Digital Imaging and Communications in Medicine (DICOM)

Supplement XXX: Application Request

Prepared by:

DICOM Standards Committee, Working Group 23 (Artificial Intelligence/Application Hosting)

1300 N. 17th Street

Rosslyn, Virginia 22209 USA

VERSION: Version 1

March 11, 2025

Developed pursuant to DICOM Work Item 2020-08-A

Table of Contents

Document History

Document Version	Date	Content
01		Initial Draft for discussion at WG-9

Open Issues for Public Comment

1		
2		
3		
4		
5		
6		
7		
8		

9		

Closed Issues

1	
2	

Scope and Field of Application

This Supplement proposes an updated Application Program Interface (API) to DICOM PS3.19 and retiring the previous SOAP API. The use of this API will allow a client to request an Application to perform work on given data. This API was first proposed by the American College of Radiology's Data Science Institute (ACR DSI) and has since been extended by the MONAI Deploy Informatics Gateway (MIG) for inference requests. This API fills the gap in IHE's AI Workflow for Imaging (AIW-I) profile where a Task Performer (Proxy) communicates and exchanges input data to perform inference and exchange results. In the profile there are no defined transactions between the Task Performer (Proxy) to request services from an AI Model (Proxied). For the purposes of this supplement the Task Performer (Proxy) would be considered a client and the AI Model (Proxied) would be an Application.

Out of scope are all DICOM UPS-RS transactions as well as any exchange of capabilities between a client and the Application. Also there is no restriction on what role a client plays in the overall workflow when using this API for requesting services of an Application.

Figure SF-1 shows how this API would fit in the IHE AIW-I Proxied AI Model use case.

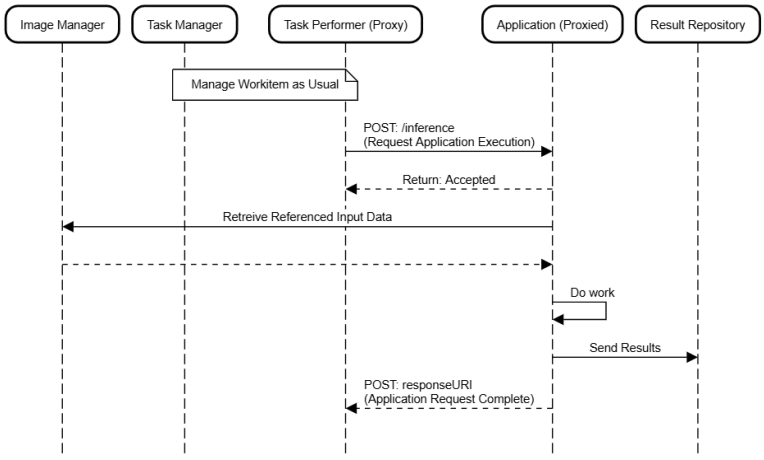

Figure SF-1 API Workflow Interaction

```
rparticipant Image Manager
rparticipant Task Manager
rparticipant Task Performer (Proxy)
rparticipant Application (Proxied)
rparticipant Result Repository
note over Task Manager,Task Performer (Proxy):Manage Workitem as Usual
Task Performer (Proxy)->Application (Proxied):POST: /inference\n(Request Application
Execution)
Task Performer (Proxy)<--Application (Proxied): Return: Accepted
Application (Proxied)->Image Manager:Retreive Referenced Input Data
Application (Proxied)<--Image Manager:
Application (Proxied)->Application (Proxied): Do work
Application (Proxied)->Result Repository: Send Results
Application (Proxied)-->Task Performer (Proxy):POST: responseURI\n(Application Request
Complete)
```

Changes to NEMA Standards Publication PS 3.19

Update PS3.19 Foreword as follows:

Foreword

…

HL7®**, FHIR®** and CDA® are the registered trademarks of Health Level Seven International, all rights reserved.

…

Replace PS3.19 Section 1 Scope and Field of Application with the following:

1 Scope and Field of Application

This Part of the DICOM Standard defines an Application Program Interface (API) for asynchronous workflow which is designed to support potentially long-running Application or AI model (also known as algorithm) inference requests (see Figure 1-1). Some trigger event, for example notification of the availability of clinical data or user request, causes a client to request services from one or more Applications which will execute on the data referenced in the request. The request issued to the Application by the client consists largely of metadata and endpoint information associated with acquired clinical data, allowing the Application to fetch data as needed. Once the Application is complete and all results have been successfully sent to the appropriate result repositories, the Application shall issue a

message to the response Uniform Resource Identifier (URI) included in the request body, enumerating the resources it created, signaling its completion.

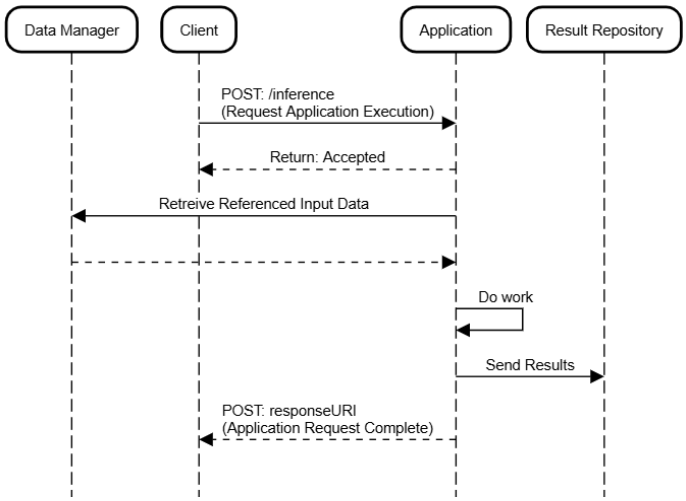

Figure 1-1 API Workflow Interaction

```
rparticipant Data Manager
rparticipant Client
rparticipant Application
rparticipant Result Repository
Client->Application:POST: /inference\n(Request Application Execution)
Client<--Application: Return: Accepted
Application->Data Manager:Retreive Referenced Input Data
Application<--Data Manager:
Application->Application: Do work
Application->Result Repository: Send Results
Application-->Client:POST: responseURI\n(Application Request Complete)
```

For those clients unable to provide a response URI, Figure 1-2 shows how the Application providing a status endpoint as defined in Section 8 interacts with client requests. In this implementation, the Application is not responsible for notifying the client upon completion of its execution request.

