MyTardis @ Swinburne

Administrator guide for the BPsyC Tardis system

Background

MyTARDIS is a multi-institutional collaborative venture that facilitates the archiving and sharing of data and metadata collected at major facilities such as the Australian Synchrotron and ANSTO and within Institutions.

An example of the benefit of a system such as MyTARDIS in the protein crystallography community is that while the model coordinates and (less often) the structure factors (processed experimental data) are stored in the community Protein Data Bank (PDB) the raw diffraction data is often not available. There are several reasons why this is important, which can be summarised as:

- The availability of raw data is extremely useful for the development of improved methods of image analysis and data processing.
- Fostering the archival of raw data at an institutional level is one of the best ways of ensuring that this data is not lost (laboratory archives are typically volatile).
- Better data management for instrument users: reliable storage of all data and easily finding data in the future
- Streamlined instrument use: no need to burn CDs

MyTardis @Swinburne

Official Website: http://www.research.swinburne.edu.au/researchers/managedata/tardis/

Key People:

- **VeRSI Developers:** Steve Bennet – solution architect and project manager, and Cyrus Keong – software developer for metadata filters. (→end 2013)
- **Swinburne Research support:** Arna Karick – akarick@swin.edu.au, and Shaun Peterson – speterson@swin.edu.au
- **BPsyC Lead Project Scientist:** Will Woods

The BPsyC MyTardis Store

For ACCESS point your web browser to: http://bpsyctardis-dev.swin.edu.au:8000
Local administrator and lead project scientist: Will Woods - wwoods@swin.edu.au
Swinburne Research administrators: Arna Karick – akarick@swin.edu.au & Shaun Peterson – speterson@swin.edu.au

Architecture

MyTARDIS is built on the Django web framework, which itself is built on Python, thus MyTARDIS follows the architectural model of Django.

At the simplest level, the experimental data is simply a collection of files (datafiles), which are grouped in to datasets, which are grouped in to Experiments/ At each level, experiment, dataset and datafile, administrator defined parameters may be added, grouped in to parameter sets.

MyTardis doesn't impose any interpretation on what is considered an experiment or dataset. Examples of how datasets may be grouped are: by sample, by instrument settings, or as a time sequence, e.g. artificially aging a material and investigating the effects.

