourcing, allow-lists).]

N.5.2.7.8 SCP of the Study Root Q/R Information Model - MOVE SOP Class

[If this SOP Class is supported, fill in the section as indicated in Section N.5.2.7.7.]

Add to section N.5.3 Supported DICOM Web Services for the Send Transactions

N.5 Service and Interoperability Description

…

N.5.3 DICOM Web Services

…

N.5.3.2 Studies Web Service

…

N.5.3.2.X Send Transactions (SEND-RS)

The Studies Web Service Send Transactions are also known as SEND-RS.

N.5.3.2.X.1 User Agent

[Indicate whether your product supports sending matching instances to itself and/or to a different endpoint.]

[Indicate, only when your product supports sending matching instances to itself, its behavior in case no Store Request is received after a specific time, e.g., <Product> expects to receive the Store Request in a configurable time frame after the Send Request is sent. If no Store Requests are received within this configurable timeframe, it repeats the Send Request.]

[Indicate, only when your product supports sending matching instances to itself, whether it accepts C-STORE or STOW or both kinds of Store Requests. This can be done by referring to the applicable sections in this conformance statement.]

N.5.3.2.X.2 Origin Server

[Indicate whether <Product> performs Store sub-operations by using DIMSE (with C-STOREs) or by using DICOMweb (with STOW). Indicate, when applicable, how <Product> determines what to use and whether this is configurable.]

[In case the Store sub-operations are performed using DIMSE, indicate the relationship between the incoming Send Request and the C-STORE sub-operation, e.g., is each instance sent on a separate Association or are all instances sent on the same Association. Indicate whether, and how, this behavior is configurable. Also indicate <Product>'s function to come to an AE Title given the URI destination.]

…

N.5.3.4 Non-Patient Instance Web Service

…

N.5.3.4.X Send Transactions

N.5.3.4.X.1 User Agent

[Indicate whether your product supports sending matching instances to itself and/or to a different endpoint.]

[Indicate, only when your product supports sending matching instances to itself, its behavior in case no Store Request is received after a specific time, e.g., <Product> expects to receive the Store Request in a configurable time frame after the Send Request is sent. If no Store Requests are received within this configurable timeframe, it repeats the Send Request.]

[Indicate, only when your product supports sending matching instances to itself, whether it accepts C-STORE or STOW or both kinds of Store Requests. This can be done by referring to the applicable sections in this conformance statement.]

N.5.3.4.X.2 Origin Server

[Indicate whether <Product> performs Store sub-operations by using DIMSE (with C-STOREs) or by using DICOMweb (with STOW). Indicate, when applicable, how <Product> determines what to use and whether this is configurable.]

[In case the Store sub-operations are performed using DIMSE, indicate the relationship between the incoming Send Request and the C-STORE sub-operation, e.g., is each instance sent on a separate Association or are all instances sent on the same Association. Indicate whether, and how, this behavior is configurable. Also indicate <Product>'s function to come to an AE Title given the URI destination.]

Add to section N.6.3 Configuration of DICOM Web Services for the Send Transactions

N.6.3 Configuration of DICOM Web Services

…

N.6.3.2 Studies Web Service Configuration

…

N.6.3.2.X Send Transactions (SEND-RS) Configuration

The Send Transactions are also known as SEND-RS. Table N.6-X lists configuration parameters for the Send Transactions of the Studies Web Service:

[Remove the unsupported parameters from the local and remote configuration parameters.]

Table N.6-X. Send Transaction Parameters

Local Configuration Parameters - Send Transactions			
Parameter	**Configurable**	**Default Value**	**Comments**
	<<USER SERVICE FIXED>>	*[If there is no default, leave blank]*	*[Provide comments or Values/ranges if applicable]*
Send local Origin Server URL (Base URI)	*FIXED*	*http://<hostname>:<port>/send*	
Port	*SERVICE*	*8080*	
Secured Send local Origin Server URL (Base URI)		*https://<hostname>:<securedport>/send*	
Secured Port		*8081*	
<Specific Send Transaction parameter>			
Remote Configuration Parameters - Send Transactions			