Kommentiert [LS1]: I've been thinking should be somehow describe the governance of the inference models. If we think in production, model versioning should be recorded, and may be not only an implementation detail, but it could be part of standard somehow?

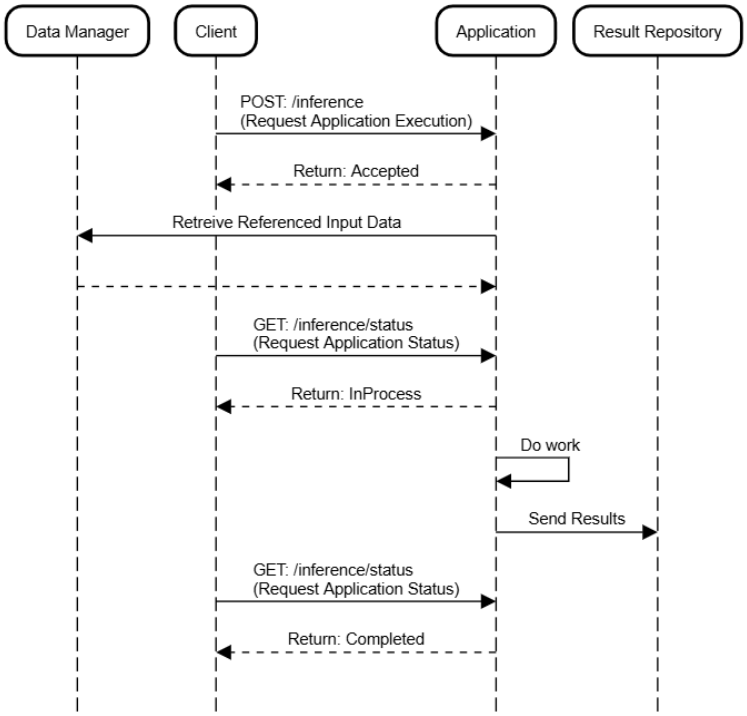

Figure 1-2 Client Requesting Status Updates

```
rparticipant Data Manager
rparticipant Client
rparticipant Application
rparticipant Result Repository
Client->Application:POST: /inference\n(Request Application Execution)
Client<--Application: Return: Accepted
Application->Data Manager:Retreive Referenced Input Data
Application<--Data Manager:
Client->Application:GET: /inference/status\n(Request Application Status)
Client<--Application: Return: InProcess
Application->Application: Do work
Application->Result Repository: Send Results
Client->Application:GET: /inference/status\n(Request Application Status)
Client<--Application: Return: Completed
```

Update PS3.19 Section 2 Normative References as follows:

2 Normative References

The following standards contain provisions that, through reference in this text, constitute provisions of this Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this Standard are encouraged to investigate the possibilities of applying the most recent editions of the standards indicated below.

[IEEE 9274.1.1 2033] IEEE. October 2023. ***IEEE Standard for Learning Technology--JavaScript Object Notation (JSON) Data Model Format and Representational State Transfer (RESTful) Web Service for Learner Experience Data Tracking and Access.***
https://standards.ieee.org/ieee/9274.1.1/7321/

…

[RFC3986] IETF. *Uniform Resource Identifiers (URI): Generic Syntax*. http://tools.ietf.org/html/rfc3986 .

[RFC6749] IETF. October 2012. ***The OAuth 2.0 Authorization Framework.***
https://datatracker.ietf.org/doc/html/rfc6749

[RFC7807] IEFT. March 2016. ***Problem Details for HTTP APIs***
https://datatracker.ietf.org/doc/html/rfc7807

RFC9700] IEFT. January 2025. ***Best Current Practice for OAuth 2.0 Security***
https://datatracker.ietf.org/doc/html/rfc9700

Update PS3.19 Section 3 Definitions as follows:

3 Definitions

3.7 Application Hosting Definitions

The following definitions are commonly used in this Part of the Standard:

Application Programming Interface (API)

A set of interface methods that ~~Hosted~~ Applications and ~~Hosting Systems~~ **Clients** use to communicate with each other.

~~**Hosted Application**~~

~~**An application launched and controlled by a Hosting System. The Hosted Application may utilize services offered by the Hosting System.**~~

~~**Hosting System**~~

~~**The application used to launch and control Hosted Applications. The Hosting System provides a variety of services such as DICOM object retrieval and storage for the Hosted Application. The Hosting System provides the infrastructure in which the Hosted Application runs and interacts with the external environment. This includes network access, database and security.**~~

Application

Software that operates or performs inference on data that it is either provided directly or tasked to operate upon by a client through an API request.

Client

Software that interacts and makes requests of an Application through the use of an API.

Update PS3.19 Section 4 Symbols and Abbreviations as follows:

4 Symbols and Abbreviations

The following symbols and abbreviations are used in this Part of the Standard.

…

ISO

International Standards Organization

JSON

JavaScript Object Notation

…

Replace PS3.19 Section 6 Application Hosting Overview with the following:

6 Application Request Overview

A client, with knowledge of an Application's capabilities and endpoint details, chooses to invoke the Application on a given dataset. The Application is assumed to be aware of the environment in which it is deployed such as the details of data retrieval endpoints and thus those details are not communicated in the Application request.

