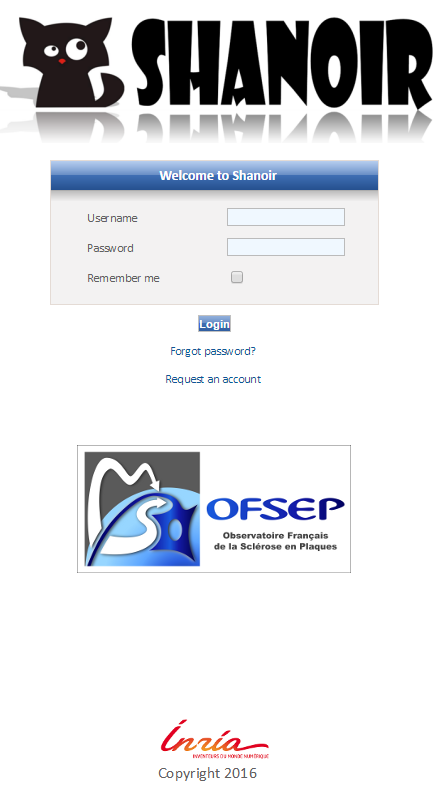
ShanoirUploader Documentation

Shanoir version 1.1.1

ShanoirUploader version 4.0



# ShanoirUploader V4

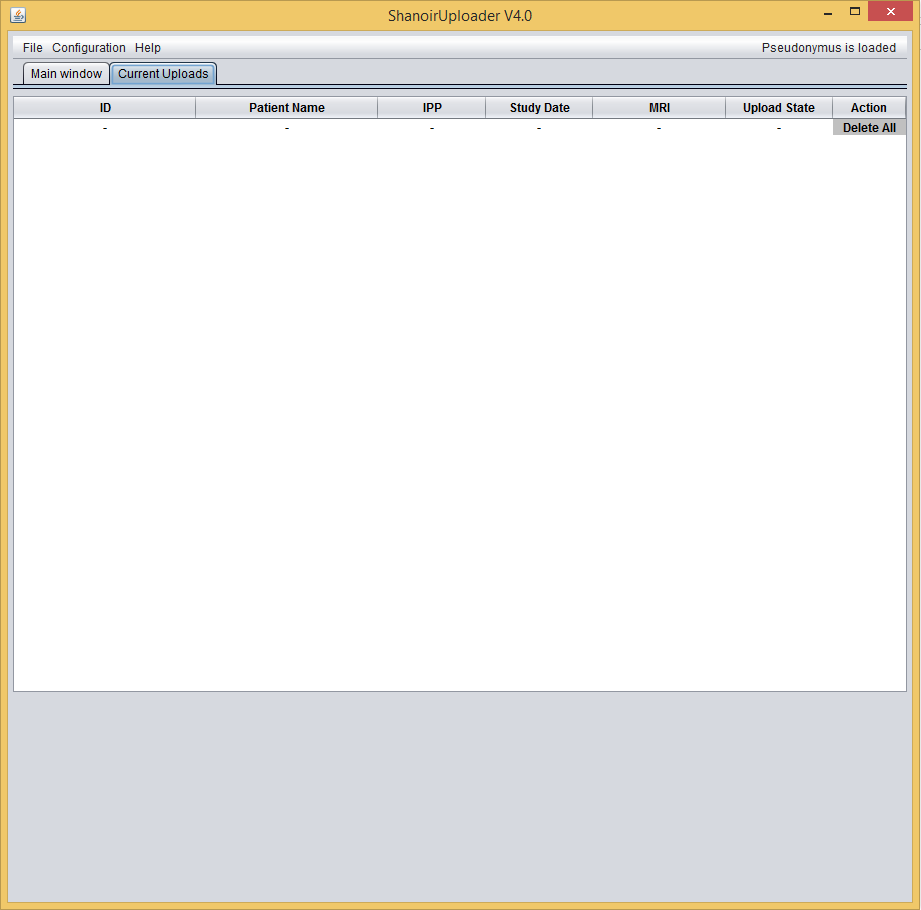


Table of contents

[2](#_Toc481162161)

[Context 5](#_Toc481162162)

[Goal 5](#_Toc481162163)

[Version 5](#_Toc481162164)

[Conventions 5](#_Toc481162165)

[Abbreviations 5](#_Toc481162166)

[Shanoir Uploader description 6](#_Toc481162167)

[What is the ShanoirUploader? 6](#_Toc481162168)

[Workflows 7](#_Toc481162169)

[Installation and configuration 7](#_Toc481162170)

[Shanoir Uploader features 9](#_Toc481162171)

[Use case for the user on SU 14](#_Toc481162172)

[Technical functions of ShanoirUploader 15](#_Toc481162173)

[Reading properties files 15](#_Toc481162174)

[DICOM server echo 15](#_Toc481162175)

