OpenREM Documentation

Release 0.4.3

Ed McDonagh

October 01, 2014

Contents

OpenREM is an opensource framework created for the purpose of radiation exposure monitoring. The software is capable of importing and displaying data from a wide variety of x-ray dose related sources, and then enables easy export of the data in a form that is suitable for further analysis by suitably qualified medical physics personnel.

Please see openrem.org for more details.

Contents:

CHAPTER 1

Installation instructions

OpenREM can be installed with a single command; however, there are two prerequisites that need to be installed first – python and pip – and RabbitMQ needs to be installed for exports and patient size imports to work.

Once installed, there are a few configuration choices that need to be made, and finally a couple of services that need to be started. Then you are ready to go!

1.1 Install the prerequisites

1.1.1 Install Python 2.7.x

- Linux – likely to be installed already
- Windows – https://www.python.org/downloads

Add Python and the scripts folder to the path

Windows only – this is usually automatic in linux

Add the following to the end of the `path` environment variable (to see how to edit the environment variables, see http://www.computerhope.com/issues/ch000549.htm):

```
;C:\Python27\;C:\Python27\Scripts\
```

1.1.2 Setuptools and pip

Install setuptools and pip – for details go to http://www.pip-installer.org/en/latest/installing.html. The quick version is as follows:

Linux

Download the latest version using the same method as for Windows, or get the version in your package manager, for example:

```
sudo apt-get install python-pip
```

Windows

Download the installer script get-pip.py and save it locally – right click and *Save link as...* or equivalent.

Open a command window (Start menu, cmd.exe) and navigate to the place you saved the getpip.py file:

```
python get-pip.py
```

Quick check of python and pip

To check everything is installed correctly so far, type the following in a command window/shell. You should have the version number of pip returned to you:

```
pip -V
```

1.1.3 Install RabbitMQ

(New for version 0.4.3)

- Linux - Follow the guide at http://www.rabbitmq.com/install-debian.html
- Windows - Follow the guide at http://www.rabbitmq.com/install-windows.html

For either install, just follow the defaults – no special configurations required.

Note: Before continuing, consider virtualenv

1.2 Install and configure OpenREM

1.2.1 Install

```
pip install openrem
```

Will need ''sudo'' or equivalent if installing on linux without using a virtualenv

1.2.2 Configure

Locate install location

- Linux: `/usr/local/lib/python2.7/dist-packages/openrem/` or `/usr/lib/python2.7/site-packages/openrem/`
- Windows: `C:\Python27\Lib\site-packages\openrem\`

There are three files that need renaming: *(changed for 0.4.0)*

- `openremproject/local_settings.py.example` to `openremproject/local_settings.py`
- `openremproject/wsgi.py.example` to `openremproject/wsgi.py`
- `openremproject/settings.py.new` to `openremproject/settings.py` *Not applicable from 0.4.3 onwards, and for 0.4.2 folder is called openrem*

In the `local_settings.py` file, set the database details, the `MEDIA_ROOT` path, the secret key and the `ALLOWED_HOSTS`.

Note: Windows notepad will not recognise the Unix style line endings. Please use an editor such as Notepad++ or Notepad2 if you can, else use WordPad – on the View tab you may wish to set the Word wrap to 'No wrap'

Database settings

For testing you can use the SQLite3 database

```
'ENGINE': 'django.db.backends.sqlite3',
'NAME': '/ENTER/PATH/WHERE/DB/FILE/CAN/GO',
```

- Linux example: `'NAME': '/home/user/openrem/openrem.db',`
- Windows example: `'NAME': 'C:/Users/myusername/Documents/OpenREM/openrem.db',` *Note use of forward slash in configuration files*

For production use, see Database options below

Location setting for imports and exports

Csv and xlsx study information exports and patient size csv imports are written to disk at a location defined by `MEDIA_ROOT`.

The path set for `MEDIA_ROOT` is up to you, but the user that runs the webserver must have read/write access to the location specified because it is the webserver than reads and writes the files. In a debian linux, this is likely to be www-data for a production install. Remember to use forward slashes for the config file, even for Windows.

Linux example:

```
MEDIA_ROOT = "/var/openrem/media/"
```

Windows example:

```
MEDIA_ROOT = "C:/Users/myusername/Documents/OpenREM/media/"
```

Secret key

Generate a new secret key and replace the one in the `local_settings.py` file. You can use http://www.miniwebtool.com/django-secret-key-generator/ for this.

Allowed hosts

The `ALLOWED_HOSTS` needs to be defined, as the `DEBUG` mode is now set to `False`. This needs to contain the server name or IP address that will be used in the URL in the web browser. For example:

```
ALLOWED_HOSTS = [
    '192.168.56.102',
    '.doseserver.',
    'localhost',
]
```

A dot before a hostname allows for subdomains (eg www.doseserver), a dot after a hostname allows for FQDNs (eg doseserver.ad.trust.nhs.uk)

1.2.3 Create the database

Linux:

```
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py syncdb
```

Windows:

```
python C:\Python27\Lib\site-packages\openrem\manage.py syncdb
```

Answer each question as it is asked, do setup a superuser. This username and password wil be used to log into the admin interface to create the usernames for using the web interface. See the Start using it! section below.

Help! I get a `value too long for type character varying(50)` error! This error with part of the Django auth_permissions system that we are not using, and can safely be ignored. This is being tracked as Issue 62

For production installs, convert to South

(What is south?)

Linux:

```
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py convert_to_south remapp
```

Windows:

```
python C:\Python27\Lib\site-packages\openrem\manage.py convert_to_south remapp
```

1.3 Start all the services

1.3.1 Start test web server

Linux:

```
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py runserver --insecure
```

Windows:

```
python C:\Python27\Lib\site-packages\openrem\manage.py runserver --insecure
```

If you are using a headless server and need to be able to see the web interface from another machine, use `python /usr/lib/python2.7/dist-packages/openrem/manage.py runserver x.x.x.x:8000 --insecure` (or Windows equivalent) replacing the `x` with the IP address of the server and `8000` with the port you wish to use.

Open the web addesss given, appending `/openrem` (http://localhost:8000/openrem)

Note: Why are we using the `--insecure` option? With `DEBUG` mode set to `True` the test web server would serve up the static files. In this release, `DEBUG` mode is set to `False`, which prevents the test web server serving those files. The `--insecure` option allows them to be served again.

1.3.2 Start the Celery task queue

(New for version 0.4.3)

Celery will have been automatically installed with OpenREM, and along with RabbitMQ allows for asynchronous task processing for imports and exports.

Note: The webserver and Celery both need to be able to read and write to the `MEDIA_ROOT` location. Therefore you might wish to consider starting Celery using the same user or group as the webserver, and setting the file permissions accordingly.

In a new shell:

Linux:

```
cd /usr/local/lib/python2.7/dist-packages/openrem/
celery -A openremproject worker -l info
```

Windows:

```
cd C:\Python27\Lib\site-packages\openrem\
celery -A openremproject worker -l info
```

For production use, see Daemonising Celery below

1.3.3 Start using it!

Add some data!

```
openrem_rdsr.py rdsrfile.dcm
```

Add some users *(New in version 0.4.0)*

- Go to the admin interface (eg http://localhost:8000/admin) and log in with the user created when you created the database (`syncdb`)
- Create some users and add them to the appropriate groups (if there are no groups, go to the OpenREM homepage and they should be created).
  - `viewgroup` can browse the data only
  - `exportgroup` can do as view group plus export data to a spreadsheet
  - `admingroup` can delete studies and import height and weight data in addition to anything the export group can do
- Return to the OpenREM interface (eg http://localhost:8000/openrem) and log out of the superuser in the top right corner and log in again using one of the new users you have just created.

1.4 Further instructions

1.4.1 Database options

SQLite is great for getting things running quickly and testing if the setup works, but is really not recommended for production use on any scale. Therefore it is recommended to use a different database such as PostgreSQL or MySQL.

Here are instructions for installing PostgreSQL on linux and on Windows:

Installing PostgreSQL for OpenREM on Ubuntu linux

Install PostgreSQL and the python connector

- `sudo apt-get install postgresql`
- `sudo apt-get build-dep python-psycopg2`

The second command installed a lot of things, at least some of which are necessary for this to work!

If you are using a virtualenv, make sure you are in it and it is active (`source bin/activate`)

- `pip install psycopg2`

Create a user for the database

- `sudo passwd postgres`
- Enter password, twice
- `sudo -u postgres createuser -P openrem_user`
- Enter password, twice
- Superuser, *No*
- Create databases, *No*
- Create new roles, *No*

Optional: Specify the location for the database You might like to do this if you want to put the database on an encrypted location

For this example, I'm going to assume all the OpenREM programs and data are in the folder `/var/openrem/`:

- `sudo /etc/init.d/postgresql stop`
- `mkdir /var/openrem/database`
- `sudo cp -aRv /var/lib/postgresql/9.1/main /var/openrem/database/`
- `sudo nano /etc/postgresql/9.1/main/postgresql.conf`

Change the line

- `data_directory = '/var/lib/postgresql/9.1/main'` to
- `data_directory = '/var/openrem/database/main'`

- `sudo /etc/init.d/postgresql start`

Create the database

- `su postgres`
- `psql template1`
- `CREATE DATABASE openrem_db OWNER openrem_user ENCODING 'UTF8';`
- `\q`
- `exit`

Change the security configuration

The default security settings are too restrictive to allow access to the database.

- `sudo nano /etc/postgresql/9.1/main/pg_hba.conf`
- **Add the following line:**
  - `local openrem_db openrem_user md5`
- `sudo /etc/init.d/postgresql restart`

Configure OpenREM to use the database

Find and edit the settings file, eg

- `nano local/lib/python2.7/site-packages/openrem/openrem/settings.py`

Set the following (changing name, user and password as appropriate):

- `'ENGINE': 'django.db.backends.postgresql_psycopg2',`
- `'NAME': 'openrem_db',`
- `'USER': 'openremuser',`
- `'PASSWORD': 'openrem_pw',`

Fire it up with OpenREM

- `python path/to/openrem/manage.py syncdb`
- `python path/to/openrem/manage.py convert_to_south remapp`

Installing PostgreSQL for OpenREM on Windows

Note: Author JA Cole

Get PostgreSQL and the python connector

- Download the installer from http://www.enterprisedb.com/products-services-training/pgdownload#windows
- Download psycopg2 from http://www.lfd.uci.edu/~gohlke/pythonlibs/. Make sure it matches your python and Windows version.

Install PostgreSQL

Run the the postgresql installer. It will ask for a location. Ensure the "data" directory is *not* under "Program Files" as this can cause permissions errors. Enter a superuser password when prompted. Make sure you keep this safe as you will need it.

Create a user and database

Open pgAdmin III

- Click on servers to expand
- Double click on PostgreSQL 9.3
- Enter your superuser password
- Right click on "login roles" and choose "New login role"
- Create the openremuser (or whatever you want your user to be called) and under definition add a password.
- Click OK
- Right click on databases and choose "New database"
- Name the database (openremdb is fine) and assign the the owner to the user you just created.

Install psycopg2

Run the installer you downloaded for psycopg2 earlier.

Configure OpenREM to use the database

Find and edit the settings file (notepad works fine). The path depends on your python install, but could be something like:

- `C:\lib\python2.7\site-packages\openrem\openrem\settings.py`

Set the following (changing name, user and password as appropriate):

- `'ENGINE': 'django.db.backends.postgresql_psycopg2',`
- `'NAME': 'openrem_db',`
- `'USER': 'openremuser',`
- `'PASSWORD': 'openrem_pw',`

Fire it up with OpenREM

- `python path/to/openrem/manage.py syncdb`
- `python path/to/openrem/manage.py convert_to_south remapp`

Fix '' value too long for type character varying(50)" error

This error is caused by the django auth_permissions system not being able to cope with long names in the models.

- Open pgAdmin III
- Open Servers
- Open databases
- Open the openrem database
- Open schemas

- Open public
- Open tables
- right click on auth_permission
- Select properties
- Change ''name" to ''varying(100)" from ''varying(50)''

Then run `python path/to/openrem/manage.py syncdb` again.

1.4.2 Database migrations

South is a django application to manage database migrations. Using South means that future changes to the database model can be calculated and executed automatically with simple commands when OpenREM is upgraded.

1.4.3 Production webservers

Unlike the database, the production webserver can be left till later and can be changed again at any time.

For performance it is recommended that a production webserver is used instead of the inbuilt 'runserver'. Popular choices would be either Apache or you can do as the cool kids do and use Gunicorn with nginx.

The django website has instructions and links to get you set up with Apache.

1.4.4 Daemonising Celery

(New for version 0.4.3)

In a production environment, Celery will need to start automatically and not depend on a particular user being logged in. Therefore, much like the webserver, it will need to be daemonised. For now, please refer to the instructions and links at http://celery.readthedocs.org/en/latest/tutorials/daemonizing.html.

1.4.5 Virtualenv and virtualenvwrapper

If the server is to be used for more than one python application, or you wish to be able to test different versions of OpenREM or do any development, it is highly recommended that you use virtualenv or maybe virtualenvwrapper

Virtualenv sets up an isolated python environment and is relatively easy to use.

