MEDIRAD

Project title: Implications of Medical Low Dose Radiation Exposure

Grant Agreement Number: 755523

Call identifier: NFRP-2016-2017

Topic: NFRP-9

Deliverable D2.1

Implementation of the prototype of the central DICOM repository

Lead partner:	ITMI - UniGe
Author(s):	Osman Ratib, Nicolas Roduit, Joël Spaltenstein
Work Package:	WP2
Estimated delivery:	31/5/2018
Actual delivery:	1/6/2018
Type:	Report
Dissemination level:	Public

This project has received funding from the Euratom research and training programme 2014-2018 under grant agreement No 755523.

Table of contents

List of figures

Abbreviations

API	Application Programming Interface
BFO	Basic Formal Ontology
CID	(DICOM) Context Group Identifier
CT	Computed Tomography
CTDI	Computed Tomography Dose Index
DCM	DICOM Content Mapping resource
DICOM	Digital Imaging and Communications in Medicine
FHIR	Fast Healthcare Interoperability Resources
IOD	Information Object Definition
IRDBB	Image and Radiation Dose BioBank
IRI	Internationalized Resource Identifier
LOINC	Logical Observation Identifiers Names and Codes
NM	Nuclear Medicine
PACS	Picture Archiving and Communication System
PET	Positron Emission Tomography
RDF	Resource Description Framework
SNOMED	Systematized Nomenclature of Medicine
SPARQL	Simple Protocol And RDF Query Language
SPECT	Single Photon Emission Computed Tomography
SOP	(DICOM) Service Object Pair
SQL	Structured Query Language
SR	(DICOM) Structured Reporting
TID	(DICOM) Template Identifier
UML	Unified Modeling Language
W3C	World Wide Web Consortium
WADO	(DICOM) Web Access to DICOM Objects
XML	eXtensible Markup Language
XSD	XML Schema Definition

Technical terms glossary

DICOM Context Group Identifier (CGI)	DICOM Context Groups specify Value Set restrictions for the use of codes in the DICOM standard. DICOM Context Groups are referred to using Context Group Identifiers.
DICOM Content Mapping resource (DCM)	DICOM uses as much as possible existing terminology resources (such as SNOMED CT, LOINC, UCUM). However, DICOM manages its own terminology resource called DICOM Content Mapping resource.
Information Object Definition (IOD)	In DICOM, an Information Object Definition is an information object that can be exchanged between DICOM Application Entities.
Image and Radiation Dose BioBank (IRDBB)	The Image and Radiation Dose BioBank is a resource for managing image and dose data in an integrated way. IRDBB supports both DICOM data and non-DICOM data. The IRDBB software supports importation, internal management and query/retrieval of MEDIRAD research data.
Ontology Web Language (OWL)	The Ontology Web Language is one of the main standard languages developed by the World Wide Web Consortium (W3C) for representing ontologies. It is based on Description Logics (DL), a term that denotes a family of knowledge representation languages.
Resource Description Framework (RDF)	The Resource Description Framework (RDF) is a family of W3C specifications used as a general method for conceptual description or modeling of information that is implemented in web resources, using a variety of syntaxes. In practice, RDF is a directed, labeled graph data format for representing information in the Web, that is also used in knowledge management applications. [adapted from Wikipedia]
Simple Protocol And RDF Query Language (SPARQL)	The SPARQL specification defines the syntax and semantics of the SPARQL query language for RDF. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. SPARQL also supports extensible value testing and constraining queries by source RDF graph. The results of SPARQL queries can be results sets or RDF graphs. [adapted from Wikipedia]
DICOM Service Object Pair (SOP)	A DICOM Service Object Pair is a specification of a service in the DICOM standard. It associates an object class (i.e. an Information Object Definition) to a set of services that can be applied to this object.
DICOM Structured Reporting (SR)	DICOM Structured Reporting is a specification of a class of objects in the DICOM standard, based on a hierarchical organization of observations (called SR tree). DICOM SR is used for representing a broad range of documents such as reports of imaging procedures, procedure logs, reports of Computed Assisted Detection (CAD) analyzes, and radiation dose structured reports.
DICOM Template Identifier (TID)	A DICOM SR Template specifies the structure of a sub-graph of a DICOM SR tree. It is represented as a table describing the

	characteristics of each node (called content item) of the SR sub-graph, such as: nesting level, type of node (e.g. CODE, TEXT, PNAME, etc), type of relation with parent node, multiplicity (e.g. unique or multiple), requirement type (i.e. mandatory or optional), restrictions on content (for CODE nodes). DICOM SR Templates are referred to by means of DICOM Template Identifiers, e.g. to specify their use in a particular Structured Report IOD.
Unified Modeling Language (UML)	Unified Modeling Language is a general-purpose, developmental, modeling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system. Various kinds of models can be represented in UML, including Class Diagrams, Sequence Diagrams, Use Case Diagrams etc. [adapted from Wikipedia]
DICOM Web Access to DICOM Objects (WADO) or DICOMweb	DICOM Web Access to DICOM Objects (also called DICOMweb) is a term applied to the family of RESTful DICOM services defined for sending, retrieving and querying for medical images and related information. DICOMweb provide a light-weight mobile device and web browser friendly mechanism for accessing images, which can be implemented by developers who have minimal familiarity with the DICOM standard and which uses consumer application friendly mechanisms like http, JSON and media types (like "image/jpeg") to the maximum extent possible.The standard is formally defined in DICOM PS3.18 Web Services. [adapted from Wikipedia]
eXtensible Markup Language (XML)	The Extensible Markup Language (XML) is a markup language of the W3C that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services. Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many application programming interfaces (APIs) to aid the processing of XML data. [adapted from Wikipedia]
XML Schema Definition (XSD)	XSD (XML Schema Definition) is a recommendation of the W3C, that specifies how to formally describe the elements in an Extensible Markup Language (XML) document. It can be used by programmers to verify each piece of item content in a document. They can check if it adheres to the description of the element it is placed in. Like all XML schema languages, XSD can be used to express a set of rules to which an XML document must conform in order to be considered "valid" according to that schema. [adapted from Wikipedia]

1. Introduction

This deliverable describes the progress of development and implementation of the DICOM repository that will host the images and data of the *Image and Radiation Dose BioBank* (IRDBB). This development of the ITMI team of the University of Geneva was undertaken in close collaboration with the INSERM team from Rennes with the support of two partners: b<>com who developed the RDF repository component (also called Metadata repository) and Vital-IT team from Geneva who assisted in the set up and implementation of the server.

