OpenREM Documentation

Release 0.6.2

Ed McDonagh

January 27, 2016

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.

New to this version, NumPy needs to be installed if you want charts, and a specific version of PyNetDICOM needs to be installed for the DICOM C-Store function.

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

- 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*

1.1.4 Install NumPy

(New for version 0.6.0)

Numpy is required for charts. OpenREM will work without NumPy, but charts will not be displayed.

For linux:

```
sudo apt-get install python-numpy
# If using a virtualenv, you might need to also do:
pip install numpy
```

For Windows, there are various options:

1. Download executable install file from SourceForge:
   - Download a pre-compiled Win32 .exe NumPy file from http://sourceforge.net/projects/numpy/files/NumPy/. You need to download the file that matches the Python version, which should be 2.7. At the time of writing the latest version was 1.9.2, and the filename to download was `numpy-1.9.2-win32-superpack-python2.7.exe`. The filename is truncated on SourceForge, so you may need to click on the *i* icon to see which is which. It's usually the third *superpack*.
   - Run the downloaded binary file to install NumPy.
2. Or download a `pip` installable wheel file:
   - Download NumPy from http://www.lfd.uci.edu/~gohlke/pythonlibs/#numpy - `numpy1.9.2+mklcp27nonewin32.whl` is likely to be the right version, unless you have 64bit Python installed, in which case use the `numpy1.9.2+mklcp27nonewin_amd64.whl` version instead.
   - Install using pip:

```
        pip install numpy1.9.2+mklcp27nonewin32.whl
```

1.1.5 Install pynetdicom

(New for version 0.6.0)

Pynetdicom is used for the new DICOM store SCP function that is available as a preview in this release. See DICOM Networking in OpenREM for details.

```
pip install https://bitbucket.org/edmcdonagh/pynetdicom/get/default.tar.gz#egg=pynetdicom-0.8.2b2
```

1.2 Install and configure OpenREM

```
pip install openrem
```

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

1.2.1 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 two files that need renaming:

- `openremproject/local_settings.py.example` to `openremproject/local_settings.py`
- `openremproject/wsgi.py.example` to `openremproject/wsgi.py`

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). Alternatively, a single `'*'` allows any host, but removes the security the feature gives you.

DICOM networking

See DICOM Networking in OpenREM

1.2.2 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.

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

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 Preview feature: Start the DICOM Store SCP

See DICOM Networking in OpenREM

1.3.4 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/openremproject/local_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\openremproject\local_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

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_pr
```

1.5.3 Backing up a PostgreSQL database

Note: Content contributed by DJ Platten

These instructions are based on PostgreSQL 9.1 and OpenREM 0.5.0 running on Windows Server 2008. The database restore has been tested on Ubuntu 12.04 LTS.

As a one-off, create a PostgreSQL user called `backup` with a password of `backup`. This is easiest to do using the `pgAdminIII` tool: you'll need to create a new login role. In the role privileges ensure that `Can initiate streaming replication and backups` is checked.

The `pgAdminIII` tool is available by default on Windows, but needs to be explicitly installed if using Ubuntu with the following command:

```
sudo apt-get install pgadmin3
```

For the remainder of this article I'm going to assume that your OpenREM database is called `openrempostgresql`.

To backup the contents of `openrempostgresql` to a file called `backup.sql` run the following at the command line in a command prompt (Windows), or terminal window (Ubuntu):

```
pg_dump -i -U backup -F c -b -v -f backup.sql openrempostgresql
```

Note that the `pg_dump` command needs to be in your path for this to work exactly as written. The `-U backup` indicates that the `backup` user is to carry out the task. The `-F c` option archives in a suitable format for input into the `pg_restore` command. Further information on `pg_dump` and backing up a PostgreSQL database can be found here: http://www.postgresql.org/docs/9.3/static/app-pgdump.html and here: http://www.postgresql.org/docs/9.3/static/backup-dump.html

1.5.4 Restoring a PostgreSQL database

The `pg_restore` command can be used to restore the database using one of the backed-up SQL files that were produced using the `pg_dump` command.

Use the `pgAdminIII` tool to ensure that there is a PostgreSQL user called `openremuser`.

Use `pgAdminIII` to create a database called `openrempostgresql`; set the owner to `openremuser` and the encoding to `UTF8`.

Run the following command in a command prompt window (Windows) or terminal window (Ubuntu) to restore the contents of `backupFile` to the `openrempostgresql` database, where `backupFile` is the file created by the `pg_dump` command:

```
pg_restore -U postgres -d openrempostgresql backupFile
```

Ensure that `openremuser` has an entry in PostgreSQL's `pg_hpa.conf` file for md5 authentication:

```
local all openremuser md5
```

The PostgreSQL server will need to be restarted if you have changed `pg_hpa.conf`.

See http://www.postgresql.org/docs/9.3/static/backup-dump.html#BACKUP-DUMP-RESTORE for further details.

1.5.5 Configuring Conquest DICOM server to automatically forward data to OpenREM

The Conquest DICOM server can be configured to automatically run tasks when it receives specific types of DICOM object. For example, a script can be run when a DX image is received that will extract dose information into OpenREM; Conquest will then delete the original image.

These actions are set up in the `dicom.ini` file, located in the root of the Conquest installation folder.

For example:

```
ImportModality1   = MG
ImportConverter1  = save to C:\conquest\dosedata\mammo\%o.dcm; system C:\conquest\openrem-mam-la
```

`ImportModality1 = MG` tells Conquest that modality 1 is MG. The commands listed in the `ImportConverter1` line are then run on all incoming MG images.

The `ImportConverter` instructions are separated by semicolons; the above example has three commands:

- `save to C:\conquest\dosedata\mammo\%o.dcm` saves the incoming MG image to the specified folder with a file name set to the SOP instance UID contained in the image
- `system C:\conquest\openrem-mam-launch.bat C:\conquest\dosedata\mammo\%o.dcm` runs a DOS batch file, using the newly saved file as the argument. On my system this batch file runs the OpenREM `openrem_mg.py` import script
- `destroy` tells Conquest to delete the image that it has just received.

My system has three further import sections for DX, CR, and structured dose report DICOM objects:

```
# Import of DX images
ImportModality2   = DX
ImportConverter2  = save to C:\conquest\dosedata\dx\%o.dcm; system C:\conquest\openrem-dx-launch

# Import of CR images
```

```
    ImportModality3    = CR
    ImportConverter3   = save to C:\conquest\dosedata\dx\%o.dcm; system C:\conquest\openrem-dx-launch

    # Import of structured dose reports (this checks the DICOM tag 0008,0016 to see if it matches th
    ImportConverter4   = ifequal "%V0008,0016","1.2.840.10008.5.1.4.1.1.88.67"; {save to C:\conquest\
```

1.5.6 Configuring Conquest DICOM server to accept x-ray radiation dose structured reports

The Conquest DICOM server only accepts incoming DICOM objects if they have a service-object pair (SOP) unique identifier that appears in the `dgatesop.lst` file, located in the root of the Conquest installation folder. This file is created by the Conquest DICOM server automatically when it is first run.

By default this file does not contain the SOP information for x-ray radiation dose structured reports (RDSRs). It is easy to add this: open the `dgatesop.lst` file in a text editor and add the following line:

```
    XRayRadiationDoseSR 1.2.840.10008.5.1.4.1.1.88.67    sop
```

After a restart, Conquest will now accept incoming RDSR objects.

1.5.7 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/openremproject/wsgi.py"
        WSGIPythonPath "c:/Python27/Lib/site-packages/openrem"

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

```
    </Files>
    </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/openremproject 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", "openremproject.settings")

# Apply WSGI middleware here.

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

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

- Create a file monitor.py in the openrem/openremproject 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

Release Notes v0.6.0

2.1 Headline changes

- Charts
- Preview of DICOM Store SCP functionality
- Exports available to import into OpenSkin
- Modalities with no data are hidden in the user interface
- Mammography import compression force behaviour changed
- Import of Toshiba planar RDSRs fixed

2.1.1 Changes for 0.6.2

Minor update due prevent new installs from installing a non-compatible version of `django-filter`. The link to OpenSkin has also been updated in the fluoroscopy detail page.

There is no advantage to updating to this version over 0.6.0

Release 0.6.1 was just a documentation only change to update the link to OpenSkin.

2.2 Preparing for the upgrade

2.2.1 Convert to South

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: Debian/Ubuntu and derivatives
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py convert_to_south remapp
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
python /usr/local/lib/python2.7/site-packages/openrem/manage.py convert_to_south remapp

# Windows, assuming no virtualenv
python C:\Python27\Lib\site-packages\openrem\manage.py convert_to_south remapp
```

2.2.2 Additional installs

OpenREM requires two additional programs to be installed to enable the new features: *Numpy* for charts, and *pynetdicom* for the DICOM Store Service Class Provider. Note that the version of pynetdicom must be later than the current pypi release!

Install NumPy

For linux:

```
sudo apt-get install python-numpy
# If using a virtualenv, you might need to also do:
pip install numpy
```

For Windows, there are various options:

1. Download executable install file from SourceForge:
    - Download a pre-compiled Win32 .exe NumPy file from http://sourceforge.net/projects/numpy/files/NumPy/. You need to download the file that matches the Python version, which should be 2.7. At the time of writing the latest version was 1.9.2, and the filename to download was `numpy-1.9.2-win32-superpack-python2.7.exe`. The filename is truncated on SourceForge, so you may need to click on the *i* icon to see which is which. It's usually the third *superpack*.
    - Run the downloaded binary file to install NumPy.
2. Or download a `pip` installable wheel file:
    - Download NumPy from http://www.lfd.uci.edu/~gohlke/pythonlibs/#numpy - `numpy1.9.2+mklcp27nonewin32.whl` is likely to be the right version, unless you have 64bit Python installed, in which case use the `numpy1.9.2+mklcp27nonewin_amd64.whl` version instead.
    - Install using pip:

```
pip install numpy1.9.2+mklcp27nonewin32.whl
```

Install pynetdicom

```
pip install https://bitbucket.org/edmcdonagh/pynetdicom/get/default.tar.gz#egg=pynetdicom-0.8.2b2
```

2.2.3 Upgrading from versions prior to 0.5.1

You must upgrade to 0.5.1 first. Instructions for doing this can be found in the OpenREM Release Notes version 0.5.1.

2.2.4 Upgrading from version 0.5.1

- Back up your database
    - For PostgreSQL you can refer to Backing up a PostgreSQL database
    - For a non-production SQLite3 database, simply make a copy of the database file
- The 0.6.0 upgrade must be made from a 0.5.1 (or later) database, and a schema migration is required:

```
pip install openrem==0.6.0

# Linux: Debian/Ubuntu and derivatives
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
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
python /usr/local/lib/python2.7/site-packages/openrem/manage.py schemamigration --auto remapp
python /usr/local/lib/python2.7/site-packages/openrem/manage.py migrate remapp
# Windows:
python C:\Python27\Lib\site-packages\openrem\manage.py schemamigration --auto remapp
python C:\Python27\Lib\site-packages\openrem\manage.py migrate remapp
```

- Restart the services
    - Restart the webserver
    - Restart Celery

2.3 Summary of new features

2.3.1 Charts

Release 0.6.0 has a range of charting options available for CT and radiographic data. These charts allow visualisation of trends and frequencies to inform surveys and monitor performance. For more information, please see Charts.

2.3.2 DICOM Store Service Class Provider

OpenREM can now act as the DICOM Store service, allowing direct sending of DICOM objects from modalities to OpenREM without needing to use Conquest or any other DICOM Store SCP. This feature is a preview as it hasn't been extensively tested, but it is expected to work. For more information, please see DICOM Networking in OpenREM.

2.3.3 Exports for OpenSkin

Fluoroscopy studies can now be exported in a format suitable for importing into Jonathan Cole's OpenSkin software. The export link is on the fluoroscopy study detail page. The software for creating the exposure incidence map can be downloaded from https://bitbucket.org/jacole/openskin/downloads (choose the zip file), and information about the project can be found on the OpenSkin wiki. The software allows the user to choose between a 2D phantom that would represent the dose to a film laying on the couch surface, or a simple 3D phantom made up of a cuboid and two semi-cylinders (these can be seen on the Phantom design section of the wiki). For both options the output is an image of the dose distribution in 2D, along with calculated peak skin dose information.

2.3.4 Automatic hiding of unused modality types

A fresh install of OpenREM will no longer show any of the four modality types in the tables or in the navigation bar at the top. As DICOM objects are ingested, the appropriate tables and navigation links are created.

Therefore a site that has no mammography for example will no longer have that table or navigation link in their interface.

2.3.5 Mammography import compression force change

Prior to version 0.6, the compression force extracted from the mammography image header was divided by ten before being stored in the database. This was because the primary author only had access to GE Senograph DS units, which store the compression force in dN, despite claiming using Newtons in the DICOM conformance statement.

The code now checks for the term *senograph ds* contained in the model name. If it matches, then the value is divided by ten. Otherwise, the value is stored without any further change. We know that later GE units, the GE Senograph Essential for example, and other manufacturer's units store this value in N. If you have a case that acts like the Senograph DS, please let us know and we'll try and cater for that.

If you have existing non-GE Senograph mammography data in your database, the compression force field for those studies is likely to be incorrect by a factor of ten (it will be too small). Studies imported after the upgrade will be correct. If this is a problem for you, please let us know and we'll see about writing a script to correct the existing data.

2.3.6 Import of Toshiba Planar RDSRs fixed

Toshiba include Patient Orientation and Patient Orientation Modifier information in their cath lab RDSRs. The extractor code was deficient for this as the RDSRs previously used didn't have this information. This has now been fixed. There might however be an issue with Station Name not being provided - it is not yet clear if this is a configuration issue.

CHAPTER 3

Previous Release Notes and Change Log

3.1 Version history change log

3.1.1 OpenREM version history

0.6.2 (2016-01-27)

- **‘#347‘_** Django-filter v0.12 has minimum Django version of 1.8, fixed OpenREM 0.6.2 to max django-filter 0.11
- **‘#341‘_** Changed references to the OpenSkin repository for 0.6 series.

0.6.1 (2015-10-30

- #303 Corrected name of Store SCP command in docs

0.6.0 (2015-05-14)

- #227 Fixed import of RDSRs from Toshiba Cath Labs
- #226 Charts: Updated Highcharts code and partially fixed issues with CTDIvol and DLP combined chart
- #225 Charts: Added link from mAs and kVp histograms to associated data
- #224 Charts: Added link from CTDIvol histograms to associated data
- #221 Charts: Fixed issue where filters at acquisition event level were not adequately restricting the chart data
- #219 Charts: Fixed issue where some charts showed data beyond the current filter
- #217 Charts: Code optimised to speed up calculation time
- #216 Fixed typo that prevented import of RSDR when DICOM store settings not present
- #215 Charts: Fixed x-axis labels for mean dose over time charts
- #214 Charts: Improved consistency of axis labels
- #213 Fixed admin menu not working
- #212 Charts: Created off-switch for charts

- #210 OpenSkin exports documented
- #209 Charts: Fixed server error when CT plots switched off and filter form submited
- #208 Charts: Fixed blank chart plotting options when clicking on histogram tooltip link
- #205 Charts: Fixed issue of histogram tooltip links to data not working
- #204 Charts: Fixed issue of not being able to export with the charts features added
- #203 Charts: Fixed display of HTML in plots issue
- #202 Charts: Added mean CTDIvol to charts
- #200 Charts: Now exclude Philips Ingenuity SPRs from plots
- #196 Added comments and entrance exposure data to DX export
- #195 Fixed error with no users on fresh install
- #194 Added more robust extraction of series description from DX
- #193 Charts: Fixed reset of filters when moving between pages
- #192 Created RF export for OpenSkin
- #191 Charts: Factored out the javascript from the filtered.html files
- #190 Charts: Added time period configuration to dose over time plots
- #189 Charts: Fixed plotting of mean doses over time when frequency not plotted
- #187 Charts: Merged the charts work into the main develop branch
- #186 Fixed duplicate data in DX exports
- #179 Charts: Added kVp and mAs plots for DX
- #177 Charts: Fixed issue with date ranges for DX mean dose over time charts
- #176 Charts: Added link to filtered dataset from mean dose over time charts
- #175 Charts: Allowed configuration of the time period for mean dose trend charts to improve performance
- #174 Charts: Fixed number of decimal places for mean DLP values
- #173 Charts: Fixed plot of mean DLP over time y-axis issue
- #170 Charts: Added plot of mean dose over time
- #169 Charts: Improved chart colours
- #157 Charts: Added chart showing number of studies per day of the week, then hour in the day
- #156 Charts: Fixed issue with some protocols not being displayed
- #155 Charts: Added chart showing relative frequency of protocols and study types
- #140 Charts: Added configuration options
- #139 Charts: Link to filtered dataset from histogram chart
- #138 Charts: Number of datapoints displayed on tooltip
- #135 Mammography compression force now only divides by 10 if model contains *senograph ds* **Change in behaviour**
- #133 Documented installation of NumPy, initially for charts
- #41 Preview of DICOM Store SCP now available

- #20 Modality sections are now suppressed until populated

0.5.1 (2015-03-12)

- #184 Documentation for 0.5.1
- #180 Rename all reverse lookups as a result of #62
- #178 Added documentation regarding backing up and restoring PostgreSQL OpenREM databases
- #172 Revert all changes made to database so #62 could take place first
- #165 Extract height and weight from DX, height from RDSR, all if available
- #161 Views and exports now look for accumulated data in the right table after changes in #159 and #160
- #160 Created the data migration to move all the DX accumulated data from TID 10004 to TID 10007
- #159 Modified the DX import to populate TID 10007 rather than TID 10004. RDSR RF already populates both
- #158 Demo website created by DJ Platten: http://demo.openrem.org/openrem
- #154 Various decimal fields are defined with too few decimal places - all have now been extended.
- #153 Changed home page and modality pages to have whole row clickable and highlighted
- #150 DJ Platten has added Conquest configuration information
- #137 Carestream DX multiple filter thickness values in a DS VR now extracted correctly
- #113 Fixed and improved recording of grid information for mammo and DX and RDSR import routines
- #62 Refactored all model names to be less than 39 characters and be in CamelCase to allow database migrations and to come into line with PEP 8 naming conventions for classes.

0.5.0 (2014-11-19)

- Pull request from DJ Platten: Improved display of DX data and improved export of DX data
- #132 Fixed mammo export error that slipped in before the first beta
- #130 Only creates ExposureInuAs from Exposure if Exposure exists now
- #128 Updated some non-core documentation that didn't have the new local_settings.py reference or the new openremproject folder name
- #127 DX IOD studies with image view populated failed to export due to lack of conversion to string
- #126 Documentation created for the radiographic functionality
- #125 Fixes issue where Hologic tomo projection objects were dropped as they have the same event time as the 2D element
- #123 Fixed issue where filters came through on export as lists rather than strings on some installs
- #122 Exports of RF data should now be more useful when exporting to xlsx. Will need refinement in the future
- #26 Extractors created for radiographic DICOM images. Contributed by DJ Platten
- #25 Views and templates added for radiographic exposures - either from RDSRs or from images - see #26. Contributed by DJ Platten

- #9 Import of *.dcm should now be available from Windows and Linux alike

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

3.2 Release notes and upgrade instructions

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

3.2.1 Version specific information

OpenREM Release Notes version 0.5.1

Headline changes

- Major database modification to remove table name length errors
- Extended the field value lengths to better incorporate all possible values and decimal places
- Improved import of grid and filter information from DX images
- Improved DX summary and detail web pages
- Any item in a row can now be clicked to move between the home and filtered pages

Upgrades: Convert to South

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: Debian/Ubuntu and derivatives
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py convert_to_south remapp
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
python /usr/local/lib/python2.7/site-packages/openrem/manage.py convert_to_south remapp

# Windows, assuming no virtualenv
python C:\Python27\Lib\site-packages\openrem\manage.py convert_to_south remapp
```

Upgrading from before 0.5.0

Upgrading from version 0.3.9 or earlier

- Back up your database
  - For PostgreSQL you can refer to Backing up a PostgreSQL database
  - For a non-production SQLite3 database, simply make a copy of the database file
- `pip install openrem==0.4.2`
- Migrate the schema

```
    # Linux: Debian/Ubuntu and derivatives
    python /usr/local/lib/python2.7/dist-packages/openrem/manage.py schemamigration --auto r
    # Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
    python /usr/local/lib/python2.7/site-packages/openrem/manage.py schemamigration --auto r
    # Windows:
    python 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, please select option 3, and then execute the migration:

```
# Linux: Debian/Ubuntu and derivatives
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py migrate remapp
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
python /usr/local/lib/python2.7/site-packages/openrem/manage.py migrate remapp

# Windows, assuming no virtualenv
python C:\Python27\Lib\site-packages\openrem\manage.py migrate remapp
```

- Create and populate the database settings in the new `local_settings.py` file

  The `openrem/openrem` folder can be found at:

```
# Linux: Debian/Ubuntu and derivatives
/usr/lib/python2.7/dist-packages/openrem/openrem
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
/usr/lib/python2.7/site-packages/openrem/openrem
# Windows:
C:\Python27\Lib\site-packages\openrem\openrem
```

  In the `openrem/openrem` folder, create a new file called `local_settings.py` and copy the contents of this link into a the file and save it. Alternatively, rename `local_settings.py.example` to `local_settings.py` - this is an older version of the file.

  Copy the database details from `settings.py` into `local_settings.py`

- Change the secret key - you can use http://www.miniwebtool.com/django-secret-key-generator/ to generate a new one
- Move the existing `settings.py` out of the python directories (delete or move somewhere as a backup)
- Rename the `settings.py.new` to `settings.py`
- Restart your webserver to check everything looks ok
- 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 (the `manage.py syncdb` command - *Do you want to create a superuser*)
  - Create some users and add them to the appropriate groups (if there are no groups, go to the OpenREM homepage and they should be there when you go back to admin).
    - `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

Upgrading from versions 0.4.0 - 0.4.2

- Back up your database

  - For PostgreSQL you can refer to Backing up a PostgreSQL database
  - For a non-production SQLite3 database, simply make a copy of the database file
- Install version 0.5.0
  - `pip install openrem==0.5.0`
- Install RabbitMQ
  - 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 `local_settings.py` details from `openrem` to `openremproject`

  The inner `openrem` Django project folder has now been renamed `openremproject` The customised `local_settings.py` file and the `wsgi.py` file have remain in the old `openrem` folder. The `openrem/openrem` folder can be found as detailed in the upgrade from '0.3.9 or earlier' instructions above, and the new `openrem/openremproject` folder is in the same place.

  - Move `local_settings.py` to `openremproject`. If you have kept the older local_settings file, you may like to instead rename the `local_settings.py.example` file instead, then transfer the database settings and change the secret key.
  - Set the path for `MEDIA_ROOT`. The webserver needs to be able to write to this location - it is where OpenREM will store export files etc so that they can be downloaded. For suggestions, see the main _install instructions.
  - Set `ALLOWED_HOSTS`. For details see the Django docs A `'*'` allows any host - see the Django docs for the risk of this.
- Move `wsgi.py` from `openrem` to `openremproject` or rename `wsgi.py.example` in `openremproject`

  If you haven't edited it, simply rename the new version in `openremproject`. Otherwise, move the old version and edit the following line as follows:

```
# Old:
os.environ.setdefault("DJANGO_SETTINGS_MODULE", "openrem.settings")
# New:
os.environ.setdefault("DJANGO_SETTINGS_MODULE", "openremproject.settings")
```

- Tidying up - you should delete the old `openrem` folder - you might like to take a backup first!
- Update web server configuration

  The configuration of the webserver will need to be updated to reflect the new location for the `settings.py` file and the `wsgi.py` file.

  If you are using the built-in test webserver, static files will no-longer be served unless you use the `insecure` option:

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

- Migrate the schema

```
# Linux: Debian/Ubuntu and derivatives
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py schemamigration --auto r
python /usr/local/lib/python2.7/dist-packages/openrem/manage.py migrate remapp
```

```
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
python /usr/local/lib/python2.7/site-packages/openrem/manage.py schemamigration --auto re
python /usr/local/lib/python2.7/site-packages/openrem/manage.py migrate remapp
# Windows:
python C:\Python27\Lib\site-packages\openrem\manage.py schemamigration --auto remapp
python C:\Python27\Lib\site-packages\openrem\manage.py migrate remapp
```

After restarting the webserver, you should now have OpenREM 0.5.0 up and running. If you wish to test export functionality at this stage, start the Celery task queue - instructions in the Installation instructions docs or at the end of this guide.

Now move to *Upgrading from version 0.5.0*.

Upgrading from version 0.4.3

- Back up your database
  - For PostgreSQL you can refer to Backing up a PostgreSQL database
  - For a non-production SQLite3 database, simply make a copy of the database file
- The 0.5.1 upgrade *must* be made from a 0.5.0 database, so a schema migration is required:

```
pip install openrem==0.5.0

    # Linux: Debian/Ubuntu and derivatives
    python /usr/local/lib/python2.7/dist-packages/openrem/manage.py schemamigration --au
    python /usr/local/lib/python2.7/dist-packages/openrem/manage.py migrate remapp
    # Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
    python /usr/local/lib/python2.7/site-packages/openrem/manage.py schemamigration --au
    python /usr/local/lib/python2.7/site-packages/openrem/manage.py migrate remapp
    # Windows:
    python C:\Python27\Lib\site-packages\openrem\manage.py schemamigration --auto remapp
    python C:\Python27\Lib\site-packages\openrem\manage.py migrate remapp
```

Upgrading from version 0.5.0

- Back up your database
  - For PostgreSQL you can refer to Backing up a PostgreSQL database
  - For a non-production SQLite3 database, simply make a copy of the database file
- Install 0.5.1:

```
pip install openrem==0.5.1
```

- Find out how many migration files you have

  Method 1:

  Use a file browser or terminal to list the contents of the `migrations` folder, eg:

```
# Linux: Debian/Ubuntu and derivatives
ls /usr/local/lib/python2.7/dist-packages/openrem/remapp/migrations/
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
ls /usr/local/lib/python2.7/site-packages/openrem/remapp/migrations/
```

```
    # Windows (alternatively use the file browser):
    dir C:\Python27\Lib\site-packages\openrem\remapp\migrations\
```

Method 2:

Use the Django `manage.py` program to list the existing migrations:

```
    # Linux: Debian/Ubuntu and derivatives
    python /usr/local/lib/python2.7/dist-packages/openrem/manage.py migrate --list remapp
    # Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
    python /usr/local/lib/python2.7/site-packages/openrem/manage.py migrate --list remapp
    # Windows
    python C:\Python27\Lib\site-packages\openrem\manage.py migrate --list remapp
```

The output should look something like this - there can be any number of existing migrations (though 0001_initial will always be present):

```
    remapp
    (*) 0001_initial
    (*) 0002_auto__chg_field_ct_accumulated_dose_data_ct_dose_length_product_total_
    (*) 0003_auto__chg_field_general_equipment_module_attributes_station_name
    (*) 0004_auto__chg_field_ct_radiation_dose_comment__chg_field_accumulated_proje
    (*) 0005_auto__add_exports__add_size_upload
    (*) 0006_auto__chg_field_exports_filename
    (*) 0007_auto__add_field_irradiation_event_xray_detector_data_relative_xray_exp
    ( ) 000x_051datamigration
    ( ) 000x_051schemamigration
```

- Rename the two 051 migration files to follow on from the existing migrations, for example `0008_051schemamigration.py` and `0009_051datamigration.py` for the existing migrations above, or `0002_051schemamigration.py` and `0003_051datamigration.py` if the only migration is the initial migration. The `051schemamigration` **must** come before the `051datamigration`.

  If you are using linux, you might like to do it like this (from within the `openrem` folder):

```
mv remapp/migrations/000{x,8}_051schemamigration.py
mv remapp/migrations/000{x,9}_051datamigration.py
```

- If you now re-run `migrate --list remapp` you should get a listing with the `051schemamigration` and the `051datamigration` listed at the end:

```
remapp
 (*) 0001_initial
 (*) 0002_auto__chg_field_ct_accumulated_dose_data_ct_dose_length_product_total_
 (*) 0003_auto__chg_field_general_equipment_module_attributes_station_name
 (*) 0004_auto__chg_field_ct_radiation_dose_comment__chg_field_accumulated_proje
 (*) 0005_auto__add_exports__add_size_upload
 (*) 0006_auto__chg_field_exports_filename
 (*) 0007_auto__add_field_irradiation_event_xray_detector_data_relative_xray_exp
 ( ) 0008_051schemamigration
 ( ) 0009_051datamigration
```

  The star indicates that a migration has already been completed. If you have any that are not completed apart from the `051schemamigration` and the `051datamigration` then please resolve these first.

- Now execute the migrations:

```
    # Linux: Debian/Ubuntu and derivatives
    python /usr/local/lib/python2.7/dist-packages/openrem/manage.py migrate remapp
    # Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
    python /usr/local/lib/python2.7/site-packages/openrem/manage.py migrate remapp
    # Windows
    python C:\Python27\Lib\site-packages\openrem\manage.py migrate remapp
```

Restart the web server If you are using the built-in test web server (*not for production use*):

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

Otherwise restart using the command for your web server

Restart the Celery task queue For testing, in a new shell:

```
# Linux: Debian/Ubuntu and derivatives
cd /usr/local/lib/python2.7/dist-packages/openrem/
# Linux: other distros. In a virtualenv replace all up to lib/ as appropriate
cd /usr/local/lib/python2.7/site-packages/openrem/
# Windows
cd C:\Python27\Lib\site-packages\openrem\

# All
celery -A openremproject worker -l info
```

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

OpenREM Release Notes version 0.5.0

Headline changes

- Import, display and export of CR/DX data from image headers
- Export of study data from fluoroscopy to xlsx files
- Importing data from Windows using *.dcm style wildcards
- Hologic tomography projection images are no longer excluded if part of a Combo exposure

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 versions before 0.4.3 If you are upgrading from 0.3.9 or earlier, you will need to upgrade to version 0.4.2 first. See the OpenREM Release Notes version 0.4.3.

If you are upgrading from 0.4.0 or later, the instructions in OpenREM Release Notes version 0.4.3 still need to be followed to install/setup RabbitMQ and Celery and to update the configuration files, but you can go straight to 0.5.0 rather than installing 0.4.3.

Upgrading from version 0.4.3

```
pip install openrem==0.5.0
```

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

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
```

Restart the web server If you are using the built-in test web server (*not for production use*):

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

Otherwise restart using the command for your web server

Restart the Celery task queue 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 0.4.3

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). Alternatively, a single `'*'` allows any host, but removes the security the feature gives you.

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, yo
          ? 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

3.3 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 4

Importing data to OpenREM

4.1 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:

4.1.1 Radiation Dose Structured Reports

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

You can use wildcards to process a number of files at once, ie:

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

4.1.2 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.

4.1.3 For radiographic DICOM images

```
openrem_dx.py filename.dcm
```

It is now possible with OpenREM version 0.5.0 to import dose information from DX and CR images from digital radiography units. As yet, it hasn't been tested extensively.

4.1.4 For CT dose summary files from Philips CT scanners

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

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

4.2.1 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 5

DICOM Networking in OpenREM

5.1 Functionality available

This is an initial preview release in 0.6.0, with the following features:

- DICOM Store service class provider
- DICOM objects are fed directly into appropriate routine for data extraction
- Extraction jobs are handled by Celery
- Configuration is via the `local_settings.py` file

5.2 Configuration

The following settings need to be in your `local_settings.py` file:

```
STORE_AET = "STOREOPENREM"
STORE_PORT = 8104
RM_DCM_NOMATCH = True
RM_DCM_RDSR = False
RM_DCM_MG = False
RM_DCM_DX = False
RM_DCM_CTPHIL = False
```

5.2.1 STORE_AET

This is the AET you use when configuring send nodes from your modalities.

Set this to your chosen value - any combination of letters and numbers up to 16 characters. No spaces or other characters allowed.

In the current implementation, the actual value is not actually important - the AET is not checked when a DICOM object is received.

5.2.2 STORE_PORT

This is the port you send DICOM objects to. The standard port for DICOM servers is port 104. However, on many operating systems starting a service on a port lower than 1025 requires additional privileges. That is why the suggested port is 8104.

Depending on your network setup you may have to configure the firewall accordingly.

5.2.3 RM_DCM_NOMATCH

When DICOM objects are received they are checked for suitability to have dose related data extracted using any of the current extraction routines.

If you want the DICOM object to be deleted if it can't be used, set `RM_DCM_NOMATCH` to `True`. Otherwise set this to `False`.

Setting this to `True` is advisable as otherwise your disk can fill up very quickly if enture CT studies get sent through for example.

5.2.4 RM_DCM_RDSR, RM_DCM_MG, RM_DCM_DX and RM_DCM_CTPHIL

Set these to `True` to delete the DICOM objects once the dose related data had been extracted. Otherwise set them to `False`, and the DICOM objects will be stored in a folder called `dicom_in` in the `MEDIA_ROOT` folder.

The `RDSR` setting is for Radiation Dose Structured Reports (usually from CT or fluoroscopy), `MG` is for mammography images, `DX` is for radiography images and `CTPHIL` is for Philips CT dose screen images.

Setting these to `True` is advisable, especially for the images as again the disk can fill up quickly.

5.3 How to use the DICOM store service

Open a command window or shell:

```
openrem_store.py
```

You should see the following output, depending on your configuration:

```
starting AE... AET:STOREOPENREM, port:8104 done
```

Make sure that the Celery task manager is running, as all extraction jobs are passed to Celery.

5.4 Planned functionality for future releases

5.4.1 DICOM Store

- Configuration will move to the database with a web interface
- Web interface view of activity and logs

5.4.2 DICOM Query-Retrieve

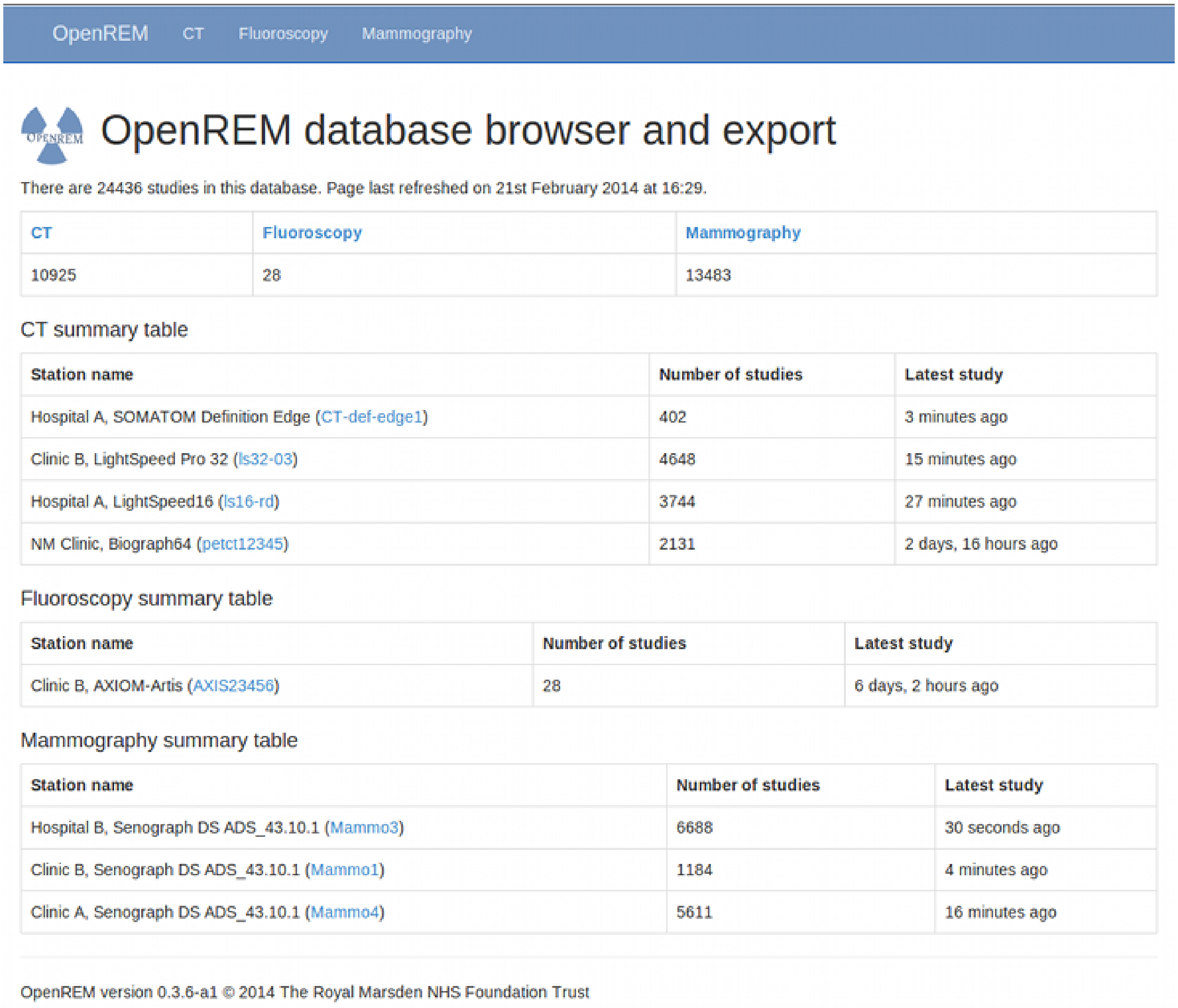
- Function to query retrieve the PACS or modality
- Ad hoc or scheduled
- Web interface for configuration, activating, monitoring success and logs

Navigating, filtering and study details

6.1 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 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

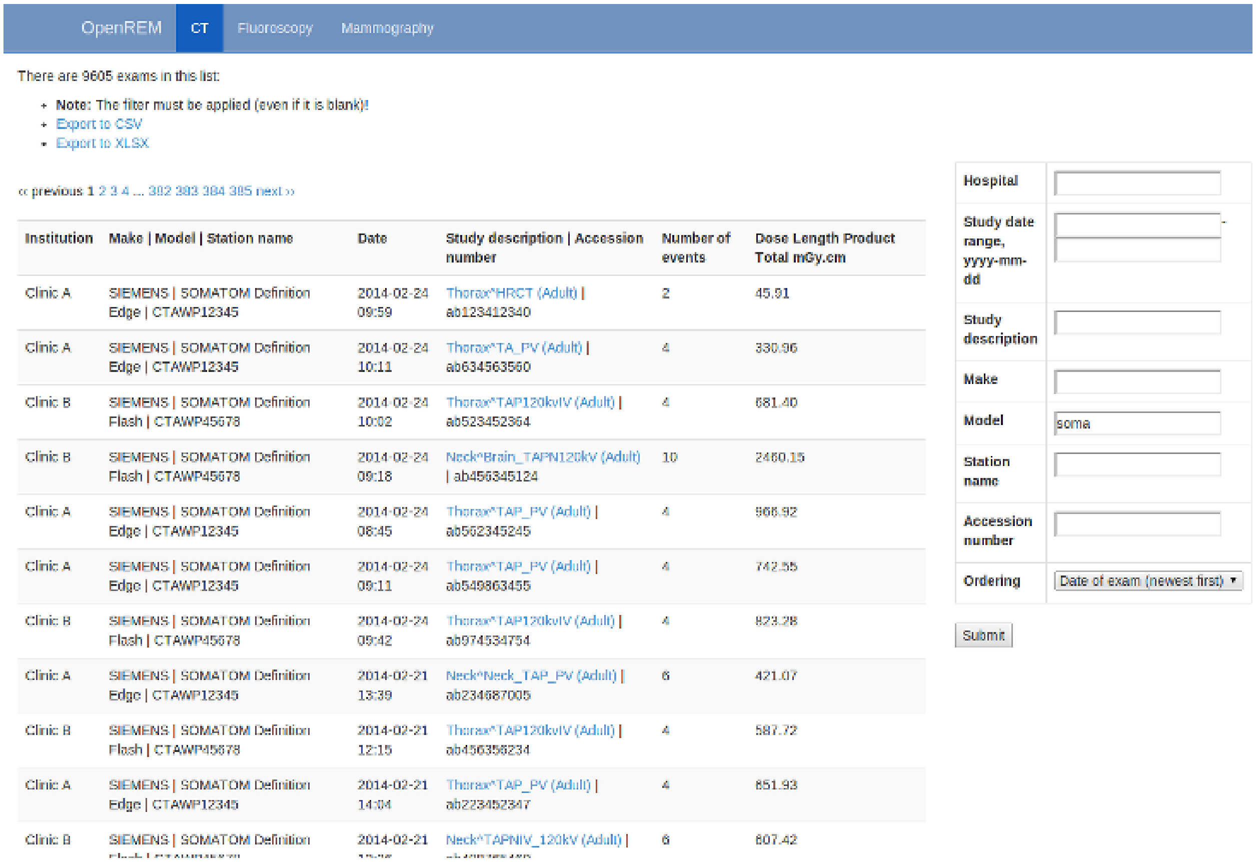
OpenREM version 0.3.6-a1 © 2014 The Royal Marsden NHS Foundation Trust

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.