[Shanoir server echo 15](#_Toc481162176)

[DICOM server query & retrieve 15](#_Toc481162177)

[Shanoir server upload 15](#_Toc481162178)

[Logging 16](#_Toc481162179)

[Finishing import from Shanoir Uploader on Shanoir Platform 17](#_Toc481162180)

[Subject creation/selection 18](#_Toc481162181)

[Neurinfo Algorithm 21](#_Toc481162182)

[OFSEP Algorithm 21](#_Toc481162183)

[Integrating indexation part in Shanoir Uploader 22](#_Toc481162184)

[Selection of a study 22](#_Toc481162185)

[Selection of a study card 22](#_Toc481162186)

[Selection of a subject 22](#_Toc481162187)

[StudyCard conversion 22](#_Toc481162188)

# Context

## Goal

This technical specification collects all information regarding the data import and export of the Shanoir platform, from a technical point of view. It helps to understand already existing functions better and to document them.

## Version

This document is version 2.0.

## Conventions

* Font types:
* Courier New is used for source code.
* *Italic letters* are used for file paths and file names.
* Synonyms:
* DICOM server == PACS (we try to use only DICOM server here)

## Abbreviations

**AET** Application Entity Title

**LOC** Lines Of Code

**PACS** Picture Archiving and Communication System

**JWS** Java WebStart

**SU** ShanoirUploader

**ShS** Shanoir Server

# Shanoir Uploader description

## What is the ShanoirUploader?

ShanoirUploader is a Java desktop application that transfers data securely between a PACS and a Shanoir server instance (e.g., within a hospital). It uses a DICOM query/retrieve connection to search and download images from a local PACS. After retrieval, the DICOM files are locally anonymized and then uploaded to the Shanoir server.

The primary goals of that application are to enable mass data transfers between different remote server instances and therefore reduce the waiting time of the users, when importing data into Shanoir. Most of the time during import is spent with data transfers.

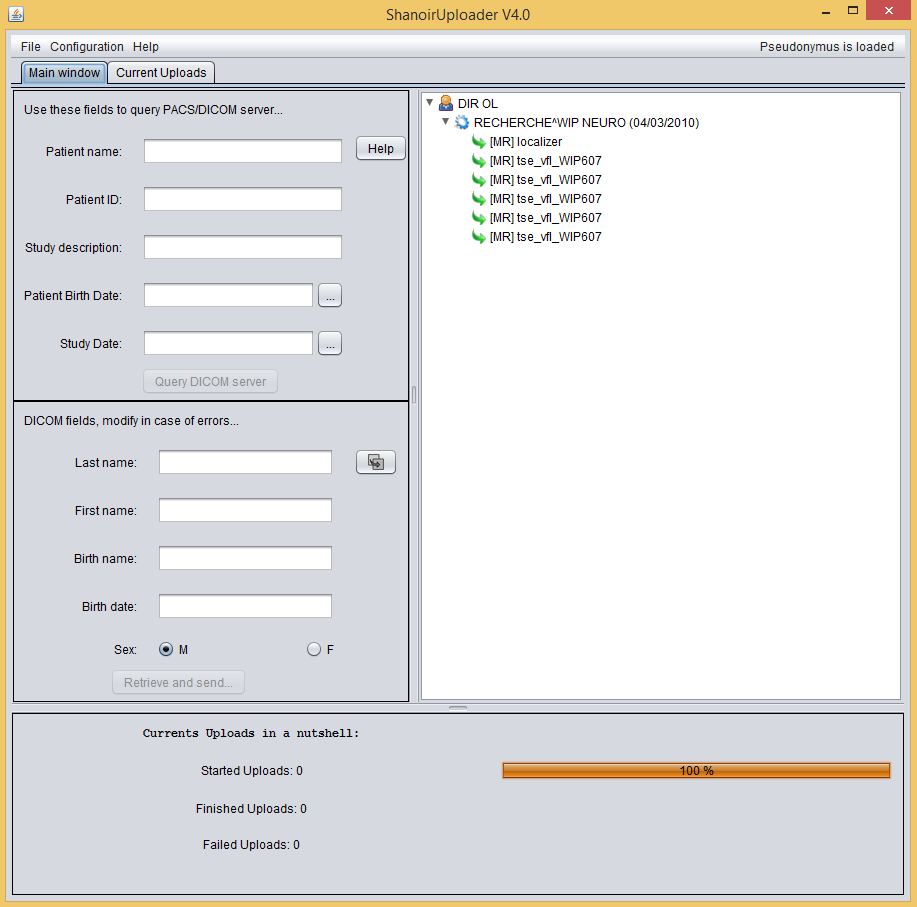


Figure 1: Screenshot of the ShanoirUploader application

## Workflows

This chapter describes the user workflows using the ShanoirUploaderOFSEP.