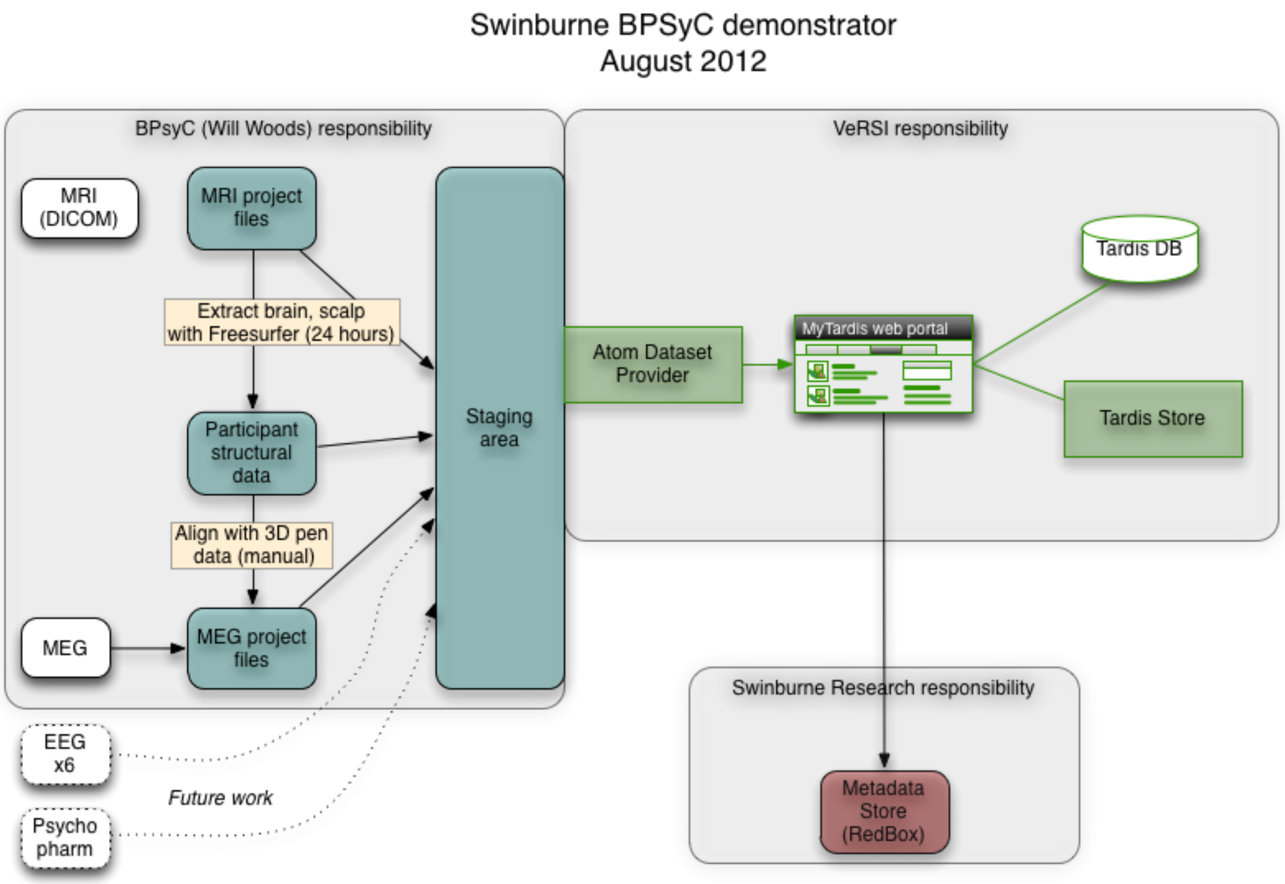

Figure 1 –*Swinburne BpsyC demonstrator – Draft Architecture*:
Source: (created by Steve Bennett): https://sites.google.com/site/bpsycdatamanagement/

Figure 1 is similar to the final implementation except that the staging area, Tardis store and database are mounted on Swinburne's Research Storage. The metadata link to ReDBox has not yet been implemented. Unlike CAOUS Tardis, this is a multi-harvest system. The first staging area is separate area on Research Storage

The components:

- **RSync server** running on each instrument machine. (On Windows, use DeltaCopy.)
- **Harvester script** which collates data from the RSync servers to a **staging area**.
- **Atom provider** which serves the data in the staging area as an Atom feed.
- **MyTardis**, which uses the **Atom Ingest app** to pull data from the Atom feed. It saves data to the **MyTardis store** and metadata to the **MyTardis database**.
- **Networking**:Each instrument needs to be reachable on at least one port from the harvester.

RSync Server:
We assume each PC is currently operating in "stand-alone" mode (users save data to a local hard disk) but is reachable by network. If your users have authenticated access to network storage, you'll do something a bit different.
Install an Rsync server on each PC. For Windows machines, use DeltaCopy. It's free. Estimate 5 minutes per machine. Make it auto-start, running in the background, all the time, serving up the user data directory on a given port.

MEG Data:
For a full description of this section see; https://sites.google.com/site/bpsycdatamanagement/meg-data
MEG data is stored in **.fif** files. These can by read using the MNE library. MNE has pre-requisites such as numpy, scipy and matplotlib. To install it, the easiest way seems to be use the Enthought EPD bundle: http://www.enthought.com/products/edudownload.php
Much metadata can be found using the raw.info object.