6.2 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’:



There are 9605 exams in this list:

- **Note:** The filter must be applied (even if it is blank)!
- Export to CSV
- Export to XLSX

« previous 1 2 3 4 ... 382 383 384 385 next »

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	651.93
Clinic B	SIEMENS | SOMATOM Definition	2014-02-21	Neck^TAPNIV_120kV (Adult) |	6	607.42

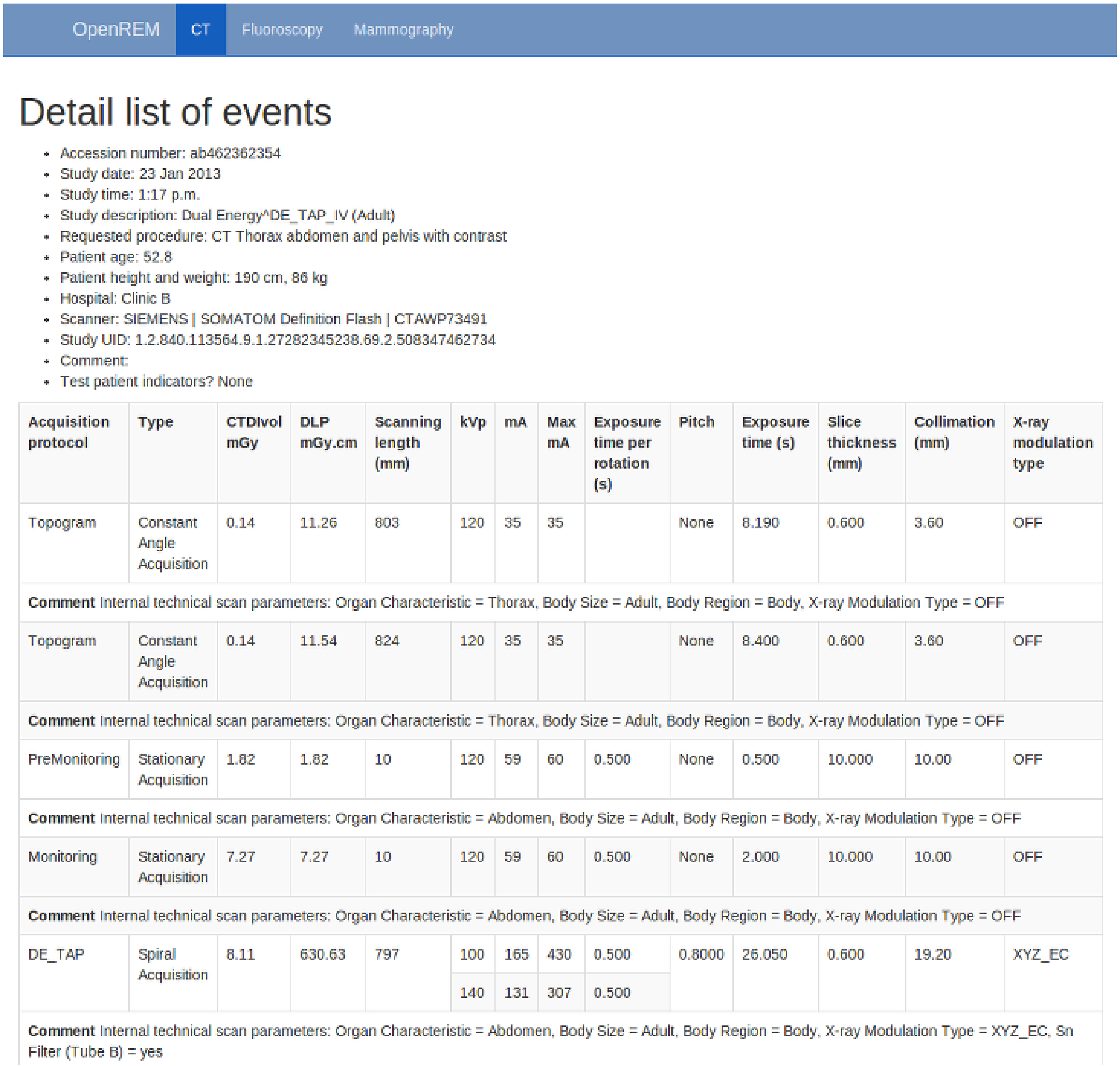
Hospital	
Study date range, yyyy-mm-dd	-
Study description	
Make	
Model	soma
Station name	
Accession number	
Ordering	Date of exam (newest first) ▾

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.

6.3 Viewing study details

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



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.

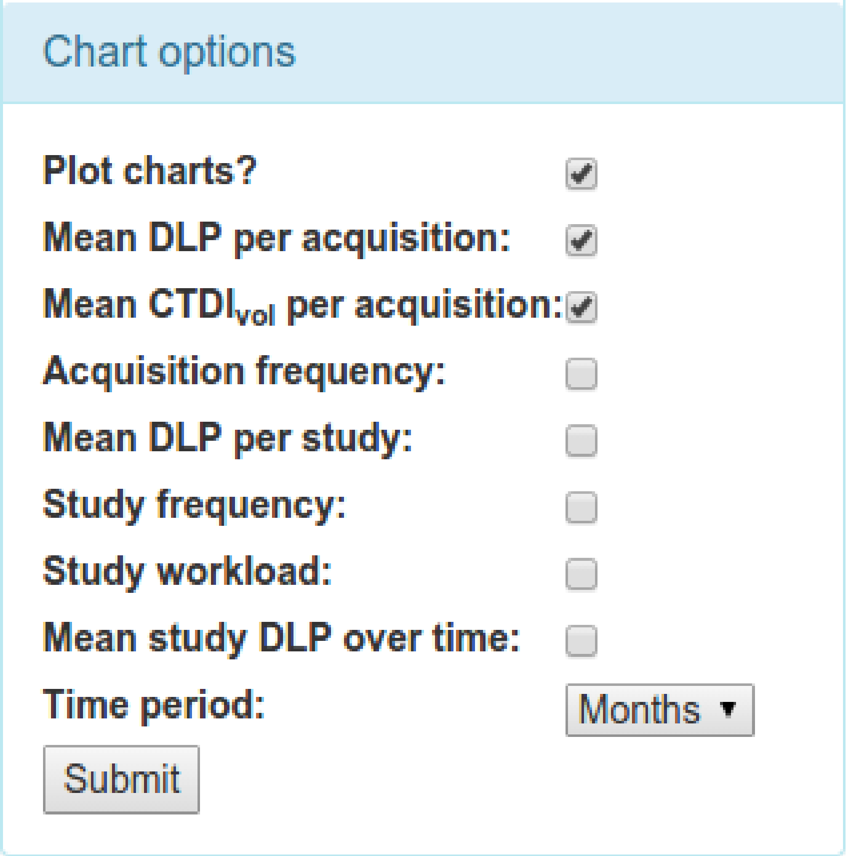
CHAPTER 7

Charts

Charts of the currently filtered data can now be shown for CT and radiographic data. The user can configure which plots are shown using the `Chart options` on the CT and radiographic pages.

The charts are automatically updated to reflect any filters that the user applies to the data.

7.1 Chart options



Plot charts? ☑
Mean DLP per acquisition: ☑
Mean CTDI_{vol} per acquisition: ☑
Acquisition frequency: ☐
Mean DLP per study: ☐
Study frequency: ☐
Study workload: ☐
Mean study DLP over time: ☐
Time period: Months ▾

The first option, `Plot charts?`, determines whether any plots are shown. This also controls whether the data for the plots is calculated by OpenREM. Some plot data is slow to calculate when there is a large amount of data: some users may prefer to leave `Plot charts?` off for performance reasons. `Plot charts?` can be switched on and activated with a click of the `submit` button after the data has been filtered.

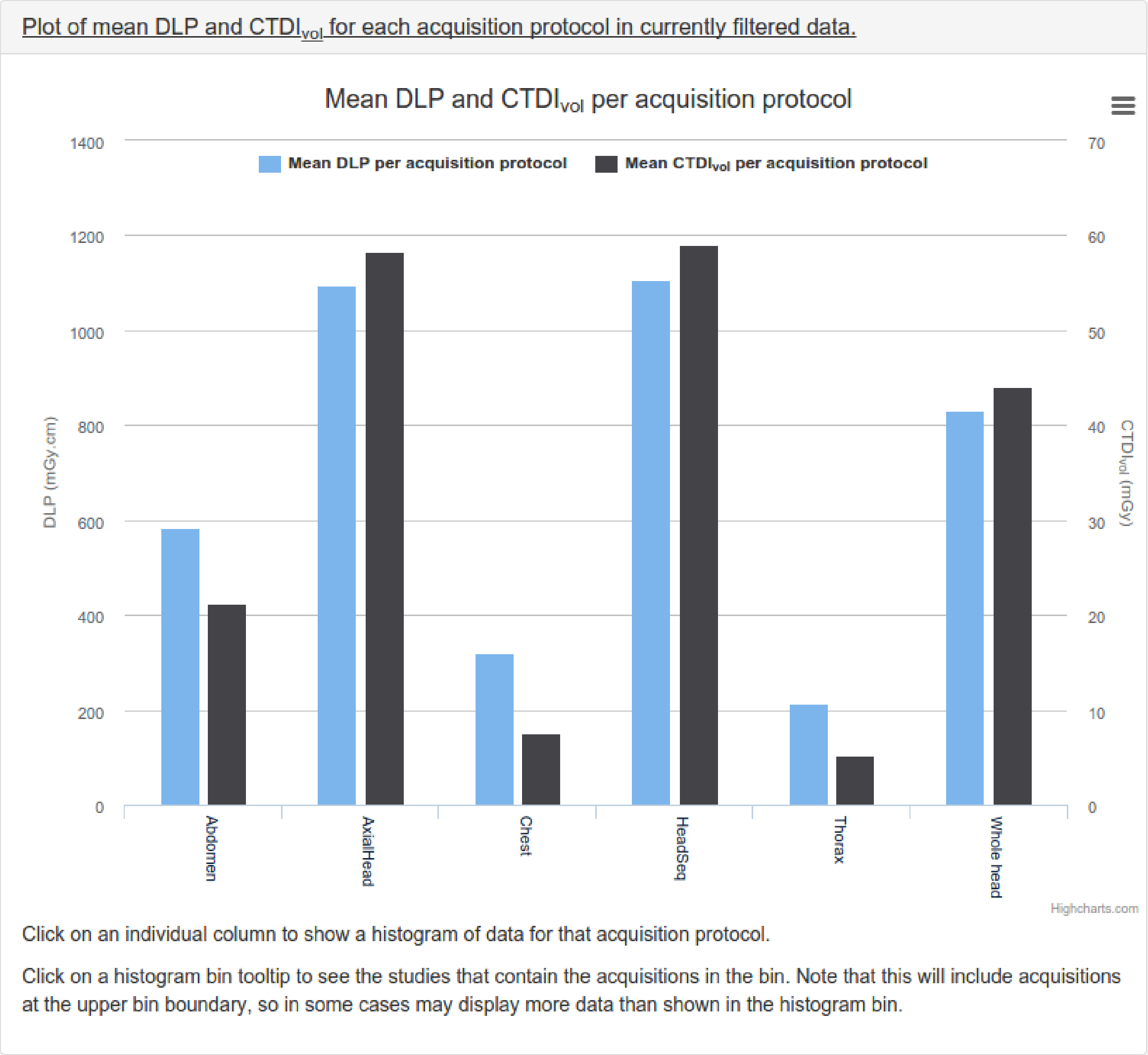
The user can switch off all chart plotting by clicking on the `Switch charts off` link in the `User options` menu in the navigation bar at the top of any OpenREM page. Clicking on this link takes the user back to the home page.

A user's chart options can also be configured by an administrator via OpenREM's user administration page.

7.2 Chart types

The available charts for CT data are as follows:

- Bar chart of mean DLP and CTDI$_{\text{vol}}$ for each acquisition protocol:



Plot of mean DLP and CTDI$_{\text{vol}}$ for each acquisition protocol in currently filtered data.

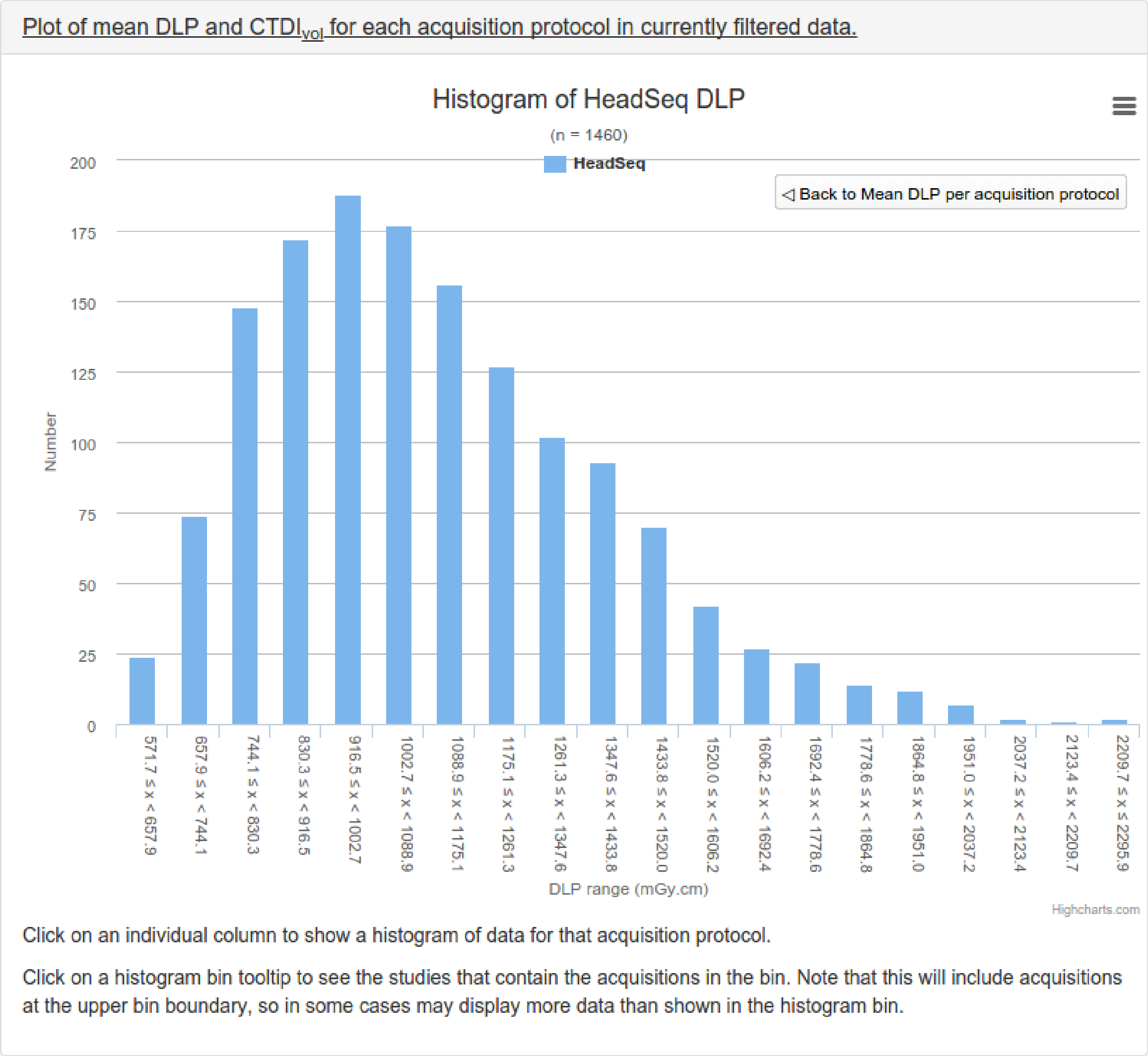
Mean DLP and CTDI$_{\text{vol}}$ per acquisition protocol
Mean DLP per acquisition protocol
Mean CTDI$_{\text{vol}}$ per acquisition protocol
DLP (mGy.cm)
1400
1200
1000
800
600
400
200
0
CTDI$_{\text{vol}}$ (mGy)
70
60
50
40
30
20
10
0
Abdomen
AxialHead
Chest
HeadSeq
Thorax
Whole head
Highcharts.com

Click on an individual column to show a histogram of data for that acquisition protocol.