If you do use virtualenv, all the paths referred to in the documentation will be changed to:

- Linux: `lib/python2.7/site-packages/openrem/`
- Windows: `Lib\site-packages\openrem`

In Windows, even when the virtualenv is activated you will need to call *python* and provide the full path to script in the *Scripts* folder. If you call the script (such as *openrem_rdsr.py*) without prefixing it with *python*, the system wide Python will be used instead. This doesn't apply to Linux, where once activated, the scripts can be called without a *python* prefix from anywhere.

1.5 Related guides

1.5.1 Running Conquest on Windows as a service

Note: Content contributed by DJ Platten, with edit by ET McDonagh

This guide assumes Conquest has already been installed and runs ok. These instructions are based on Windows XP. **Run as a service**

1. Make sure conquest isn't running.
2. Open a file browser and navigate to your conquest folder.
3. Right-hand click on the "ConquestDICOMServer.exe" file and choose "Run as..."
4. Enter the username and password of a Windows user with administrator rights.
5. Once conquest is running, click on the "Install server as NT service" on the "Configuration" tab.
6. Close the conquest Window.
7. Log in to Windows as a user with administrator rights.
8. Go to "Control panel" -> "Administrative Tools" -> "Services".
9. There will be a service with the same name as conquest's AE title. Right-hand click the mouse on this service and select "Properties".
10. On the "Log On" tab check the box that says "Allow service to interact with the desktop".
11. Click "Apply" then "OK".
12. Right-hand click on the service again and click "Restart".

The "Allow service to interact with the desktop" seems to be necessary for the batch to run that puts the dose report into OpenREM.

Firewall settings

Windows is able to change its firewall settings after you think everything is working ok! Assuming you have control of the firewall, add three port exceptions to the Windows firewall on the server computer: ports 80 and 443 for Apache, and whichever port that was chosen for conquest (104 is *the* port allocated to DICOM, but you may have used a higher port above 1024 for permissions reasons).

The firewall instructions at portforward.com were found to be a useful guide for this.

1.5.2 Backing up MySQL on Windows

Note: Content contributed by DJ Platten

These instructions are based on Windows XP.

As a one-off, create a MySQL user called `backup` with a password of `backup` that has full rights to the database:

```
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe -u root -p -e "CREATE USER 'backup'@'local
```

Grant the `backup` user full privileges on the database called `openremdatabasemysql`:

```
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe -u root -p -e "GRANT ALL PRIVILEGES ON ope
```

Grant the `backup` user privileges to create databases:

```
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe -u root -p -e "GRANT CREATE ON *.* TO 'bac
```

Reload the privileges to ensure that MySQL registers the new ones:

```
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe -u root -p -e "FLUSH PRIVILEGES";
```

To backup the contents of the database from the command line to a file called `backup.sql` (note that the lack of spaces after the `-u` and `-p` is not a typo):

```
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqldump.exe -ubackup -pbackup openremdatabasemysql
```

To restore the database, assuming that it doesn't exist anymore, first it needs to be created:

```
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe -ubackup -pbackup -e "CREATE DATABASE open
```

Then restore the contents of the database from a file called `backup.sql`:

```
C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe -ubackup -pbackup openremdatabasemysql < b
```

An example DOS batch file to back up the contents of the `openremdatabasemysql` database using a time stamp of the form `yyyy-mm-dd_hhmm`, zip up the resulting file, delete the uncompressed version and then copy it to a network location (the network copy will only work if the user that runs the batch file has permission on the network):

```
@echo off
For /f "tokens=1-4 delims=/ " %%a in ('date /t') do (set mydate=%%c-%%b-%%a)
For /f "tokens=1-2 delims=/:" %%a in ('time /t') do (set mytime=%%a%%b)

"C:\Program Files\MySQL\MySQL Server 5.6\bin\mysqldump.exe" -ubackup -pbackup openremdatabasemys

"C:\Program Files\7-Zip\7z.exe" a F:\OpenREMdatabase\backup\%mydate%_%mytime%_openremdatabasemys

del F:\OpenREMdatabase\backup\%mydate%_%mytime%_openremdatabasemysql.sql

copy F:\OpenREMdatabase\backup\%mydate%_%mytime%_openremdatabasemysql.zip "\\Srv-mps-001\xls_prc
```

1.5.3 Advanced guides for developers

Installing Apache on Windows Server 2012 with auto-restart

Note:

Author JA Cole

These instructions are for installing OpenREM under Apache on Windows as a developers alternative to the built-in HTTP server. They have been written using Windows Server 2012, and feature automatic restarts of the Apache server when the code changes, much as the built-in server does. **Get and Install Apache**

- Download the zip of the appropriate version from https://www.apachelounge.com/
- Extract the zip somewhere useful. For this guide we will assume `C:\apache24\`

Get and Install MOD_WSGI

- Download mod_wsgi that matches your Windows, Apache and Python versions from http://www.lfd.uci.edu/~gohlke/pythonlibs/#mod_wsgi
- Extract the mod_wsgi.so file to `C:\apache24\modules\`
- Add the following module `C:\apache24\conf\httpd.conf`:

```
LoadModule wsgi_module modules/mod_wsgi.so
```

- At the end of `C:\apache24\conf\httpd.conf` add the following:

```
WSGIScriptAlias / "c:/Python27/Lib/site-packages/openrem/openrem/wsgi.py"
WSGIPythonPath "c:/Python27/Lib/site-packages/openrem"

<Directory "c:/Python27/Lib/site-packages/openrem/openrem">
<Files wsgi.py>
Order deny,allow
Require all granted
</Files>cd c:\py
</Directory>
```

Get and Install wsgi.py and monitor.py

Detailed instructions are available here: https://code.google.com/p/modwsgi/wiki/ReloadingSourceCode

- Change wsgi.py in the openrem/openrem folder to the following

```
"""
WSGI config for OpenREM project.

This module contains the WSGI application used by Django's development server
and any production WSGI deployments. It should expose a module-level variable
named ``application``. Django's ``runserver`` and ``runfcgi`` commands discover
this application via the ``WSGI_APPLICATION`` setting.

Usually you will have the standard Django WSGI application here, but it also
might make sense to replace the whole Django WSGI application with a custom one
that later delegates to the Django one. For example, you could introduce WSGI
middleware here, or combine a Django application with an application of another
framework.

"""
import os
import sys

path = 'C:/Python27/Lib/site-packages/openrem'
if path not in sys.path:
sys.path.append(path)

os.environ.setdefault("DJANGO_SETTINGS_MODULE", "openrem.settings")

# Apply WSGI middleware here.

import django.core.handlers.wsgi
application = django.core.handlers.wsgi.WSGIHandler()
```

```
import openrem.monitor
openrem.monitor.start(interval=1.0)
```

- Create a file monitor.py in the openrem/openrem folder with the following contents

```
# Code from the modwsgi wiki at https://code.google.com/p/modwsgi/wiki/ReloadingSourceCode
# Copyright 2007-2011 GRAHAM DUMPLETON
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#     http://www.apache.org/licenses/LICENSE-2.0
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#


import os
import sys
import time
import signal
import threading
import atexit
import Queue

_interval = 1.0
_times = {}
_files = []

_running = False
_queue = Queue.Queue()
_lock = threading.Lock()

def _restart(path):
    _queue.put(True)
    prefix = 'monitor (pid=%d):' % os.getpid()
    print >> sys.stderr, '%s Change detected to \'%s\'.' % (prefix, path)
    print >> sys.stderr, '%s Triggering Apache restart.' % prefix
    import ctypes
    ctypes.windll.libhttpd.ap_signal_parent(1)

def _modified(path):
    try:
        # If path doesn't denote a file and were previously
        # tracking it, then it has been removed or the file type
        # has changed so force a restart. If not previously
        # tracking the file then we can ignore it as probably
        # pseudo reference such as when file extracted from a
        # collection of modules contained in a zip file.

        if not os.path.isfile(path):
            return path in _times

        # Check for when file last modified.
```

```
            mtime = os.stat(path).st_mtime
            if path not in _times:
                _times[path] = mtime

            # Force restart when modification time has changed, even
            # if time now older, as that could indicate older file
            # has been restored.

            if mtime != _times[path]:
                return True
        except:
            # If any exception occured, likely that file has been
            # been removed just before stat(), so force a restart.

            return True

        return False

    def _monitor():
        while 1:
            # Check modification times on all files in sys.modules.

            for module in sys.modules.values():
                if not hasattr(module, '__file__'):
                    continue
                path = getattr(module, '__file__')
                if not path:
                    continue
                if os.path.splitext(path)[1] in ['.pyc', '.pyo', '.pyd']:
                    path = path[:-1]
                if _modified(path):
                    return _restart(path)

            # Check modification times on files which have
            # specifically been registered for monitoring.

            for path in _files:
                if _modified(path):
                    return _restart(path)

            # Go to sleep for specified interval.

            try:
                return _queue.get(timeout=_interval)
            except:
                pass

    _thread = threading.Thread(target=_monitor)
    _thread.setDaemon(True)

    def _exiting():
        try:
            _queue.put(True)
        except:
            pass
        _thread.join()

    atexit.register(_exiting)
```

```
def track(path):
    if not path in _files:
        _files.append(path)

def start(interval=1.0):
    global _interval
    if interval < _interval:
        _interval = interval

    global _running
    _lock.acquire()
    if not _running:
        prefix = 'monitor (pid=%d):' % os.getpid()
        print >> sys.stderr, '%s Starting change monitor.' % prefix
        _running = True
        _thread.start()
_lock.release()
```

Install Micosoft C++ Distributable

Install the microsoft C++ distributable making sure the version number matches the version number for the apache and mod_wsgi downloads. http://www.microsoft.com/en-us/download/details.aspx?id=30679#

Optional: Install apache as a service

Run a terminal as administrator.:

```
c:\apache24\bin\httpd -k install
```

Setup the URLs

Add the following to the openrem urls.py file:

```
from django.conf import settings
if settings.DEBUG:
    urlpatterns += patterns('django.contrib.staticfiles.views',
        url(r'^static/(?P<path>.*)$', 'serve'),
    )
```

Collect the static files

Collect your static files by running:

```
python manage.py collectstatic
```

If this fails because openrem lacks a static folder either copy the static folder from remapp to the openrem directory, adjust the openrem settings or set up a link. To setup a link run:

```
mklink /D c:\python27\lib\site-packages\openrem\static c:\python27\lib\site-packages\openrem\rem
```

CHAPTER 2

OpenREM Release Notes

2.1 Version history change log

2.1.1 OpenREM version history

0.4.3 (2014-10-01)

- #119 Fixed issue where Celery didn't work on Windows. Django project folder is now called openremproject instead of openrem
- #117 Added Windows line endings to patient size import logs
- #113 Fixed units spelling error in patient size import logs
- #112 File system errors during imports and exports are now handled properly with tasks listed in error states on the summary pages
- #111 Added abort function to patient size imports and study exports
- #110 Converted exports to use the FileField handling for storage and access, plus modified folder structure.
- #109 Added example `MEDIA_ROOT` path for Windows to the install docs
- #108 Documented ownership issues between the webserver and Celery
- #107 Documented process for upgrading to 0.4.2 before 0.4.3 for versions 0.3.9 or earlier
- #106 Added the duration of export time to the exports table. Also added template formatting tag to convert seconds to natural time
- #105 Fixed bug in Philips CT import where `decimal.Decimal` was not imported before being used in the age calculation
- #104 Added documentation for the additional study export functions as a result of using Celery tasks in task #19 as well as documentation for the code
- #103 Added documentation for using the web import of patient size information as well as the new code
- #102 Improved handling of attempts to process patient size files that have been deleted for when users go back in the browser after the process is finished
- #101 Set the security of the new patient size imports to prevent users below admin level from using it

- #100 Logging information for patient size imports was being written to the database - changed to write to file
- #99 Method for importing remapp from scripts and for setting the *DJANGO_SETTINGS_MODULE* made more robust so that it should work out of the box on Windows, debian derivatives and virtualenvs
- #98 Versions 0.4.0 to 0.4.2 had a settings.py.new file to avoid overwriting settings files on upgrades; renaming this file was missing from the installation documentation for new installs
- #97 Changed the name of the export views file from ajaxviews as ajax wasn't used in the end
- #96 Changed mammo and fluoro filters to use named fields to avoid needing to use the full database path
- #93 Set the security of the new exports to prevent users below export level from creating or downloading exports
- #92 Add NHSBSP specific mammography csv export from Jonathan Cole - with Celery
- #91 Added documentation for Celery and RabbitMQ
- #90 Added delete function for exports
- #89 Added the Exports navigation item to all templates, limited to export or admin users
- #88 Converted fluoroscopy objects to using the Celery task manager after starting with CT for #19
- #87 Converted mammography objects to using the Celery task manager after starting with CT for #19
- #86 Digital Breast Tomosynthesis systems have a projections object that for Hologic contains required dosimetry information
- #85 Fix for bug introduced in #75 where adaption of ptsize import for procedure import broke ptsize imports
- #74 'Time since last study' is now correct when daylight saving time kicks in
- #39 Debug mode now defaults to False
- #21 Height and weight data can now be imported through forms in the web interface
- #19 Exports are now sent to a task manager instead of locking up the web interface

Reopened issue

- #9 Issue tracking import using *.dcm style wildcards reopened as Windows `cmd.exe` shell doesn't do wildcard expansion, so this will need to be handled by OpenREM in a future version

0.4.2 (2014-04-15)

- #83 Fix for bug introduced in #73 that prevents the import scripts from working.