**Figure 2: General use case of the import with SU and ShS**

## Installation and configuration

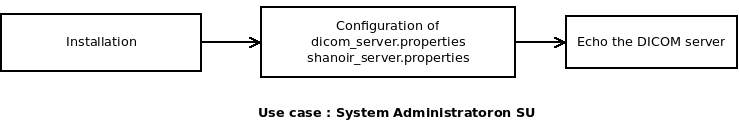


Figure 3: Use case for the system administrator on SU

1. The user downloads and installs the ShanoirUploaderOFSEP via JWS from a Shanoir server instance where the corresponding version will be deployed with or via a .zip containing the last version of ShanoirUploaderOFSEP.
2. After the installation and the first startup of the application the user will have to
   * + edit two properties files: *dicom\_server.properties* and *shanoir\_server.properties*.

These files will automatically be created in a *.su* folder within the *user.home* (according to each operating system). During every call of a DICOM server or the Shanoir server the values of these two properties files will be used. To modify the parameters of these two files, users can edit them directly or use the GUI windows developed for this purpose.

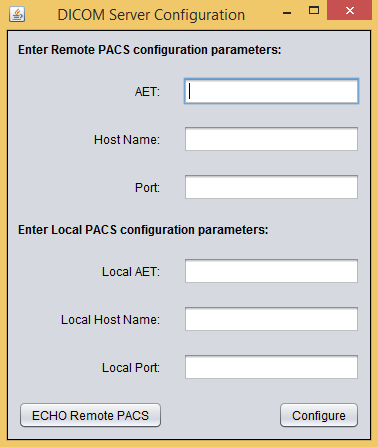


Figure 4: Dicom Server Configuration Window

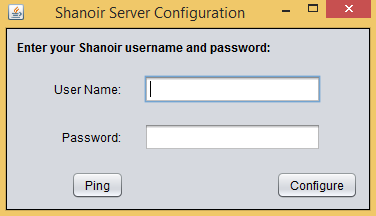


Figure 5: Shanoir Server Configuration Window

* + - Add ShanoirUpload Application Entity Title (AET) in your DICOM server configuration
  1. Example for *dicom\_server.properties*

########################################################

# Fichier de configuration pour accéder au serveur DICOM

########################################################

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Configurez votre PACS local ici

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# AET de votre PACS local

dicom.server.aet.called=DCM4CHEE

dicom.server.host=localhost

dicom.server.port=11112

dicom.server.web.port=8180

# Paramètres de sécurité de votre PACS

# NB: dicom.server.protocol = "dicom" ou "dicom-tls.3des"

dicom.server.protocol=dicom

dicom.server.enableTLS3DES=false

# Ne remplir la section suivante que si enableTLS3DES=true

# Chemin ou URL vers votre keystore

dicom.server.keystore.url=/folder/file.jks

dicom.server.keystore.password=password

# Chemin ou URL vers votre truststore

dicom.server.truststore.url=/folder/file.jks

dicom.server.truststore.password=password

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# AET de ShanoirUploaderOFSEP - NE PAS MODIFIER!

# Ces données vous servent à configurer votre PACS local

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

local.dicom.server.aet.calling=SHANOIR-UPLOADER

local.dicom.server.host=localhost

local.dicom.server.port=44105

* 1. Example for *shanoir\_server.properties*

##########################################################

# Fichier de configuration pour accéder au serveur Shanoir

##########################################################

# Profil utilisateur

shanoir.server.user.name=guest

shanoir.server.user.password=guest

# Serveur Shanoir auquel vous souhaitez accéder

shanoir.server.upload.service.url=https://localhost:8443/Shanoir-Shanoir/FileUploader?wsdl

# NE PAS MODIFIER! Merci

shanoir.server.upload.service.qname.namespace.uri=http://upload.impl.webservices.shanoir.org/

shanoir.server.upload.service.qname.local.part=UploadFileService

* 1. How to configure your AET

Go to the URL of your DICOM server, find the AET tab and use the following lines of *dicom\_server.properties* to configure it:

local.dicom.server.aet.calling=SHANOIR-UPLOADER

local.dicom.server.host=localhost

local.dicom.server.port=44105

## 

## Shanoir Uploader features

* Import DICOM Data: Users can import Dicom Data from CD/DVD or by querying the DICOM Server already configured in the DICOM Server configuration file. Several fields can be used to query DICOM Server which are the patient name, the patient ID, the study description, the patient birth date and the study date. The wildcard “\*” can be used with Patient Name and Study Description Criteria. A “help” button explains the patient Name forms as it should be introduced in corresponding field and the wildcard’s usage way.