1. A client chooses to invoke the Application on a given dataset

2. The client uses the API to initiate an inference task by the Application. The initiation request with details about the input data and endpoint location
3. The Application uses the API to respond to the client's inference request
4. The Application retrieves its input data from the specified endpoint for processing
5. The Application may also use the API to inform the client of the status of the inference task, for example completion, warnings, or errors encountered
6. In cases where the Application informs the client of the status using the API, the client uses the API to respond to the Application's completion message
7. The Application, as it processes the input data, might create output objects, and use the API to inform the client of their existence
8. The Application uses the API to provide the health of its API endpoint using liveliness and readiness endpoints
9. The Application may also use the API to provide the client the status of its requested inference task, for example completed, in progress, or queued

Replace PS3.19 Section 7 Hosted Application Life Cycle with the following:

7 Inference Request

The inference endpoint provides a RESTful interface for triggering a new inference request to an Application. The API supports, but does not mandate, the use of the authorization header in the request. Given the wide variety of authentication mechanisms available, if present, the authentication scheme is dictated by the Task Performer (Proxy). Support of secure DICOM TLS (Transport Layer Security) per DICOM P3.15 for API calls as well as interactions with data endpoints should be considered best practice. The Application is not mandated to perform any authentication of requests. For input and output connections that require credentials, ensure that all the connections are secured and encrypted.

Kommentiert [BB2]: Provide proper reference

When secure authentication is not present the Application should only be used within a secure environment with a firewall, router protection, VPN, and other network security provisions.

7.1 POST /inference

This POST API call triggers a new inference job for execution by an Application using the dataset specified by the client.

7.1.1 Inference Request Body Parameters

Table 7.1.1-1 provides the parameters for an inference request to an Application.

A transaction ID must be included in all requests for correlation. This is used by the Task Performer (Proxy) to track subsequent requests from the Application and shall be included in all requests to clinical datastores (where applicable). Additionally, it shall be included in the response.

A response URI shall be provided to the Application. This URI is where the response is to be posted upon completion of inference or to signal failure, which then successfully completes the request.

Priority may be included in the request body. If priority is not specified in the request, but is used by the Application, the Application will use a default priority of 128. Priority is available for Applications that handle multiple requests or can scale resources. Larger values indicate higher priority with the valid range being 0-255.

Input metadata provides the details surrounding the inference request.

Input resources shall be provided in the inference request and list where the Application may find the additional data it needs to perform inference. The supported interface types include XDS, DICOMWeb, DIMSE, FHIR, S3 and file folders. The order in which these are listed in the request, should there be more than one endpoint that can provide data, suggests the order in which the Application should query. The connection details must be provided and enumerate a list of operations the endpoint offers to the Application as well as additional necessary connection information. These resources may or may not differ from output resources.

Output resources shall be provided and list where the Application shall store artifacts and resources it creates as a result of its inference. These may or may not be different from the input resources, therefore artifacts should only be sent to locations specified as output resources. The same list of interface types are available as outputs, and the valid operations for the endpoint must be specified along with connection information.

Request Content Type: JSON

Table 7.1.1-1 Parameters

Name	Type	Required	Description
transactionId	string	Y	A user-provided transaction ID for correlating an inference request
responseUri	uri	Conditionally	URI to which Application responds to signal completion. Required if Application does not support retrieval of status by transaction ID. See Section 8.2
priority	integer	N	The priority at which the request should be treated; valid range is 0-255 with 255 being the highest priority; if not specified default is 128
inputMetadata	inputMetadata object	Y	Details of the data associated with the inference request. See Section 7.1.1.1
inputResources	array of	Y	Data sources from which to retrieve the

	inputResource objects		specified dataset. See Section 7.1.1.2
outputEndpoint	array of outputEndpoint objects	Y	Output destinations where results are exported to. See Section 7.1.1.3

See example Annex B, B.1 Inference Request.

7.1.1.1 inputMetadata Object

Table 7.1.1.1-1 inputMetadata Object Parameters

Name	Type	Required	Description
type	enumerated value	Y	Details the object information for parsing Defined Terms: DICOM_UID PATIENT_ID FHIR ACCESSION_NUMBER See Section 7.1.1.1.1
object parameters	type specific object	Y	DICOM_UID – See Section 7.1.1.1.2 PATIENT_ID – See Section 7.1.1.1.3 FHIR – See Section 7.1.1.1.4 ACCESSION_NUMBER – See Section 7.1.1.1.5

7.1.1.1.1 type Descriptions

Details a list of terms which define the parameters for the object which is to be parsed. These parameters are used to define the input data which the Application will use for its inference.

Defined Terms:

DICOM_UID

Provides study, series, instance and/or frame information to be used for Application inference.

PATIENT_ID

Patient ID (0010,0020) which shall used to retrieving data from the defined input resources

FHIR

Defines the FHIR resource which shall used to retrieving data from the defined input resources

ACCESSION_NUMBER

Accession Number ((0008,0050) or (FHIR reference.identifier type 'ASCN') which shall used to retrieving data from the defined input

7.1.1.1.2 DICOM_UID Object

Table 7.1.1.1.2-1 DICOM_UID Object Parameters

Name	Type	Required	Description
studies	array	Y	A set of images produced in single study
>studyInstanceUid	UID	Y	Unique identifier for the Study. Only a single UID is permitted.
>>series	array	N	Identify and describe general information about the Series within a Study.
>>>seriesInstanceUid	UID	Y	Unique identifier of the Series.
>>>instances	array	N	Identify and describe one or more image within a particular Series.
>>>>sopInstanceUid	array	Y	Uniquely identifies the SOP Instance.
>>>>frames	array	N	Identify and describe one or more frames within a particular Image.
>>>>>frameNumber	array	Y	Identifies the frame numbers within the SOP Instance.