More information:
http://martinos.org/mne/stable/python_tutorial.html
http://www.martinos.org/mne/stable/getting_started.html

Installation

BpsyC Development environment;

- (web) bpsyctardis-dev.cc.swin.edu.au
- ssh mytardis@bpsyctardis-dev.cc.swin.edu.au

BpsyC Production environment:

- (web) bpsyctardis.cc.swin.edu.au
- ssh mytardis@bpsyctardis.cc.swin.edu.au

According to ITS both VMs have the same specifications.

- **home directory:**
  [mytardis@caoustardis-vm1 current]$ /opt/mytardis

- **releases directory:** /opt/mytardis/releases
  There are three different Tardis software releases on the development and production VMs. I'm not sure how they differ but it's

  a2cccad8edc96123321856fcadc9481261178d61/
  a549cd05272afe8f16c2fe5efe8158490acbde82/
  3.5/

  within these directories are a whole bunch of files and sub directories.

- current → points to -- releases/a549cd05272afe8f16c2fe5efe8158490acbde82/

  When Tardis is upgraded the new version should be unpacked in the releases/ and a new symbolic link created called 'current'. The old version should be retained in the releases/ directory. Version releases are downloaded from github: https://github.com/mytardis/mytardis

- mytardis → points to – current (this seems superfluous)

- **atom directory:** /opt/mytardis/atom/ → turns the files on disk into a feed

- **node directory:**

- **nodesrc directory**:

- **shared directory:**   /opt/mytardis/shared/apps   → contains the directory that turns the feed into and ingest – also confusingly called atom AND also contains the all important 'settings.py' file.

The 'INSTALL.rst' file contains the link to the online documentation: http://mytardis.readthedocs.org/en/latest/install.html

/apache - django stuff
/bin –
/docs – a bunch of text files providing limited documentation on various aspects of Tardis.

du –sh *   → disk usage/directory sizes

Database

The database is a postgress database under the user mytardis.

- **Create Database:** Type 'createdb' while logged in as the mytardis user. This will also prompt you to create a super-user for initial access to the database.

- **Populate MyTardis Specific tables**: To populate the database with the necessary tables and initial data needed to run mytardis, use the command './bin/django migrate' While in the 'current' directory.

- **Log in:** To log into the database type 'psql' while logged in as the mytardis user no username and password will be required.

Starting and Stopping Tardis

starttardis.sh

Start Tardis by running the ./starttardis.sh script. The script includes a 'nohup' command which is a non-specific terminal start up. 'celeryd' (daemon) and 'celerybeat' are servers that run background tasks – e.g. scheduled tasks. On start up, one of the first things that happens is that uWSI starts. A bunch of stuff of tardis directories/files are also added to the python path.

stoptardis.sh

Stop Tardis by running the ./stoptardis.sh script. This kills celery, django etc. It's a little buggy and sometimes doesn't kill the uwsgi process. You can check if this is still running by typing;

```
ps aux | grep uwsgi
```

Staging directories

Each time Tardis is upgraded, you need to create new symbolic likes to the staging/ and store/ directories (/projects/mytardis/staging/). These are mounded on ITS Research Storage. Both CAOUS and BpsyC data are mounted as independent shares on Research Storage. ** these are not write on the new CAOUS dev – Shaun to check?
CAOUS:
The Tardis directory is /opt/mytardis/current/var/
CAOUS doesn't use Research Storage, the data is pulled directly from the instrument to the PCs using delta copy.
BPSYC:
The staging/ directory holds the sinues_MEG_data/ **does it?? In this directory are a bunch of sym links that point to the relevant directories. There is also a staging directory in the /opt/mytardis/staging/lhugrass/MEG. Ideally datasets should be stored in the following format: |Username|Instrument| Experiment |Dataset

Atom Server

The atom server turns the files on disk into a feed (from port 4000): /opt/mytardis/atom/atom-dataset-provider. The provider.sh script includes directory definitions and formatting. The kill_provider.sh script writes to a log file: 'atom-dataset-provider.log'

[NOTES: atom-dataset-provider for one of the CAOUS machines. Steve reckons he hasn't configured the properly - the staging directory STAGING="/mnt/np_staging" is wrong???? not sure if he fixed this]

You can check this is working properly by typing;
[mytardis@caoustardis]$ curl localhost:4000
For CAOUS dev (new) there is an error: curl: (7) couldn't connect to host
atom is a form of XML which has open <entry> and closing </entry> tags for each dataset. There are also a bug of other tags like <id>, <tardis:uniqueid>, <tardis:experimentTitle>, <author> etc.