Click on a histogram bin tooltip to see the studies that contain the acquisitions in the bin. Note that this will include acquisitions at the upper bin boundary, so in some cases may display more data than shown in the histogram bin.

The tooltip of each bar shows the user the name of the protocol, the number of acquisitions of that type and also the mean DLP and/or CTDI$_{\text{vol}}$ value.

Clicking on an individual bar takes the user to a histogram of DLP or CTDI$_{\text{vol}}$ for that protocol. The tooltip for each histogram bar shows the number of acquisitions. The DLP histogram tooltip also includes a link that will take the user to the list of studies that contain the acquisitions represented by that histogram bar:

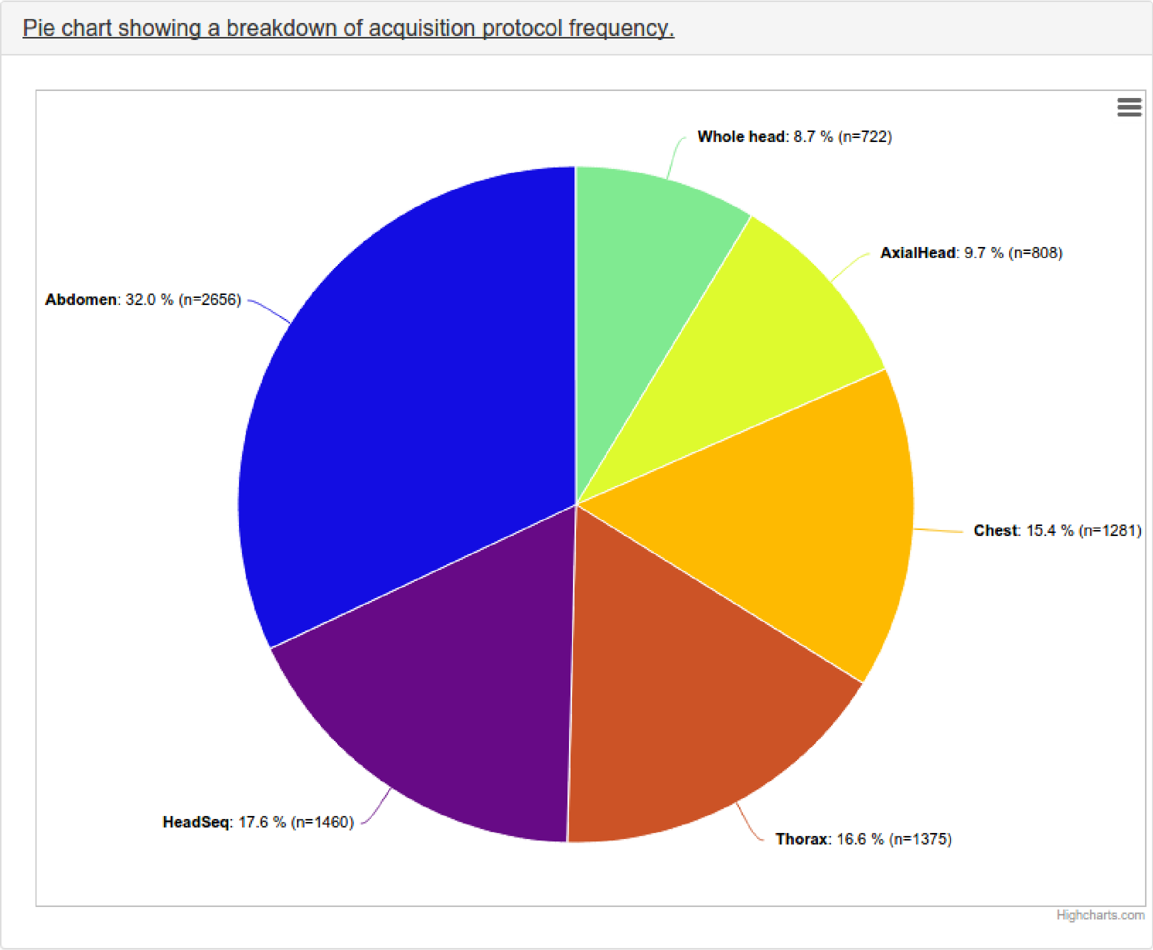


Plot of mean DLP and CTDI$_{vol}$ for each acquisition protocol in currently filtered data.

Click on an individual column to show a histogram of data for that acquisition protocol.

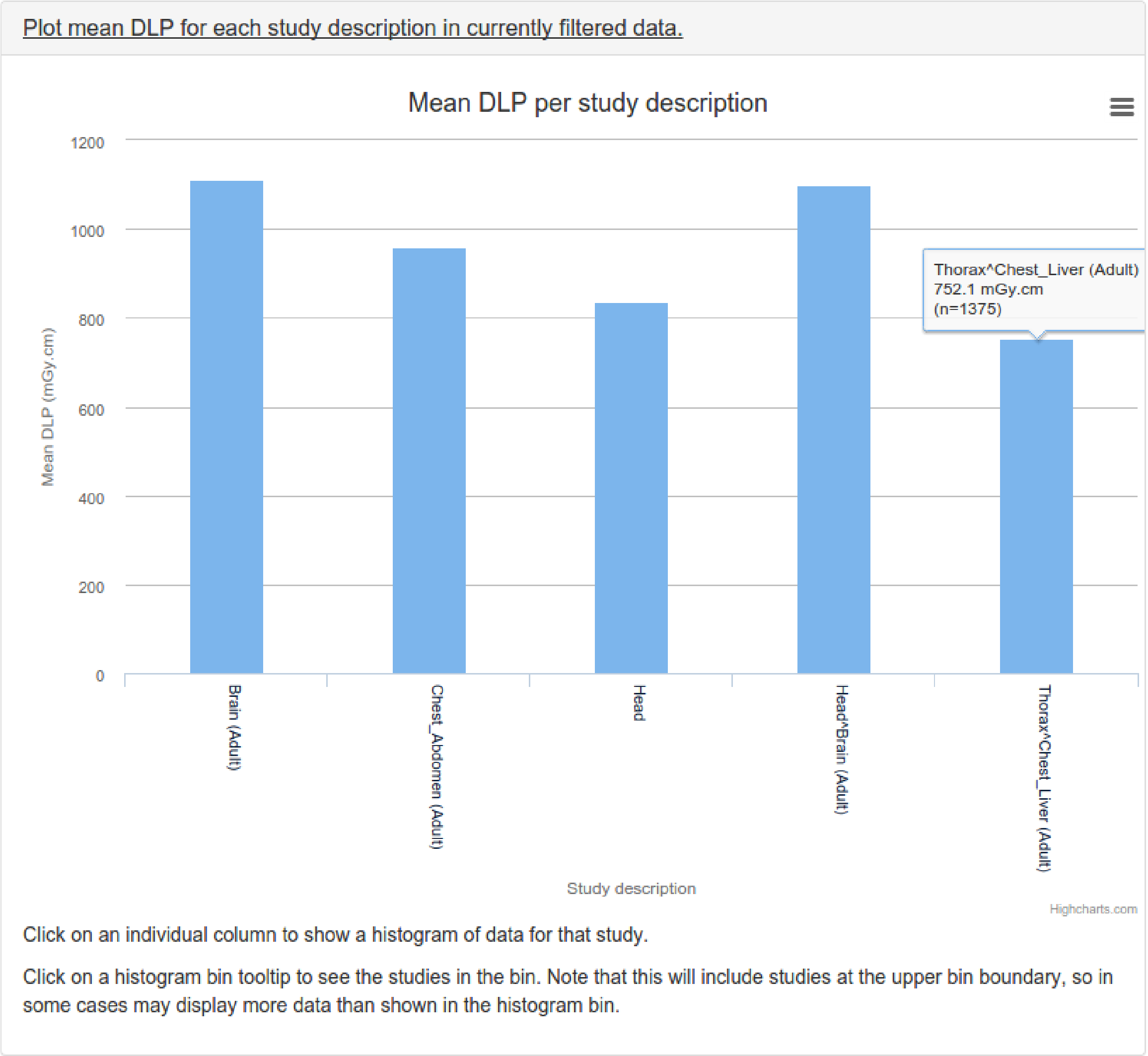
Click on a histogram bin tooltip to see the studies that contain the acquisitions in the bin. Note that this will include acquisitions at the upper bin boundary, so in some cases may display more data than shown in the histogram bin.

- Pie chart of the frequency of each acquisition protocol. Clicking on a segment of the pie chart takes the user to the list of studies that contain the acquisitions in that segment.



Pie chart showing a breakdown of acquisition protocol frequency.

- Bar chart of mean DLP for each study name. Clicking on a bar takes the user to a histogram of DLP for that study name. Clicking on a histogram bar tooltip link takes the user to the list of studies that correspond to the data represented by that bar.

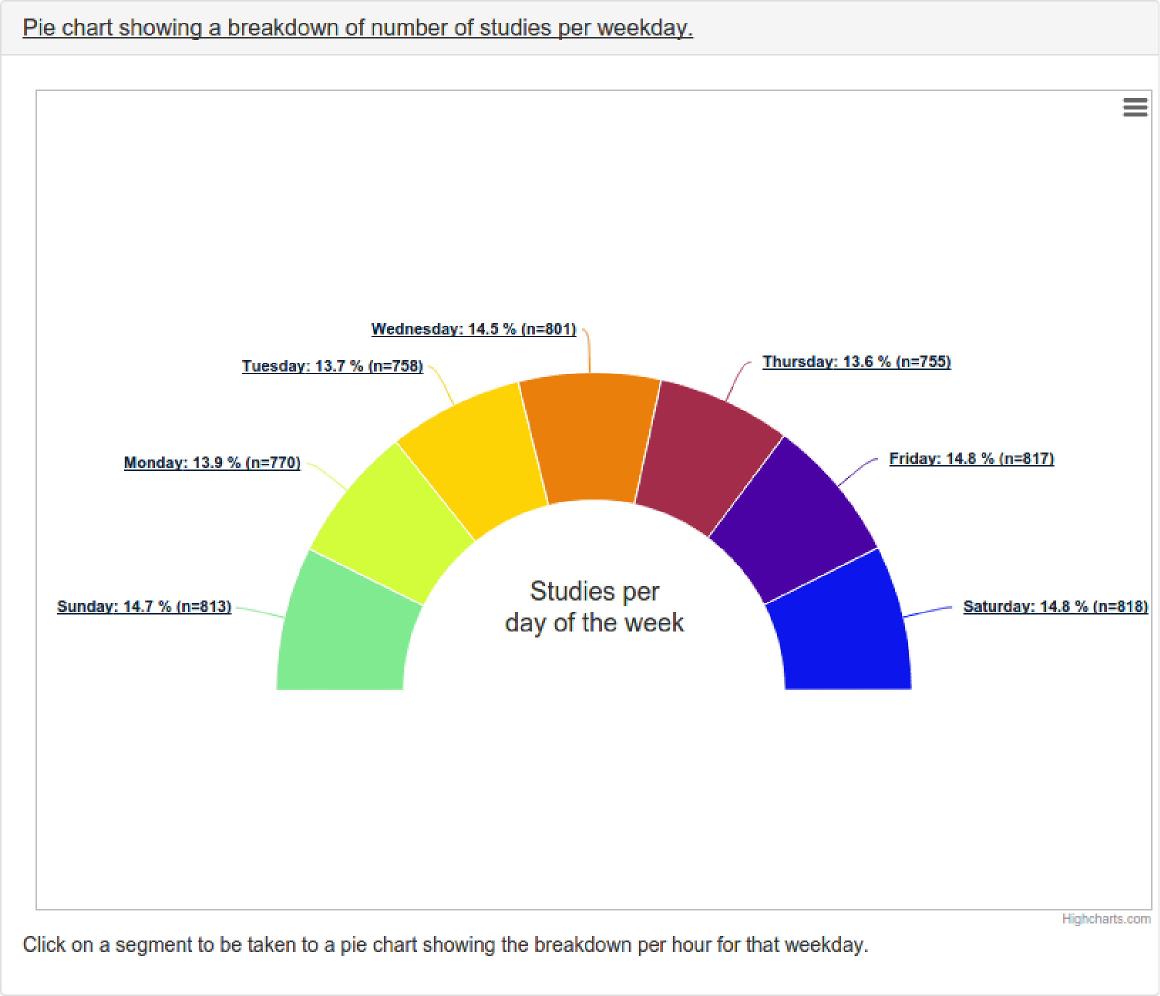


Plot mean DLP for each study description in currently filtered data.

Click on an individual column to show a histogram of data for that study.

Click on a histogram bin tooltip to see the studies in the bin. Note that this will include studies at the upper bin boundary, so in some cases may display more data than shown in the histogram bin.