Example:

```
inputMetadata: {
      "type": "DICOM_UID",
      "studies": [{
            "studyInstanceUid":
      "2.16.840.1.114488.0.4.123489834087.13300714",
            "series": [{
                  "seriesInstanceUid":
      "2.16.840.1.114488.0.4.123489834087.13300714.2",
                        "instances": [
                              "sopInstanceUid":
      ["2.16.840.1.114488.0.4.123489834087.13300714.2.1"],
                              "frames": [
                                    "frameNumber": ["10","11","12"]
                              ]
                        ]
            }]
      }]
}
```

7.1.1.1.3 PATIENT_ID Object

Table 7.1.1.1.3-1 PATIENT_ID Object Parameters

Name	Type	Required	Description
patientId	string	Y	Primary identifier for the Patient.

Example:

```
inputMetadata: {
      "type": " PATIENT_ID",
      "patientId": "ABC123"
}
```

7.1.1.1.4 FHIR Object

Table 7.1.1.1.4-1 FHIR Object Parameters

Name	Type	Required	Description
resources	array	Y	One or more FHIR resources that contain the object data.
>resourceType	enumerated value	Y	The FHIR resource type for the data.
>id	string	Y	The unique identifier for the resource in the FHIR server.

Example:

```
inputMetadata: {
      "type": "FHIR",
      "resources": [{
            "resourceType": "Observation",
            "id": "12345"
      }]
}
```

7.1.1.1.5 ACCESSION_NUMBER Object

Table 7.1.1.1.5-1 ACCESSION_NUMBER Object Parameters

Name	Type	Required	Description
accessionNumber	string	Y	A departmental Information System generated number that identifies the Imaging Service Request.

Example:

```
inputMetadata: {
      "type": "ACCESSION_NUMBER",
      "accessionNumber": "ABC123"
}
```

7.1.1.2 inputResources Object

Table 7.1.1.2-1 inputResources Object Parameters

Name	Type	Required	Description
interface	enumerated value	Y	The interface method for data retrieval. Defined Terms: DIMSE DICOMweb FHIR

			Application See Section 7.1.1.2.1
connectionDetails	connectionDetails object	Y	DIMSE – See Section 7.1.1.2.2 DICOMweb – See Section 7.1.1.2.3 FHIR – See Section 7.1.1.2.3 Application – See Section 7.1.1.2.3

7.1.1.2.1 interface Descriptions

Details a list of terms which define the interface method used by an Application for data exchange.

Defined Terms:

DIMSE
: DICOM Message Service Element

DICOMweb
: DICOM Standard for web-based medical imaging

FHIR
: RESTful interface for exchanging clinical and administrative data

Application
: Specifies another Application

7.1.1.2.2 DIMSE connectionDetails Object

Table 7.1.1.2.2-1 DIMSE connectionDetails Object Parameters

Name	Type	Required	Description
aet	string	Y	AET the DIMSE application uses for inbound data requests
hostname	string	N	The hostname or ip address used for inbound data requests
port	integer	N	The port number used for inbound data requests

Example:

```
inputResources: [{
      "interface": "DIMSE",
      "connectionDetails": {
            "aet": "MYPACSAET",
            "hostname": "mypacs.domain.org",
            "port": "104"
      }
}]
```

7.1.1.2.3 DICOMweb, FHIR and Application connectionDetails Object

Table 7.1.1.2.3-1 DICOMweb, FHIR and Application connectionDetails Object Parameters

Name	Type	Required	Description
uri	uri	Y	The resource URI of the connection.

Kommentiert [BB3]: Add examples at the end of each table

Kommentiert [BB4]: This should not be required and removed as we have stated it would not be required

Kommentiert [BB5]: Discuss leaving this in as optional to provide robustness to the interface

Kommentiert [BB6R5]: Discuss both client and application end, is it required to use the provided information or can it use data it may already have or a different endpoint

Kommentiert [BB7]: Examples here

Example:

```
inputResources: [{
      "interface": "DICOMweb",
      "connectionDetails": {
            "uri": "http://mypacs.domain.org:8000/dicom-web"
      }
},
{
      "interface": "FHIR",
      "connectionDetails": {
            "uri": " http://myehr.domain.org:8000/fhir"
      }
},
{
      "interface": "Application",
      "connectionDetails": {
            "uri": " http://mysidecar.domain.org/inference"
      }
}]
```

7.1.1.3 outputEndpoints Object

Table 7.1.1.3-1 outputEndpoints Object Parameters

Name	Type	Required	Description
interface	enumerated value	Y	The interface method for data output. Defined Terms: DIMSE DICOMweb FHIR Application See Section 7.1.1.2.1
connectionDetails	connectionDetails object	Y	DIMSE – See Section 7.1.1.2.2 DICOMweb – See Section 7.1.1.2.3 FHIR – See Section 7.1.1.2.3 Application – See Section 7.1.1.2.3

Example:

```
outputEndpoints: [{
      "interface": "DICOMweb",
      "connectionDetails": {
            "uri": "http://mypacs.domain.org:8000/dicom-web"
      }
},
{
```

```
            "interface": "FHIR",
            "connectionDetails": {
                  "uri": " http://myehr.domain.org:8000/fhir"
            }
      },
      {
            "interface": "Application",
            "connectionDetails": {
                  "uri": " http://mysidecar.domain.org/inference"
            }
      },
      {
            "interface": "DIMSE",
            "connectionDetails": {
                  "aet": "MYPACSAET",
                  "hostname": "mypacs.domain.org",
                  "port": "104"
            }
      }]
```

7.1.2 Connection Security

For information about securing communications for DICOM DIMSE and DICOMweb transactions refer to PS3.15 Chapter B Secure Transport Connection Profiles (Normative). (https://dicom.nema.org/medical/dicom/current/output/html/part15.html#chapter_B)

For information about securing communications for FHIR transactions refer to HL7 FHIR Security. (https://hl7.org/fhir/security.html)

For information about securing communications for Application transactions refer to Best Current Practice for OAuth 2.0 Security (https://datatracker.ietf.org/doc/html/rfc9700) or it's update.

7.2 Inference Request Response

Upon receiving a request, the Application shall respond to the API with a valid HTTP status code indicating whether the request has been accepted for processing.