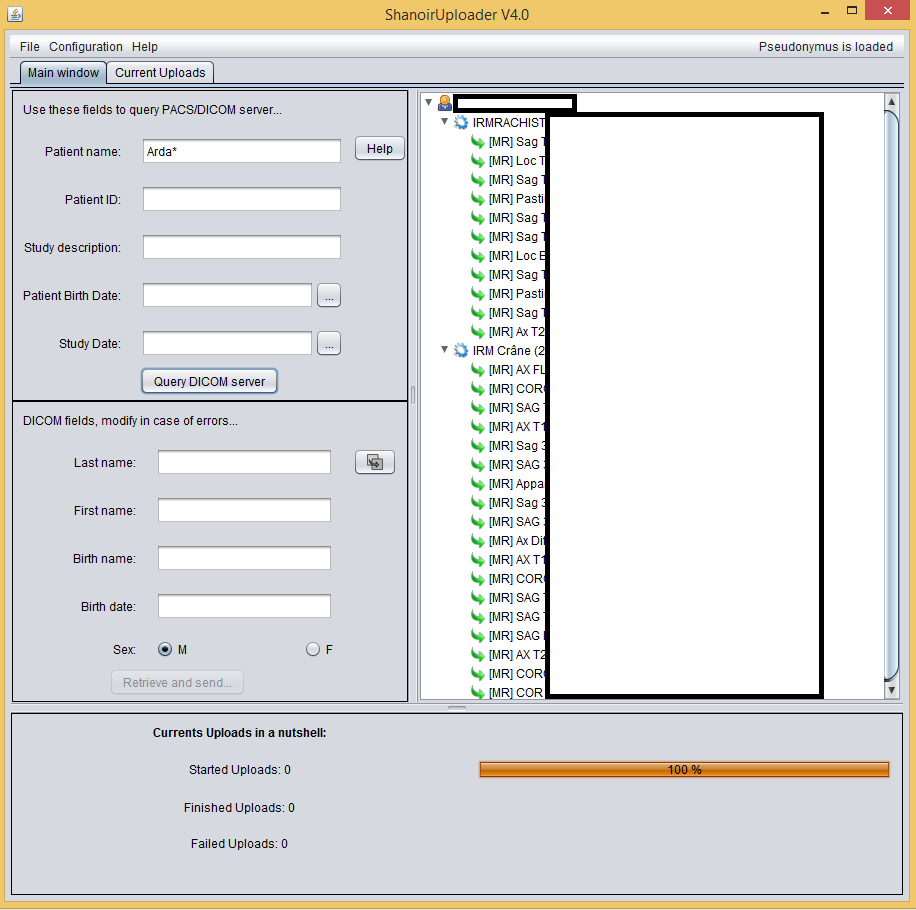


Figure 6: DICOM Server query example

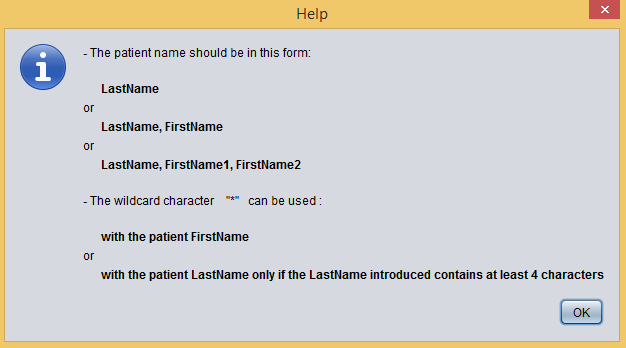


Figure 7: Patient name field’s help button

* Upload DICOM data to Shanoir Server: Users can select the datasets that will be stored in Shanoir database. When selecting datasets, the DICOM fields used to create the Pseudonymus hash are automatically filled. The user have to verify the exactitude of those information and introduce the patient birth date that does not exist in DICOM fields, then click on the “Retrieve and Send” button. Thus, Shanoir Uploader creates the Pseudonymus hash, downloads data from PACS, anonymises DICOM data, and prepare DICOM files to be sent to Shanoir Server. The upload process to Shanoir is done on background, so, users can import more data to Shanoir. A pop-up window appears after clicking on the button “Retrieve and Send” to inform user when he can start a new upload as illustrated by figure 9.