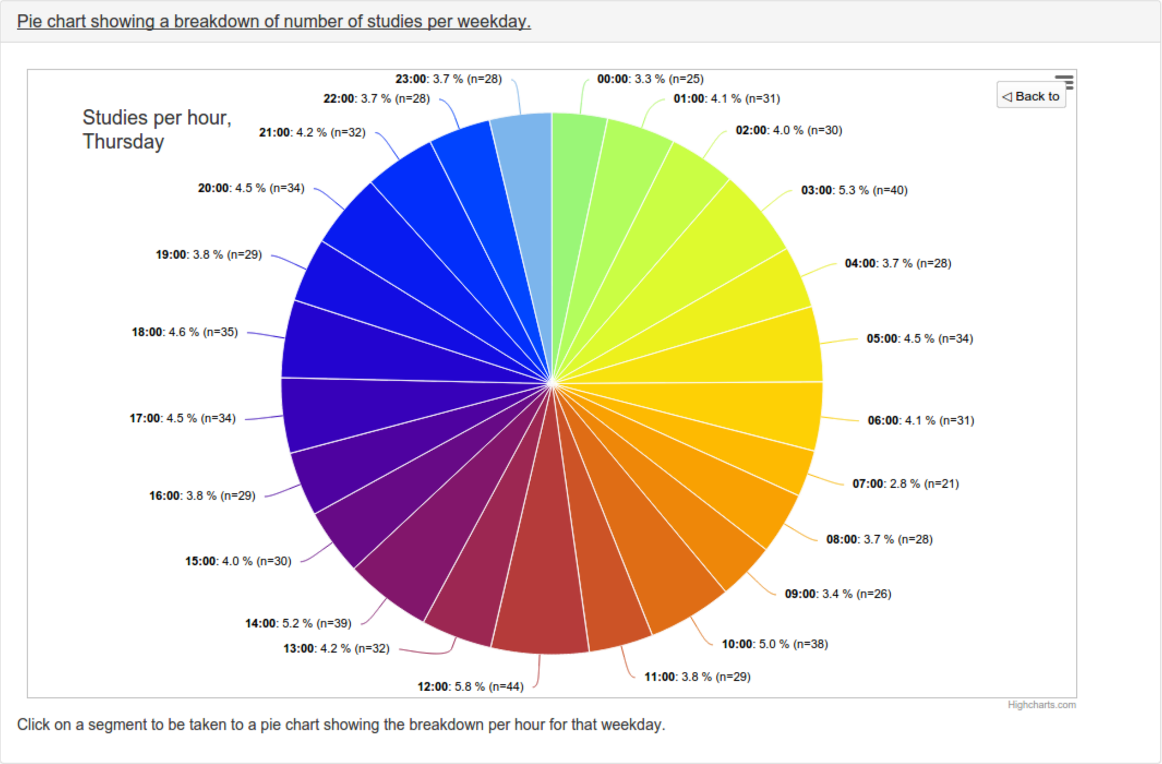
- Pie chart of the frequency of each study name. Clicking on a segment of the pie chart takes the user to the list of studies that correspond to the data in that segment.
- Pie chart showing the number of studies carried on each daty of the week:



Pie chart showing a breakdown of number of studies per weekday.

Click on a segment to be taken to a pie chart showing the breakdown per hour for that weekday.

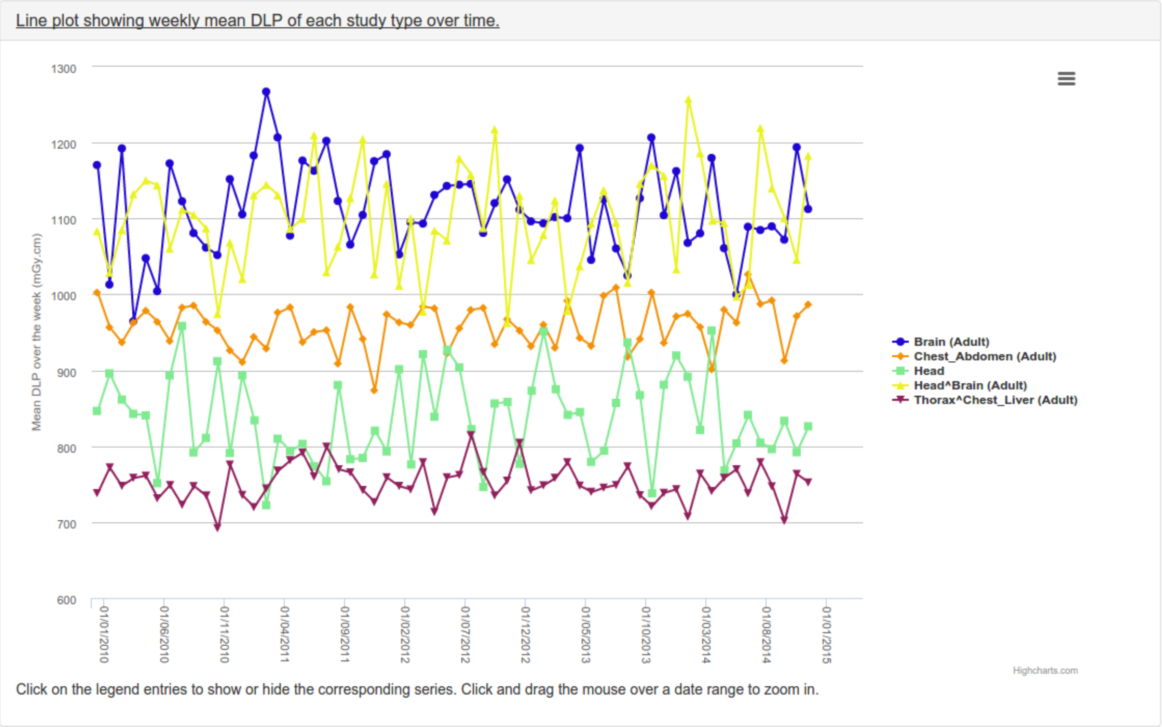
Clicking on a segment of the pie chart takes the user to a pie chart showing the studies for that weekday broken down per hour:



Pie chart showing a breakdown of number of studies per weekday.

Click on a segment to be taken to a pie chart showing the breakdown per hour for that weekday.

- Line chart showing the mean DLP of each study name over time. The time period per data point is chosen by the user in the `Chart options`. Note that selecting a short time period may result in long calculation times. The user can zoom in to the plot by clicking and dragging the mouse to select a date range. The user can also click on items in the chart legend to show or hide individual lines.



Line plot showing weekly mean DLP of each study type over time.

Click on the legend entries to show or hide the corresponding series. Click and drag the mouse over a date range to zoom in.

The available charts for radiographic data are as follows:

- Bar chart of mean DAP for each acquisition protocol. Clicking on a bar takes the user to a histogram of DAP for that protocol. Clicking on the tooltip link of a histogram bar takes the user to the list of studies that contain the acquisitions in the histogram bar.
- Pie chart of the frequency of each acquisition protocol. Clicking on a segment of the pie chart takes the user to the list of studies that contain the acquisitions in that segment.
- Bar chart of mean kVp for each acquisition protocol. Clicking on a bar takes the user to a histogram of kVp for that protocol. Clicking on the tooltip link of a histogram bar takes the user to the list of studies that contain the acquisitions in the histogram bar.
- Bar chart of mean mAs for each acquisition protocol. Clicking on a bar takes the user to a histogram of mAs for that protocol. Clicking on the tooltip link of a histogram bar takes the user to the list of studies that contain the acquisitions in the histogram bar.
- Pie chart showing the number of studies carried out per weekday. Clicking on a segment of the pie chart takes the user to a pie chart showing the studies for that weekday broken down per hour.
- Line chart showing how the mean DAP of each acquisition protocol varies over time. The time period per data point can be chosen by the user in the `Chart options`. Note that selecting a short time period may result in long calculation times. The user can zoom in to the plot by clicking and dragging the mouse to select a date range. The user can also click on items in the legend to show or hide individual lines.

7.3 Exporting chart data

An image file of a chart can be saved using the menu in the top-right hand side of any of the charts. The same menu can be used to save the data used to plot a chart: the data can be downloaded in either csv or xls format.

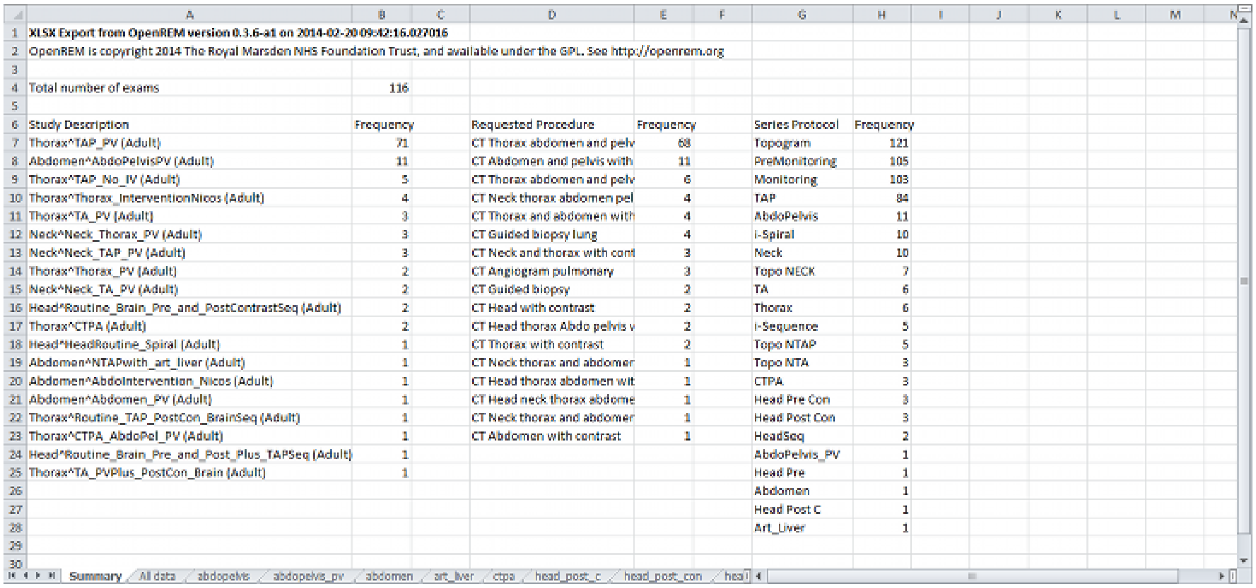
Exporting study information

8.1 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 0.5.0)

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

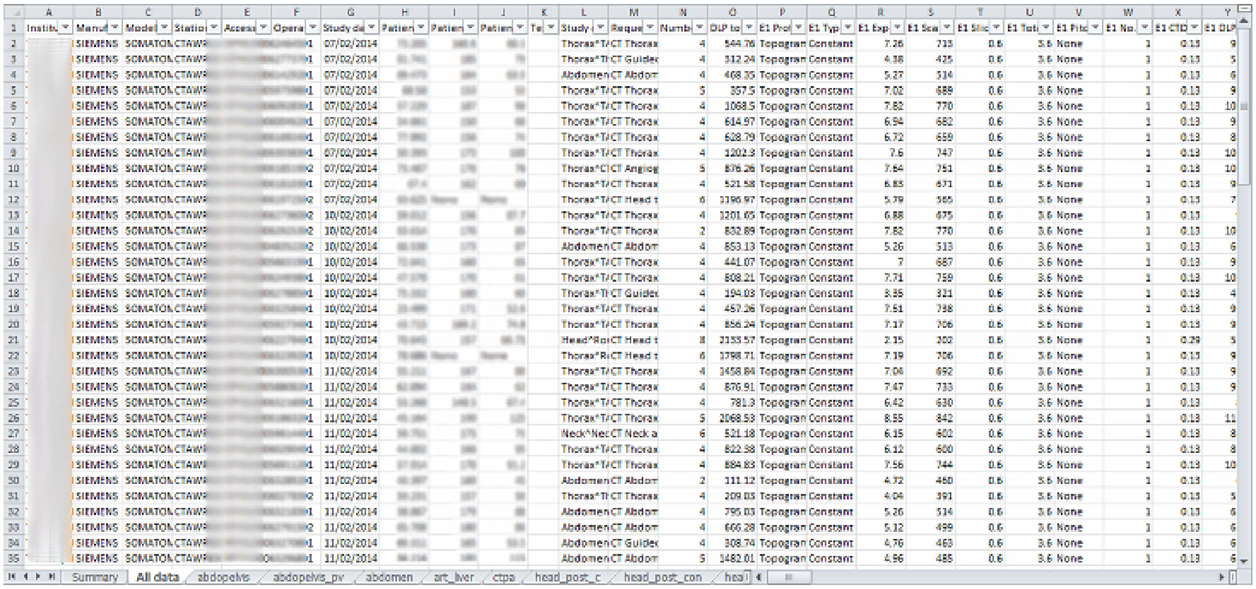
For CT and radiographic, the XLSX export has multiple sheets. The first sheet contains a summary of all the study descriptions, requested procedures and series protocol names contained in the export:



	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								

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.