For Applications that support inference status request per Section 9.2, returning the URI for such an endpoint can be provided in the response.

```
      HTTP/1.1 200 OK
      Content-Type: application/json

      {
        "status": "'https://myapp.myhospital.org/inference/status/75847"
      }
```

7.2.1 Response Codes

Valid response codes are listed in Table 7.2.1-1.

Table 7.2.1-1 Response Codes

Code	Description
200	Inference request received and scheduled for processing.
409	A request with the same transaction ID already exists.
422	Request contains invalid data or is missing required fields.
500	Server error.

The Application may additionally provide a message with further information regarding the status code following the Problem Details JSON Object [RFC7807] with error details.

```
HTTP/1.1 422 Unprocessable Content
Content-Type: application/json

{
  "type": "https://example.org/validation-error",
  "title": "Your request parameters didn't validate.",
  "details": "API Key Missing"
}
```

8 Inference Complete

Upon the completion or failure of Application inference, the Application issues a POST message to the URI specified in the inference request where they claim conformance to this feature of the API. The POST of the status allows the client to close the transaction request to Application and shall include the original transaction ID as well as a status code indicating whether the Application successfully completed its computation. Status codes are listed in PS3.18 Table 8.5-1. An optional human readable message may be included. Additionally, if the Application created any artifacts which it sent to the output data source, the request shall include a list of created resources. DICOM artifacts will follow the structure as defined in Section 7.1.1.1.2 DICOM UID Object. FHIR resources will be listed as a tuple of pairs defined as a resource type and id. The location of these resources need not be specified as the Transaction ID links the created resource to the available output resources that were defined in the request.

For Application requests where a response URI was not provided in the request or they do not claim conformance to this feature, see Section 9.2 for additional information about how the client may determine the status of a request.

An Application shall comply with the requirements of at least one of the status communication methods, either that specified in Section 8.1 or Section 9.2, but may claim compliance to both.

8.1 POST /responseURI

This POST call signals the completion of an inference job from by Application specified by the client. This request is only made once the Application has submitted all output resources to the output endpoints.

8.1.1 Inference Complete Body Parameters

Request Content Type: JSON

Table 7.1.1-1 Parameters

Name	Type	Required	Description
transactionID	string	Y	A user-provided transaction ID for correlating an inference request
status	enumerated value	Y	Status code indicating whether the Application successfully completed its computation. Status codes are listed in PS3.18 Table 8.5-1
message	string	N	A human-readable message providing additional information
outputResources	array of outputResources objects	N	A list of created resources

See example Annex B, B.2 Inference Complete Request.

8.1.1.1 outputResource Object

Table 7.1.1.1-1 outputResource Object Parameters

Name	Type	Required	Description
type	enumerated value	Y	Details the object information for parsing Defined Terms: DICOM_UID FHIR See Section 8.1.1.1.1
object parameters	type specific object	Y	DICOM_UID – See Section 7.1.1.1.2 FHIR – See Section 7.1.1.1.4

Example:

```
"outputResources": [
{
"type": "DICOM_UID",
"studies": [
{
"StudyInstanceUID": "2.16.840.1.114488.0.4.123489834087.13300714",
```

```
            "series": [{
                  "SeriesInstanceUID":
      "2.16.840.1.114488.0.4.123489834087.13300714.2.1",
                  "instances": [
                        "SOPInstanceUID":
      ["2.16.840.1.114488.0.4.123489834087.5000"],
                              "frames": [
                                    "FrameNumber": ["1","2"]
                              ]
                  ]
            }]
      }]
      },
      {
      "type": "FHIR",
      "resources": [{
            "resourceType": "Observation",
            "id": "98765"
      }]
      },
      ]
      }
```

8.1.1.1.1 type Descriptions

Details a list of terms which define the parameters for the object which is to be parsed. These parameters are used to define the output data which the Application created during inference.

Defined Terms:

DICOM_UID

Provides study, series, instance and/or frame information which the Application created during inference.

FHIR

Defines the FHIR resources which the Application created during inference.

8.2 Inference Completion Response

The client shall respond with a valid HTTP status code to indicate whether it successfully processed the completion request. Valid response codes are listed in PS3.18 Table 8.5-1.

9 Health/Status Request

9.1 Health

Applications must provide liveness and readiness endpoints for clients to be able to use.
The health endpoint provides the following APIs to get the status of the internals of the Application.

Kommentiert [BB8]: KH - It might be worth using HTTP codes only for the health check, rather than relying on a custom code. For instance, Nginx assumes that any health check response outside the range 200-399, or which arrives after a configurable timeout, indicates an unhealthy service. That gives us access to a wide range of potentially useful HTTP status codes, and again is more in tune with a "native" HTTP / RESTful API.

9.1.1 GET /heath/live

The liveness endpoint returns the liveness of the Application. If the check does not return the expected response, it means that the Application is unhealthy or dead.

9.1.1.1 /health/live responses

The /health/live endpoint shall signal LIVE, even though the Application may not be ready to accept incoming requests. No response from the /health/live endpoint means the Application itself is down or stopped.

Example:

```
HTTP/1.1 200 Success Content
Content-Type: application/json
{
  "status": "LIVE"
}
```

9.1.2 GET /health/ready

The readiness endpoint returns the readiness state to accept incoming requests to the Application from the client.

9.1.2.1 /health/ready responses

Table 9.1.2.1-1 provides the valid responses for the /health/ready endpoint and their description.

Table 9.1.2.1-1 Responses

Response	Description
READY	Incoming requests will be accepted by the Application.
NOT_READY	The Application although running is unable to process new requests currently.

Example:

```
HTTP/1.1 200 Success Content
Content-Type: application/json
{
  "status": "READY"
}
```

9.2 Status

For Application requests where a response URI was not provided in the request or they do not claim conformance to this feature, the client shall be able to determine the status of a request from the /inference/status endpoint using the transactionId.