For example, in CAOUS prod curl localhost:4000 gives you;

```
<entry>
<id>Ed/20130822</id>
  <tardis:uniqueid>http://caoustardis-
vm2.cc.swin.edu.au:4000/a6e1d5a689b4683a5426f806ce6dd06c0b308f5e9f02fa12211f844134a1a8ba@2013-08-
22T05:05:45.000Z</tardis:uniqueid>
  <author>
   <name>jhartley</name>
  </author>
  <tardis:experimentTitle>Ed</tardis:experimentTitle>
  <title>20130822</title>
  <tardis:instrument>Raman</tardis:instrument>
  <content type="xhtml">
   <div xmlns="http://www.w3.org/1999/xhtml">
    <ul>
          <li>jhartley/Raman/Ed/20130822/100_1mT_N_50_10x_4.spc</li>
          <li>jhartley/Raman/Ed/20130822/100_1mT_N_50_10x_4.txt</li>
          <li>jhartley/Raman/Ed/20130822/100_1mT_N_50_10x_4.wxd</li>
```

The directory /lib/atom-data/provider/templates/feed.atom provides the metadata template/definitions.
Documentation for this is all on Github

Turning the feed into the Tardis Ingest

BPSYC Harvesting:
Confusingly this is also called atom. /opt/mytardis/shared/apps/atom contains the ingest app (a whole bunch of Python) which is configured through 'options.py'. PARAM lines don't normally get changed. but you can change the 'ALLOW_EXPERIMENT_BLAH' parameters.
Typical parameters;

```
  # Names of parameters, must match fixture entries.
  # Some are also used for <category> processing in the feed itself.
  PARAM_ENTRY_ID = 'EntryID'
  PARAM_EXPERIMENT_ID = 'ExperimentID'
  PARAM_UPDATED = 'Updated'
  PARAM_EXPERIMENT_TITLE = 'ExperimentTitle'

  ALLOW_EXPERIMENT_CREATION = True       # Should we create new experiments –useful for generating users, still need to
setup usernames and passwords as well
  ALLOW_EXPERIMENT_TITLE_MATCHING = True   # If there's no id, is the title enough to match on –normally the default
  ALLOW_UNIDENTIFIED_EXPERIMENT = True    # If there's no title/id, should we process it as "uncategorized"?
  DEFAULT_UNIDENTIFIED_EXPERIMENT_TITLE = "Uncategorized Data"
  ALLOW_UNNAMED_DATASETS = True          # If a dataset has no title, should we ingest it with a default name
  DEFAULT_UNNAMED_DATASET_TITLE = '(assorted files)'

  # Should we always examine every dataset entry in the feed, even after encountering "old" entries?
  ALWAYS_PROCESS_FULL_FEED = False  - Feed generated form newest to oldest. It first looks at datasets and checks if they
have already  been ingested. Sometimes new files will be added but will look like they are old and so Tardis doesn't pick them
up.
```

There is also an associated settings.py file in the shared directory: /opt/mytardis/shared/settings.py
This file;
- defines the link to the MNE filter
- debug mode should be off because it takes up loads of memory. When it's on you get loads of debug information but this may be viewed as a security risk.
- lists the installed apps (e.g. ingest app)
- celerybeat schedule - the bit that works the feed. At the moment it looks like we don't have a full harvest set up (Steve was going to look into this.)

settings.py;

REQUIRE_DATAFILE_SIZES = False **
REQUIRE_DATAFILE_CHECKSUMS = False **
** these two are important flags (checksum not really helpful but Tardis requires these)
REQUIRE_VALIDATION_ON_INGESTION = False

The logfile for 'settings.py' is symlinked (→ /var/log/mytardis/). There is also an empty data/ file which isn't used. The changeme file should not be changed. ** I can't remember what the changeme file refers to..?