This deliverable also describes the methodology for the development and integration of different software components into the IRDBB system. It also provides the basic description of software interfaces for developer and system user interface.

2. Content

2.1 Introduction

2.1.1 Specifications and functionalities for the DICOM repository

2.1.1.1 System architecture and overall description

The *DICOM repository* implemented in Geneva is a large database storage server that will host DICOM-compliant imaging data but also an *RDF repository* (Resource Description Framework), supporting the semantic descriptions of both DICOM and non-DICOM data (developed by the INSERM team in Rennes). The goal of this server is to provide an integrated repository of dosimetric and image data (called *Image and Radiation Dose Biobank* or IRDBB). This system will support the storage and sharing of image and radiation dose data that are produced and used in the Clinical research studies involved in the MEDIRAD project.

This repository will be composed of two main components:

1) a *DICOM data repository* suitable for managing radiological images and Radiation Dose Structured Reports, and
2) a *RDF repository*, supporting the semantic (i.e. ontology-based) descriptions of both DICOM and non-DICOM data, and facilitating the querying of the IRDBB repository.

The two components will reside on the same server in Geneva, but duplicate of the two software platforms will also be implemented at INSERM in Rennes for development and testing.

All the software components are designed to be easily transferred to other servers at the end of the project allowing other sites to implement a similar server locally in their institution.

2.1.1.2 DICOM repository architecture

The DICOM repository is a standards DICOM-compliant image database that supports standard DICOM query and retrieve protocols and provides storage of imaging data in DICOM format. It is only intended as storage for “anonymized” imaging data. Imaging arrives can be partitioned un subsets of PACS servers for clusters of users on specific sets of data.

Hardware platform

The main IT infrastructure will be hosted in a new research facility, the “Campus Biotech” in Geneva (http://campusbiotech.ch), a large philanthropic foundation under the leadership of three universities (UniGE, Unil, EPFL) linked to the two main university hospitals of Geneva and Lausanne.

The server configuration as described below provides high-end computing performance and fast storage access while offering reliable up-time of the services. The main configuration consist of the HSM (Hierarchical Storage Management) that allows near-line management of **more than 7 PB of imaging-data** at a low cost and low energy consumption.

The current configuration installed at Campus Biotech facility consist of a central unit of 16 dual-processor machines **(E5-2680-v3)** with 128 GB of RAM and 960 GB of SSD storage. The frontal server is also a dual processor (E5-2680-v3) unit with 256 GB of RAM each and four 960GB SSD disks. The storage unit (HSM) is currently at its minimum capacity of 256 TB extensible to 7 PB in near-inline storage. A backup HSM tape storage is installed to ensure continuing security backups. On full operation a second remote HSM tape storage is planned for redundant off-site backups.

The current unit can be configured in number of “virtual machines” that can be tailored to the amount of CPU performance and storage capacity depending on the requirement of individual projects.

Software platform

The PACS repository is based on Open-Source platform dcm4che (http://www.dcm4che.org), dcm4che is a collection of open source applications and utilities developed in Java programming language for performance and portability.

A specially designed “Web-portal” platform is designed to allow access to the server settings for remote configuration and management. The web-portal can be accessed through standard Web browsers (through a specific URL provided by the system manager).

To configure a new node the system manager can connect to the server from a web browser, enter the URL of the web portal and add a new AE Title (AET) of the destination node. A special security protocol allows to identify and authorize specific nodes from different external centers.

The general architecture of the system which has the code name of KHEOPS) is an open system that allows input and output of imaging data from different sources, either PACS system from different sources, or directly from imaging modalities. It supports different DICON-compliant protocols including WADO, which is the web-based DICOM protocol.

2.1.1.3 RDF (Metadata) repository architecture

The design of the RDF repository took into account the IRDBB architecture, defined in collaboration with b<>com and ITMI, and described in deliverable D 2.2 provided by the INSERM group.

2.1.2.1 Integration of DICOM and RDF storage architecture

2.1.2.1.1 Global IRDBB architecture

Figure 1 details the proposed architecture of the IRDBB system. The metadata repository will consist in a RDF database implemented in a RDF Triple store. This RDF database will contain data that are instances of the classes of the ontology as well as relationships connecting these instances.

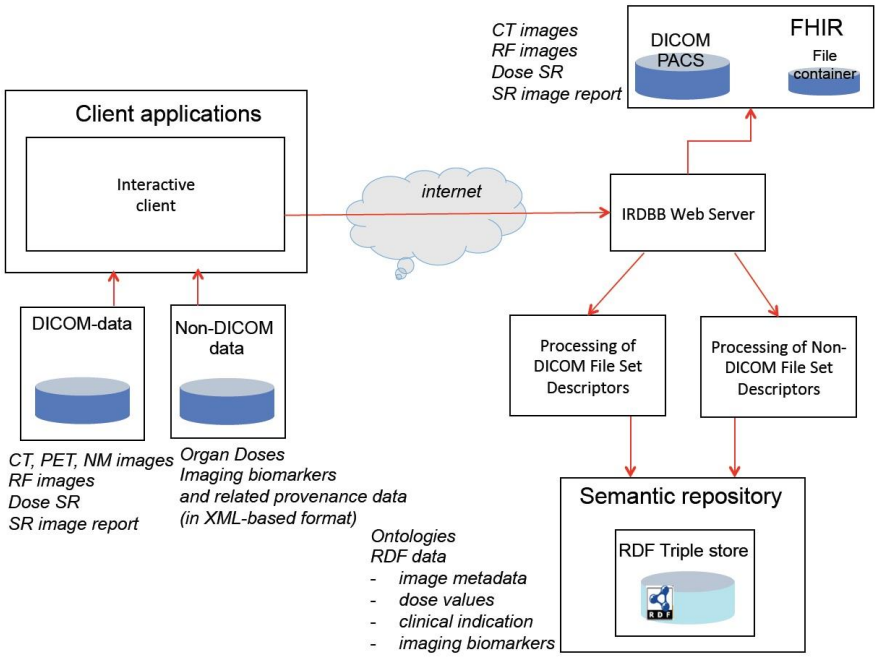


Figure 1. Proposed architecture of the IRDBB system

The IRDBB system is represented in the right part of Figure 1. It is composed of two main components: a *PACS*, managing the DICOM data, and the *Metadata repository*, supporting the semantic database, hosted on a RDF Triple store. Both are fed through a Web server (IRDBB Web Server) in interaction with the client applications, which function is to import or query and retrieve the image and dosimetric data.