Local Configuration Parameters - Send Transactions			
Parameter	**Configurable**	**Default Value**	**Comments**
[Either document the number of supported remote hosts, e.g., <Product> supports configuration of up to <X> remote hosts, or state that there is no limitation other than those mandated by the operating system.]			
Parameter	**Configurable**	**Default Value**	**Comments**
	<<USER SERVICE FIXED>>	***[If there is no default, leave blank]***	***[Provide comments or Values/ranges if applicable]***
Send remote Origin Server URL	***SERVICE***	***http://<hostname>:<port>/send***	
Port	***SERVICE***	***8080***	
Secured Send remote Origin Server URL	***SERVICE***	***https://<hostname>:<securedport>/send***	
Secured Port	***SERVICE***	***8081***	
<Specific Send Transaction parameter>			

...

Add to section N.7.3.3 DICOM Web Services for the Send Transactions

N.7 Network and Media Communication Details

…

N.7.3 Status Codes

…

N.7.3.3 DICOM Web Services

…

N.7.3.3.3 Studies Service

…

N.7.3.3.3.X1 Send Transaction as Origin Server

Table N.7.3.3.3.X1-1 lists the Status Codes that an origin server supports for the Send Transaction of the Studies Service and the condition in which any of the listed Status Codes is sent.

[Describe below the condition in which the application sends the specific Status Codes in the Send Transaction response as origin server.]

Table N.7.3.3.3.X1-1. Status Codes of Origin Server for Send Transaction

Status	Code	Condition
Success	*200 (OK)*	*The origin server finished processing the send request*
	202 (Accepted)	*The origin server has not finished processing the send request yet*
Failure	*400 (Bad Request)*	*The origin server cannot handle the send request because of errors in the request headers or parameters*
	409 (Conflict)	*The origin server cannot handle the send request because the provided Transaction UID is already in use*
	503 (Service Unavailable)	*The origin server cannot handle the send request; this may be a temporal or permanent state*

N.7.3.3.3.X2 Send Transaction as User Agent

Table N.7.3.3.3.X2-1 lists the Status Codes that a user agent supports for the Send Transaction of the Studies Service and defines the application behavior, when encountering any of the listed Status Codes.

[Describe below the behavior of the application when it receives various Status Codes in the Request Transaction response.]

Table N.7.3.3.3.X2-1. Status Codes of User Agent for Send Transaction

Status	Code	Behavior
Success	*200 (OK)*	*Continue with completion behavior*
	202 (Accepted)	*Query later to get the result of the request*
Failure	*400 (Bad Request)*	*Report the failure*
	409 (Conflict)	*Retry with another Transaction UID*
*	*Any other code*	*Report the failure*

N.7.3.3.3.X3 Check Send Result Transaction as Origin Server

Table N.7.3.3.3.X3-1 lists the Status Codes that an origin server supports for the Check Send Result Transaction of the Studies Service and the condition in which any of the listed Status Codes is sent.

[Describe below the condition in which the application sends the specific Status Codes in the Check Send Result Transaction response as origin server.]

Table N.7.3.3.3.X3-1. Status Codes of Origin Server for Check Send Result Transaction

Status	Code	Condition
Success	*200 (OK)*	*The origin server finished processing the send request*
	202 (Accepted)	*The origin server has not finished processing the send request yet*

Status	Code	Condition
Failure	*404 (Not Found)*	*The origin server cannot find the send request result*
	410 (Gone)	*The origin server can no longer provide the send request result*
	503 (Service Unavailable)	*The origin server cannot handle the check send result request; this may be a temporary or permanent state*

1063

1064 **N.7.3.3.3.X4 Check Send Result Transaction as User Agent**

1065 **Table N.7.3.3.3.X4-1 lists the Status Codes that a user agent supports for the Check Send Result Transaction**
1066 **of the Studies Service and defines the application behavior when encountering any of the listed Status Codes.**

1067 ***[Describe below the behavior of the application when it receives various Status Codes in the Check Send***
1068 ***Result Transaction response.]***

1069 **Table N.7.3.3.3.X4-1. Status Codes of User Agent for Check Send Result Transaction**
1070

