Mastr-MS Documentation

Release 1.12.4

Centre for Comparative Genomics

May 26, 2016

Contents

Mastr-MS is a web-based tool for experimental design, sample metadata configuration, and sample data acquisition.

CHAPTER 1

Metabolomics Australia Sample Tracking Repository

Mastr-MS is a collaborative project undertaken by the Australian Bioinformatics Facility (ABF), located at the Centre for Comparative Genomics, for Metabolomics Australia (MA).

Mastr-MS is comprised of a number of modules:

User Management Module

A secure online user management module for the Metabolomics Australia Data and Sample Management System.

Sample Management Module

An online sample management module allowing users to submit samples, track samples through experimental workflows and receive results.

Data Management Module

Provision of secure, online metabolomics repository to capture all relevant MA experimental workflows. It includes storage of raw data files as well as all relevant meta data associated with an experiment.

CHAPTER 2

Table of Contents

2.1 Mastr-MS Server

The Mastr-MS server is a Django application which provides a web interface for user management, sample management, and access to experiment data.

2.1.1 Installation

Yum repository setup

Mastr-MS is distributed as RPM, tested on Centos 6.x (x86_64). To satisfy dependencies, Epel and IUS repos need to be enabled:

```
sudo rpm -Uvh http://repo.ccgapps.com.au/repo/ccg/centos/6/os/noarch/CentOS/RPMS/ccg-release-6-2.noa
sudo rpm -Uvh http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
sudo rpm -Uvh http://dl.iuscommunity.org/pub/ius/stable/CentOS/6/x86_64/ius-release-1.0-11.ius.centos
```

Dependencies

The database server package isn't a dependency of the Mastr-MS RPM, so it has to be installed manually:

```
sudo yum install postgresql-server
```

Note: These instructions are written for PostgreSQL. With minor alterations (issue MAS-32) Mastr-MS could work with MySQL/MariaDB, but at present only PostgreSQL is supported.

Install

Install the Mastr-MS RPM, replacing `X.X.X` with the desired version:

```
sudo yum install mastrms-X.X.X
```

2.1.2 Server Configuration

Database Setup

If starting from a fresh CentOS install, you will need to configure PostgreSQL:

```
service postgresql initdb
service postgresql start
chkconfig postgresql on
```

To enable password authentication in PostgreSQL, you need to edit `/var/lib/pgsql/data/pg_hba.conf`. As described in the documentation, add the following line to `pg_hba.conf`:

```
# TYPE  DATABASE    USER        CIDR-ADDRESS          METHOD
host    all         all         127.0.0.1/32          md5
```

Then restart postgresql.

Database Creation

Create the database in the normal way for Django/PostgreSQL:

```
sudo su postgres
createuser -e -DRS -P mastrms
createdb -e -O mastrms mastrms
exit
```

The default database, username, password are all set to *mastrms*. These settings can be changed, see (*Django Settings File*).

Database Population

Run Django syncdb and South migrate:

```
sudo mastrms syncdb
sudo mastrms migrate
```

Django will prompt to create a superuser. If you choose to create a superuser, ensure the username and e-mail address are exactly the same, otherwise you won't be able to log in.

Alternatively, you can use the preconfigured user `admin@example.com` with password `admin` to log in. Once you have set up your own users, the `admin@example.com` user can be deleted.

Apache Web Server

The Mastr-MS RPM installs an example Apache config file at `/etc/httpd/conf.d/mastrms.ccg`. This config is designed to work out of the box with an otherwise unconfigured CentOS Apache installation. All that is needed is to rename `mastrms.ccg` to `mastrms.conf` so that Apache will pick it up.

If you have already made changes to the default Apache configuration, you may need to tweak `mastrms.conf` so that the existing setup is not broken. It may be necessary to consult the Apache and mod_wsgi documentation for this.

User Creation

It is a good idea to create a special user and group for data syncing:

```
SYNCUSER=maupload
adduser $SYNCUSER
su $SYNCUSER -c "mkdir --mode=700 /home/$SYNCUSER/.ssh"
```

Data Repository and Permissions

By default, the data repository is located at `/var/lib/mastrms/scratch`.