0.4.1 (2014-04-15)

- #82 Added instructions for adding users to the release notes

0.4.0 (2014-04-15)

Note:

- #64 includes **changes to the database schema and needs a user response** - see version 0.4.0 release notes
- #65 includes changes to the settings file which **require settings information to be copied** and files moved/renamed - see version 0.4.0 release notes

---

- #80 Added docs for installing Apache with auto-start on Windows Server 2012. Contributed by JA Cole
- #79 Updated README.rst instructions
- #78 Moved upgrade documentation into the release notes page
- #77 Removed docs builds from repository
- #76 Fixed crash if exporting from development environment
- #75 Fixed bug where requested procedure wasn't being captured on one modality
- #73 Made launch scripts and ptsizecsv2db more robust
- #72 Moved the secret key into the local documentation and added instructions to change it to release notes and install instructions
- #71 Added information about configuring users to the install documentation
- #69 Added documentation about the new delete study function
- #68 Now checks sequence code meaning and value exists before assigning them. Thanks to JA Cole
- #67 Added 'Contributing authors' section of documentation
- #66 Added 'Release notes' section of documentation, incuding this file
- #65 Added new `local_settings.py` file for database settings and other local settings
- #64 Fixed imports failing due to non-conforming strings that were too long
- #63 The mammography import code stored the date of birth unnecessarily. Also now gets decimal_age from age field if necessary
- #60 Removed extraneous colon from interface data field
- #18 Studies can now be deleted from the web interface with the correct login
- #16 Added user authentication with different levels of access
- #9 Enable import of `*.dcm`

0.3.9 (2014-03-08)

Note: #51 includes changes to the database schema – make sure South is in use before upgrading. See http://docs.openrem.org/page/upgrade.html

---

- #59 CSS stylesheet referenced particular fonts that are not in the distribution – references removed
- #58 Export to xlsx more robust - limitation of 31 characters for sheet names now enforced
- #57 Modified the docs slightly to include notice to convert to South before upgrading

- #56 Corrected the mammography target and filter options added for issue #44
- #53 Dates can now be selected from a date picker widget for filtering studies
- #52 Split the date field into two so either, both or neither can be specified
- #51 Remove import modifications from issue #28 and #43 now that exports are filtered in a better way after #48 and #49 changes.
- #50 No longer necessary to apply a filter before exporting – docs changed to reflect this
- #49 CSV exports changed to use the same filtering routine introduced for #48 to better handle missing attributes
- #48 New feature – can now filter by patient age. Improved export to xlsx to better handle missing attributes
- #47 Install was failing on pydicom – fixed upstream

0.3.8 (2014-03-05)

- – File layout modified to conform to norms
- #46 Updated documentation to reflect limited testing of mammo import on additional modalities
- #45 mam.py was missing the licence header - fixed
- #44 Added Tungsten, Silver and Aluminum to mammo target/filter strings to match – thanks to DJ Platten for strings
- #43 Mammography and Philips CT import and export now more robust for images with missing information such as accession number and collimated field size
- #42 Documentation updated to reflect #37
- #37 Studies now sort by time and date

0.3.7 (2014-02-25)

- #40 Restyled the filter section in the web interface and added a title to that section
- #38 Column titles tidied up in Excel exports
- #36 openrem_ptsizecsv output of log now depends on verbose flag
- #35 Numbers no longer stored as text in Excel exports

0.3.6 (2014-02-24)

- #34 Localised scripts that were on remote web servers in default Bootstrap code
- #33 Documentation now exists for adding data via csv file
- #24 Web interface has been upgraded to Bootstrap v3
- #5 Web interface and export function now have some documentation with screenshots

0.3.5-rc2 (2014-02-17)

- #32 Missing sys import bug prevented new patient size import from working

0.3.5 (2014-02-17)

- – Prettified this document!
- #31 Promoted patient size import from csv function to the scripts folder so it will install and can be called from the path
- #30 Improved patient size import from csv to allow for arbitary column titles and study instance UID in addition to accession number.
- #29 Corrected the docs URL in the readme

0.3.4-rc2 (2014-02-14)

- #28 XLSX export crashed if any of the filter fields were missing. Now fills on import with 'None'
- #27 Use requested procedure description if requested procedure code description is missing

0.3.4 (2014-02-14)

- – General improvements and addition of logo to docs
- #23 Added Windows XP MySQL backup guide to docs
- #22 Added running Conquest as a Windows XP service to docs
- #15 Added version number and copyright information to xlsx exports
- #14 Added version number to the web interface
- #13 Improve the docs with respect to South database migrations

0.3.3-r2 (2014-02-04)

- #12 Added this version history
- #11 Documentation is no longer included in the tar.gz install file – see http://openrem.trfd.org instead

0.3.3 (2014-02-01)

Note: Installs of OpenREM earlier than 0.3.3 will break on upgrade if the scripts are called from other programs. For example openrem_rdsr is now called openrem_rdsr.py

- – Added warning of upgrade breaking existing installs to docs
- #10 Added .py suffix to the scripts to allow them to be executed on Windows (thanks to DJ Platten)
- #8 Removed superfluous '/' in base html file, harmless on linux, prevented Windows loading stylesheets (thanks to DJ Platten)
- #7 Added windows and linux path examples for test SQLite database creation
- #6 Corrected renaming of example files installation instruction (thanks to DJ Platten)
- #4 Added some text to the documentation relating to importing files to OpenREM
- #3 Corrected copyright notice in documentation

0.3.2 (2014-01-29)

- Initial version uploaded to bitbucket.org

2.2 Release notes and upgrade instructions

Each release comes with specific upgrade instructions, so please follow the links below for the appropriate version.

2.2.1 Version specific information

OpenREM Release Notes version 0.4.3

Headline changes

- Export of study information is now handled by a task queue - no more export time-outs.
- Patient size information in csv files can now be uploaded and imported via a web interface.
- Proprietary projection image object created by Hologic tomography units can now be interrogated for details of the tomosynthesis exam.
- Settings.py now ships with its proper name, this will overwrite important local settings if upgrade is from 0.3.9 or earlier.
- Time since last study is no longer wrong just because of daylight saving time!
- Django release set to 1.6; OpenREM isn't ready for Django 1.7 yet
- The inner `openrem` Django project folder is now called `openremproject` to avoid import conflicts with Celery on Windows
- DEBUG mode now defaults to False

Specific upgrade instructions

Always make sure you have converted your database to South before attempting an upgrade

Quick reminder of how, if you haven't done it already:

Linux:

```
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py convert_to_south remapp
```

Windows:

```
python C:\Python27\Lib\site-packages\openrem\manage.py convert_to_south remapp
```

Upgrading from 0.3.9 or earlier **It is essential that you upgrade to at least 0.4.0 first**, then upgrade to 0.4.3. Otherwise the settings file will be overwritten and you will lose your database settings. There is also a trickier than usual database migration and instructions for setting up users. *Fresh installs should start with the latest version.*

Upgrade to version 0.4.2

```
pip install openrem==0.4.2
```

(Will need `sudo` or equivalent if using linux without a virtualenv)

Then follow the instructions in *OpenREM Release Notes version 0.4.0* from migrating the database onwards, before coming back to these instructions.

Upgrading from 0.4.0 or above

Install OpenREM version 0.4.3

```
pip install openrem==0.4.3
```

(Will need `sudo` or equivalent if using linux without a virtualenv)

RabbitMQ The message broker RabbitMQ needs to be installed to enable the export and upload features

- Linux - Follow the guide at http://www.rabbitmq.com/install-debian.html
- Windows - Follow the guide at http://www.rabbitmq.com/install-windows.html

Move and edit local_settings.py file and wsgi.py files The inner `openrem` Django project folder has now been renamed `openremproject` to avoid import confusion that prevented Celery working on Windows.

When you upgrade, the `local_settings.py` file and the `wsgi.py` file will remain in the old `openrem` folder. Both need to be moved across to the `openremproject` folder, and edited as below.

The new and old folders will be found in:

- Linux: `/usr/local/lib/python2.7/dist-packages/openrem/`
- Linux with virtualenv: `/home/myname/openrem/lib/python2.7/site-packages/openrem/`
- Windows: `C:\Python27\Lib\site-packages\openrem\`

Edit the local_settings.py file The `MEDIA_ROOT` path needs to be defined. This is the place where the study exports will be stored for download and where the patient size information csv files will be stored temporarily whilst they are bing processed.

The path set for `MEDIA_ROOT` is up to you, but the user that runs the webserver must have read/write access to the location specified because it is the webserver than reads and writes the files. In a debian linux, this is likely to be `www-data` for a production install. Remember to use forward slashes in the config file, even for Windows.

Linux example:

```
MEDIA_ROOT = "/var/openrem/media/"
```

Windows example:

```
MEDIA_ROOT = "C:/Users/myusername/Documents/OpenREM/media/"
```

The `ALLOWED_HOSTS` needs to be defined, as the `DEBUG` mode is now set to `False`. This needs to contain the server name or IP address that will be used in the URL in the web browser. For example:

```
ALLOWED_HOSTS = [
    '192.168.56.102',
    '.doseserver.',
    'localhost',
]
```

A dot before a hostname allows for subdomains (eg www.doseserver), a dot after a hostname allows for FQDNs (eg doseserver.ad.trust.nhs.uk)

Edit the wsgi.py file with the new project folder name If you aren't using the wsgi.py file as part of your webserver setup, you might like to simply rename the `wsgi.py.example` file in the `openremproject` folder.

If you are using it, edit the line:

```
os.environ.setdefault("DJANGO_SETTINGS_MODULE", "openrem.settings")
```

to read:

```
os.environ.setdefault("DJANGO_SETTINGS_MODULE", "openremproject.settings")
```

Tidying up Finally, you should delete the old `openrem` folder - you might like to take a backup first!

Database migration *Assuming no virtualenv*

Linux:

```
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py schemamigration --auto remapp
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py migrate remapp
```

Windows:

```
C:\Python27\Lib\site-packages\openrem\manage.py schemamigration --auto remapp
C:\Python27\Lib\site-packages\openrem\manage.py migrate remapp
```

Web server If you are using a production webserver, you will probably need to edit some of the configuration to reflect the change in location of `settings.py`, for example `DJANGO_SETTINGS_MODULE = openremproject.settings`, and then restart the web server. You may need to do something similar for the location of `wsgi.py`.

If you are using the built-in test web server (*not for production use*), then the static files will not be served unless you add `--insecure` to the command. This is one of the consequences of setting `DEBUG` to `False`:

```
python manage.py runserver x.x.x.x:8000 --insecure
```

Start the Celery task queue

Note: The webserver and Celery both need to be able to read and write to the `MEDIA_ROOT` location. Therefore you might wish to consider starting Celery using the same user or group as the webserver, and setting the file permissions accordingly.

For testing, in a new shell: *(assuming no virtualenv)*

Linux:

```
cd /usr/local/lib/python2.7/dist-packages/openrem/
celery -A openremproject worker -l info
```

Windows:

```
cd C:\Python27\Lib\site-packages\openrem\
celery -A openremproject worker -l info
```

For production use, see http://celery.readthedocs.org/en/latest/tutorials/daemonizing.html

OpenREM Release Notes version 0.4.2

Headline changes

- This release fixes a major bug introduced in 0.4.0 regarding the import scripts.

Specific upgrade instructions

Upgrading from 0.3.9 or earlier Follow the instructions in *OpenREM Release Notes version 0.4.0*

Upgrading from 0.4.0 or above Move straight to version 0.4.3 and follow the instructions in *OpenREM Release Notes version 0.4.3*

OpenREM Release Notes version 0.4.1

Headline changes

- This release is exacly the same as 0.4.1 bar some documentation corrections

Specific upgrade instructions

Please use the 0.4.0 release notes for upgrades from 0.3.9

OpenREM Release Notes version 0.4.0

OpenREM Release Notes version 0.4.0

Headline changes

- User authentication has been added
- Studies can be deleted from the web interface
- Import scripts can now be passed a list of files, eg `python openrem_rdsr.py *.dcm`
- Date of birth no longer retained for mammography (bug fix - correct behaviour already existed for other imports)
- General bug fixes to enable import from wider range of sources
- Improved user documentation

Specific upgrade instructions

- `pip install openrem==0.4.2` *Go straight to 0.4.2*
- Migrate the database

  > **Warning:** A database migration is required that will need a choice to be made

  - Linux: `python /usr/lib/python2.7/dist-packages/openrem/manage.py schemamigration --auto remapp`
  - Windows: `C:\Python27\Lib\site-packages\openrem\manage.py schemamigration --auto remapp`

  When South has considered the changes to the schema, you will see the following message:

  ```
  ? The field 'Observer_context.device_observer_name' does not have a default specified, y
  ? Since you are making this field nullable, you MUST specify a default
  ? value to use for existing rows. Would you like to:
  ?  1. Quit now.
  ?  2. Specify a one-off value to use for existing columns now
  ?  3. Disable the backwards migration by raising an exception; you can edit the migratio
  ? Please select a choice: 3
  ```

  As per the final line above, the correct choice is `3`. The fields that are now nullable previously weren't. Existing data in those fields will have a value, or those tables don't exist in the current database. The problem scenario is if after the migration these tables are used and one of the new nullable fields is left as null, you will not be able to migrate back to the old database schema without error. This is not something that you will want to do, so this is ok.

  Do the migration:

  - Linux: `python /usr/lib/python2.7/dist-packages/openrem/manage.py migrate remapp`
  - Windows: `C:\Python27\Lib\site-packages\openrem\manage.py migrate remapp`

- Update the settings files

  > **Warning:** The settings file has changed and will need to be manually edited.