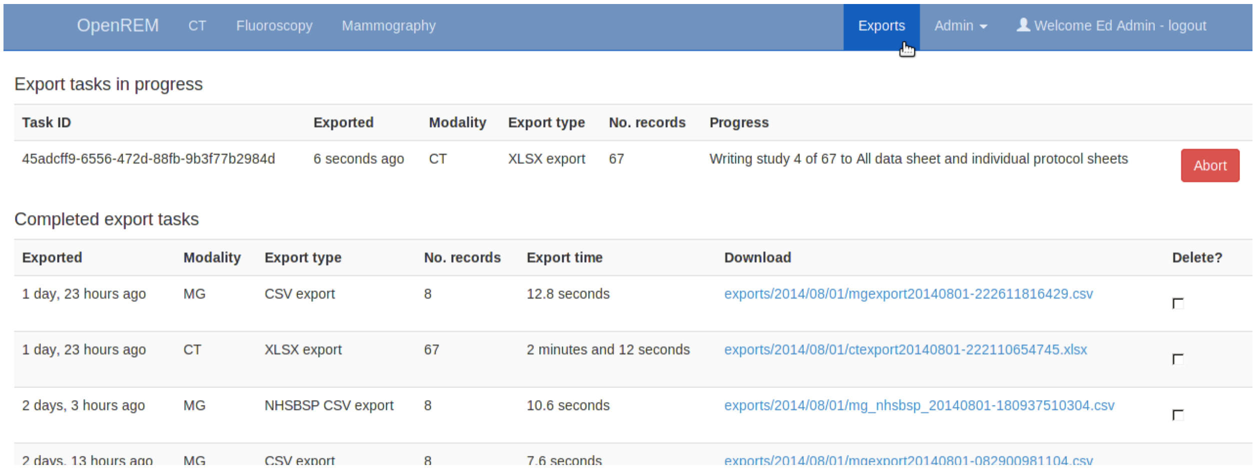
The second sheet of the exported file lists all the studies, with each study taking one line and each series in the study displayed in the columns to the right.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	Institu	Manuf	Model	Statio	Access	Opera	Study da	Patien	Patien	Patien	Te	Study	Reque	Numb	DLP to	E1 Prot	E1 Typ	E1 Exp	E1 Sca	E1 Slic	E1 Tot	E1 Pitc	E1 No.	E1 CTD	E1 DLP
2	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	544.76	Topogran	Constant	7.26	715	0.6	3.6	None	1	0.13	9
3	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^Th	CT Guidec	4	312.24	Topogran	Constant	4.38	425	0.6	3.6	None	1	0.13	5
4	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Abdomen	CT Abdom	4	468.35	Topogran	Constant	5.27	514	0.6	3.6	None	1	0.13	6
5	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	5	357.5	Topogran	Constant	7.02	689	0.6	3.6	None	1	0.13	9
6	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	1068.5	Topogran	Constant	7.82	770	0.6	3.6	None	1	0.13	10
7	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	614.97	Topogran	Constant	6.94	682	0.6	3.6	None	1	0.13	9
8	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	628.79	Topogran	Constant	6.72	659	0.6	3.6	None	1	0.13	8
9	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	1202.3	Topogran	Constant	7.6	747	0.6	3.6	None	1	0.13	10
10	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]2	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^C	CT Angiog	5	876.26	Topogran	Constant	7.64	751	0.6	3.6	None	1	0.13	10
11	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	521.58	Topogran	Constant	6.83	671	0.6	3.6	None	1	0.13	9
12	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]2	07/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Head t	6	1196.97	Topogran	Constant	5.79	565	0.6	3.6	None	1	0.13	7
13	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]2	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	1201.65	Topogran	Constant	6.88	675	0.6	3.6	None	1	0.13	
14	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]2	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	2	832.89	Topogran	Constant	7.82	770	0.6	3.6	None	1	0.13	10
15	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]2	10/02/2014	[redacted]	[redacted]	[redacted]		Abdomen	CT Abdom	4	853.13	Topogran	Constant	5.26	513	0.6	3.6	None	1	0.13	6
16	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	441.07	Topogran	Constant	7	687	0.6	3.6	None	1	0.13	9
17	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	808.21	Topogran	Constant	7.71	759	0.6	3.6	None	1	0.13	10
18	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^Th	CT Guidec	4	194.03	Topogran	Constant	3.35	321	0.6	3.6	None	1	0.13	4
19	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	457.26	Topogran	Constant	7.51	738	0.6	3.6	None	1	0.13	9
20	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	856.24	Topogran	Constant	7.17	706	0.6	3.6	None	1	0.13	9
21	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	10/02/2014	[redacted]	[redacted]	[redacted]		Head^Ro	CT Head t	8	2153.57	Topogran	Constant	2.15	202	0.6	3.6	None	1	0.29	5
22	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	10/02/2014	[redacted]	[redacted]	[redacted]		Thorax^R	CT Head t	6	1798.71	Topogran	Constant	7.19	706	0.6	3.6	None	1	0.13	9
23	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	1458.84	Topogran	Constant	7.04	692	0.6	3.6	None	1	0.13	9
24	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	876.91	Topogran	Constant	7.47	733	0.6	3.6	None	1	0.13	9
25	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	781.3	Topogran	Constant	6.42	630	0.6	3.6	None	1	0.13	
26	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	5	2068.53	Topogran	Constant	8.55	842	0.6	3.6	None	1	0.13	11
27	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Neck^Nec	CT Neck a	6	521.18	Topogran	Constant	6.15	602	0.6	3.6	None	1	0.13	8
28	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	822.38	Topogran	Constant	6.12	600	0.6	3.6	None	1	0.13	8
29	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Thorax^T/	CT Thorax	4	884.83	Topogran	Constant	7.56	744	0.6	3.6	None	1	0.13	10
30	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Abdomen	CT Abdom	2	111.12	Topogran	Constant	4.72	460	0.6	3.6	None	1	0.13	
31	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]2	11/02/2014	[redacted]	[redacted]	[redacted]		Thorax^Th	CT Thorax	4	209.03	Topogran	Constant	4.04	391	0.6	3.6	None	1	0.13	5
32	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Abdomen	CT Abdom	4	795.03	Topogran	Constant	5.26	514	0.6	3.6	None	1	0.13	6
33	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]2	11/02/2014	[redacted]	[redacted]	[redacted]		Abdomen	CT Abdom	4	666.28	Topogran	Constant	5.12	499	0.6	3.6	None	1	0.13	6
34	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Abdomen	CT Guidec	4	308.74	Topogran	Constant	4.76	463	0.6	3.6	None	1	0.13	6
35	[redacted]	SIEMENS	SOMATOM	CTAW[redacted]	[redacted]	[redacted]1	11/02/2014	[redacted]	[redacted]	[redacted]		Abdomen	CT Abdom	5	1482.01	Topogran	Constant	4.96	485	0.6	3.6	None	1	0.13	6

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:



Export tasks in progress

Task ID	Exported	Modality	Export type	No. records	Progress	
45adcff9-6556-472d-88fb-9b3f77b2984d	6 seconds ago	CT	XLSX export	67	Writing study 4 of 67 to All data sheet and individual protocol sheets	Abort

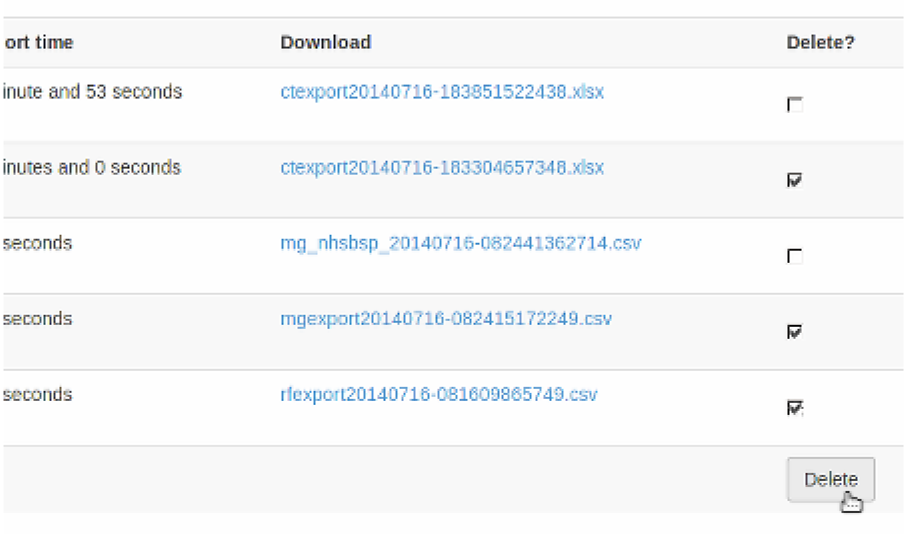
Completed export tasks

Exported	Modality	Export type	No. records	Export time	Download	Delete?
1 day, 23 hours ago	MG	CSV export	8	12.8 seconds	exports/2014/08/01/mgexport20140801-222611816429.csv	☐
1 day, 23 hours ago	CT	XLSX export	67	2 minutes and 12 seconds	exports/2014/08/01/ctexport20140801-222110654745.xlsx	☐
2 days, 3 hours ago	MG	NHSBSP CSV export	8	10.6 seconds	exports/2014/08/01/mg_nhsbsp_20140801-180937510304.csv	☐
2 days, 13 hours ago	MG	CSV export	8	7.6 seconds	exports/2014/08/01/mgexport20140801-082900981104.csv	

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.

CHAPTER 9

Exporting studies to OpenSkin

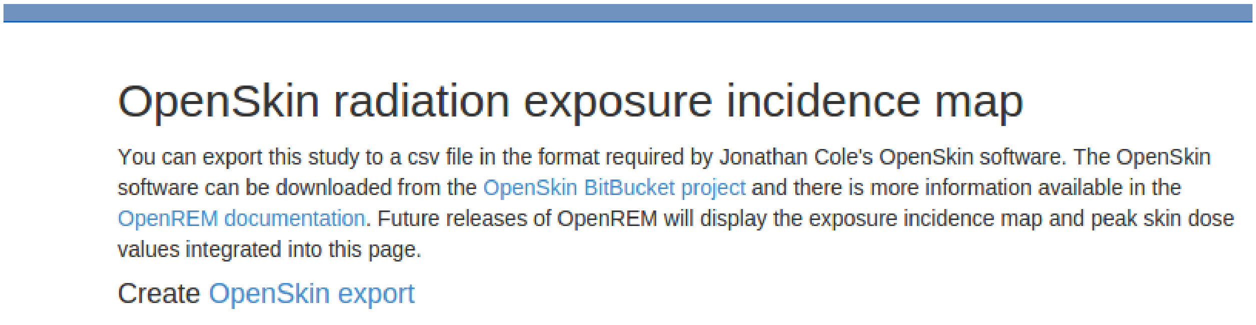
9.1 Functionality available

This is a temporary solution for 0.6.0; future versions will have OpenSkin integrated into the web interface.

- Fluoroscopy study export in a format suitable for OpenSkin
- Currently OpenSkin must be downloaded and run independently

9.2 Instructions for OpenREM

Select the fluoroscopy study you wish to create the exposure incidence map for and go to the detail view. Then click on link to create the OpenSkin export:



OpenSkin radiation exposure incidence map

You can export this study to a csv file in the format required by Jonathan Cole's OpenSkin software. The OpenSkin software can be downloaded from the OpenSkin BitBucket project and there is more information available in the OpenREM documentation. Future releases of OpenREM will display the exposure incidence map and peak skin dose values integrated into this page.

Create OpenSkin export

9.3 Instructions for OpenSkin

Download the latest version as a zip file from https://bitbucket.org/openskin/openskin/downloads. At the time of release for OpenREM 0.6.0, the current OpenSkin release was 0.4, 26th March 2015. The application referred to here will only work on Windows. When OpenSkin is built into OpenREM, it will work on both Linux and Windows servers.

- Extract the contents of the zip file into a folder on your computer and run the openSkin.exe executable
- Choose a phantom type: 3D or flat. See Phantom design for details, but in summary:
    - Flat represents the exposure incidence if the X-rays had been delivered to a film placed flat on the couch

  - 3D represents the exposure incidence if the X-rays had been delivered to a phantom consisting of a cuboid with one semi-cylinder on each side

- Select the source csv file - this should be the one exported from OpenREM
- Select the output folder - this should already exist as it can't be created in the dialogue
- Wait! Depending on the number of events in the export and the power of your machine, this can take a few minutes

Two files will be produced - a textfile called `skin_dose_results.txt` and a small image called `skin_dose_map.png`

9.3.1 Results text file

It should look something like this:

```
File created    : 04/21/15 17:42:45
Data file       : C:/Users/[...]/exports-2015-04-21-OpenSkinExport20150421-162805246134.csv
Phantom         : 90.0x70.0 3d phantom
Peak dose (Gy)  :                  0.50844405521
Cells > 3 Gy    :                              0
Cells > 5 Gy    :                              0
Cells > 10 Gy   :                              0
```

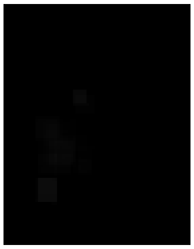
The peak dose is the peak incident dose delivered to any one-cm-square area. If any of these 1 cm^2 areas (referred to as cells) are above 3 Gy, then the number of cells in this category, or the two higher dose categories, are listed in the table accordingly.

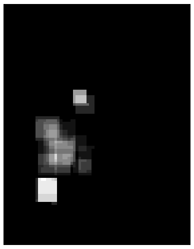
9.3.2 Incidence map image file

The image file will be a small 70x90 px PNG image if you used the 3D phantom, or 150 x 50 px PNG if you used the 2D phantom. With both, the head end of the table is on the left.

The image is scaled so that black is 0 Gy and white is 10 Gy. For most studies, this results in an incidence map that is largely black! However, if you use GIMP or ImageJ or similar to increase the contrast, you will find that the required map is there.

A native and 'colour equalised' version of the same export are shown below:





9.4 Limitations

OpenSkin is yet to be validated independently - if this is something you want to do, please do go ahead and feed back your findings to Jonathan Cole at https://bitbucket.org/jacole/

CHAPTER 10

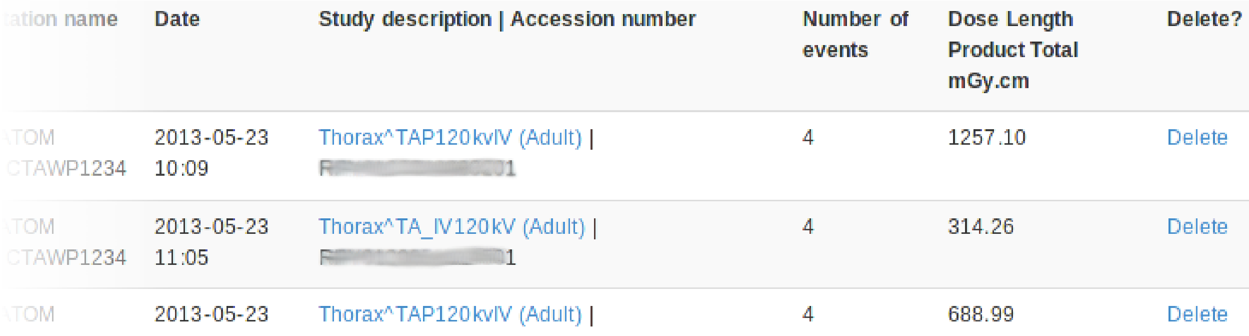
OpenREM administration (deleting studies, importing patient size data)

Contents:

10.1 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]	4	1257.10	Delete
TOM CTAWP1234	2013-05-23 11:05	Thorax^TA_IV120kV (Adult) | [redacted]	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.