There should be ample disk space on this filesystem and some data redundancy would be desirable. If this is not the case, then you could mount a suitable file system at this path. If the data repository needs to be at another location, its path can be configured in the settings file.

The data sync user needs to be able to write to this directory, and the web server user needs to be able to read from this directory, so:

```
DATADIR=/var/lib/mastrms/scratch
mkdir -p $DATADIR/files $DATADIR/quotes
chown maupload:maupload $DATADIR $DATADIR/*
chmod 2770 $DATADIR $DATADIR/*
```

Also add the web server user to the `maupload` group so that it can read/write the data which `maupload` has rsynced in:

```
usermod -a -G maupload apache
```

Django Settings File

The config file for Mastr-MS is installed at `/etc/mastrms/mastrms.conf`. It contains basic settings that need to be changed for most sites, for example the database password. There are comments and example values for each setting within this config file.

Option	Description
dbtype	Database backend – always use pgsql.
dbname	The rest are standard database connection options.
dbuser	
dbpass	
dbserver	Optional settings for remote database connection.
dbport	
memcache	Optional connection string(s) for memcached. Multiple servers are separated by spaces.
allowed_hosts	Space-separated list of address permitted to access the server. Wildcards can be used as in the ALLOWED_HOSTS docs.
server_email	"From" e-mail address for server-generated error mails.
email_host	Details for SMTP server. User and password are optional.
email_port	
email_host_user	
email_host_password	
alert_email	Where error messages are sent.
return_email	The "From" address on e-mail sent by mastr-ms.
logs_to_email	E-mail address to receive datasync client log notifications.
keys_to_email	E-mail address to receive datasync key upload notifications.
registration_to_email	E-mail address to receive registration requests.
repo_user	Mastr-MS will attempt to change ownership of data files to this user and group.
repo_group	
repo_files_root	Location of data files for experiments and quotes.
quote_files_root	
secret_key	Needs to be a secret random string, can be generated by a key generator program.

More advanced options appear in /etc/mastrms/settings.py. Any of the Django Settings can be changed in this file.

SELinux and Mastr-MS

Mastr-MS does not yet ship with a SELinux policy (issue MAS-21). For Mastr-MS to function correctly on a CentOS server, SELinux must be disabled.

The CentOS wiki contains instructions on how to disable SELinux.

2.1.3 Upgrading to a new version

New versions of Mastr-MS are made available in the CCG yum repository.

Warning: Some versions will require "database migrations" to update the database. While every care is taken to ensure smooth upgrades, we still advise to make a backup of the database before upgrading. This can be done with a command such as:

```
su - postgres -c "pg_dump mastrms | gzip > /tmp/mastrms-$(date +%Y%m%d).sql.gz"
```

Before upgrading, please check the *Release Notes* for any special information relating the new version.

Install the Mastr-MS RPM, replacing X.X.X with the desired version:

```
sudo yum install mastrms-X.X.X
```

Run Django syncdb and South migrate:

```
sudo mastrms syncdb
sudo mastrms migrate
```

2.1.4 Testing

After changing the configuration or upgrading, start/restart the web server with:

```
service httpd restart
```

Mastr-MS is available at https://your-web-host/mastrms/. A login page should be visible at this URL.

2.1.5 Administration

There are two levels of administration necessary for Mastr-MS.

- **Management**

  This involves administrating users, projects, quotes, experiments, etc. The URL for management is the normal Mastr-MS address, but only users who are in the admin group can see the interface.

  https://your-web-host/mastrms/

  The management interface is described in *Using the System*.

- **Django Admin**

  This involves manipulation of database objects to configure the data sync system. Only admin users can access the address:

  https://your-web-host/mastrms/repoadmin/

  The Django Admin site can also be accessed from *Dashboard* → *Repository* → *Admin*.

Data Sync Node Client Configuration

Configuration of a new site is done by adding a *Node client* using the **Django Admin**. The fields should be set as follows.

Field	Description
Organisation name	These values determine how the node is visible in the data sync client.
Site name	
Station name	
Default data path	This should be a subdirectory of `$DATADIR` (see *Data Repository and Permissions*).
Username	This should be the data sync user (see *User Creation*).
Hostname	The hostname or IP address of the Mastr-MS server.
Flags	This controls the options the data sync client will pass to rsync. They should always be set to `--protocol=30 -rzv --chmod=ug=rwX`.