Duration of the availability of the status for the transaction ID is implementation specific. Each Application shall disclose the length of time from the end of a transaction for which the status of that transaction will be available at the /inference/status endpoint. For compliance purposes, the minimum time shall be 24 hours.

9.2.1 GET /inference/status/{transactionId}

Returns the status of an inference request. Table 9.2.1-1 includes the parameters that are used in the inference status request.

Table 9.2.1-1 Parameters

Name	Type	Required	Description
transactionId	string	Y	The transaction ID of the original inference request

Example:

```
curl --location 'https://myapp.myhospital.org/inference/status/68743698'
```

9.2.2 Inference Status Response

Valid response codes are listed in Table 9.2.2-1.

Table 9.2.2-1 Response Codes

Code	Description
200	Inference request status is available. In the response detail, Table 9.2.2-2 provides the values representing the status itself.
404	Inference request not found.
500	Server error.

The Application may additionally provide a message with further information regarding the status code following the Problem Details JSON Object [RFC7807] with error details.

```
HTTP/1.1 500 Internal Server Error Content
Content-Type: application/json

{
  "type": "https://example.org/server-error",
  "title": "The server was unable to complete your request.",
  "details"; "Data access permission denied."
}
```

Table 9.2.2-2 Response Detail

Response Detail	Description
Completed	Indicates the Application has downloaded and processed the data, as well

	as successfully placed any outputs to the specified endpoint
InProcess	The inference request is being processed by the Application
Queued	Indicates that an inference request is currently queued for data retrieval

```
HTTP/1.1 200 Success Content
Content-Type: application/json

{
  "details": "Completed"
}
```

Replace PS3.19 B. Interface Definitions

Annex B – Schema Definitions

B.1 Inference Request

The following section provides inference request schemas for DICOM UID, Patient ID, FHIR and Accession Number input data.

B.1.1 Inference Request DICOM UID

```
{
"transactionID": "MY_TRANSACTION_ID",
"responseURI": "http://app.endpoint",
"priority": 128,
"inputMetadata":
{
"type": "DICOM_UID",
"studies": [{
      "studyInstanceUid": "2.16.840.1.114488.0.4.123489834087.13300714",
      "series": [{
            "seriesInstanceUid":
"2.16.840.1.114488.0.4.123489834087.13300714.2",
                  "instances": [
                        "sopInstanceUid":
["2.16.840.1.114488.0.4.123489834087.13300714.2.1"],
                        "frames": [
                              "frameNumber": ["10","11","12"]
                        ]
                  ]
      }]
}]
},
"inputResources": [
{
"interface": "DIMSE",
```

```
        "connectionDetails": {
        "aet": "MYPACSAET",
        "hostname": "mypacs.domain.org",
        "port": "104"
        }},
        {
        "interface": "DICOMweb",
        "connectionDetails": {
        "uri": "http://mypacs.domain.org:8000/dicom-web"
        }},
        {
        "interface": "FHIR",
        "connectionDetails": {
        "uri": "http://myehr.domain.org:8000/fhir"
        }},
        {
        "interface": "Application",
        "connectionDetails": {
        "uri": "http://mysidecar.domain.org/inference"
        }}
        ],
        "outputEndpoints": [
        {
        "interface": "DIMSE",
        "connectionDetails": {
        "aet": "MYPACSAET",
        "hostname": "mypacs.domain.org",
        "port": "104"
        }},
        {
        "interface": "DICOMweb",
        "connectionDetails": {
        "uri": http://mypacs.domain.org:8000/dicom-web
        }},
        {
        "interface": "FHIR",
        "connectionDetails": {
        "uri": "http://myehr.domain.org:8000/fhir"
        }},
        {
        "interface": "Application",
        "connectionDetails": {
        "uri": "http://mysidecar.domain.org/inference"
        }}
        ]
        }
```

B.1.2 Inference Request Patient ID

```
        {
        "transactionID": "MY_TRANSACTION_ID",
        "responseURI": "http://app.endpoint",
```

```
        "priority": 128,
        "inputMetadata":
        {
        "type": " PATIENT_ID",
        "patientId": "ABC123"
        },
        "inputResources": [
        {
        "interface": "DIMSE",
        "connectionDetails": {
        "aet": "MYPACSAET",
        "hostname": "mypacs.domain.org",
        "port": "104"
        }},
        {
        "interface": "DICOMweb",
        "connectionDetails": {
        "uri": "http://mypacs.domain.org:8000/dicom-web"
        }},
        {
        "interface": "FHIR",
        "connectionDetails": {
        "uri": "http://myehr.domain.org:8000/fhir"
        }},
        {
        "interface": "Application",
        "connectionDetails": {
        "uri": "http://mysidecar.domain.org/inference"
        }}
        ],
        "outputEndpoints": [
        {
        "interface": "DIMSE",
        "connectionDetails": {
        "aet": "MYPACSAET",
        "hostname": "mypacs.domain.org",
        "port": "104"
        }},
        {
        "interface": "DICOMweb",
        "connectionDetails": {
        "uri": http://mypacs.domain.org:8000/dicom-web
        }},
        {
        "interface": "FHIR",
        "connectionDetails": {
        "uri": "http://myehr.domain.org:8000/fhir"
        }},
        {
        "interface": "Application",
        "connectionDetails": {
        "uri": "http://mysidecar.domain.org/inference"
        }}
```

```
]
}
```