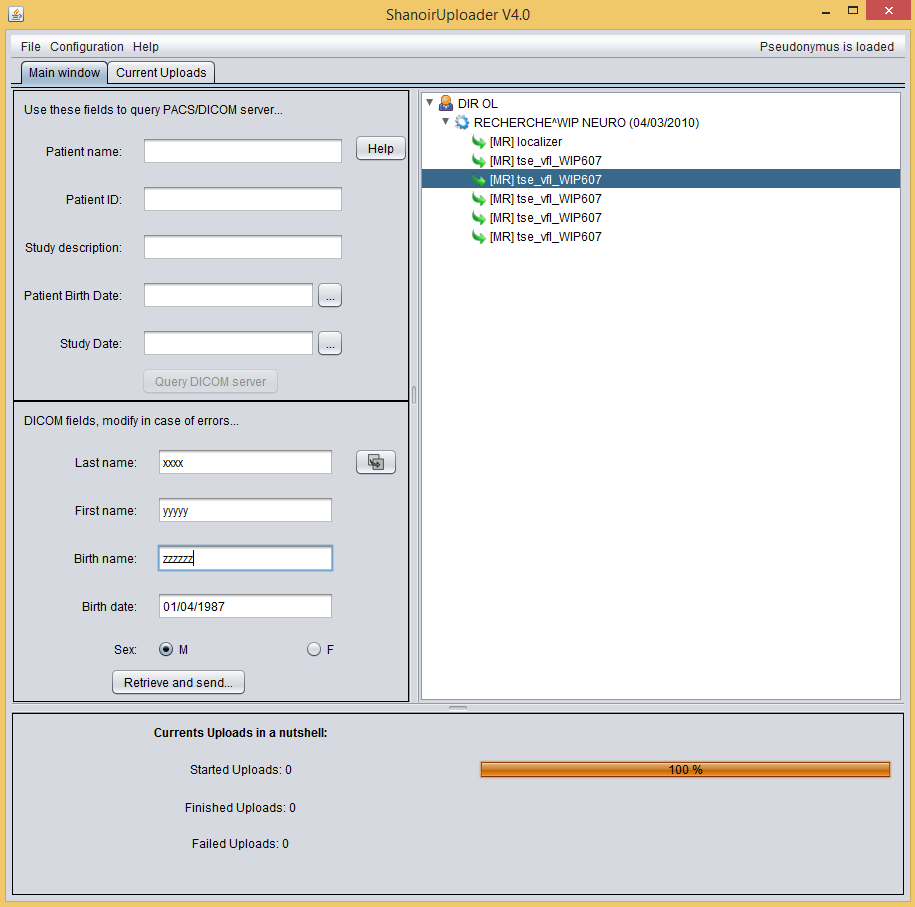


Figure 8: DICOM data retrieve and send example

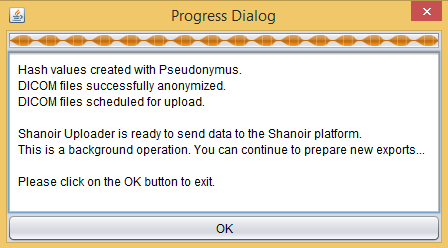


Figure 9: Shanoir Uploader progress dialog

* Several features are introduced to give users an overview of current uploads:
  + show nominative data of current uploads in the current uploads tab
  + show the state of each upload (start/finished/error) in the current uploads tab
  + show the percentage of started uploads in the current uploads tab
  + show the state of current uploads in the main window (nb start uploads, nb finished uploads, nb failed uploads)
  + show the global percentage of uploads in the main window tab
  + alert user in case of failed uploads in the main window tab
* Delete the DICOM data temporary folders created in the user work Folder:
  + user can delete the temporary folder of (finished/error) upload in the work folder
  + user can delete all finished uploads by doing a single operation which is clicking on the link ‘delete all”.