Instrument Method Configuration

Before runs can be created, an *Instrument method* must be created using the **Django Admin**. At present, the Instrument Method object isn't used, but it must be set. The fields should be set as follows.

Field	Description
Title	Default Method
Method path	A folder path on the lab machine, e.g. `D:\mastrms`
Method name	Default Method
Version	1
Creator	*Your own username*
Template	CSV
The other fields	*Blank*

Standard Operating Procedure Documents

If you would like to make SOP documents available for viewing, you can create objects in the Django Admin within the Repository / Standard operation procedures page.

Once the documents are uploaded, they can be attached to experiments and viewed through the Experiment Sample Preparation screen.

SSH Key Management

When the data sync clients hit *Send Key*, it sends the client's public key via a HTTP post to a URL at the Mastr-MS site, and a view handles this, saving it to the `publickeys` directory on the server. It then sends an e-mail to the admins configured for the site, telling them that a new key has been uploaded, and they should append it on to the `authorized_keys` for the data sync user.

To install the key, run:

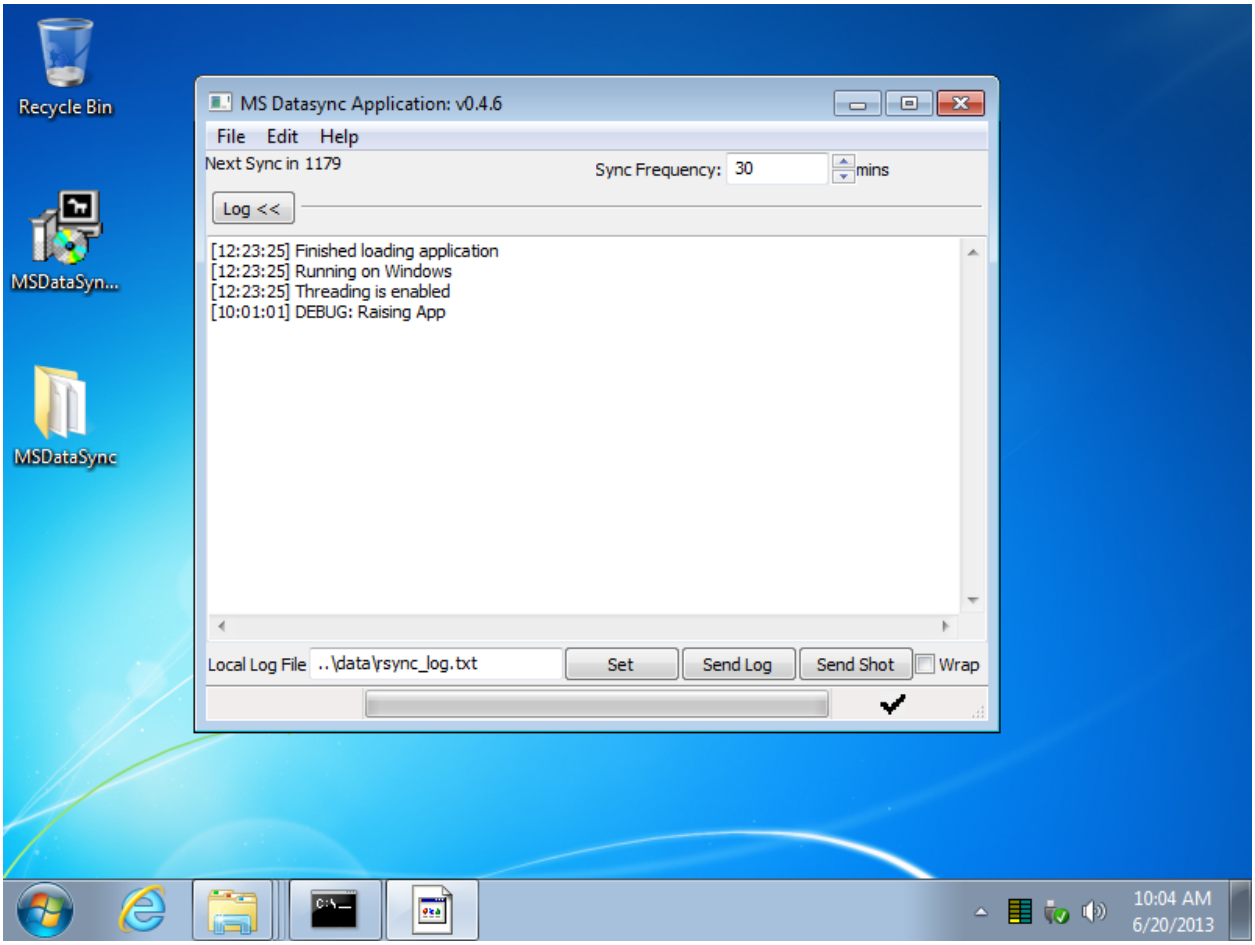
```
cat $DATADIR/files/publickeys/$ORG.$SITE.$STATION_id_rsa.pub \
    >> /home/$SYNCUSER/.ssh/authorized_keys
```