2.1.2.1.2 DICOM server control and configuration

A specially designed "Web-portal" allows access to the DICOM database settings for remote configuration and management. The web-portal can be accessed through standard Web browsers (through a specific URL provided by the system manager).

To configure a new node, the system administrator must connect to the server from a web browser, enter the URL of the web portal and you credentials. Then add a new AE Title (AET).

The receiving workstation must be configured to match the parameters to the listener of the workstation. For instance, in Osirix, this information is available in Osirix > Preferences… > Listener

2.1.2.1.3 IRDBB import workflow

The general workflow of data import software components is shown in the diagram below. The software components in the yellow box at the bottom of the diagram reside directly on the database server.

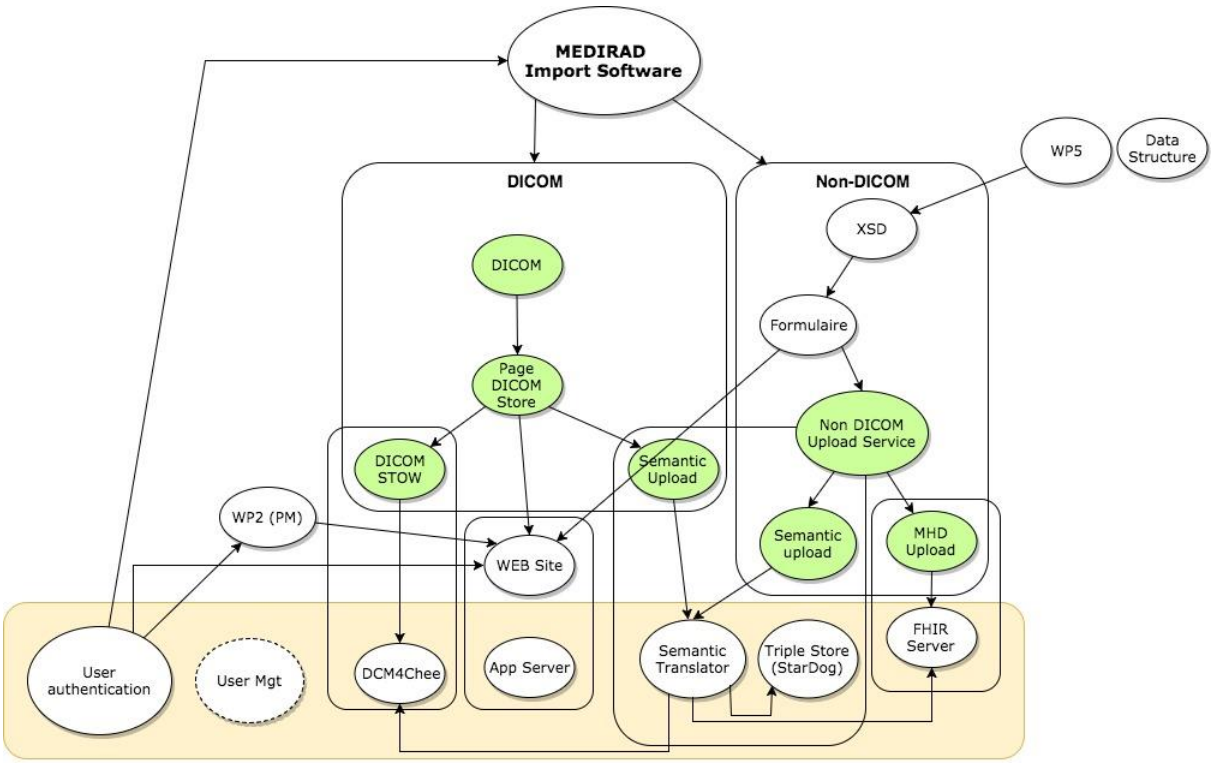


Figure 2. IRDBB data import software components and workflow

2.1.2.1.4 Data and files management

All images and related data will be imported into IRDBB in *File sets*. Each File set pertains to one patient, and to his/her participation in one Clinical research study. Two kinds of File sets are distinguished:

- *DICOM File sets*, that are composed of DICOM files only and will be indexed in the DICOM database (DCM4CHE)
- *Non-DICOM File sets*, that are composed of:
    - non-DICOM files (files that will be stored in the IRDBB system but their detailed content will not be stored in the Metadata repository)
    - and of one XML Description file (containing the detailed data to be stored into the Metadata repository).

Management of DICOM File sets: The importation software will import DICOM files into the DICOM repository and will automatically create an XML description file (called *DICOM File set Descriptor*). As explained in [2] "this XML Description File is used to inform the Metadata repository which DICOM files have been uploaded to the PACS and to which Clinical research study they should be associated. This XML Description file will contain: 1) the essential reference links with the Clinical research study and between DICOM entities; 2) the essential DICOM data items and metadata items". This content will be translated into RDF and stored in the RDF repository.

Management of Non-DICOM File sets: The importation software will import non-DICOM files into the file manager of the Metadata repository. It will import the XML description file (called *Non-DICOM File set Descriptor*) into the Metadata repository. This content will be translated into RDF and stored in the RDF repository.

2.2 Progress report

2.2.1 Design of the PACS server architecture

The IRDBB server architecture included several software components that are interlinked to allow user authentication, user access, data transfer, and data retrieval through a set of modules that connect to the PACS database (dcm4che open source DICOM archive management system).