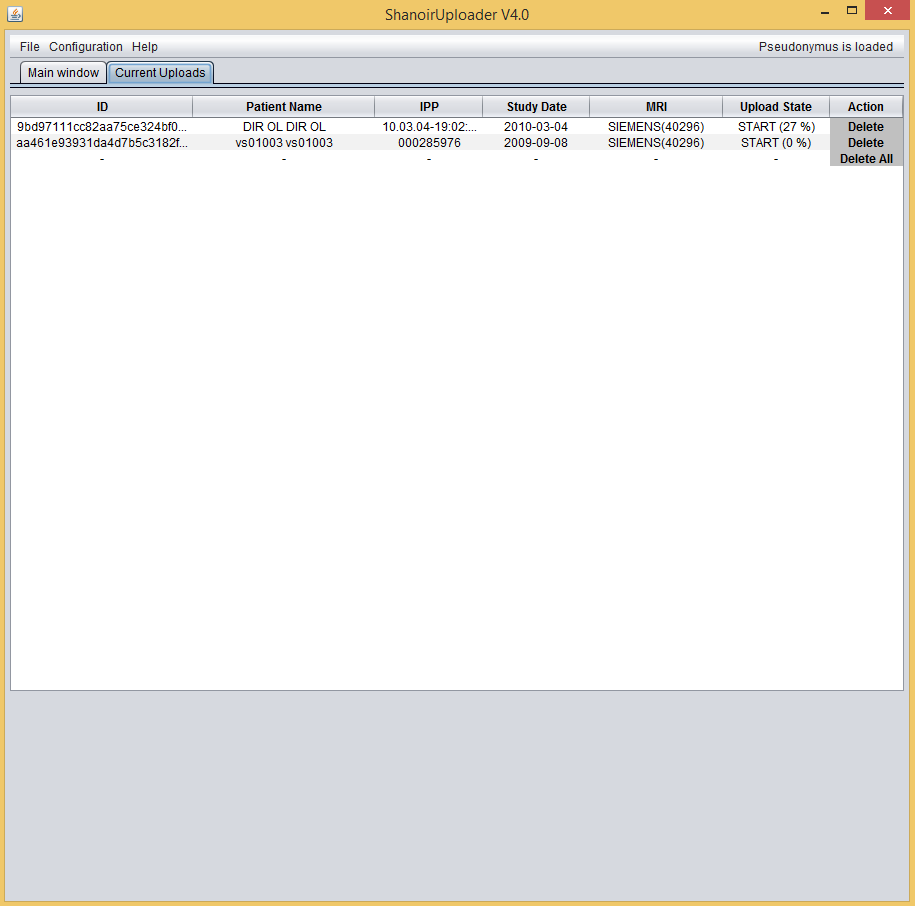


Figure 10: Shanoir Uploader’s Current Uploads tab

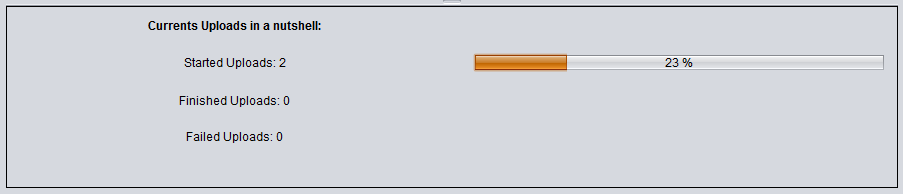
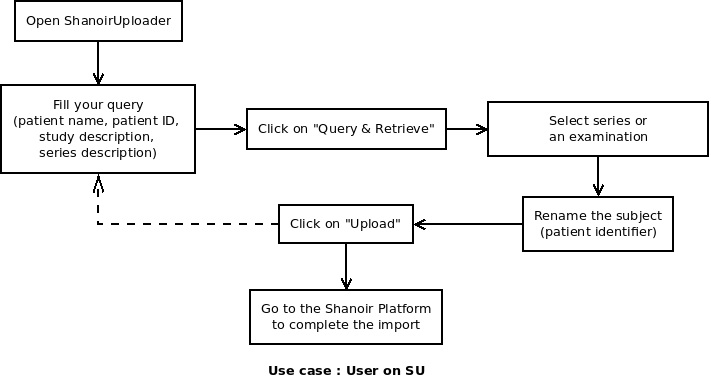


Figure 11: Information about current uploads in main window tab of Shanoir Uploader

* Inform users about Shanoir Uploader’s version, Shanoir Uploader’s copyright, Pseudonymus copyright and the user support address. Those information are available in the About Shanoir Uploader window.



**Figure 12: About Shanoir Uploader Window**



## Use case for the user on SU

# Technical functions of ShanoirUploader

## Reading properties files

This function enables the ShanoirUploader to read the two properties files *dicom\_server.properties* and *shanoir\_server.properties*.

## DICOM server echo

With this function a menu item will be provided to echo an existing DICOM server using the configuration provided in the file *dicom\_server.properties*. This function can be used to test the current DICOM server configuration.

## Shanoir server echo

With this function a menu item will be provided to ping Shanoir server.

## DICOM server query & retrieve

With this function a DICOM server will be queried and selected data will be retrieved.

## Shanoir server upload

This function is required to transfer data from the ShanoirUploader to a Shanoir server instance.

1. After the retrieval of the DICOM files from the DICOM server all files are stored locally in a temporary folder on your computer.
2. The ShanoirUploader will contact a web-service of the Shanoir server instance to upload his files using this service. A web-service, SOAP over HTTP(S), will be used to extend / reuse the already existing frameworks. When contacting this web-service an authentication will be required. Therefore the user will have to configure his account credentials in the *Shanoir.properties* file (in a later version using the ShanoirUploader GUI). For transfer security reasons all images will be uploaded separately.
3. The server will receive all the files and store them in a specific folder in his *pre-archive* folder.
4. Additionally to the DICOM files a file called *upload-job.properties* will be created on the ShanoirUploader. This file will contain additional information concerning the retrieve/import/upload. It will represent the state engine of the data transfer. This file will be uploaded to the Shanoir server instance as well and will be used by the server for continuing the import process on the web. It will e.g. contain informations concerning the series, later needed by the Shanoir server, which are listed below. Later this file will be extended with more informations entered in the ShanoirUploader to fasten the import process on the web, e.g. like the selected study and selected subject.

* Series Instance UID
* Series Description
* Series Date
* Series Number
* Modality
* Protocol Name
* Images Count
* (setSelected, which will be set automatically on the server, but is still necessary for later treatement on the server)

## Logging

ShanoirUploader will have his own logging, build with log4j. All logging will be written in the file *su.log* in the folder *.su*. The log file will be used to store important information of the data treatment, e.g. like the mapping of the subjects real name to his common name within Shanoir.