B.1.3 Inference Request FHIR

```
{
"transactionID": "MY_TRANSACTION_ID",
"responseURI": "http://app.endpoint",
"priority": 128,
"inputMetadata":
{
"type": "FHIR",
"resources": [{
      "resourceType": "Observation",
      "id": "12345"
}]
},
"inputResources": [
{
"interface": "DIMSE",
"connectionDetails": {
"aet": "MYPACSAET",
"hostname": "mypacs.domain.org",
"port": "104"
}},
{
"interface": "DICOMweb",
"connectionDetails": {
"uri": "http://mypacs.domain.org:8000/dicom-web"
}},
{
"interface": "FHIR",
"connectionDetails": {
"uri": "http://myehr.domain.org:8000/fhir"
}},
{
"interface": "Application",
"connectionDetails": {
"uri": "http://mysidecar.domain.org/inference"
}}
],
"outputEndpoints": [
{
"interface": "DIMSE",
"connectionDetails": {
"aet": "MYPACSAET",
"hostname": "mypacs.domain.org",
"port": "104"
}},
{
"interface": "DICOMweb",
"connectionDetails": {
```

```
        "uri": http://mypacs.domain.org:8000/dicom-web
        }},
        {
        "interface": "FHIR",
        "connectionDetails": {
        "uri": "http://myehr.domain.org:8000/fhir"
        }},
        {
        "interface": "Application",
        "connectionDetails": {
        "uri": "http://mysidecar.domain.org/inference"
        }}
        ]
        }
```

B.1.4 Inference Request Accession Number

```
        {
        "transactionID": "MY_TRANSACTION_ID",
        "responseURI": "http://app.endpoint",
        "priority": 128,
        "inputMetadata":
        {
        "type": "ACCESSION_NUMBER",
        "accessionNumber": "ABC123"
        },
        "inputResources": [
        {
        "interface": "DIMSE",
        "connectionDetails": {
        "aet": "MYPACSAET",
        "hostname": "mypacs.domain.org",
        "port": "104"
        }},
        {
        "interface": "DICOMweb",
        "connectionDetails": {
        "uri": "http://mypacs.domain.org:8000/dicom-web"
        }},
        {
        "interface": "FHIR",
        "connectionDetails": {
        "uri": "http://myehr.domain.org:8000/fhir"
        }},
        {
        "interface": "Application",
        "connectionDetails": {
        "uri": "http://mysidecar.domain.org/inference"
        }}
        ],
        "outputEndpoints": [
        {
        "interface": "DIMSE",
        "connectionDetails": {
```

```
784       "aet": "MYPACSAET",
          "hostname": "mypacs.domain.org",
786       "port": "104"
          }},
788       {
          "interface": "DICOMweb",
790       "connectionDetails": {
          "uri": http://mypacs.domain.org:8000/dicom-web
792       }},
          {
794       "interface": "FHIR",
          "connectionDetails": {
796       "uri": "http://myehr.domain.org:8000/fhir"
          }},
798       {
          "interface": "Application",
800       "connectionDetails": {
          "uri": "http://mysidecar.domain.org/inference"
802       }}
          ]
804       }
```

806 **B.1.5 Inference Request Response**

```
          HTTP/1.1 422 Unprocessable Content
808       Content-Type: application/json

810       {
            "type": "https://example.org/validation-error",
812         "title": "Your request parameters didn't validate.",
            "details": "API Key Missing"
814       }
```

816 **B.2 Inference Complete Request**

```
          {
818       "transactionID": "MY_TRANSACTION_ID",
          "status": 200,
820       "message": "Success",
          "resources": [
822       {
          "type": "DICOM_UID",
824       "studies": [
          {
826       "StudyInstanceUID": "2.16.840.1.114488.0.4.123489834087.13300714",
                "series": [{
828                   "SeriesInstanceUID":
          "2.16.840.1.114488.0.4.123489834087.13300714.2.1",
830                 "instances": [
                          "SOPInstanceUID":
832       ["2.16.840.1.114488.0.4.123489834087.5000"],
                               "frames": [
```

```
                                        "FrameNumber": ["1","2"]
                                    ]
                    ]
              }]
        }]
        },
        {
        "type": "FHIR",
        "resources": [{
              "resourceType": "Observation",
              "id": "98765"
        }]
        },
        ]
        }
```

B.2.1 Inference Complete Request Response

```
HTTP/1.1 200 OK
```

B.3 Liveness Request

```
GET /health/live
```

B.3.1 Liveness Request Response

```
HTTP/1.1 200 Success Content
Content-Type: application/json

{
  "status": "LIVE"
}
```

B.4 Readiness Request

```
GET /health/ready
```

B.4.1 Readiness Request Response

```
HTTP/1.1 200 Success Content
Content-Type: application/json

{
  "status": "READY"
}
```

B.5 Status Request

```
GET /inference/status/MY_TRANSACTION_ID
```

B.5.1 Status Request Response

```
HTTP/1.1 200 Success Content
Content-Type: application/json

{
```

```
        "details": "Completed"
    }
```

Changes to NEMA Standards Publication PS 3.17

Replace all of XX. Use Cases for Application Hosting

XX. Use Cases for Application Request

XX.1 Use Case – Simple

The inference endpoint of an Application provides a RESTful interface for triggering a new inference request that is compatible with the PS3.15 Application Request API. A client, knowing the Application capabilities and data input requirements, requests inference, specifies the output endpoint and has a response URI compliant with PS3.15 Section 8.1.

XX.1.1 Example

An interface engine, acting as a client, is configured to request inference of VendorX's Pneumothorax detection Application for every chest x-ray stored to VNA. VendorX provides an IHE, AIR (AI Result) compliant DICOM SR for all successful inference requests, which shall be stored to the local PACS.

The interface engine is aware of the VendorX API endpoint information and has been configured to pass inference requests to the API, requesting the results be stores in the VNA along with the chest x-ray.