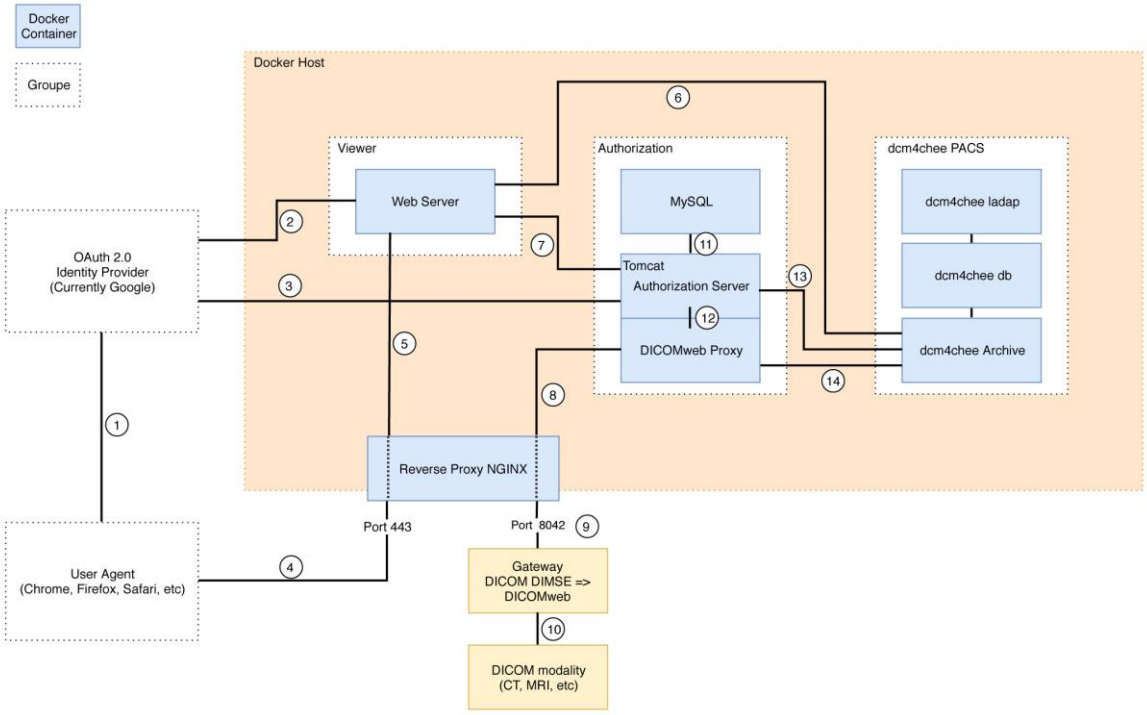


Figure 3. General architecture of the software components of the PACS database server

2.2.2 Description of software components

2.2.2.1 OAuth 2.0 Identity Provider

Referred to as an "Authorization Server" in RFC6749. This is a service that authenticates and provides the identity of Users. Any OpenID Connect compliant Identity Provider can be used. The current implementation uses Google as the Identity Provider.

2.2.2.2 User Agent

The User's web browser (Chrome, Firefox, Safari, etc.)

2.2.2.3 Reverse Proxy (NGINX)

Provides security between the platform and the outside world. All incoming traffic goes through the Reverse Proxy. Provides TLS (https) for all incoming connections.

2.2.2.4 Web Server

Based on a modified version of OHIF image viewing platform (http://ohif.org). The user interface provides:

- A list of studies a User has access to.
- Basic search options.
- The OHIF viewer for visualization.
- Sending studies to other users.

- Upload DICOM files using the browser.
- Download DICOM files to a .zip of DICOM files.
- Removal of studies from the User's list.

Currently the Web Server serves as a Proxy to access the dcm4chee Archive and validates that the user making the request has access to the study/series. In future versions of the platform the dcm4chee archive will be protected. Since it will be safe to give direct access to the dcm4chee server, the proxy capability of the Web Server will be deprecated.

2.2.2.5 Authorization server

Maintains the list of access authorizations to series for each User. Provides QIDO and WADO services that require knowledge of which series a User has access to (all QIDO requests and WADO requests such as metadata retrieval at the study level). Provides a REST API to share studies with other Users. Provides a REST API to obtain secret Capability Tokens that the User can use in DICOMweb Proxy URLs. Currently served by an Apache Tomcat instance shared with the DICOMweb Proxy.

2.2.2.6 DICOMweb Proxy

Provides DICOMweb STOW services to Service URLs that contain Capability Tokens. Instances stored to the proxy are added to the appropriate User's list of Series. Currently served by an Apache Tomcat instance shared with the Authorization Server.

2.2.2.7 Links between entities

1. The User Agent (web browser) redirects the User to the OAuth 2.0 Identity Provider (referred to as the Authorization Server in RFC6749) so that the User can provide login credentials. Once done, Identity Provider will then redirect the User Agent to a resource on the Web Server and provide an Authorization Code as a query parameter. Login credential are never given to any entity other than the Identity Provider.
2. The Web Server connects to the OAuth 2.0 Authorization Server, and presents the Authorization Code along with a shared secret in order to obtain an Assertion (referred to as the Access Token in RFC6749)
3. The Authorization Server connects to the Identity Provider to retrieve the RSA Public key needed to validate the Assertion.

4/5. The User Agent connects to the Web Server (through the NGINX proxy that provides TLS, ie. https). The Web Server provides the User Agent with all displayed content.

6. The Web Server connects to the dcm4chee PACS when it needs to serve DICOM content to the User Agent.

7. The Web Server connects to the Authorization Server in order to find what series and studies a User has access to.

8/9/10. A DICOM AE can send DICOM instances to the Gateway using a DIMSE C-Store operation. The Gateway then forwards the DICOM instances to the DICOMweb proxy (through the NGINX proxy that provides TLS, ie. https) using a DICOMweb STOW operation. All data is streamed so that any errors are reported to the DICOM AE.

11. The Authorization Server connects to the MySQL to store and retrieve it's data.

12. The DICOMweb Proxy connects to the Authorization Server in order to obtain and Access Token for the User based on the Capability path (part of the URL that is kept secret) of the URL to which the STOW request was made. The DICOMweb Proxy will then use this Access Token to connect to the Authorization server again in order to add the stored series to the list of series the User has access to.

13. The Authorization Server connects to the DICOMweb server in order to retrieve metadata necessary in order to provide QIDO services.
14. The DICOMweb Proxy connects to the dcm4chee PACS in order to store DICOM instances.

2.2.3 Data navigation cockpit

The set of software components are assembled into an interactive user interface (referred in our project as the *Râ-cockpit*) that allows the users to retrieve and navigate through image datasets and also initiate and perform data transfers from and to different users located on different nodes of the network.