# Finishing import from Shanoir Uploader on Shanoir Platform

This chapter describes the user workflow on the server side (on the Shanoir platform), when the SU has finished his upload successfully, which will be displayed in the SU.

1. Log in with the same user account credentials, which have been used within the ShanoirUploader.
2. Select Import Data -> From ShanoirUploader...

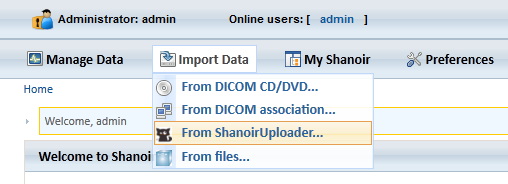


Figure 6: Screenshot of ShS new menu item

1. It directly opens the "MRI (quick)" view, as this faster version seems to be used mostly. The next process steps will be the following, as on the screen scribble below. After pressing the button "Import DICOM data" all following process steps will remain the same. In this screen only a list of series will be displayed in the tree view.

The information box on the right side of the tree will only display informations of series, if clicked on (not as a study as in the screenshot).

In a first version always all series will be selected, as they were previously already selected with the ShanoirUploader. The GUI part for querying the DICOM server will NOT be displayed as this has already be done in the ShanoirUploader. The user can import data for multiple subjects and multiple examinations during the import process at the same time with ShanoirUploader, but all data will be mapped to one subject and one examination as it is today on the ShS.

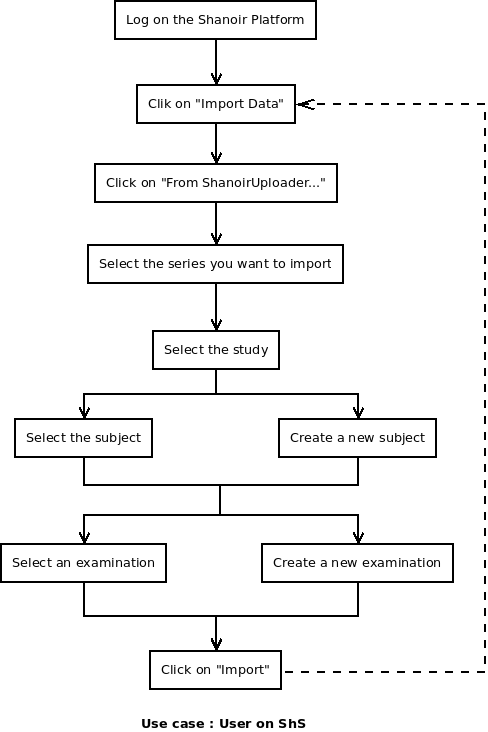


Figure 8: Use case for user on ShS

One-step that need more specification is the subject creation/selection

## Subject creation/selection

During the upload, the SU will transfer the patient's hash key (PHK) to Shanoir server. The PHK will be used to compare patients and to find duplicate patients in the Shanoir database. Then, during the indexation phase on shanoir, a common name (PID) will be attributed to the subject either by the user or by the system depending on the platform used.

For the OFSEP project, the patient’s hash key (PHK) is calculated based on the patient’s first name, birth name and his birth date. It has a length of 64.

*PHK = SHA256(hashP1(first\_name)|| hashP1(birth\_name)|| hashP1(birth\_date))*

*«SHA256 »: SHA\_256 bits,*

*«hashP1 » : Pseudonymus hash with soundex « 0 »,*

*« || »:  concatenation symbol.*

For the other cases, the patient’s hash key has a length of 14 and is calculated with the first name, the last name and the birth date.

*PHK = SHA(first\_name || last\_name || birth\_date)«SHA»: SHA\_algorithm 160 bits \*,*

*« || »:  concatenation symbol.*

*\** <https://www.cs.clemson.edu/course/cpsc424/material/Cryptography/Message%20Digest%20Functions.pdf>

https://fr.wikipedia.org/wiki/SHA-0

There are also some features about the patient identifier (PID) for the OFSEP project. In the other cases except OFSEP, an input field “*NewPatientIDTF”* is present in ShanoirUploader interface to let the user enter a patient’s identifier. This PID-SU is then transferred to the SHS. In case of the OFSEP project, the input field “*NewPatientIDTF*” is disable in the SU, and the PID-SU is calculated automatically in Shanaoir platform based on the Id of the center to which the subject is attached. The PIS is a string of seven characters. The first tree characters represent the Id of the center formatted to a string of three characters. The last four characters are incremental numbers increased when we add a new subject to a study which study card is attached to the given center.