(Replace `$DATADIR`, `$SYNCUSER` and `$ORG.$SITE.$STATION` with your actual settings and the information from the e-mail.)

Once the key is added, the client should be able to "Handshake" with the server (see *Configuration*).

If the key isn't working, try checking the authorized_keys permissions.

2.2 Data Sync Client



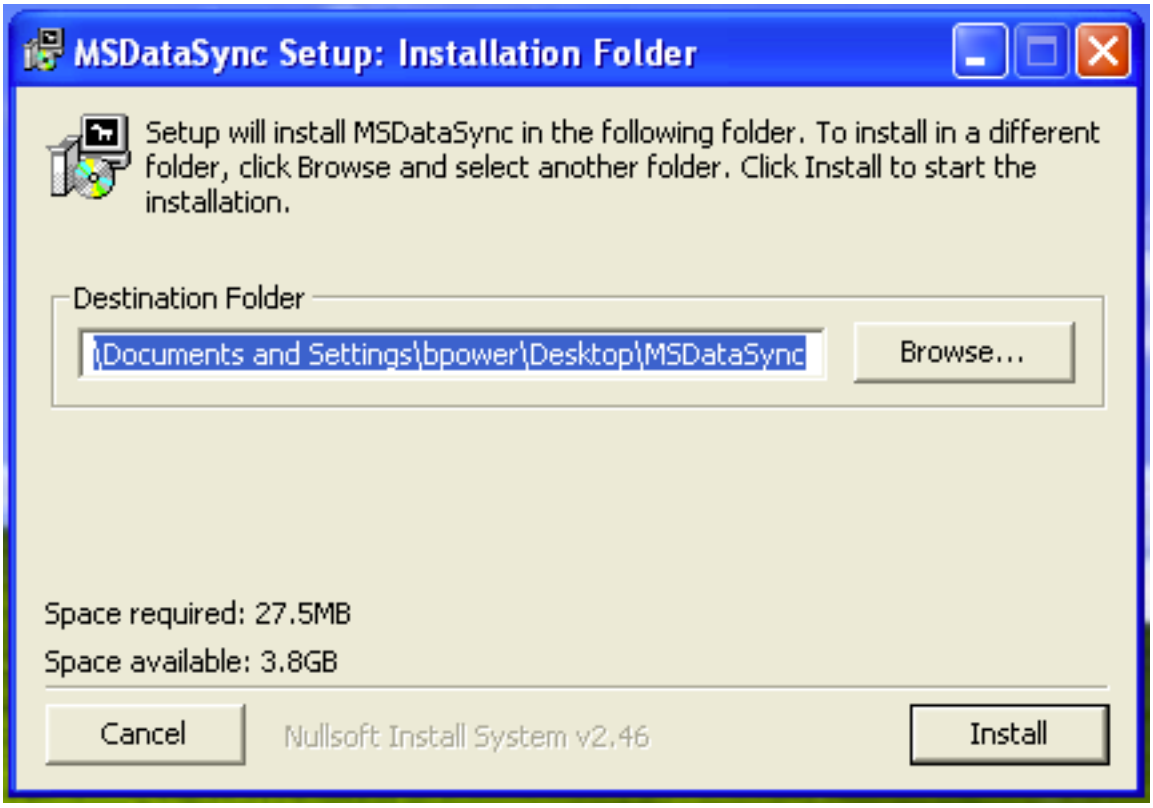
The Mastr-MS data sync client should be installed on the Windows machine which is connected to the lab instrument. It runs in the background and uploads experiment data as it is produced by the lab instrument software.