1. A 2 View Chest X-Ray is received by the interface engine, which stores the imaging study to the VNA and in its roll as the Application client, creates an inference request to VendorX using the Accession Number.

```
    {
    "transactionID": "5150",
    "responseURI": "http://app.vendors.myhospital.org",
    "priority": 128,
    "inputMetadata":
    {
    "type": "ACCESSION_NUMBER",
    "accessionNumber": "ABC123"
    },
    "inputResources": [
    {
    "interface": "DIMSE",
    "connectionDetails": {
    "aet": "MYVNAAET",
    "hostname": "myvna.myhospital.org",
    "port": "104"
```

```
}}
],
"outputEndpoints": [
{
"interface": "DIMSE",
"connectionDetails": {
"aet": "MYVNAAET",
"hostname": "myvna.myhospital.org",
"port": "104"
}}
]
}
```

2. VendorX responds to the request that it has been received successfully.

```
HTTP/1.1 200 OK
```

3. VendorX issues a C-MOVE request to the VNA to retrieve the images from the 2 View Chest X-Ray for processing.

```
(0008,0052) CS [STUDY]      # Query/Retrieve Level
(0008,0050) SH [ABC123]     # Accession Number
```

4. Images are processed and VendorX creates a DICOM SR containing the results. This DICOM SR is sent to the VNA via C-Store and VendorX responds to the interface engine client URI provided in the request with details of the DICOM SR.

```
{
"transactionID": "5150",
"status": 200,
"message": "Success",
"resources": [
{
"type": "DICOM_UID",
"studies": [
{
"StudyInstanceUID": "2.16.840.1.114488.0.4.123489834087.13300714",
      "series": [{
            "SeriesInstanceUID":
"2.16.840.1.114488.0.4.123489834087.13300714.2.1",
            "instances": [
                  "SOPInstanceUID":
["2.16.840.1.114488.0.4.123489834087.5000"]
            ]
      }]
}]
}
]
}
```

XX.2 Use Case – Pipeline

The inference endpoint of an Application provides a RESTful interface for triggering a new inference request that is compatible with the PS3.15 Application Request API. A client, knowing the Application capabilities and data input requirements, includes additional endpoints to be used by the Application based on findings in its inference request, specifies the output endpoints, which contain the input to another Application and only supports checking inference status per PS3.15 Section 9.2.

XX.2.1 Example

A client is made aware of an Application which provides lung nodule detection on CT CHEST exams. This Application will use prior exams if available. Should the first Application detect any lung nodules, it passes its output, along with EHR endpoint information to a second Application to provide a LUNG-RADS assessment.

1. A client is made aware of a CT CHEST exam which was stored to the hospital's VNA. The client creates an inference request for the lung nodule detection Application using the Accession Number. This request contains FHIR endpoint information which will be passed along to the second Application for use in LUNG-RADS determination if needed as well as endpoint information for the second Application if needed.

```
{
"transactionID": "5151",
"priority": 128,
"inputMetadata":
{
"type": "ACCESSION_NUMBER",
"accessionNumber": "ABC124"
},
"inputResources": [
{
"interface": "DIMSE",
"connectionDetails": {
"aet": "MYVNAAET",
"hostname": "myvna.myhospital.org",
"port": "104"
}},
{
"interface": "FHIR",
"connectionDetails": {
"uri": "http://myehr.myhospital.org:8000/fhir"
}}
],
"outputEndpoints": [
{
"interface": "Application",
"connectionDetails": {
"uri": "http://mysidecar.myhospital.org/inference"
}},
{
"interface": "DIMSE",
```

```
"connectionDetails": {
"aet": "MYVNAAET",
"hostname": "myvna.myhospital.org",
"port": "104"
}},
{
"interface": "FHIR",
"connectionDetails": {
"uri": "http://myehr.myhospital.org:8000/fhir"
}}
]
}
```

2. The Application responds to the request that it has been received successfully.

```
HTTP/1.1 200 OK
```

3. The Application issues a C-FIND and C-MOVE requests to the VNA to retrieve the images and relevant priors from the CT CHEST exam for processing.

```
(0008,0052) CS [STUDY]      # Query/Retrieve Level
(0008,0050) SH [ABC124]     # Accession Number
```

4. Images are processed and the Application creates a DICOM SR containing the lung nodule detection results. This DICOM SR is sent to the VNA via C-Store and an inference request made of the second Application to provide a LUNG-RADS assessment.

```
{
"transactionID": "5152",
"responseURI": "http://lung_nodule.myhospital.org",
"priority": 128,
"inputMetadata":
{
"type": "ACCESSION_NUMBER",
"accessionNumber": "ABC124"
},
"inputResources": [
{
"interface": "FHIR",
"connectionDetails": {
"uri": "http://myehr.myhospital.org:8000/fhir"
}},
{
"interface": "Application",
"connectionDetails": {
"uri": "http://lung_nodule.myhospital.org/result"
}},
],
"outputEndpoints": [
{
"interface": "FHIR",
"connectionDetails": {
"uri": "http://myehr.myhospital.org:8000/fhir"
```

```
}}
]
}
```

5. The second Application responds to the request that it has been received successfully.

```
HTTP/1.1 200 OK
```

6. The original client making the request is made aware that the DICOM SR was stored to the VNA and requests the status of the inference request to the lung nodule detection Application.

```
GET /inference/status/5151
```

7. The lung nodule detection Application responds “InProcess” as the second Application has not yet completed.

```
HTTP/1.1 200 Success Content
Content-Type: application/json

{
  "details": "InProcess"
}
```

8. The second Application continues its process and needs smoking history for the patient from the EHR to finalize its LUNG-RADS assessment.

```
GET http://myehr.myhospital.org:8000/fhir/Observation?
patient=6786&code=72166-2,11367-0,68535-4/inference/status/5151
```

9. After completion of the LUNG-RADS assessment, the second Application creates a FHIR resource and stores it to the EHR’s FHIR endpoint and then informs the first Application, lung nodule which invoked this second Application to close out the task.

```
{
"transactionID": "5152",
"status": 200,
"message": "Success",
"resources": [
{
"type": "FHIR",
"resources": [{
      "resourceType": "Observation",
      "id": "987654"
}]
}
```

10. The original client making the request requests the status of the inference request to the lung nodule detection Application.

```
GET /inference/status/5151
```

11. The lung nodule detection Application responds “Completed” as now the second Application has completed its task and the complete pipeline has been executed.

```
HTTP/1.1 200 Success Content
Content-Type: application/json

{
  "details": "Completed"
}
```