*PID= XXXYYYY*

*«XXX»: Center Id*

*«YYYY»: Incremental number*

Regarding all these features and differences, the following table lists all of the use cases of the import and the anonymization. Each use case is explained in detail below.

|  |  |
| --- | --- |
|  | **Import From SU** |
| **Neurinfo** | |  |  |  | | --- | --- | --- | | **PHK-SU = PHK-SHS** | **PHK-SU ≠ PHK-SHS** | | | Case #1 | **PID-SU = PID-SHS** | **PID-SU ≠ PID-SHS** | | Case #2 | Case #3 |   Case A |
| **OFSEP** | Case B   |  |  | | --- | --- | | **PHK-SU = PHK-SHS** | **PHK-SU ≠ PHK-SHS** | | Case #1 | Case #2 | |

* **Case A: Import from ShanoirUploader with the Shanoir Neurinfo platform**

Four cases may be present here:

|  |  |  |
| --- | --- | --- |
|  | **PHK-SU = PHK-SHS** | **PHK-SU ≠ PHK-SHS** |
| **PID-SU = PID-SHS** | Case #1   the patient correspond with the one in the database  | Case #2    Warning   (two different patients?) |
| **PID-SU ≠ PID-SHS** | Case #3   the patient is new  |

1. **Case #1**: PHK-SU = PHK-SHS

The patient could be found within Shanoir database using the PHK. The existing patient (subject) will be preselected on Shanoir or Shanoir uplader interface. The user will not be able to change the subject or create a new subject. No local anonymization in SHS is required as the patient is already correctly anonymized by the SU.

1. **Case #2**: PHK-SU ≠ PHK-SHS and PID-SU = PID-SHS

Since the subject’s PHK does not exist in SHS, the user has to create a new subject. The creation of a new subject will already be proposed. During the subject creation, the system detects that the patient could be found within Shanoir database using the PID, but the PHK is different. We display a warning that the PID is already used: “Are you sure the patient identifier is correct? There exists already one in the database, please change the common name”. So, the user has to change the *Common Name* since the same one exists already for another subject. If the user change other fields like the birth date, another PHK will be calculated and stored in the SHS. In any case an anonymization will have to be done since the common name is changed.

**Case #3**: PID-SU ≠ PID-SHS and PHK-SU ≠ PHK-SHS

The patient could not be found in Shanoir database using the PHK. A new subject creation page will be proposed. The *Birth Date* field will not be editable; The *Common Name* and the *Sex* fields will be editable. The common name introduced by the user is not used for another subject, so the new subject could be created and saved in the database of SHS

### Neurinfo Algorithm

1. **if (PHK-SU == PHK-ShS)** then
2. dicom.patient <- patient-shs
3. else
   1. **if (PID-SU == PID-ShS)** then
4. echo 'Warning: We found a patient with the same PID in the Shanoir database. Please modify the common name attributed to this subject'
5. **else**
6. create(patient-su)
7. dicom.patient <- patient-su

* **Case B: Import from ShanoirUploader with the Shanoir OFSEP platform**

Since the PID-SU for OFSEP is empty and the PHK-OFSEP is already calculated by the SU, only two cases will be present here:

1. **Case #1:** PHK-OPSEP = PHK-SHS: the subject exists already in SHS

The subject will be preselected and the creation tab will be disable.

1. **Case #2:** PHK-OFSEP ≠ PHK-SHS: the subject does not exist before

The subject selection tab will be disabled and a new subject creation page will be open. The *Sex* and the *Birth Date* fields will be prefilled with the elements in the *upload-job.xml* file and are not modifiable. PID-SHS will be calculated automatically as described earlier. The PHK-OFSEP and the 10 hash values will be saved in the database of the SHS.

Both of these two cases need no more local anonymization in SHS as the patient was already correctly anonymized by the SU.

### OFSEP Algorithm

1. **if (PHK-SU == PHK-ShS)** then
2. dicom.patient <- patient-shs
3. **else**
4. create(patient-su)with incremental PID
5. dicom.patient <- patient-su

# Integrating indexation part in Shanoir Uploader

Several steps should be integrated in Shanoir uploader to integrate indexation part originally done in Shanoir:

## Selection of a study

During the import process the user can select a study from a drop-down-box. Only studies for which the user has the right to import data will be displayed (Role/position “can see, download and import data” for the study). For this a web-service client call is implemented to get all studies for which a user can import data.

## Selection of a study card

Once the user selects a study, a list of corresponding study cards will be displayed. The user should select the study card that will be used for indexation.

## Selection of a subject

During the import process the user have to create/select a subject ( see Subject creation/selection part of Finishing import from Shanoir Uploader on Shanoir Platform section)

***NIfTI conversion***

The NIfTI conversion will be performed on the server, as a service, because the dependencies to other components are high. Multiple native libraries would be required.

## StudyCard conversion

The StudyCard conversion will be performed on the server during the final import of the data.