The data sync client includes the well-known rsync tool and uses it to transfer data.

2.2.1 Installation

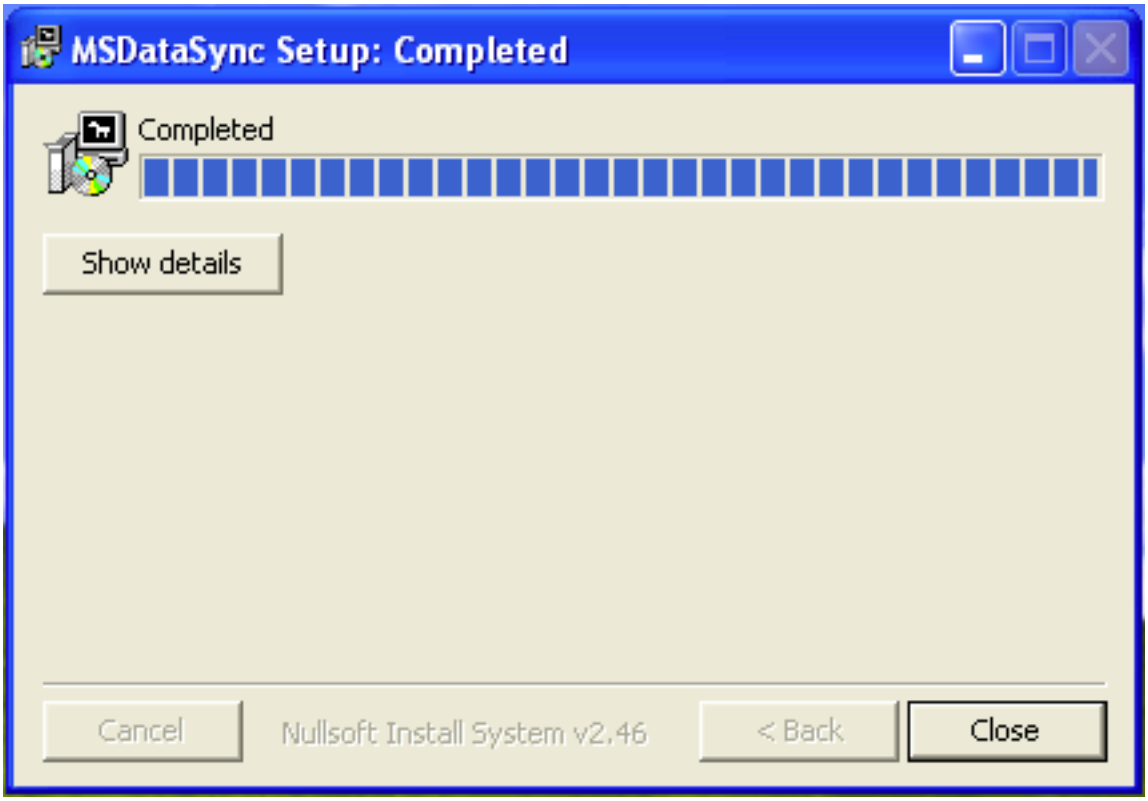
The Mastr-MS client is packaged as a Windows installer .EXE file. The latest version should be downloaded from:

http://repo.ccgapps.com.au/ma/



By default the client will be installed to the MSDataSync folder on the desktop.

During installation, a SSH key will be created in `C:\Users\USER\Desktop\MSDataSync\id_rsa.pub` (depends on chosen install folder). This will be used for authentication to the MastrMS repository.