2.2.3.1 Data access and navigation platform

The web interface provided allow users to access and view image series by performing the following steps (currently using Google's authentication server for demonstration):

- The User connects to the Web Server, and clicks on "Connect with Google"
- The User is redirected by the Web Server to Google's authentication server.
- The User enters the appropriate credentials.
- The User is redirected by Google's authentication server to the Web Server using a URL with an Authorization Code as a URL parameter.
- The Web Server connects with Google's authentications server and exchanges the Authorization Code for an Assertion.
- The Web Server connects with the Authorization Server and provides the Assertion.
- The Authorization Server connects with Google's authentication server to recover Google's RSA public key.
- The Authorization Server validates the Assertion using the RSA public key and returns an Access Token to the Web Server.
- The Web Server issues a QIDO request to the User's service URL on the Authorization Server and provides the Access Token as credentials.
- The Authorizations server looks up which series/studies the User has access to on the MySQL database, and returns a list of studies.
- The Web Server displays the list of studies to the User.
- The User selects a study to visualize.
- The User's browser makes WADO requests to the Web Server.
- The Web Server Connects to the Authentication Server to obtain an Access Token for the study/series specified in the WADO request.
- The Web Server Proxies the WADO request to the dcm4chee Archive, which will then return the instances to the User's browser.

2.2.3.2 Image viewer

The current user cockpit provides a modified version of the Open Source image viewing OHIF (http://ohif.org). OHIF is an HTML5 zero-footprint (no software installed on the user's side) that provide image viewing and manipulation tools on any hardware platform (Mac, Windows, iOS tablets or iPhone, Android etc.). This viewer is integrated to our platform allowing users to visualize and navigate through the images using a standard web browser.

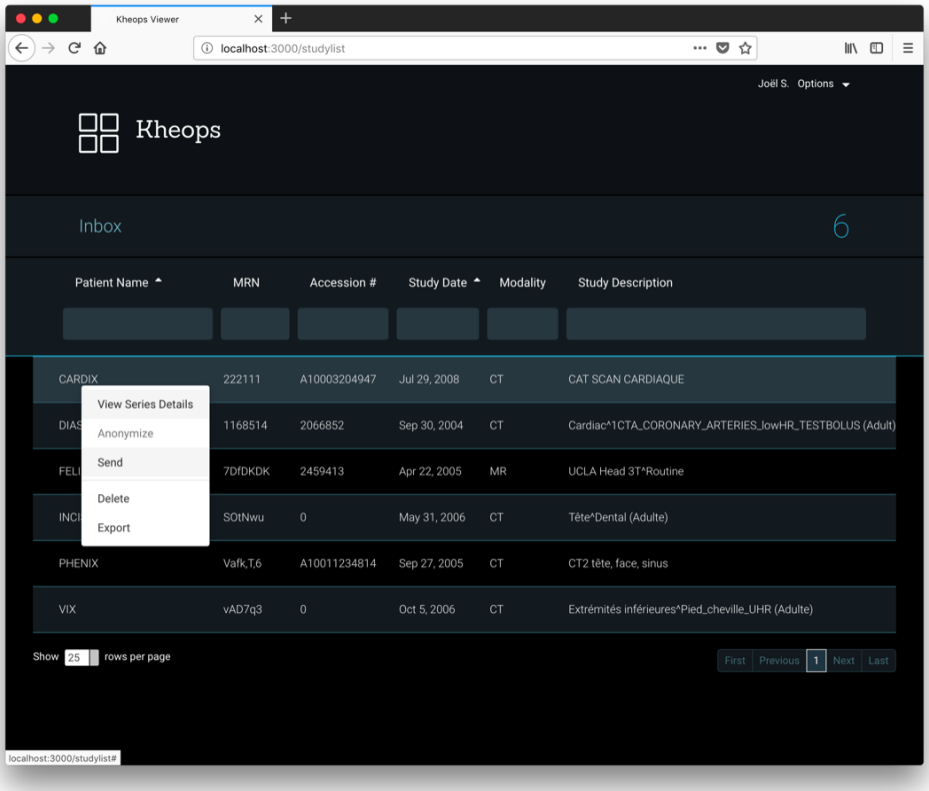


Figure 4. Selection dialog window of studies and series

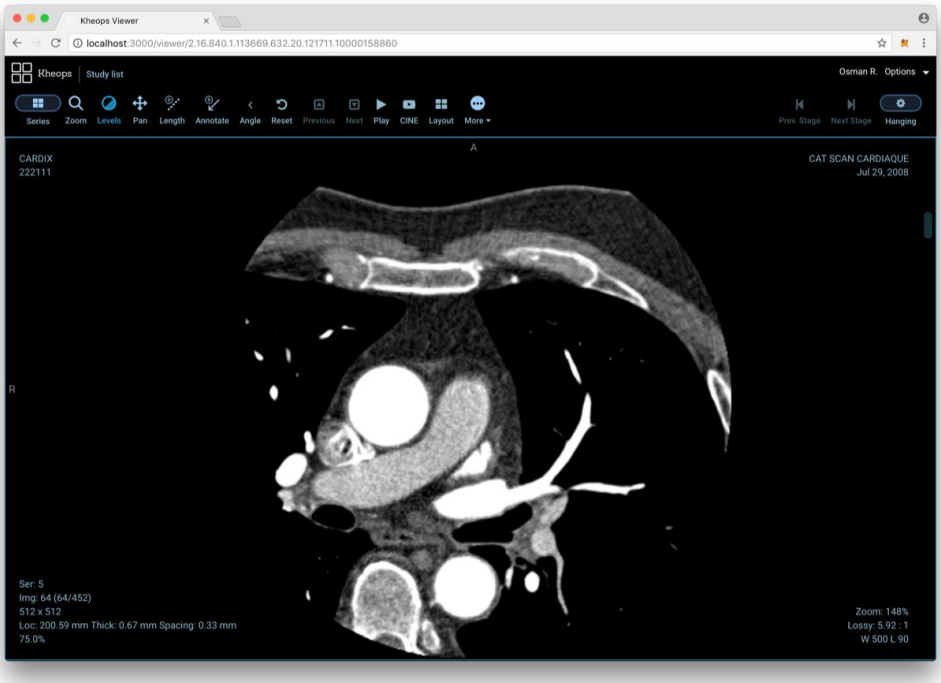


Figure 5. User interface of the OHIF viewer for image display and processing

2.2.3 Implementation and testing on the central server (KHEOPS)

A demonstration version of the software components was implemented and tested on the central server (KHEOPS) in Geneva. A copy of the software components was also installed on a local server at the INSERM site in Rennes.

2.2.4 Integration of the DRF components from (INSERM software)

The implementation and integration of the different software components relies on a set of RESTful Services packaged in Docker[1] containers and communicating through http. The list of these components are shown in figure X below. The Docker solution for software installation is a simple and robust method to implement software on different virtual machines (VM). It will greatly facilitate the transfer of these software components to other sites on local servers when needed at the end of the project.