  Changes need to be made to the `settings.py` file where the database details are stored. Normally upgrades don't touch this file and the copy in the upgrade has a `.example` suffix. **This upgrade and potentially future ones will need to change this file**, so the format has been changed. The `settings.py` file will now be replaced each time the code is upgraded. In addition, there is a new `local_settings.py` file that contains things that are specific to your installation, such as the database settings.

  This upgrade will include a file called `settings.py.new` and the `local_settings.py.example` file. You will need to do the following:

  - Copy the database settings from your current `settings.py` file to the `local_settings.py.example` file and rename it to remove the `.example`. Both of these files are in the `openrem/openrem` directory, which will be somewhere like
    - Linux: `/usr/lib/python2.7/dist-packages/openrem/openrem/`

    * Windows: `C:\Python27\Lib\site-packages\openrem\openrem\`
  - Move the existing `settings.py` out of the python directories
  - Rename the `settings.py.new` to `settings.py`
- Create a new secret key

  All versions of openrem ship with the same secret key. This key is used for web security checks, and should be unique (and secret) for each installation.

  - Generate a new secret key - http://www.miniwebtool.com/django-secret-key-generator/ is a suitable method of creating a new key.
  - Copy the new key and use it to replace the default key in the `local_settings.py` file
- Restart your webserver
- Add some users
  - Go to the admin interface (eg http://localhost:8000/admin) and log in with the user created when you originally created the database (`manage.py syncdb`)
  - Create some users and add them to the appropriate groups (if there are no groups, go to the OpenREM homepage and they should be created).
    * `viewgroup` can browse the data only
    * `exportgroup` can do as view group plus export data to a spreadsheet, and will be able to import height and weight data in due course (See Issue #21)
    * `admingroup` can delete studies in addition to anything the export group can do

2.3 Contributing authors

The following people have contributed to OpenREM - either with code, documentation or ideas.

- Ed McDonagh
- David Platten
- Jonathan Cole
- Elly Castellano
- Laurence King
- Daniel Gordon

CHAPTER 3

Importing data to OpenREM

3.1 Importing dose related data from DICOM files

If you are using linux, or for Windows if you have put `C:\Python27\;C:\Python27\Lib\site-packages;C:\Python27\` onto your system path, you should be able to import from the command line:

3.1.1 Radiation Dose Structured Reports

```
openrem_rdsr.py filename.dcm
```

In linux, or using an advanced shell in Windows, you can use wildcards to process a number of files at once, ie:

```
openrem_rdsr.py *.dcm
```

This feature will be added for Windows users with the standard cmd.exe shell in a future version, and is tracked as issue #9

3.1.2 For mammography DICOM images

```
openrem_mg.py filename.dcm
```

The facility for extracting dose information from mammography DICOM images has been designed and tested with images created with the GE Senographe DS. It has now been tested in a limited fashion with the images generated by the following systems:

- GE Senographe Essential
- Hologic Selenia
- Siemens Inspiration

See *Mammography module* for testing restrictions.

3.1.3 For CT dose summary files from Philips CT scanners

```
openrem_ctphilips.py filename.dcm
```

3.2 Importing dose related data from csv files using the command line

3.2.1 Patient height and weight information

If height and weight data is not available in the DICOM data, but is available from another source, then it can be imported into the database using the `openrem_ptsizecsv.py` function. Normally the key to match the size information with the studies in the database will be the accession number; however in some situations this isn't available and the Study Instance UID can be used instead.

usage:

```
openrem_ptsizecsv.py [-h] [-u] [-v] csvfile id height weight
```

-h, --help Print the help text.

-u, --si-uid Use Study Instance UID instead of Accession Number.

-v, --verbose *New in 0.3.7* Print to the standard output the success or otherwise of inserting each value.

csvfile csv file containing the height and/or weight information and study identifier. Other columns will be ignored. Use quotes if the filepath has spaces.

id Column title for the accession number or study instance UID. Use quotes if the title has spaces.

height Column title for the patient height (DICOM size) - if this information is missing simply add a blank column with a suitable title. Use quotes if the title has spaces.

weight Column title for the patient weight - if this information is missing simply add a blank column with a suitable title. Use quotes if the title has spaces.

Changed in version 0.3.7: Verbosity flag added to supress printing to the standard output unless requested.

CHAPTER 4

Using the OpenREM web interface

Contents:

4.1 Navigating, filtering and study details

4.1.1 Navigating the OpenREM web interface

Depending on your web server setup, your web interface to OpenREM will usually be at http://yourserver/openrem or if you are using the test web server then it might be at http://localhost:8000/openrem.

The home page for OpenREM should look something like this when it is populated with studies:

OpenREM CT Fluoroscopy Mammography

OpenREM database browser and export

There are 24436 studies in this database. Page last refreshed on 21st February 2014 at 16:29.

CT	Fluoroscopy	Mammography
10925	28	13483

CT summary table

Station name	Number of studies	Latest study
Hospital A, SOMATOM Definition Edge (CT-def-edge1)	402	3 minutes ago
Clinic B, LightSpeed Pro 32 (ls32-03)	4648	15 minutes ago
Hospital A, LightSpeed16 (ls16-rd)	3744	27 minutes ago
NM Clinic, Biograph64 (petct12345)	2131	2 days, 16 hours ago

Fluoroscopy summary table

Station name	Number of studies	Latest study
Clinic B, AXIOM-Artis (AXIS23456)	28	6 days, 2 hours ago

Mammography summary table

Station name	Number of studies	Latest study
Hospital B, Senograph DS ADS_43.10.1 (Mammo3)	6688	30 seconds ago
Clinic B, Senograph DS ADS_43.10.1 (Mammo1)	1184	4 minutes ago
Clinic A, Senograph DS ADS_43.10.1 (Mammo4)	5611	16 minutes ago

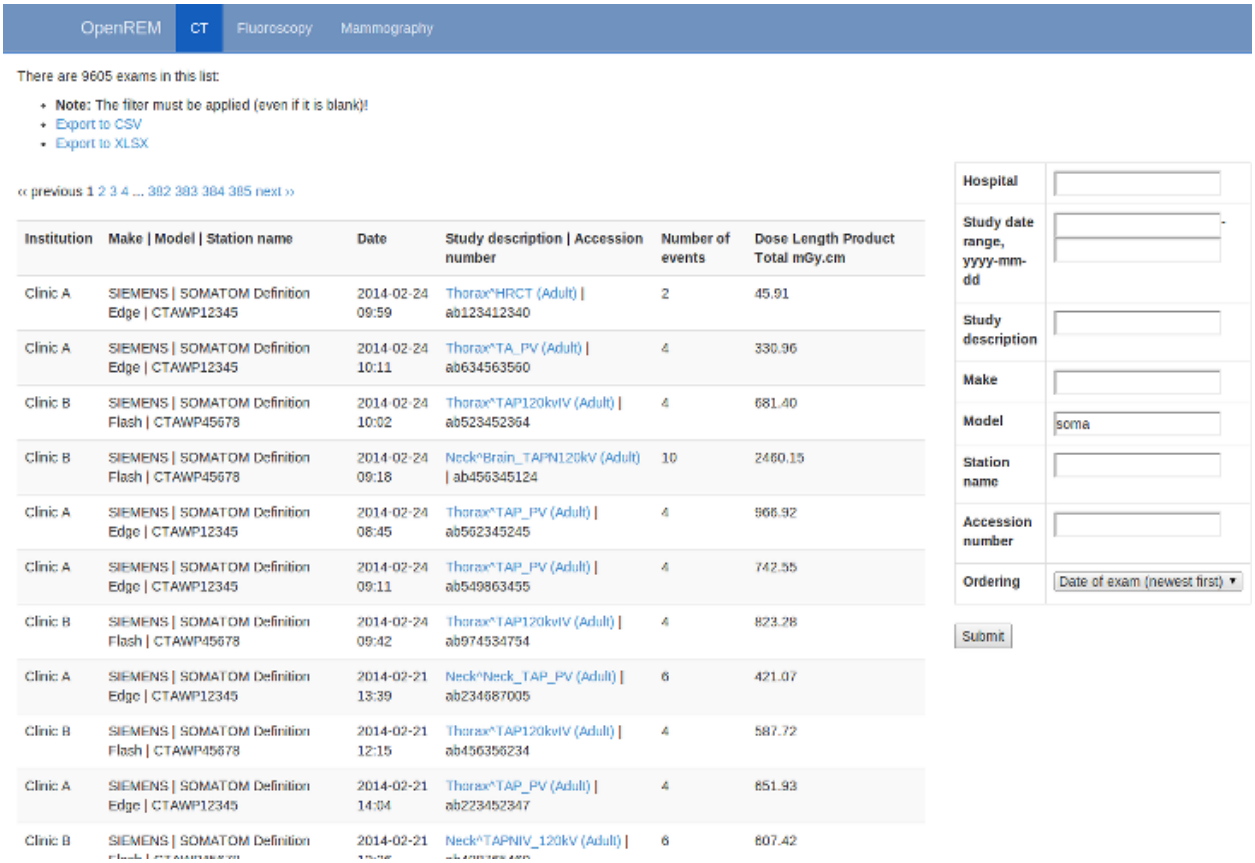
OpenREM version 0.3.6-a1 © 2014 The Royal Marsden NHS Foundation Trust

By selecting the links in the navigation bar at the top, you can view all of the CT, fluoroscopy or mammography studies. Alternatively, if you click on the station name link (in blue) you can filter to just that source modality.

New in 0.4.0: If you are not logged in, clicking any of the links will bring up the log in page.

4.1.2 Filtering for specific studies

This image shows the CT studies view, available to any logged in user, filtered by entering terms in the boxes on the right hand side to show just the studies where the modality model name includes the term 'soma':



Institution	Make | Model | Station name	Date	Study description | Accession number	Number of events	Dose Length Product Total mGy.cm
Clinic A	SIEMENS | SOMATOM Definition Edge | CTAWP12345	2014-02-24 09:59	Thorax^HRCT (Adult) | ab123412340	2	45.91
Clinic A	SIEMENS | SOMATOM Definition Edge | CTAWP12345	2014-02-24 10:11	Thorax^TA_PV (Adult) | ab634563560	4	330.96
Clinic B	SIEMENS | SOMATOM Definition Flash | CTAWP45678	2014-02-24 10:02	Thorax^TAP120kvIV (Adult) | ab523452364	4	681.40
Clinic B	SIEMENS | SOMATOM Definition Flash | CTAWP45678	2014-02-24 09:18	Neck^Brain_TAPN120kV (Adult) | ab456345124	10	2460.15
Clinic A	SIEMENS | SOMATOM Definition Edge | CTAWP12345	2014-02-24 08:45	Thorax^TAP_PV (Adult) | ab562345245	4	966.92
Clinic A	SIEMENS | SOMATOM Definition Edge | CTAWP12345	2014-02-24 09:11	Thorax^TAP_PV (Adult) | ab549863455	4	742.55
Clinic B	SIEMENS | SOMATOM Definition Flash | CTAWP45678	2014-02-24 09:42	Thorax^TAP120kvIV (Adult) | ab974534754	4	823.28
Clinic A	SIEMENS | SOMATOM Definition Edge | CTAWP12345	2014-02-21 13:39	Neck^Neck_TAP_PV (Adult) | ab234687005	6	421.07
Clinic B	SIEMENS | SOMATOM Definition Flash | CTAWP45678	2014-02-21 12:15	Thorax^TAP120kvIV (Adult) | ab456356234	4	587.72
Clinic A	SIEMENS | SOMATOM Definition Edge | CTAWP12345	2014-02-21 14:04	Thorax^TAP_PV (Adult) | ab223452347	4	851.93
Clinic B	SIEMENS | SOMATOM Definition	2014-02-21	Neck^TAPNIV_120kV (Adult) |	6	807.42

The search fields can all be used on their own or together, and they are all case insensitive 'contains' searches. The exception is the date field, where both from and to have to be filled in (if either are), and the format must be `yyyy-mm-dd`. There currently isn't any more complex filtering available, but it does exist as issue 17 for a future release.

The last box below the filtering search boxes is the ordering preference.