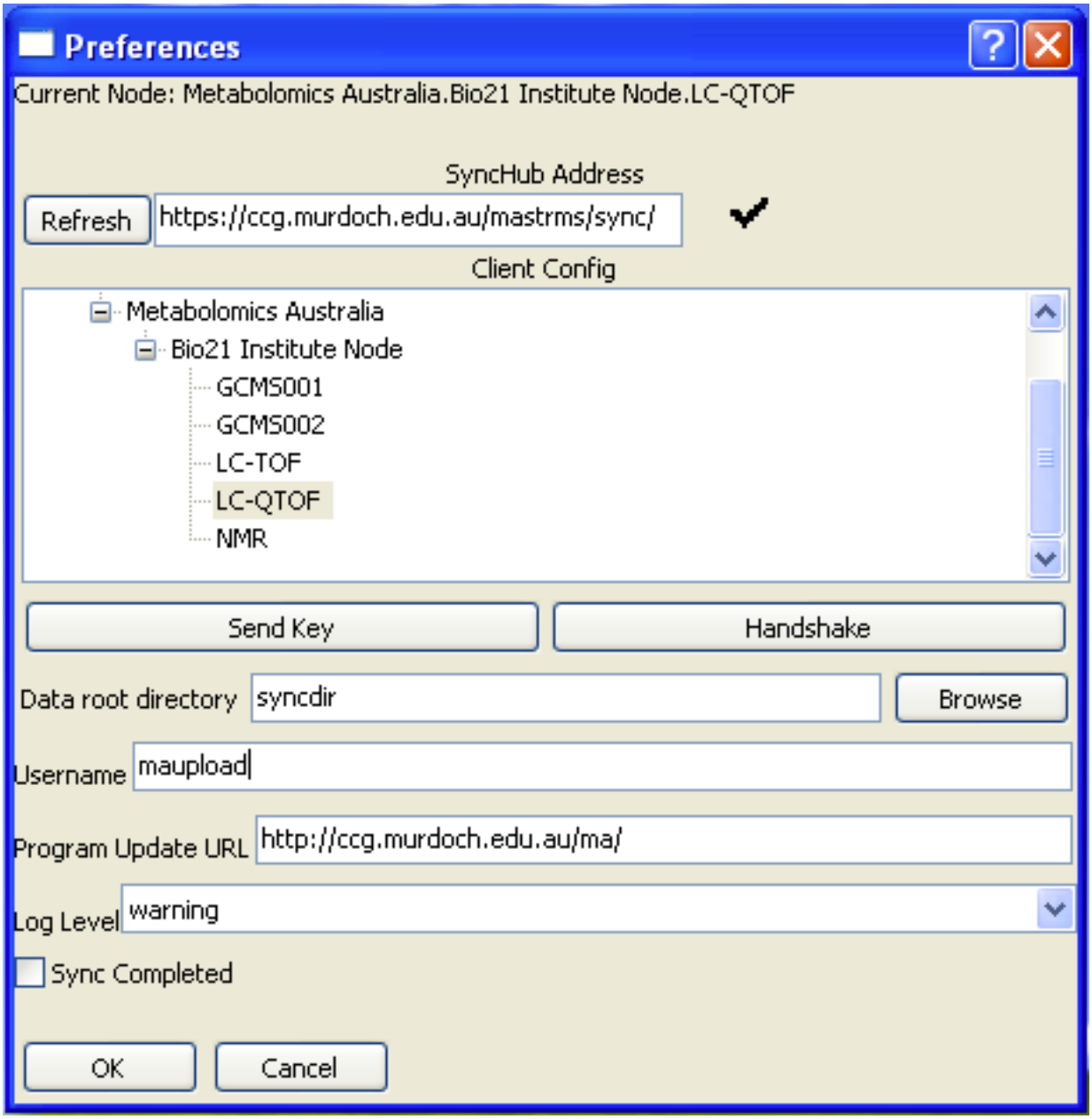


2.2.2 Starting

To start the data sync client application, open the installation folder and double click `main.exe`.

2.2.3 Configuration

Once installed, some configuration is required before the program can be used to sync data. To configure the data sync client, visit the *Edit* → *Preferences* menu.