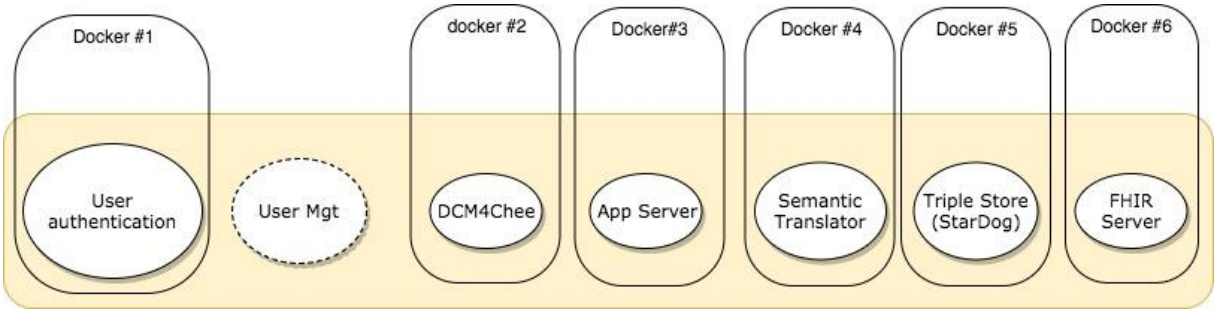


Figure 6. DRF software components (INSERM software) implemented through Docker containers

2.3 Work in progress

2.3.1 Integration and testing of software components

The software DRF components provided by the INSERM team are being finalized and implemented on the central server (KHEOPS) for further validation and testing. Remote access for uploading and downloading image sets as well as non-DICVOM data files will be tested.

The next steps will consist in interacting closely with all the project partners (of WP2, WP3 and WP5) in charge of these subtasks, in order to agree on the information to be managed, through the specification of the XML schemas (.XSD file) to be used to populate the XML DICOM and Non-DICOM File set descriptors pertaining to their specific research data.

Besides, a discussion has started with UMC-Mainz to explore how the structured reports considered in Subtask 2.4.3 should be described in the RDF repository. Indeed, it is clearly envisaged that the reports can be stored as DICOM or Non-DICOM files in the IRDBB system, but their level of semantic description in the RDF repository remains to be defined.

2.3.2 Development of user management platform

An advanced platform for user authentications and management of different user's profiles with specific access to subsets of data is being planned. This platform will allow to create communities of

[1] https://www.docker.com/

users and facilitate data and information exchange between different individuals of a group or between separate research groups.

This development will be undertaken in collaboration with the development team of Vital-IT () that has already a long experience in managing large international genomic and proteomic databases that are hosted by the Swiss Institute of Bioinformatics (). A subcontractor agreement is in progress of being finalized to allow the Vital-IT team to participate and allocate the development resources for this task

3. Conclusion

The development and implementation the IRDBB central repository is on good way. The DICOM server is already implemented and the general architecture and global framework is well defined. The development of the RDF repository platform is being consolidated but is highly dependent on the choices of the MEDIRAD users and on the Clinical research projects which type of data will eventually be managed in the IRDBB system. The current RDF components are being implemented and integrated in the central server in Geneva, and testing form remote sites will be conducted in the upcoming months to test both DICOM and non-DICOM data upload and retrieval.

To be able to finalize the specifications and implementation of the RDF components, it is of key importance that the MEDIRAD partners who plan to use the IRDBB system to manage their research data collaborate actively with those involved in the IRDBB development (i.e. INSERM, b<>com and ITMI), so that:

1. to specify in detail what information has to be managed, and so that
2. they can produce the XML File set descriptors that are needed to document the meaning and provenance of research data files.

In this respect, the M12-M18 time period is critical because it is a period in which development resources still exist to further develop and test the IRDBB system, which will be much harder after M24 (Fig. 7).

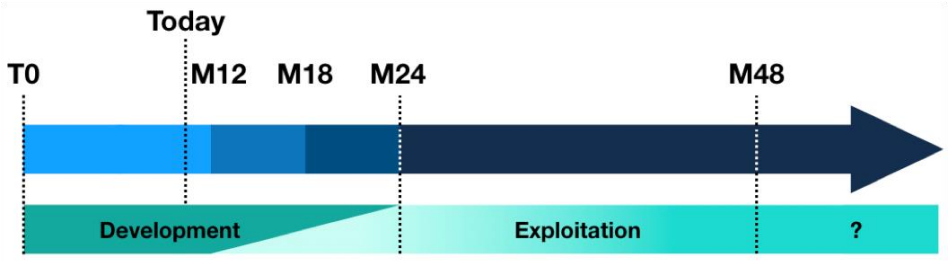
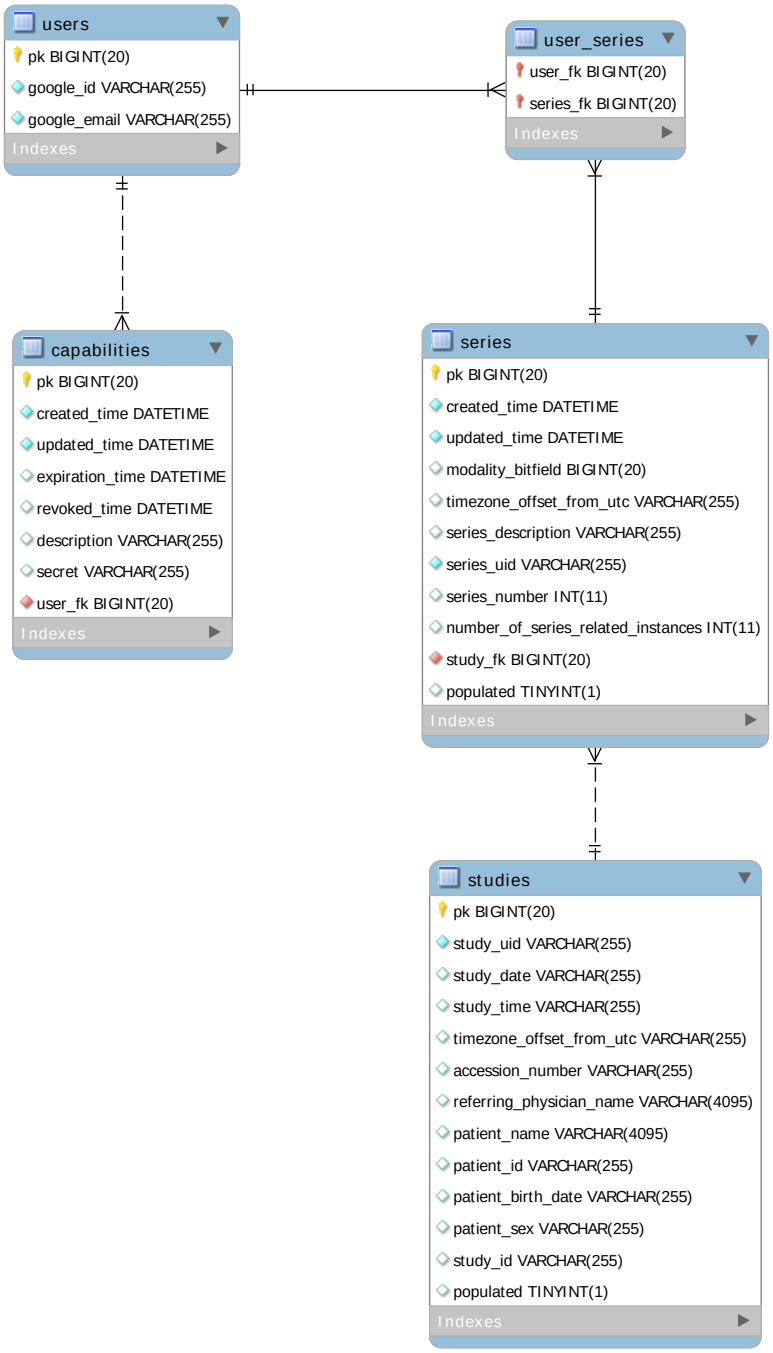