4.1.3 Viewing study details

By clicking on the study description link (in blue), you can see more details for an individual study:

OpenREM | CT | Fluoroscopy | Mammography

Detail list of events

- Accession number: ab462362354
- Study date: 23 Jan 2013
- Study time: 1:17 p.m.
- Study description: Dual Energy^DE_TAP_IV (Adult)
- Requested procedure: CT Thorax abdomen and pelvis with contrast
- Patient age: 52.8
- Patient height and weight: 190 cm, 86 kg
- Hospital: Clinic B
- Scanner: SIEMENS | SOMATOM Definition Flash | CTAWP73491
- Study UID: 1.2.840.113564.9.1.27282345238.69.2.508347462734
- Comment:
- Test patient indicators? None

Acquisition protocol	Type	CTDIvol mGy	DLP mGy.cm	Scanning length (mm)	kVp	mA	Max mA	Exposure time per rotation (s)	Pitch	Exposure time (s)	Slice thickness (mm)	Collimation (mm)	X-ray modulation type
Topogram	Constant Angle Acquisition	0.14	11.26	803	120	35	35		None	8.190	0.600	3.60	OFF
Comment Internal technical scan parameters: Organ Characteristic = Thorax, Body Size = Adult, Body Region = Body, X-ray Modulation Type = OFF													
Topogram	Constant Angle Acquisition	0.14	11.54	824	120	35	35		None	8.400	0.600	3.60	OFF
Comment Internal technical scan parameters: Organ Characteristic = Thorax, Body Size = Adult, Body Region = Body, X-ray Modulation Type = OFF													
PreMonitoring	Stationary Acquisition	1.82	1.82	10	120	59	60	0.500	None	0.500	10.000	10.00	OFF
Comment Internal technical scan parameters: Organ Characteristic = Abdomen, Body Size = Adult, Body Region = Body, X-ray Modulation Type = OFF													
Monitoring	Stationary Acquisition	7.27	7.27	10	120	59	60	0.500	None	2.000	10.000	10.00	OFF
Comment Internal technical scan parameters: Organ Characteristic = Abdomen, Body Size = Adult, Body Region = Body, X-ray Modulation Type = OFF													
DE_TAP	Spiral Acquisition	8.11	630.63	797	100	165	430	0.500	0.8000	26.050	0.600	19.20	XYZ_EC
					140	131	307	0.500					
Comment Internal technical scan parameters: Organ Characteristic = Abdomen, Body Size = Adult, Body Region = Body, X-ray Modulation Type = XYZ_EC, Sn Filter (Tube B) = yes													

Not all the details stored for any one study are displayed, just those thought to be most useful. If there are others you'd like to see, add an issue to the tracker.

The final field in the summary at the top is called 'Test patient indicators?' When studies are imported the ID and patient name fields are both ignored, but they are parsed to check if they have 'phy', 'test' or 'qa' in them to help exclude them from the data analysis. If they do, then this information is added to the field and is displayed both in the web interface as a Test patient indicator and in the Excel export. The name and ID themselves are not reproduced, simply the presence of one of the key words. Therefore a patient named 'Phyliss' would trigger this, but only 'Phy' would be reproduced in this field. Other fields will also help to confirm whether a study is for a real patient such as the lack of an Accession Number and an unusual patient age.

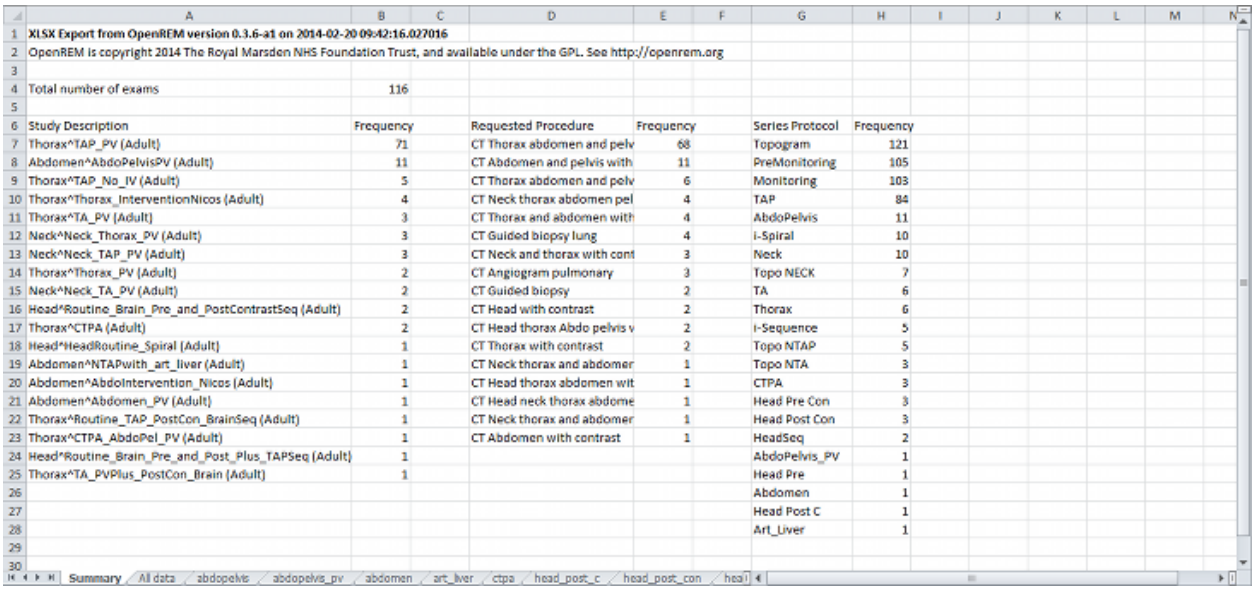
4.2 Exporting study information

4.2.1 Exporting to csv and xlsx sheets

If you are logged in as a user in the `exportgroup` or the `admingroup`, the export links will be available near the top of the modality filter pages in the OpenREM interface. The following exports are currently available (version 0.4.3)

- CT basic, single sheet csv
- CT advanced, XLSX muliple-sheets
- Fluoroscopy basic, single sheet csv
- Mammography, single sheet csv
- Mammogrpahy NHSBSP, single sheet csv designed to satisfy NHSPSB reporting

For CT, the XLSX export has multiple sheets. The first sheet contains a summary of all the study descriptions, requested procedures and series protocol names contained in the export:

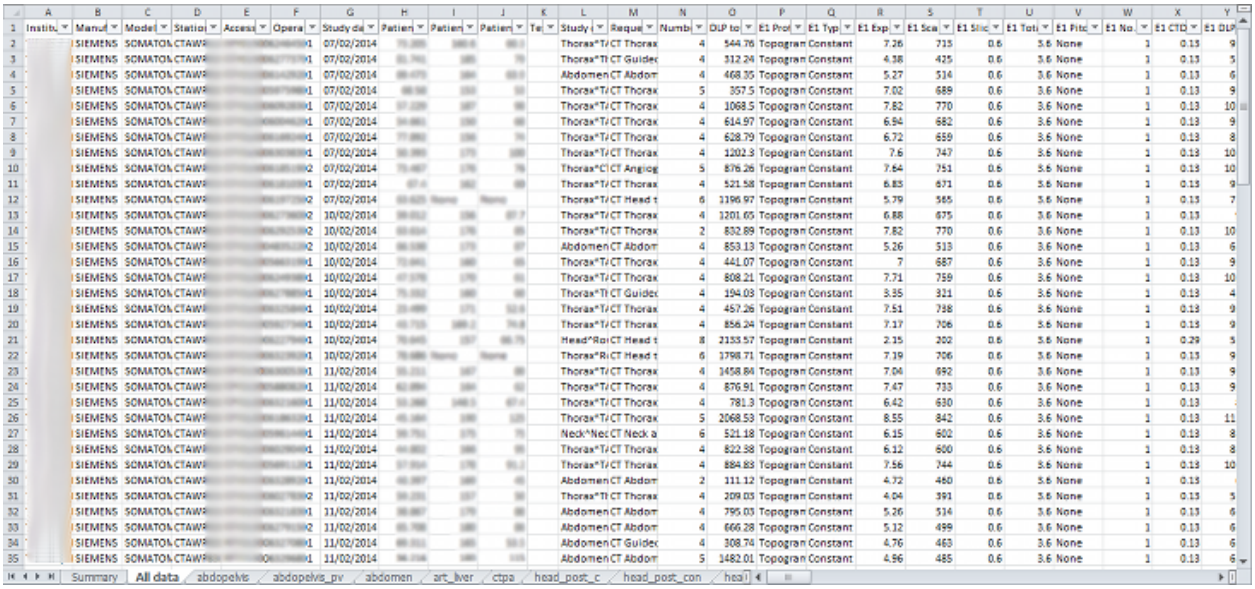


	A	B	C	D	E	F	G	H
1	XLSX Export from OpenREM version 0.3.6-a1 on 2014-02-20 09:42:16.027016							
2	OpenREM is copyright 2014 The Royal Marsden NHS Foundation Trust, and available under the GPL. See http://openrem.org							
3								
4	Total number of exams	116						
5								
6	Study Description	Frequency		Requested Procedure	Frequency		Series Protocol	Frequency
7	Thorax^TAP_PV (Adult)	71		CT Thorax abdomen and pelv	68		Topogram	121
8	Abdomen^AbdoPelvisPV (Adult)	11		CT Abdomen and pelvis with	11		PreMonitoring	105
9	Thorax^TAP_No_IV (Adult)	5		CT Thorax abdomen and pelv	6		Monitoring	103
10	Thorax^Thorax_InterventionNicos (Adult)	4		CT Neck thorax abdomen pel	4		TAP	84
11	Thorax^TA_PV (Adult)	3		CT Thorax and abdomen with	4		AbdoPelvis	11
12	Neck^Neck_Thorax_PV (Adult)	3		CT Guided biopsy lung	4		i-Spiral	10
13	Neck^Neck_TAP_PV (Adult)	3		CT Neck and thorax with cont	3		Neck	10
14	Thorax^Thorax_PV (Adult)	2		CT Angiogram pulmonary	3		Topo NECK	7
15	Neck^Neck_TA_PV (Adult)	2		CT Guided biopsy	2		TA	6
16	Head^Routine_Brain_Pre_and_PostContrastSeq (Adult)	2		CT Head with contrast	2		Thorax	6
17	Thorax^CTPA (Adult)	2		CT Head thorax Abdo pelvis v	2		i-Sequence	5
18	Head^HeadRoutine_Spiral (Adult)	1		CT Thorax with contrast	2		Topo NTAP	5
19	Abdomen^NTAPwith_art_liver (Adult)	1		CT Neck thorax and abdomer	1		Topo NTA	3
20	Abdomen^AbdoIntervention_Nicos (Adult)	1		CT Head thorax abdomen wit	1		CTPA	3
21	Abdomen^Abdomen_PV (Adult)	1		CT Head neck thorax abdome	1		Head Pre Con	3
22	Thorax^Routine_TAP_PostCon_BrainSeq (Adult)	1		CT Neck thorax and abdomer	1		Head Post Con	3
23	Thorax^CTPA_AbdoPel_PV (Adult)	1		CT Abdomen with contrast	1		HeadSeq	2
24	Head^Routine_Brain_Pre_and_Post_Plus_TAPSeq (Adult)	1					AbdoPelvis_PV	1
25	Thorax^TA_PVPlus_PostCon_Brain (Adult)	1					Head Pre	1
26							Abdomen	1
27							Head Post C	1
28							Art_Liver	1
29								
30								

Summary / All data / abdopelvis / abdopelvis_pv / abdomen / art_liver / ctpa / head_post_c / head_post_con / heal

This information is useful for seeing what data is in the spreadsheet, and can also be used to prioritise which studies or protocols to analyse based on frequency.

The second sheet of the exported file lists all the studies, with each study taking one line and each series in the study displayed in the columns to the right.