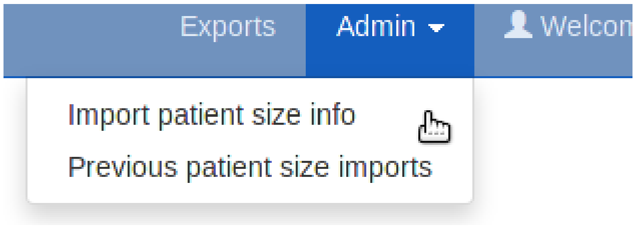
10.2 Adding patient size information from csv using the web interface

New in 0.4.3

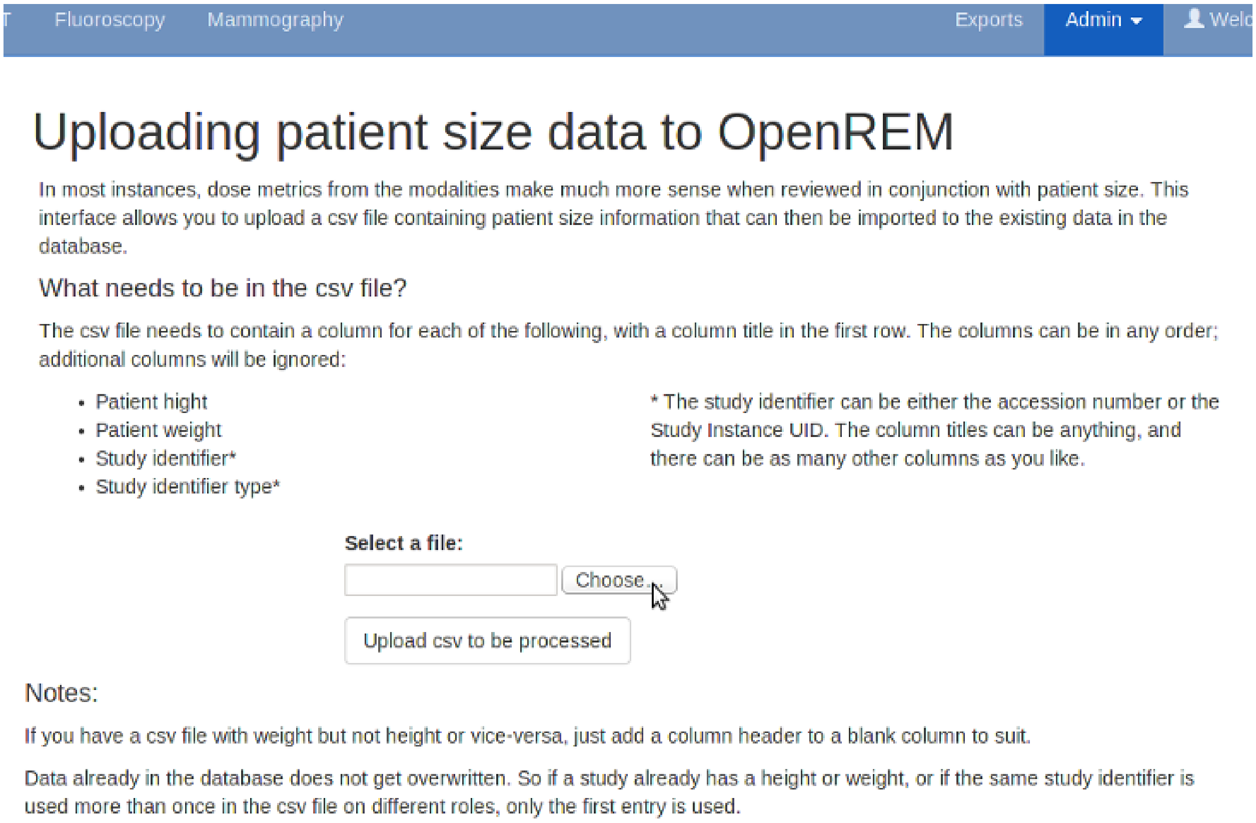
Contents

10.2.1 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.



Uploading patient size data to OpenREM

In most instances, dose metrics from the modalities make much more sense when reviewed in conjunction with patient size. This interface allows you to upload a csv file containing patient size information that can then be imported to the existing data in the database.

What needs to be in the csv file?

The csv file needs to contain a column for each of the following, with a column title in the first row. The columns can be in any order; additional columns will be ignored:

- Patient hight
- Patient weight
- Study identifier*
- Study identifier type*

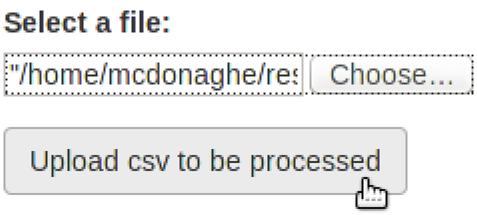
* The study identifier can be either the accession number or the Study Instance UID. The column titles can be anything, and there can be as many other columns as you like.

Select a file:

Notes:

If you have a csv file with weight but not height or vice-versa, just add a column header to a blank column to suit.

Data already in the database does not get overwritten. So if a study already has a height or weight, or if the same study identifier is used more than once in the csv file on different roles, only the first entry is used.



Select a file:

"/home/mcdonaghe/re

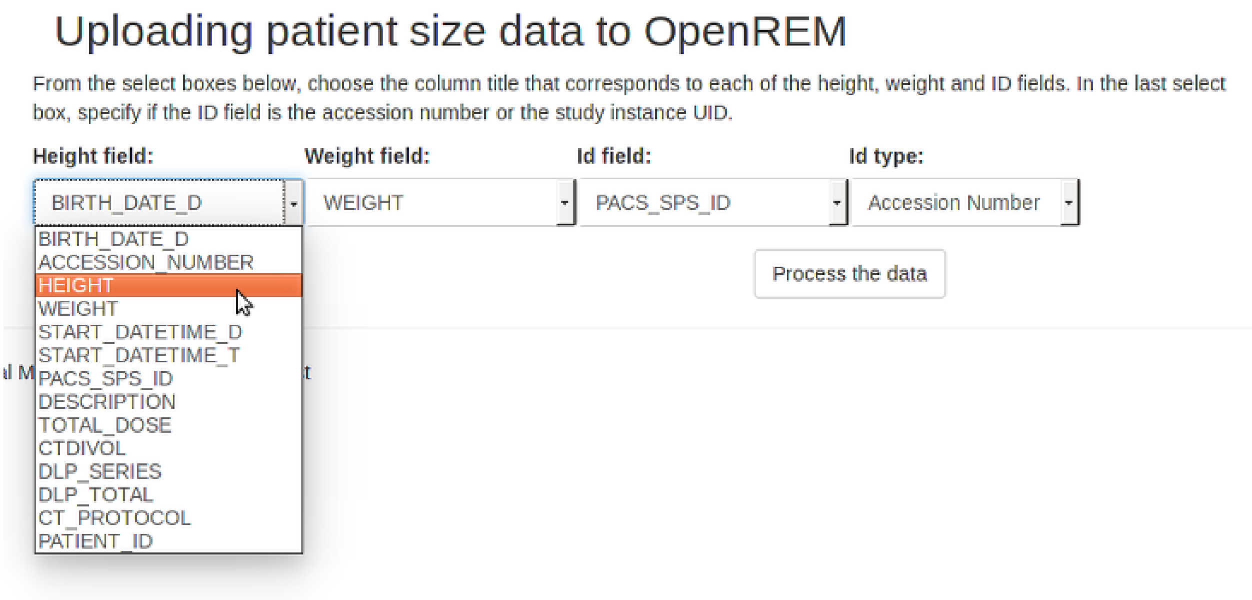
The csv file needs to have at least the required columns. Additional columns will be ignored. If your source of patient size data does not have either the height or the weight column, simply add a new empty column with just the title in the first row.

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

10.2.2 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.



Uploading patient size data to OpenREM

From the select boxes below, choose the column title that corresponds to each of the height, weight and ID fields. In the last select box, specify if the ID field is the accession number or the study instance UID.

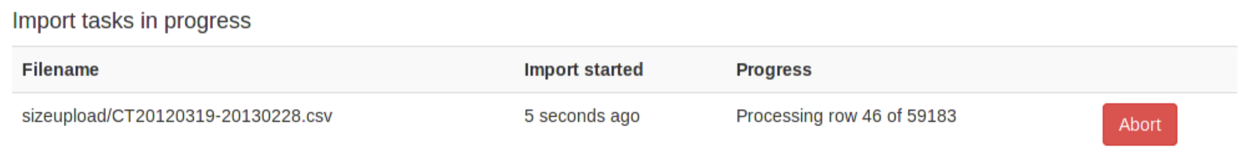
Height field: BIRTH_DATE_D

Weight field: WEIGHT

Id field: PACS_SPS_ID

Id type: Accession Number

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



Import tasks in progress

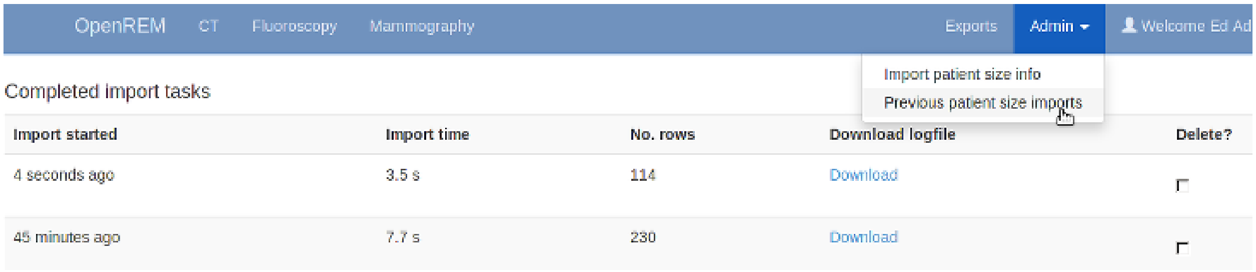
Filename	Import started	Progress	
sizeupload/CT20120319-20130228.csv	5 seconds ago	Processing row 46 of 59183	Abort

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.

10.2.3 Reviewing previous imports

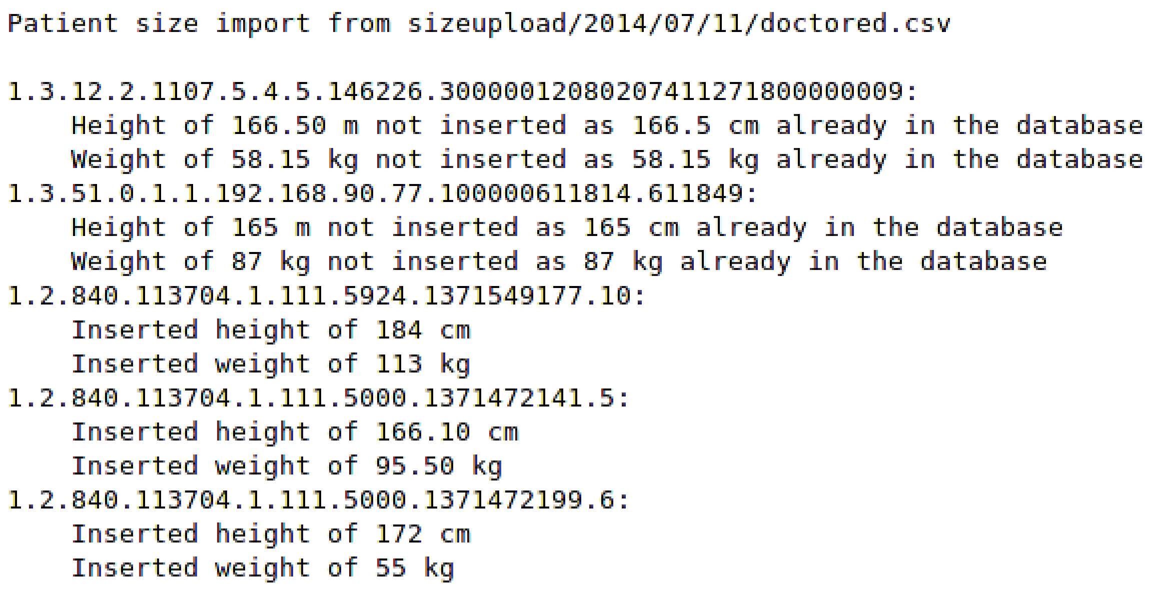
After an import is complete, it is listed in the completed import tasks table. You can also get to this page from the Admin menu:



Completed import tasks

Import started	Import time	No. rows	Download logfile	Delete?
4 seconds ago	3.5 s	114	Download	☐
45 minutes ago	7.7 s	230	Download	☐

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
```

10.2.4 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 11

Documentation for the OpenREM code

Contents:

11.1 DICOM import modules

11.1.1 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.

11.1.2 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.

11.1.3 CR and DR module

In practice this is only useful for DR modalities, but most of them use the CR IOD instead of the DX one, so both are catered for. This module makes use of the image headers much like the mammography module.

11.1.4 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.

11.2 Non-DICOM import modules

11.2.1 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`**

11.3 Export from database

11.3.1 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`**(*filterdict*)

Export filtered CT database data to multi-sheet Microsoft XSLX files.

Parameters **`filterdict`** (*`HTTP get`*) – Query parameters from the CT filtered page URL.

(task)`remapp.exports.dx_export.`**`dxxlsx`**(*filterdict*)

Export filtered DX and CR database data to multi-sheet Microsoft XSLX files.

Parameters **`filterdict`** (*`HTTP get`*) – Query parameters from the DX and CR filtered page URL.

11.3.2 Single sheet CSV exports

(task)`remapp.exports.exportcsv.`**`exportFL2excel`**(*filterdict*)

Export filtered fluoro database data to a single-sheet CSV file.

Parameters **`request`** (*`HTTP get`*) – Query parameters from the fluoro filtered page URL.

(task)`remapp.exports.exportcsv.`**`exportCT2excel`**(*filterdict*)

Export filtered CT database data to a single-sheet CSV file.

Parameters **`request`** (*`HTTP get`*) – Query parameters from the CT filtered page URL.

(task)`remapp.exports.exportcsv.`**`exportMG2excel`**(*filterdict*)

Export filtered mammography database data to a single-sheet CSV file.

Parameters **`request`** (*`HTTP get`*) – Query parameters from the mammo filtered page URL.

(task)`remapp.exports.dx_export.`**`exportDX2excel`**(*filterdict*)

Export filtered DX database data to a single-sheet CSV file.

Parameters **`request`** (*`HTTP get`*) – Query parameters from the DX filtered page URL.

Specialised csv exports - NHSBSP formatted mammography export

(task)`remapp.exports.mg_csv_nhsbsp.`**`mg_csv_nhsbsp`**(*filterdict*)

Export filtered mammography database data to a NHSBSP formatted single-sheet CSV file.

Parameters **`filterdict`** (*`dict`*) – Dictionary of query parameters from the mammo filtered page URL.

Returns None - file is saved to disk and location is stored in database

11.4 Tools and helper modules

11.4.1 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

11.4.2 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 ContextID 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 ContextID entry for code value passed

11.4.3 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

11.4.4 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

11.4.5 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.

11.5 Models

11.6 Filtering code

11.7 Views

11.8 Export Views

11.9 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

11.10 Indices and tables

- genindex
- modindex
- search

CHAPTER 12

Indices and tables

- genindex
- modindex
- search

Index