Figure 7. Scheduling of development and exploitation of the IRDBB system

ANNEX 1

This annex provides the hierarchical structure of the imaging database

Database structure



ANNEX 2

This annex provides the authentication server API Documentation

Authentication API documentation

Authentication Server API Documentation

Documentation of the Authentication Server REST API to be used as functional specification.

The authentication server APIs are based on the following RFCs.

Relevant RFCs

Core OAuth 2.0

- RFC 6749: The OAuth 2.0 Authorization Framework
- RFC 6750: The OAuth 2.0 Authorization Framework: Bearer Token Usage
- RFC 6819: OAuth 2.0 Threat Model and Security Considerations
- RFC 7235: OAuth 2.0 Hypertext Transfer Protocol (HTTP/1.1): Authentication

OAuth 2.0 Grant Requests from Assertions

- RFC 7521: Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants
- RFC 7522: Security Assertion Markup Language (SAML) 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants
- RFC 7523: JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants

Other

- RFC 7519: JSON Web Token (JWT)
- RFC 7520: Examples of Protecting Content Using JSON Object Signing and Encryption (JOSE)
- RFC 7515: JSON Web Signature (JWS)

Relevant DICOMweb Specifications

- DICOMweb
- WADO-RS Request/Response
- STOW-RS Request/Response
- QIDO-RS Request/Response
- Registry of DICOM Data Elements

Authentication API

User authentication is out of the scope of the *Authorization Server*. The *Authentication Server* will assume that the client has already been authenticated and is in possession of a SAML assertion or a JWT token provided by an authentication service.

Based on this assertion, or an *Authorization Server* provided *Capability Token*, the user can use this API to obtain *Access Tokens* that can then be used to access secured resources on this server and on the DICOMweb server.

- Request Access Token : `POST /token`

Study List

Since the DICOMWeb server doesnot keep a stateful representation of which users have access to which studies/series, it is the responsibility of the *Authorization Server* to provide both query, and study-level metadata resources.

APIs used to request a list of studies are based on QIDO-RS. Only subset of the QIDO specification is implemented.

- Get a list of studies : `GET /users/{user}/studies`

  Requires an Access Token with the user as the sub claim.

  Without specific non-QIDO parameters, this resource will return all studies the user has access to, including the user's *Inbox*, and all studies within *Albums* the user has access to.

Study-level metadata WADO-RS queries follow the WADO-RS specification.

- Get metadata for a study : `GET /{user}/studies/{studyInstanceUID}/metadata`

  Requires an Access Token with the user as the sub claim.

Will return metadata for the series the user has access to.

Working with the *Inbox*

While these resources are very similar in appearance to STOR-RS resources, they don't accept DICOM objects, and should be considered to be entirely different.

- Give a user access to a study : `PUT /users/{user}/studies/{StudyInstanceUID}`

  Requires an Access Token with a sub claim that specifies a user who has access to at least one series in the study.

  This resource can be used by a user that has access to at least one series of the specified study

to share all accessible series with another user.

- Give a user access to a specific series : `PUT /users/{user}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}`

  Requires an Access Token with a sub claim of a user that has access to the specified series, or an Access Token with the user as the sub claim.

  This resource can be used by a user that has access to a series to send the series to another user.

  Alternatively, this resource can be used by a user to claim access to a series UID that is unknown to the authorization server. The user will then be able to get an Access Token that can be used to do an STOW-RS to the DICOMweb server.

- Remove a study from the *Inbox* : `DELETE /users/{user}/studies/{StudyInstanceIUD}`

  Requires an Access Token with the user as the sub claim.

  Relinquishes access to all series in the specified study.

- Remove a specific series from the *Inbox* : `DELETE /users/{user}/studies/{StudyInstanceIUD}/series/{SeriesInstanceUID}`

  Requires an Access Token with the user as the sub claim.

  Relinquishes access to the specified series.

DICOMweb Capability Tokens

A DICOMweb proxy will provide access to a specific user's content using a service URL that will identify a specific user. For the time being this proxy will be built into the *Authenitication Server*, but in time it may be moved to it's own server. These resources give the user the ability to manage *Capability Tokens*.

- Create a Capability Token : `POST /users/{user}/capabilities`

  Requires an Access Token with the user* specified as the sub claim.*

  Creates a new *Capability Token*.

- Get Capability Tokens : `GET /users/{user}/capabilities`

  Requires an Access Token with the user* specified as the sub claim.*

  Returns a list of the the user's *Capability Tokens*.