Status	Code	Behavior
Success	*200 (OK)*	*Continue with completion behavior*
	202 (Accepted)	*Query later to get the result of the request*
Failure	*404 (Not Found)*	*Start all over with a send request*
	410 (Gone)	*Start all over with a send request*
*	*Any other code*	*Report the failure*

1071

1072 …

1073 **N.7.3.3.5 Non-Patient Instance Service**

1074 …

1075 **N.7.3.3.5.X1 Send Transactions as Origin Server**

1076 **Table N.7.3.3.5.X1-1 lists the Status Codes that an origin server supports for the Send Transaction of the Non-**
1077 **Patient Instance Service and the condition in which any of the listed Status Codes is sent.**

1078 ***[Describe below the condition in which the application sends the specific Status Codes in the Send Transaction***
1079 ***response as origin server.]***

1080 **Table N.7.3.3.5.X1-1. Status Codes of Origin Server for Send Transaction**
1081

Status	Code	Condition
Success	*200 (OK)*	*The origin server finished processing the send request*
	202 (Accepted)	*The origin server has not finished processing the send request yet*

Status	Code	Condition
Failure	*400 (Bad Request)*	*The origin server cannot handle the send request because of errors in the request headers or parameters*
	409 (Conflict)	*The origin server cannot handle the send request because the provided Transaction UID is already in use*
	503 (Service Unavailable)	*The origin server cannot handle the send request; this may be a temporal or permanent state*

N.7.3.3.5.X2 Send Transactions as User Agent

Table N.7.3.3.5.X2-1 lists the Status Codes that a user agent supports for the Send Transaction of the Non-Patient Instance Service and defines the application behavior, when encountering any of the listed Status Codes.

[Describe below the behavior of the application when it receives various Status Codes in the Request Transaction response.]

Table N.7.3.3.5.X2-1. Status Codes of User Agent for Send Transaction

Status	Code	Behavior
Success	*200 (OK)*	*Continue with completion behavior*
	202 (Accepted)	*Query later to get the result of the request*
Failure	*400 (Bad Request)*	*Report the failure*
	409 (Conflict)	*Retry with another Transaction UID*
*	*Any other code*	*Report the failure*

N.7.3.3.5.X3 Check Send Result Transaction as Origin Server

Table N.7.3.3.5.X3-1 lists the Status Codes that an origin server supports for the Check Send Result Transaction of the Non-Patient Instance Service and the condition in which any of the listed Status Codes is sent.

[Describe below the condition in which the application sends the specific Status Codes in the Check Send Result Transaction response as origin server.]

Table N.7.3.3.3.X3-1. Status Codes of Origin Server for Check Send Result Transaction

Status	Code	Condition
Success	*200 (OK)*	*The origin server finished processing the send request*
	202 (Accepted)	*The origin server has not finished processing the send request yet*
Failure	*404 (Not Found)*	*The origin server cannot find the send request result*
	410 (Gone)	*The origin server can no longer provide the send request result*

Status	Code	Condition
	503 (Service Unavailable)	*The origin server cannot handle the check send result request; this may be a temporary or permanent state*

N.7.3.3.5.X4 Check Send Result Transaction as User Agent

Table N.7.3.3.5.X4-1 lists the Status Codes that a user agent supports for the Check Send Result Transaction of the Non-Patient Instance Service and defines the application behavior when encountering any of the listed Status Codes.

[Describe below the behavior of the application when it receives various Status Codes in the Check Send Result Transaction response.]

Table N.7.3.3.5.X4-1. Status Codes of User Agent for Check Send Result Transaction

Status	Code	Behavior
Success	*200 (OK)*	*Continue with completion behavior*
	202 (Accepted)	*Query later to get the result of the send request*
Failure	*404 (Not Found)*	*Start all over with a send request*
	410 (Gone)	*Start all over with a send request*
*	*Any other code*	*Report the failure*

Changes to NEMA Standards Publications PS 3.6

No new attributes have been introduced.

Changes to NEMA Standards Publications PS 3.15

There are no new attributes to be added to table E.1-1 of annex E.