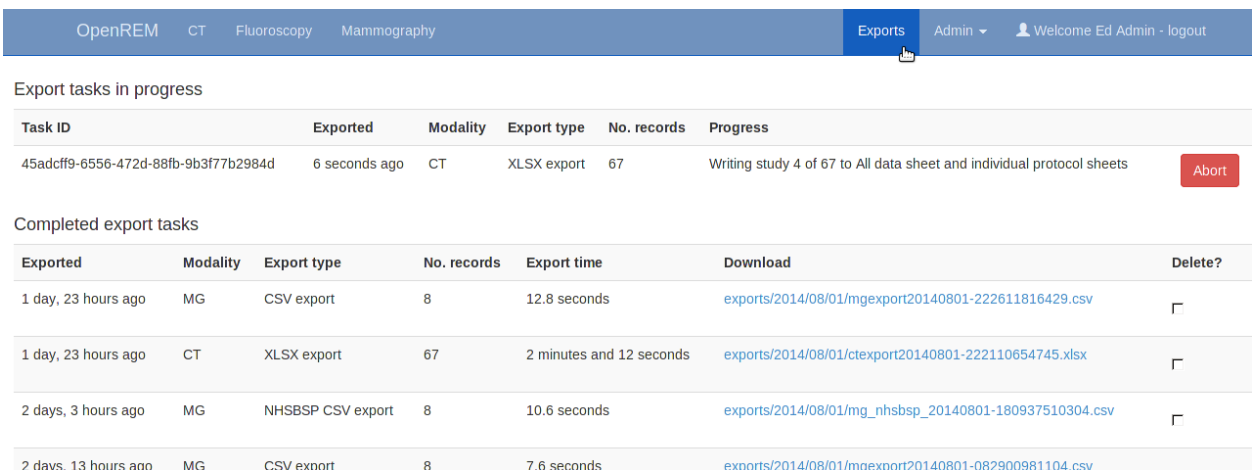


	Institu	Manuf	Model	Statio	Access	Opera	Study da	Patien	Patien	Patien	Tel	Study	Reque	Numb	DLP to	E1 Pro	E1 Typ	E1 Exp	E1 Sca	E1 Stic	E1 Toti	E1 Fitc	E1 No	E1 CTD	E1 DLP
2		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Thorax	4	544.76	Topogram	Constant	7.26	715	0.6	3.6	None	1	0.13	9
3		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Guidec	4	312.24	Topogram	Constant	4.38	425	0.6	3.6	None	1	0.13	5
4		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Abdomen	CT Abdom	4	468.35	Topogram	Constant	5.27	514	0.6	3.6	None	1	0.13	6
5		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Thorax	5	357.5	Topogram	Constant	7.02	689	0.6	3.6	None	1	0.13	9
6		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Thorax	4	1068.5	Topogram	Constant	7.82	770	0.6	3.6	None	1	0.13	10
7		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Thorax	4	614.97	Topogram	Constant	6.94	682	0.6	3.6	None	1	0.13	9
8		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Thorax	4	628.79	Topogram	Constant	6.72	659	0.6	3.6	None	1	0.13	8
9		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Thorax	4	1202.3	Topogram	Constant	7.6	747	0.6	3.6	None	1	0.13	10
10		SIEMENS	SOMATOM	CTAWS		2	07/02/2014					Thorax^C	CT Angiog	5	876.26	Topogram	Constant	7.64	751	0.6	3.6	None	1	0.13	10
11		SIEMENS	SOMATOM	CTAWS		1	07/02/2014					Thorax^T	CT Thorax	4	521.58	Topogram	Constant	6.83	671	0.6	3.6	None	1	0.13	9
12		SIEMENS	SOMATOM	CTAWS		2	07/02/2014					Thorax^T	CT Head t	6	1196.97	Topogram	Constant	5.79	565	0.6	3.6	None	1	0.13	7
13		SIEMENS	SOMATOM	CTAWS		2	10/02/2014					Thorax^T	CT Thorax	4	1201.65	Topogram	Constant	6.88	675	0.6	3.6	None	1	0.13	
14		SIEMENS	SOMATOM	CTAWS		2	10/02/2014					Thorax^T	CT Thorax	2	832.89	Topogram	Constant	7.82	770	0.6	3.6	None	1	0.13	10
15		SIEMENS	SOMATOM	CTAWS		2	10/02/2014					Abdomen	CT Abdom	4	853.13	Topogram	Constant	5.26	513	0.6	3.6	None	1	0.13	6
16		SIEMENS	SOMATOM	CTAWS		1	10/02/2014					Thorax^T	CT Thorax	4	441.07	Topogram	Constant	7	687	0.6	3.6	None	1	0.13	9
17		SIEMENS	SOMATOM	CTAWS		1	10/02/2014					Thorax^T	CT Thorax	4	808.21	Topogram	Constant	7.71	759	0.6	3.6	None	1	0.13	10
18		SIEMENS	SOMATOM	CTAWS		1	10/02/2014					Thorax^T	CT Guidec	4	194.03	Topogram	Constant	3.35	321	0.6	3.6	None	1	0.13	4
19		SIEMENS	SOMATOM	CTAWS		1	10/02/2014					Thorax^T	CT Thorax	4	457.26	Topogram	Constant	7.51	738	0.6	3.6	None	1	0.13	9
20		SIEMENS	SOMATOM	CTAWS		1	10/02/2014					Thorax^T	CT Thorax	4	856.24	Topogram	Constant	7.17	706	0.6	3.6	None	1	0.13	9
21		SIEMENS	SOMATOM	CTAWS		1	10/02/2014					Head^Ro	CT Head t	8	2133.57	Topogram	Constant	2.15	202	0.6	3.6	None	1	0.29	5
22		SIEMENS	SOMATOM	CTAWS		1	10/02/2014					Thorax^R	CT Head t	6	1798.71	Topogram	Constant	7.19	706	0.6	3.6	None	1	0.13	9
23		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Thorax^T	CT Thorax	4	1458.84	Topogram	Constant	7.04	692	0.6	3.6	None	1	0.13	9
24		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Thorax^T	CT Thorax	4	876.91	Topogram	Constant	7.47	733	0.6	3.6	None	1	0.13	9
25		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Thorax^T	CT Thorax	4	781.3	Topogram	Constant	6.42	630	0.6	3.6	None	1	0.13	
26		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Thorax^T	CT Thorax	5	2068.53	Topogram	Constant	8.55	842	0.6	3.6	None	1	0.13	11
27		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Neck^Nec	CT Neck a	6	521.18	Topogram	Constant	6.15	602	0.6	3.6	None	1	0.13	8
28		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Thorax^T	CT Thorax	4	822.38	Topogram	Constant	6.12	600	0.6	3.6	None	1	0.13	8
29		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Thorax^T	CT Thorax	4	884.83	Topogram	Constant	7.56	744	0.6	3.6	None	1	0.13	10
30		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Abdomen	CT Abdom	2	111.12	Topogram	Constant	4.72	460	0.6	3.6	None	1	0.13	
31		SIEMENS	SOMATOM	CTAWS		2	11/02/2014					Thorax^T	CT Thorax	4	209.03	Topogram	Constant	4.04	391	0.6	3.6	None	1	0.13	5
32		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Abdomen	CT Abdom	4	795.03	Topogram	Constant	5.26	514	0.6	3.6	None	1	0.13	6
33		SIEMENS	SOMATOM	CTAWS		2	11/02/2014					Abdomen	CT Abdom	4	666.28	Topogram	Constant	5.12	499	0.6	3.6	None	1	0.13	6
34		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Abdomen	CT Guidec	4	308.74	Topogram	Constant	4.76	463	0.6	3.6	None	1	0.13	6
35		SIEMENS	SOMATOM	CTAWS		1	11/02/2014					Abdomen	CT Abdom	5	1482.01	Topogram	Constant	4.96	485	0.6	3.6	None	1	0.13	6

Summary | All data | abdopelvis | abdopelvis_pv | abdomen | art_liver | ctpa | head_post_c | head_post_con | head

The remainder of the file has one sheet per series protocol name. Each series is listed one per line. If a single study has more than one series with the same protocol name, then the same study will appear on more than one line.

Clicking the link for an export redirects you to the Exports page, which you can also get to using the link at the top right of the navigation bar:



Whilst an export is being processed, it will be listed in the first table at the top. The current status is displayed to indicate export progress. If an export gets stuck for whatever reason, you may be able to abort the process by clicking the 'Abort' button. However this does not always cause an active export to terminate - you may find it completes anyway!

Completed exports are then listed in the second table, with a link to download the csv or xlsx file.

When the export is no longer needed, it can be deleted from the server by ticking the delete checkbox and clicking the delete button at the bottom:

ort time	Download	Delete?
inute and 53 seconds	ctexport20140716-183851522438.xlsx	☐
inutes and 0 seconds	ctexport20140716-183304657348.xlsx	☑
seconds	mg_nhsbsp_20140716-082441362714.csv	☐
seconds	mgexport20140716-082415172249.csv	☑
seconds	rfexport20140716-081609865749.csv	☑
		Delete

Warning: Large exports have been killed by the operating system due to running out of memory - a 6500 CT exam xlsx export was killed after 3400 studies for example. This issue is being tracked as #116 and will hopefully be addressed in the next release. It is possible that if debug mode is turned off then memory will be managed better, but I also need to modify the xlsx export to make use of the memory optimisation mode in xlsxwriter.

4.3 OpenREM administration (deleting studies, importing patient size data)

Contents:

4.3.1 Deleting studies

New in 0.4.0

If you log in as a user that is in the `admingroup`, then an extra column is appended in the filtered view tables to allow studies to be deleted:

ation name	Date	Study description | Accession number	Number of events	Dose Length Product Total mGy.cm	Delete?
TOM CTAWP1234	2013-05-23 10:09	Thorax^TAP120kvIV (Adult) | [redacted]1	4	1257.10	Delete
TOM CTAWP1234	2013-05-23 11:05	Thorax^TA_IV120kV (Adult) | [redacted]1	4	314.26	Delete
TOM	2013-05-23	Thorax^TAP120kvIV (Adult) |	4	688.99	Delete

Clicking on delete takes you to a confirmation page before the delete takes place.

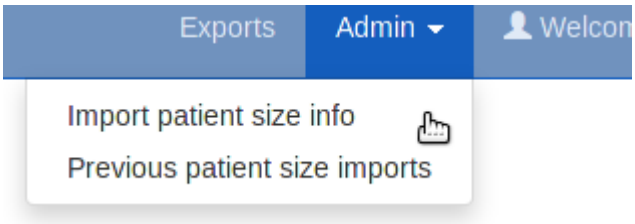
4.3.2 Adding patient size information from csv using the web interface

New in 0.4.3

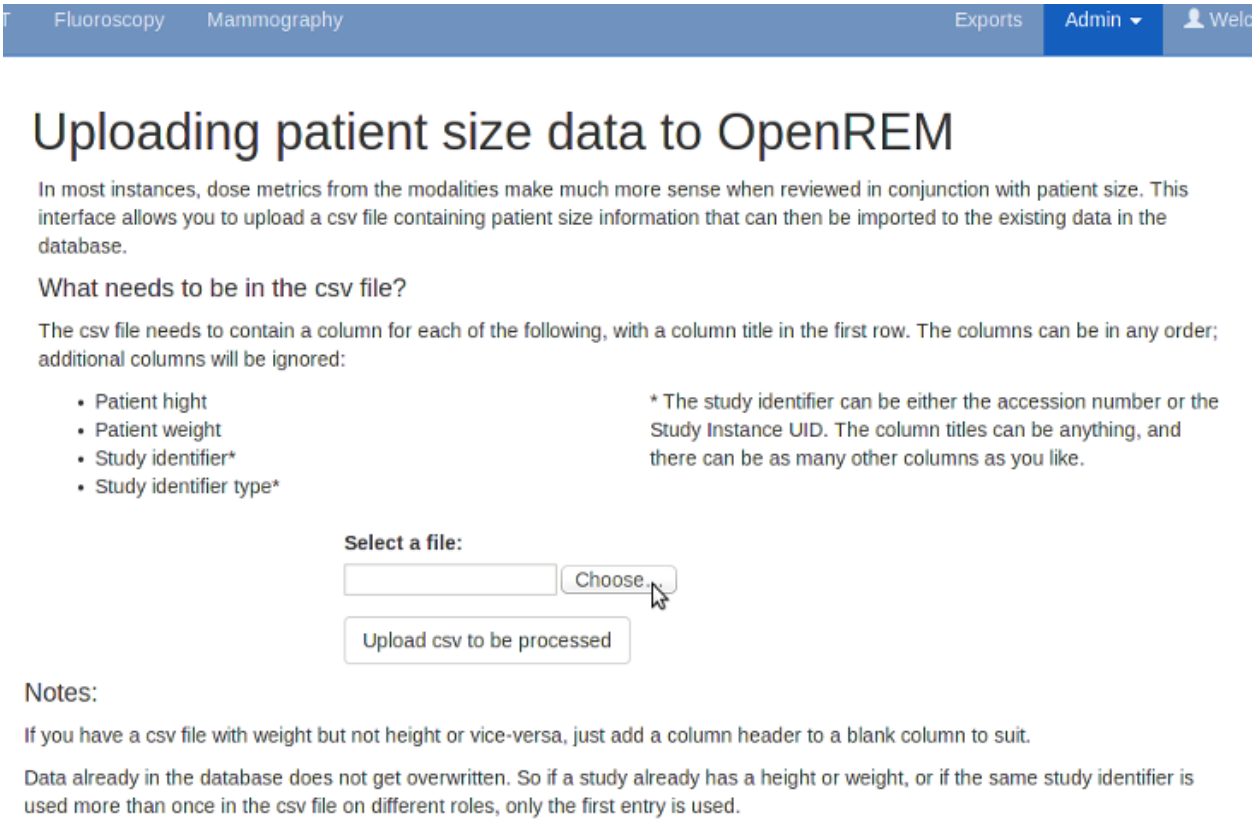
Contents

Uploading patient size data

If you log in as a user that is in the `admingroup`, then a menu is available at the right hand end of the navigation bar:



The first option takes you to a page where you can upload a csv file containing details of the patient height and weight, plus either the accession number or the Study Instance UID.



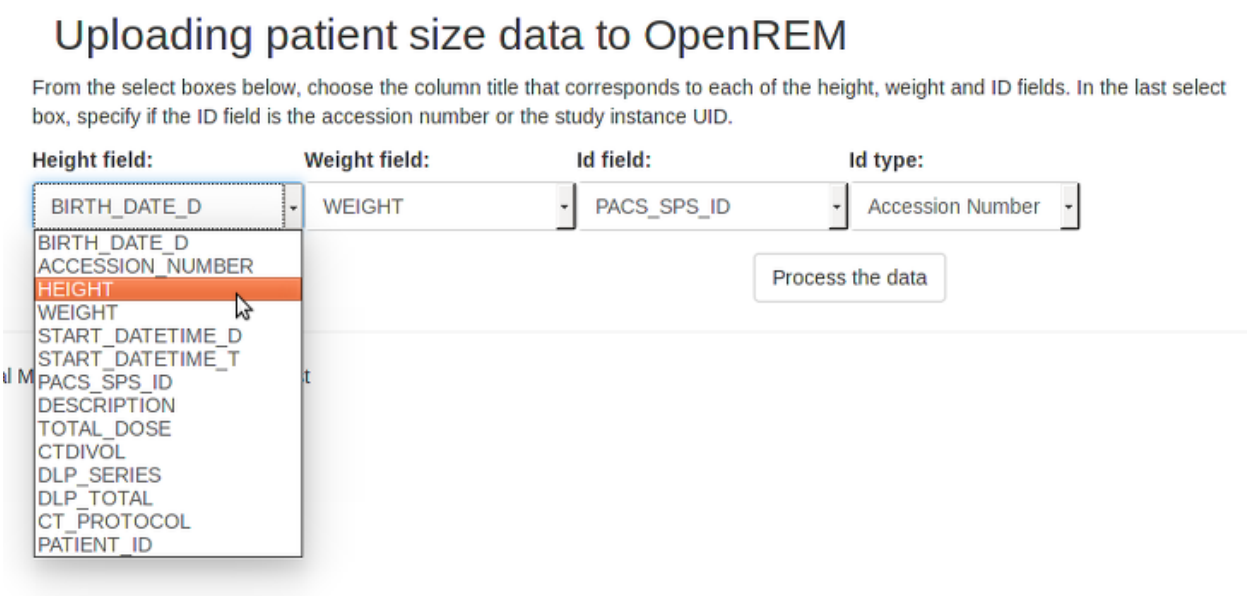
The csv file needs to have at least the required columns. Additional columns will be ignored. If your source of patient size data does not have either the height or the weight column, simply add a new empty column with just the title in the first row.