- Revoke a Capability Token : `POST users/{user}/capabilities/{secret}/revoke`

  Requires an Access Token with the user* specified as the sub claim.*

Webhooks

Not yet implemented

The *Authentication Server* will need to implement webhooks in order to accomplish tasks such as providing notifications when new series are available for a user. This will be done by implementing an IANSCP that will respond to updates from the DICOM server. The *Authentication Server* will also need to respond to these notifications in order to know to query the DICOM database to update series metadata. Until this is implemented, the *Authorization Server* will poll the DICOMweb server at regular intervals to retrieve study and series metadata.

Working with *Albums*

Not yet implemented

Albums are groups of studies.

- Create an album : `POST /albums`

  Requires an Access Token with a user* specified as the sub claim.*

  The user who creates the album is automatically set as admin.
- Delete an album : `DELETE /albums/{album}`

  Requires an Access Token with admin access to the album.
- Add a study : `PUT /albums/{album}/studies/{StudyInstanceIUD}`

  Requires an Access Token with admin access to the album.
- Remove a study : `DELETE /albums/{album}/studies/{StudyInstanceIUD}`

  Requires an Access Token with admin access to the album.
- Retrieve studies using the study list resource.
- Add a user : `PUT /albums/{album}/users/{user}`

  Requires an Access Token with admin access to the album.
- Remove a user : `DELETE /albums/{album}/users/{user}`

  *Requires an Access Token with admin access to the album, or an Access Token with the **user** as the sub claim.*
- Get a list of users : `GET /albums/{album}/users`

  Requires an Access Token with access to the album.
- Upgrade a user to admin : `PUT /albums/{album}/users/{user}/admin`

  Requires an Access Token with admin access to the album.

ANNEX 3

This annex provides the specification of different essential web services calls.

Web services calls

Give a user access to all series in a study

This resource can be used by a user that has access to at least one series of the specified study to share all accessible series with another user.

URL : `/users/{user}/studies/{StudyInstanceUID}`

Method : `PUT`

Auth Required : Requires an Access Token with a *sub* claim that specifies a user that has access to at least on series in the study.

Success Response

If the study was shared successfully.

- **Status** : `204 No Content`

Error Response

If validation of the token fails:

- **Status** : `400 Bad Request`

If no authentication token is supplied: (Don't forget to return the WWW-authenticate header)

- **Status** : `401 Unauthorized`

If the user specified in the sub claim does not have access to any series of the specified study.

- **Status** : `403 Forbidden`

Give a user access to a specific series

This resource can be used by a user that has access to a series to send the series to another user.

Alternatively, this resource can be used by a user to claim access to a series UID that is unknown to the authorization server. The user will then be able to get an Access Token that can be used to do an STOW-RS to the DICOMweb server.

URL : `/users/{user}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}`

Method : `PUT`

Auth Required : Requires an Access Token with a sub claim of a user that has access to the specified series, or an Access Token with the user as the sub claim.

Success Response

If the user bearer Access Token has a *sub* claim for a user with access to the series and the series was shared successfully.

- **Status** : `204 No Content`

If the bearer token belongs to the user to which the series is being added, and the SeriesInstanceUID is not known to the server.

- **Status** : `201 Created`

Error Response

If validation of the token fails:

- **Status** : `400 Bad Request`

If no authentication token is supplied: (Don't forget to return the WWW-authenticate header)

- **Status** : `401 Unauthorized`

If the bearer token does not give access to the SeriesInstanceUID, or if the user is trying to claim access to a SeriesInstanceUID that is already known to the *Authentication Server*.

- **Status** : `403 Forbidden`

Study Metadata

Used to get a study metadata for a given user. This API should conform to WADO-RS in the sense that a WADO-RS client capable of sending an appropriate bearer token should be able to call this resource directly.

This

URL : `/users/{user}/studies/{StudyInstanceUID}/metadata`

Method : `GET`

Auth Required : Authorization with a JWT Bearer token with the user as the *sub* claim.

Study List

Used to get a list of studies for a given user. This API should conform to QIDO-RS in the sense that a QIDO-RS client doing a study search, and capable of sending an appropriate bearer token, should be able to call this resource directly.

The original implementation will only support a very limited subset of parameters. Future versions may implement more options

URL : `/users/{user}/studies`

Method : `GET`

Auth Required : Authorization with a JWT Bearer token with the user as the *sub* claim.

Headers

- `Accept` : If present, require that this value be `application/dicom+json` *optional*

URL Parameters

Only accept the follwing QIDO-RS parameters

- `{attributeID}={value}`

  Only accept the following attributeIDs as parameters:

  - `StudyInstanceUID` or `00080020`
  - `PatientID` or `00100020`
  - `PatientName` or `00100010`
  - `PatientSex` or `00100040`
  - `PatientBirthDate` or `00100030`
  - `StudyDate` or `00080020`
  - `StudyTime` or `00080030`
  - `AccessionNumber` or `00080050`
- `limit={limit}`
- `offset={offset}`

Custom Parameters
Not yet implemented

- `albums=<albumName>` : Filter the returned studies to only include series that are in the specified album.
- `inbox` : Filter returned studies to only include series in the inbox.

Removes a study from the user's *Inbox*

Relinquishes access to all series in the specified study.

URL : `/users/{user}/studies/{StudyInstanceUID}/series/{SeriesInstanceUID}`

Method : `DELETE`

Auth Required : Requires an Access Token with the user as the sub claim.

Success Response

If the bearer token has user as the *sub*.

- **Status** : `204 No Content`

Error Response

If validation of the token fails:

- **Status** : `400 Bad Request`

If no authentication token is supplied: (Don't forget to return the WWW-authenticate header)

- **Status** : `401 Unauthorized`

If the bearer token does not have the user as the *sub* claim.

- **Status** : `403 Forbidden`

Removes a specific series from the user's *Inbox*

Relinquishes access to the specified series.

URL : `/users/{user}/studies/{StudyInstanceUID}series/{SeriesInstanceUID}`

Method : `DELETE`

Auth Required : Requires an Access Token with the user as the sub claim.

Success Response

If the bearer token has user as the *sub*.

- **Status** : `204 No Content`

Error Response

If validation of the token fails:

- **Status** : `400 Bad Request`

If no authentication token is supplied: (Don't forget to return the WWW-authenticate header)

- **Status** : `401 Unauthorized`

If the bearer token does not have the user as the *sub* claim.

- **Status** : `403 Forbidden`