CAOUS Harvesting:
The directory /opt/mytardis/MicroTardis_Harvest/ contains the ingest scripts.
domounts.sh - this script used to setup windows mounts to pull data across, now it just pings the machines to make sure things are working/mounted.

domount "deltacopy username/virtual directory" " IP address " "Raman" - internal alias for instrument names
domount "deltacopy username/virtual directory" " IP address " "Bruker"
domount "deltacopy username/virtual directory" " IP address " "Olympus"

harvest.sh - harvest script. includes path names, what the log files should be called, rsync copy options etc.

Connects once to the machine and gives the top directory view, then connects to each directory to harvest.

Sometimes you will see 'rsync completed' errors in log - usually due to the machine being switched off. Not all PC are deliberately kept on 24 hours.

set_status.sh - checks that machines are still up an running - we could implement an email saying the system is down. Subsequent downtime doesn't give an alert .You also don't get an email when the system is back up. Steve did set this up for RMIT and we should try and get this set up.

makestatus.html & **status.html** - used for Tardis home page -- a little harvesting status box to see if the system is offline or not. Monash and RMIT implemented this.

To check the logs for the harvesting go to the /opt/mytardis/MicroTardis-Harvest/logs/ directory

Bruker harvests properly at the moment.

Raman harvester is working but there are no filters. Harvesting does work for PCs (Olympus & Raman) but only when the machines are on and being used. In the logfiles it may appear that there are long periods when it's not harvesting.

Steve created temp-staging/ and temp-storage/ directories when Research Storage is down. This is now commented out so that the harvester should be pointing to the Research Storage now.

Ingest Logs

For CAOUS there are ~four ingest logs

- three are in shared → /var/log/mytardis/
- but the harvest log is running in /opt/mytardis/mytardis/

NOTE (from when Steve was going through this with us): Looks like the restarting CAOUS Tardis has found a new issue with the latest filter update: may need to talk to Cyrus (Steve doesn't know much about the filters.)

Filters

/tardis/tardis_portal/filters
- olympusoib.py

Swinburne Skin

The **/opt/mytardis/shared/** directory contains the swinburne.css file.

Miscellaneous notes:

At some point Steve was concerned about this. I can't remember what he was referring to.
/projects/mytardis/staging/mcharnely/Olympus/Nothch_inhibition/20131113
Maybe not - remnant of the RS storage downtime. I think. Gin originally set things up in /mnt/ and then changed to /projects/

Instruments hooked up to BPsyC Tardis:

Instrument	Control PC O/S	Control PC Software (Instrument)	Location	File Format 1	Metadata filter status	Shared Directory	Test Data Directory
Elekta TRIUX MEG							

Swinburne Research Setup Guide:

1. ssh caous mytardis@caoustardis-vm2.cc.swin.edu.
2. scp –rp all_relevent_directories/
3. Re-create symbolic links
4. ITS: ask to change firewall rules to allow us access to the server
5. ITS: Re-started nginx (web server) – appears we do not have access to do this.
6. We also created a blank database in the background (doesn't copy across)
7. Populated it with the tables that Tardis needs to run
8. Re-Pointed the atom-ingest process at a local directory as we do not have access to the mounted storage from this machine, and re-started the ingest process.

MyTardis contacts at other instutions:

Monash:

- Biosciences Data Platform (BDP) Manager: Steve Androulakis
- BDP Developer: Dr. Grischa Meyer

RMIT:

- BDP Developers: Dr. Iman Yusuf, Dr. Ian Thomas