When you have selected the csv file, press the button to upload it.

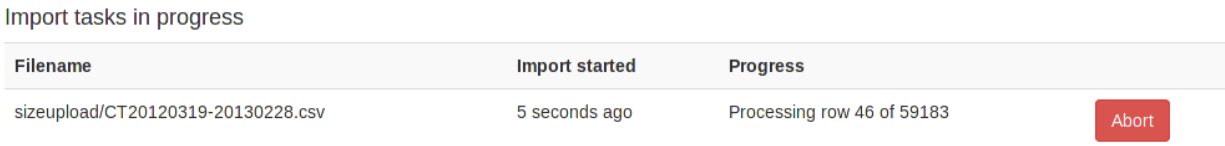
Importing the size data to the database

On the next page select the column header that corresponds to each of the head, weight and ID fields. Also select whether the ID field is an Accession number or a Study UID:

When the column headers are selected, click the 'Process the data' button.



The progress of the import is then reported on the patient size imports page:

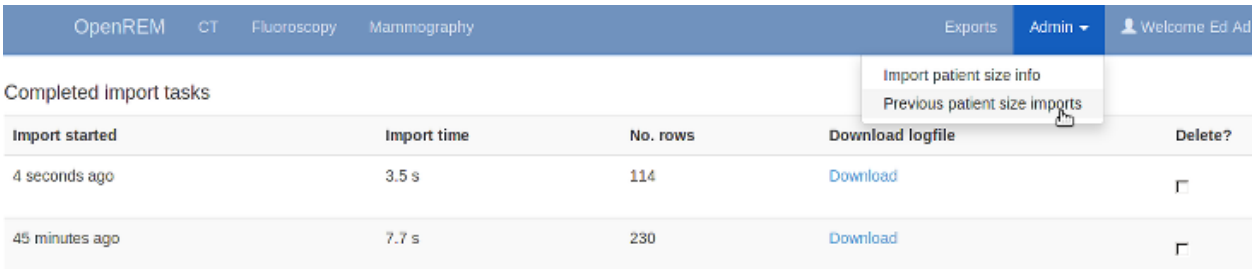


During the import, it is possible to abort the process by clicking the button seen in the image above. The log file is available from the completed table whether it completed or not - there is no indication that the import was aborted.

As soon as the import is complete, the source csv file is deleted from the server.

Reviewing previous imports

After an import is complete, it is listed in the completed import tasks table. You can also get to this page from the Admin menu:



For each import, there is a link to the logfile, which looks something like this. With this import accession numbers weren't available so the patient size information was matched to the study instance UID:

```
Patient size import from sizeupload/2014/07/11/doctored.csv

1.3.12.2.1107.5.4.5.146226.30000012080207411271800000009:
    Height of 166.50 m not inserted as 166.5 cm already in the database
    Weight of 58.15 kg not inserted as 58.15 kg already in the database
1.3.51.0.1.1.192.168.90.77.100000611814.611849:
    Height of 165 m not inserted as 165 cm already in the database
    Weight of 87 kg not inserted as 87 kg already in the database
1.2.840.113704.1.111.5924.1371549177.10:
    Inserted height of 184 cm
    Inserted weight of 113 kg
1.2.840.113704.1.111.5000.1371472141.5:
    Inserted height of 166.10 cm
    Inserted weight of 95.50 kg
1.2.840.113704.1.111.5000.1371472199.6:
    Inserted height of 172 cm
    Inserted weight of 55 kg
```

Deleting import logs

The completed import tasks table also has a delete check box against each record and a delete button at the bottom. The csv file originally imported has already been deleted - this delete function is to remove the record of the import and the log file associated with it from the database/disk.

CHAPTER 5

Documentation for the OpenREM code

Contents:

5.1 DICOM import modules

5.1.1 RDSR module

Ultimately this should be the only module required as it deals with all Radiation Dose Structured Reports. Currently this has only been tested on CT and fluoroscopy structured reports, but it also has the logic for mammography structured reports if they start to appear.

`remapp.extractors.rdsr.rdsr`(*rdsr_file*)

Extract radiation dose related data from DICOM Radiation SR objects.

Parameters **filename** (*str.*) – relative or absolute path to Radiation Dose Structured Report.

Tested with:

- CT: Siemens, Philips and GE RDSR, GE Enhanced SR.
- Fluoro: Siemens Artis Zee RDSR

5.1.2 Mammography module

Mammography is interesting in that all the information required for dose audit is contained in the image header, including patient 'size', ie thickness. However the disadvantage over an RSDR is the requirement to process each individual image rather than a single report for the study, which would also capture any rejected images.

`remapp.extractors.mam.mam`(*mg_file*)

Extract radiation dose structured report related data from mammography images

Parameters **filename** (*str.*) – relative or absolute path to mammography DICOM image file.

Tested with:

- GE Senographe DS software versions ADS_43.10.1 and ADS_53.10.10 only.
- Limited testing: GE Senographe Essential
- Limited testing: Hologic Selenia
- Limited testing: Siemens Inspiration

5.1.3 CT non-standard modules

Initially only Philips CT dose report images are catered for. These have all the information that could be derived from the images also held in the DICOM header information, making harvesting relatively easy.

`remapp.extractors.ct_philips.`**`ct_philips`**(*philips_file*)

Extract radiation dose structured report related data from Philips CT dose report images

Parameters **filename** (*str.*) – relative or absolute path to Philips CT dose report DICOM image file.

Tested with:

- Philips Gemini TF PET-CT v2.3.0
- Brilliance BigBore v3.5.4.17001.

5.2 Non-DICOM import modules

5.2.1 Patient height and weight csv import module

This module enables a csv file to be parsed and the height and weight information extracted and added to existing studies in the OpenREM database. An example may be a csv extract from a RIS or EPR system.

There needs to be a common unique identifier for the exam - currently this is limited to accession number or study instance UID.

`remapp.extractors.ptsizecsv2db.`**`csv2db`**(**args*, ***kwargs*)

Import patient height and weight data from csv RIS exports. Can be called from `openrem_ptsizecsv.py` script

Parameters

- **–si-uid** (*bool*) – Use Study Instance UID instead of Accession Number. Short form -s.
- **csvfile** (*str*) – relative or absolute path to csv file
- **id** (*str*) – Accession number column header or header if -u or –si-uid is set. Quote if necessary.
- **height** (*str*) – Patient height column header. Create if necessary, quote if necessary.
- **weight** (*str*) – Patient weight column header. Create if necessary, quote if necessary.

Example:

```
openrem_ptsizecsv.py -s MyRISExport.csv StudyInstanceUID HEIGHT weight
```

(task) `remapp.extractors.ptsizecsv2db.`**`websizeimport`**(*csv_pk=None*, **args*, ***kwargs*)

Task to import patient size data from the OpenREM web interface.

Parameters **csv_pk** – Database index key for the import record, containing the path to the import csv file and the field header details.

5.3 Export from database

5.3.1 Multi-sheet Microsoft Excel XLSX exports

This export has a summary sheet of all the requested and performed protocols and the series protocols. The next sheet has all studies on, one study per line, with the series stretching off to the right. The remaining sheets are specific to each series protocol, in alphabetical order, with one series per line. If one study has three series with the same protocol name, each one has a line of its own.

(task)`remapp.exports.xlsx.`**`ctxlsx`**

5.3.2 Single sheet CSV exports

Specialised csv exports - NHSBSP formatted mammography export

5.4 Tools and helper modules

5.4.1 OpenREM settings

Administrative module to define the name of the project and to add it to the Python path

`remapp.extractors.openrem_settings.`**`add_project_to_path`**`()`

Add project to path, assuming this file is within project

5.4.2 Get values

Tiny modules to reduce repetition in the main code when extracting information from DICOM headers using pydicom.

`remapp.tools.get_values.`**`get_value_kw`**`(`*tag*, *dataset*`)`

Get DICOM value by keyword reference.

Parameters

- **keyword** (*str.*) – DICOM keyword, no spaces or plural as per dictionary.
- **dataset** (*dataset*) – The DICOM dataset containing the tag.

Returns str. – value

`remapp.tools.get_values.`**`get_value_num`**`(`*tag*, *dataset*`)`

Get DICOM value by tag group and element number.

Always use get_value_kw by preference for readability. This module can be required when reading private elements.

Parameters

- **tag** (*hex*) – DICOM group and element number as a single hexadecimal number (prefix 0x).
- **dataset** (*dataset*) – The DICOM dataset containing the tag.

Returns str. – value

`remapp.tools.get_values.`**`get_seq_code_value`**`(`*sequence*, *dataset*`)`

From a DICOM sequence, get the code value.

Parameters

- **sequence** (*DICOM keyword, no spaces or plural as per dictionary.*) – DICOM sequence name.
- **dataset** (*DICOM dataset*) – The DICOM dataset containing the sequence.

Returns int. – code value

`remapp.tools.get_values.`**`get_seq_code_meaning`**(*sequence*, *dataset*)
From a DICOM sequence, get the code meaning.

Parameters

- **sequence** (*DICOM keyword, no spaces or plural as per dictionary.*) – DICOM sequence name.
- **dataset** (*DICOM dataset*) – The DICOM dataset containing the sequence.

Returns str. – code meaning

`remapp.tools.get_values.`**`get_or_create_cid`**(*codevalue*, *codemeaning*)
Create a code_value code_meaning pair entry in the Content_item_descriptions table if it doesn't already exist.

Parameters

- **codevalue** (*int.*) – Code value as defined in the DICOM standard part 16
- **codemeaning** – Code meaning as defined in the DICOM standard part 16

Returns Content_item_descriptions entry for code value passed

5.4.3 Check if UID exists

Small module to check if UID already exists in the database.

`remapp.tools.check_uid.`**`check_uid`**(*uid*, *level='Study'*)
Check if UID already exists in database.

Parameters **uid** (*str.*) – Study UID.

Returns 1 if it does exist, 0 otherwise

5.4.4 DICOM time and date values

Module to convert betweeen DICOM and Python dates and times.

`remapp.tools.dcmdatetime.`**`get_date`**(*tag*, *dataset*)
Get DICOM date string and return Python date.

Parameters

- **tag** (*str.*) – DICOM keyword, no spaces or plural as per dictionary.
- **dataset** (*dataset*) – The DICOM dataset containing the tag.

Returns Python date value

`remapp.tools.dcmdatetime.`**`get_time`**(*tag*, *dataset*)
Get DICOM time string and return Python time.

Parameters

- **tag** (*str.*) – DICOM keyword, no spaces or plural as per dictionary.

- **dataset** (*dataset*) – The DICOM dataset containing the tag.

Returns python time value

`remapp.tools.dcmdatetime.get_date_time(tag, dataset)`
Get DICOM date time string and return Python date time.

Parameters

- **tag** (*str.*) – DICOM keyword, no spaces or plural as per dictionary.
- **dataset** (*dataset*) – The DICOM dataset containing the tag.

Returns Python date time value

`remapp.tools.dcmdatetime.make_date(dicomdate)`
Given a DICOM date, return a Python date.

Parameters **dicomdate** (*str.*) – DICOM style date.

Returns Python date value

`remapp.tools.dcmdatetime.make_time(dicomtime)`
Given a DICOM time, return a Python time.

Parameters **dicomdate** (*str.*) – DICOM style time.

Returns Python time value

`remapp.tools.dcmdatetime.make_date_time(dicomdatetime)`
Given a DICOM date time, return a Python date time.

Parameters **dicomdate** (*str.*) – DICOM style date time.

Returns Python date time value

5.4.5 Test for QA or other non-patient related studies

`remapp.tools.not_patient_indicators.get_not_pt(dataset)`
Looks for indications that a study might be a test or QA study.

Some values that might indicate a study was for QA or similar purposes are not recorded in the database, for example patient name. Therefore this module attempts to find such indications and creates an xml style string that can be recorded in the database.

Parameters **dataset** (*dataset*) – The DICOM dataset.

Returns str. – xml style string if any trigger values are found.

5.5 Models

class `remapp.models.Size_upload(*args, **kwargs)`
Size_upload(id, sizefile, height_field, weight_field, id_field, id_type, task_id, status, progress, num_records, logfile, import_date, processtime)

class `remapp.models.Exports(*args, **kwargs)`
Table to hold the export status and filenames

class `remapp.models.Content_item_descriptions(*args, **kwargs)`
Table to hold all the context ID code values and code meanings.

- Should be renamed Context_identifiers. If it does, I think it would only need to be edited in tools.get_values.get_or_set_cid and admin, then a South migration.
- Could be prefilled from the tables in DICOM 3.16, but is actually populated as the codes occur. This assumes they are used correctly.

class `remapp.models.`**`General_study_module_attributes`**(**args*, ***kwargs*)

General Study Module C.7.2.1

Specifies the Attributes that describe and identify the Study performed upon the Patient.

From DICOM Part 3: Information Object Definitions Table C.7-3

Additional to the module definition:

- performing_physician_name
- operator_name
- modality_type
- procedure_code_value_and_meaning
- requested_procedure_code_value_and_meaning

class `remapp.models.`**`Projection_xray_radiation_dose`**(**args*, ***kwargs*)

Projection X-Ray Radiation Dose template TID 10001

From DICOM Part 16: This template defines a container (the root) with subsidiary content items, each of which represents a single projection X-Ray irradiation event entry or plane-specific dose accumulations. There is a defined recording observer (the system or person responsible for recording the log, generally the system). A Biplane irradiation event will be recorded as two individual events, one for each plane. Accumulated values will be kept separate for each plane.

class `remapp.models.`**`Accumulated_xray_dose`**(**args*, ***kwargs*)

Accumulated X-Ray Dose TID 10002

From DICOM Part 16: This general template provides detailed information on projection X-Ray dose value accumulations over several irradiation events from the same equipment (typically a study or a performed procedure step).

class `remapp.models.`**`Calibration`**(**args*, ***kwargs*)

Table to hold the calibration information

- Container in TID 10002 Accumulated X-ray dose

class `remapp.models.`**`Irradiation_event_xray_data`**(**args*, ***kwargs*)

Irradiation Event X-Ray Data TID 10003

From DICOM part 16: This template conveys the dose and equipment parameters of a single irradiation event.

class `remapp.models.`**`Image_view_modifier`**(**args*, ***kwargs*)

Table to hold image view modifiers for the irradiation event xray data table

From DICOM Part 16 Annex D DICOM controlled Terminology Definitions

- Code Value 111032
- Code Meaning Image View Modifier
- Code Definition Modifier for image view

class `remapp.models.`**`Irradiation_event_xray_detector_data`**(**args*, ***kwargs*)

Irradiation Event X-Ray Detector Data TID 10003a

From DICOM Part 16 Correction Proposal CP-1077: This template contains data which is expected to be available to the X-ray detector or plate reader component of the equipment.

class remapp.models.**Irradiation_event_xray_source_data**(**args*, ***kwargs*)
Irradiation Event X-Ray Source Data TID 10003b

From DICOM Part 16 Correction Proposal CP-1077: This template contains data which is expected to be available to the X-ray source component of the equipment.

Additional to the template:

- ii_field_size
- exposure_control_mode

class remapp.models.**Xray_grid**(**args*, ***kwargs*)
Content ID 10017 X-Ray Grid

From DICOM Part 16

class remapp.models.**Pulse_width**(**args*, ***kwargs*)
In TID 10003b. Code value 113793 (ms)

class remapp.models.**Kvp**(**args*, ***kwargs*)
In TID 10003b. Code value 113733 (kV)

class remapp.models.**Xray_tube_current**(**args*, ***kwargs*)
In TID 10003b. Code value 113734 (mA)

class remapp.models.**Exposure**(**args*, ***kwargs*)
In TID 10003b. Code value 113736 (uAs)

class remapp.models.**Xray_filters**(**args*, ***kwargs*)
Container in TID 10003b. Code value 113771

class remapp.models.**Irradiation_event_xray_mechanical_data**(**args*, ***kwargs*)
Irradiation Event X-Ray Mechanical Data TID 10003c

From DICOM Part 16 Correction Proposal CP-1077: This template contains data which is expected to be available to the gantry or mechanical component of the equipment.

Additional to the template:

- compression_force
- magnification_factor

class remapp.models.**Dose_related_distance_measurements**(**args*, ***kwargs*)
Dose Related Distance Measurements Context ID 10008

Called from TID 10003c

class remapp.models.**Accumulated_projection_xray_dose**(**args*, ***kwargs*)
Accumulated Projection X-Ray Dose TID 10004

From DICOM Part 16: This general template provides detailed information on projection X-Ray dose value accumulations over several irradiation events from the same equipment (typically a study or a performed procedure step).

convert_gym2_to_cgycm2()
Converts Gy.m2 to cGy.cm2 for display in web interface

class remapp.models.**Accumulated_mammography_xray_dose**(**args*, ***kwargs*)
Accumulated Mammography X-Ray Dose TID 10005

From DICOM Part 16: This modality specific template provides detailed information on mammography X-Ray dose value accumulations over several irradiation events from the same equipment (typically a study or a performed procedure step).

class remapp.models.**Accumulated_cassette_based_projection_radiography_dose**(**args*, ***kwargs*)

Accumulated Cassette-based Projection Radiography Dose TID 10006

From DICOM Part 16 Correction Proposal CP-1077: This template provides information on Projection Radiography dose values accumulated on Cassette- based systems over one or more irradiation events (typically a study or a performed procedure step) from the same equipment.

class remapp.models.**Accumulated_integrated_projection_radiography_dose**(**args*, ***kwargs*)

Accumulated Integrated Projection Radiography Dose TID 10007

From DICOM Part 16 Correction Proposal CP-1077: This template provides information on Projection Radiography dose values accumulated on Integrated systems over one or more irradiation events (typically a study or a performed procedure step) from the same equipment.

class remapp.models.**Patient_module_attributes**(**args*, ***kwargs*)

Patient Module C.7.1.1

From DICOM Part 3: Information Object Definitions Table C.7-1: Specifies the Attributes of the Patient that describe and identify the Patient who is the subject of a diagnostic Study. This Module contains Attributes of the patient that are needed for diagnostic interpretation of the Image and are common for all studies performed on the patient. It contains Attributes that are also included in the Patient Modules in Section C.2.

class remapp.models.**Patient_study_module_attributes**(**args*, ***kwargs*)

Patient Study Module C.7.2.2

From DICOM Part 3: Information Object Definitions Table C.7-4a: Defines Attributes that provide information about the Patient at the time the Study started.

class remapp.models.**General_equipment_module_attributes**(**args*, ***kwargs*)

General Equipment Module C.7.5.1

From DICOM Part 3: Information Object Definitions Table C.7-8: Specifies the Attributes that identify and describe the piece of equipment that produced a Series of Composite Instances.

class remapp.models.**Ct_radiation_dose**(**args*, ***kwargs*)

CT Radiation Dose TID 10011

From DICOM Part 16: This template defines a container (the root) with subsidiary content items, each of which corresponds to a single CT X-Ray irradiation event entry. There is a defined recording observer (the system or person responsible for recording the log, generally the system). Accumulated values shall be kept for a whole Study or at least a part of a Study, if the Study is divided in the workflow of the examination, or a performed procedure step. Multiple CT Radiation Dose objects may be created for one Study.

class remapp.models.**Ct_accumulated_dose_data**(**args*, ***kwargs*)

CT Accumulated Dose Data

From DICOM Part 16: This general template provides detailed information on CT X-Ray dose value accumulations over several irradiation events from the same equipment and over the scope of accumulation specified for the report (typically a Study or a Performed Procedure Step).

class remapp.models.**Ct_irradiation_event_data**(**args*, ***kwargs*)

CT Irradiation Event Data TID 10013

From DICOM Part 16: This template conveys the dose and equipment parameters of a single irradiation event.

Additional to the template:

- date_time_started
- series_description

class `remapp.models.`**`Ct_xray_source_parameters`**(**args*, ***kwargs*)
Container in TID 10013 to hold CT x-ray source parameters

class `remapp.models.`**`Scanning_length`**(**args*, ***kwargs*)
Scanning Length TID 10014

From DICOM Part 16: No description

class `remapp.models.`**`Ct_dose_check_details`**(**args*, ***kwargs*)
CT Dose Check Details TID 10015

From DICOM Part 16: This template records details related to the use of the NEMA Dose Check Standard (NEMA XR-25-2010).

class `remapp.models.`**`Observer_context`**(**args*, ***kwargs*)
Observer Context TID 1002

From DICOM Part 16: The observer (person or device) that created the Content Items to which this context applies.

class `remapp.models.`**`Device_participant`**(**args*, ***kwargs*)
Device Participant TID 1021

From DICOM Part 16: This template describes a device participating in an activity as other than an observer or subject. E.g. for a dose report documenting an irradiating procedure, participants include the irradiating device.

class `remapp.models.`**`Person_participant`**(**args*, ***kwargs*)
Person Participant TID 1020

From DICOM Part 16: This template describes a person participating in an activity as other than an observer or subject. E.g. for a dose report documenting an irradiating procedure, participants include the person administering the irradiation and the person authorizing the irradiation.

5.6 Filtering code

class `remapp.interface.mod_filters.`**`RFSummaryListFilter`**(*data=None*, *queryset=None*, *prefix=None*, *strict=None*)
Filter for fluoroscopy studies to display in web interface.

class `remapp.interface.mod_filters.`**`CTSummaryListFilter`**(*data=None*, *queryset=None*, *prefix=None*, *strict=None*)
Filter for CT studies to display in web interface.

class `remapp.interface.mod_filters.`**`MGSummaryListFilter`**(*data=None*, *queryset=None*, *prefix=None*, *strict=None*)
Filter for mammography studies to display in web interface.

5.7 Views

`remapp.views.`**`logout_page`**(*request*)
Log users out and re-direct them to the main page.

remapp.views.**rf_summary_list_filter**(*request*, **args*, ***kwargs*)

remapp.views.**ct_summary_list_filter**(*request*, **args*, ***kwargs*)

remapp.views.**mg_summary_list_filter**(*request*, **args*, ***kwargs*)

remapp.views.**openrem_home**(*request*)

remapp.views.**study_delete**(*request*, **args*, ***kwargs*)

remapp.views.**size_upload**(*request*, **args*, ***kwargs*)
Form for upload of csv file containing patient size information. POST request passes database entry ID to size_process

Parameters **request** – If POST, contains the file upload information

remapp.views.**size_process**(*request*, **args*, ***kwargs*)
Form for csv column header patient size imports through the web interface. POST request launches import task

Parameters

- **request** – If POST, contains the field header information
- **pk** (*kwarg*) – From URL, identifies database patient size import record

remapp.views.**size_imports**(*request*, **args*, ***kwargs*)
Lists patient size imports in the web interface

Parameters **request** –

remapp.views.**size_delete**(**args*, ***kwargs*)
Task to delete records of patient size imports through the web interface

Parameters **request** (*POST*) – Contains the task ID

remapp.views.**size_abort**(*request*, **args*, ***kwargs*)
View to abort current patient size imports

Parameters **request** (*POST*) – Contains the task primary key

5.8 Export Views

remapp.exports.exportviews.**ctcsv1**(**args*, ***kwargs*)
View to launch celery task to export CT studies to csv file

Parameters **request** (*GET*) – Contains the database filtering parameters. Also used to get user group.

remapp.exports.exportviews.**ctxlsx1**(**args*, ***kwargs*)
View to launch celery task to export CT studies to xlsx file

Parameters **request** (*GET*) – Contains the database filtering parameters. Also used to get user group.

remapp.exports.exportviews.**flcsv1**(**args*, ***kwargs*)
View to launch celery task to export fluoroscopy studies to csv file

Parameters **request** (*GET*) – Contains the database filtering parameters. Also used to get user group.

remapp.exports.exportviews.**mgcsv1**(**args*, ***kwargs*)
View to launch celery task to export mammography studies to csv file

Parameters **request** (*GET*) – Contains the database filtering parameters. Also used to get user group.

`remapp.exports.exportviews.`**`mgnhsbsp`**(**args*, ***kwargs*)
View to launch celery task to export mammography studies to csv file using a NHSBSP template

Parameters **request** (*GET*) – Contains the database filtering parameters. Also used to get user group.

`remapp.exports.exportviews.`**`export`**(**args*, ***kwargs*)
View to list current and completed exports to track progress, download and delete

Parameters **request** – Used to get user group.

`remapp.exports.exportviews.`**`download`**(*request*, **args*, ***kwargs*)
View to handle downloads of files from the server

Originally used for download of the export spreadsheets, now also used for downloading the patient size import logfiles.

Parameters

- **request** – Used to get user group.
- **file_name** – Passes name of file to be downloaded.

`remapp.exports.exportviews.`**`deletefile`**(**args*, ***kwargs*)
View to delete export files from the server

Parameters **request** (*POST*) – Contains the task ID

`remapp.exports.exportviews.`**`export_abort`**(*request*, **args*, ***kwargs*)
View to abort current export job

Parameters **request** (*POST*) – Contains the task primary key

5.9 Forms

class `remapp.forms.`**`SizeUploadForm`**(*data=None*, *files=None*, *auto_id=u'id_%s'*, *prefix=None*, *initial=None*, *error_class=<class 'django.forms.util.ErrorList'>*, *label_suffix=None*, *empty_permitted=False*)
Form for patient size csv file upload

class `remapp.forms.`**`SizeHeadersForm`**(*my_choice=None*, ***kwargs*)
Form for csv column header patient size imports through the web interface

5.10 Indices and tables

- *genindex*
- *modindex*
- *search*

Note: OpenREM does not currently include a DICOM Store SCP, ie you will need to install a DICOM store server in order to send RSDRs or other DICOM files from modalities.

If you have no preference, Conquest is recommended as a free, open source, scriptable DICOM server. Please note though that you will need to include the RSRD SOP in the dgatesop.lst file.

CHAPTER 6

Indices and tables

- *genindex*
- *modindex*
- *search*