Setting the station identifier

The SyncHub address is the web address of the Mastr-MS installation with `sync/` added on the end. The lab machine needs to be able to access the SyncHub address over the network.

After entering the SyncHub address, click the *Refresh* button and then select the appropriate site. The list is categorized by Organisation, then Site/Building/Division, then the machine name.

If your machine is not on the list it will need to be set up on the server first (see *Data Sync Node Client Configuration*).

Choosing the data sync folder

The next step is to set the location for the client to read data from. Everything under the chosen directory will be analysed to see if it is part of an experiment involving the target machine. Use the *Browse* button to select the folder containing your data.

Setting the username

The username should be the user which was created when Mastr-MS was set up (see *User Creation*). It's used for rsyncing the files. If in doubt, ask your system administrator or contact CCG.

Sending the public key

Click *Send Key* to upload the generated SSH key. The Mastr-MS server administrator will be e-mailed the key and must then install it (see *SSH Key Management*). Syncing will not be possible until the key is installed on the server. If the key was successfully sent, the main window will show "Key send response: ok" in the log.

After the administrator has added the key, click the "Handshake" button to test the connection. A black command console box will appear and ask you to type 'yes' or 'no' to confirm that you want to connect to the server. Type *yes* to proceed. This is a one time operation. It is a good way to test that public key authentication is working and gets the "Do you want to trust this host?" prompt out of the way.

Performing a manual sync

Click the OK button to save your changes.

The program will periodically sync according to the configured interval. However, as a test, first try a manual data sync.

Choose *File* → *Check Now*.

Depending on the contents of the sync directory, and the state of the experiments on the website, files may or may not be synced to the server.

Setting the sync frequency

You can set the frequency for syncing in minutes. The default is 30 minutes, meaning that the client attempts to sync data to the server every 30 minutes.

2.2.4 Set and forget

From now on, the client should run automatically.

If you close the client by clicking the [X] in the top corner of the window, or minimise it with *File* → *Minimize*, it will still be running in the background. You can tell if the client is running if you see this symbol in the system tray:

To re-open the client window, double click the icon, or right click and select *Open Main Window*.

To fully quit the program you must choose *File* → *Quit*.

2.2.5 Program Updates

Clicking *File* → *Program Updates* will check the server to see if any new updates are available. If a new version is available it will be downloaded and applied, and the application must be restarted for the updates to take effect.

2.2.6 Troubleshooting

If there are problems syncing, these will often appear as errors in the log. The log can sometimes reveal the source of the error. It is shown in the main window of the data sync application.

For troubleshooting purposes, the log and/or a screen capture of the client application can be sent to the server administrator using the *Send Log* and *Send Shot* buttons.

2.3 Using the System

The experiment design and sample management features of Mastr-MS are accessed through a web browser.

2.3.1 User Registration

The preferred way to create users is to have the user request an account. This can be done from the Login screen.

When someone requests an account, they are added to the Pending user list, and the registration contact is e-mailed. This e-mail address is configured with the `REGISTRATION_TO_EMAIL` setting (see *Django Settings File*).

The user can then be enabled by the "Admin Requests" screen.

Tip: User management is described in the *Admin screencast: Accepting/Rejecting users*.

An administrator user can also create users through the *Users* section of the Django Admin interface (see *Administration*). When using this interface, be careful to ensure that the username and e-mail address are the same, and all users are at least members of the *User* group.

2.3.2 User Roles

Users can have different roles which determine what they can see and change in the system.

- **Administrators** can do see all data and perform any action in the Mastr-MS system. Such users also have permission to edit objects in the *Django Admin* (see *Administration*).
- **Node Reps** can edit the list of nodes and organisations. They can also grant user account requests and edit users in their node.
- **Mastr Administrators** can edit users and edit all aspects of projects and experiments.
- **Project Leaders** can edit users and edit all aspects of projects and experiments.
- **Mastr Staff** can edit all aspects of projects and experiments.

If users do not have any of these roles, then they are considered to be **Clients**. Clients can only download experiment files.

2.3.3 Node Management

Nodes are a way of managing groups of users and directing quote requests to the correct group.

Users can view and answer quote requests sent to their node.

The list of nodes can be edited through the *Admin* → *Node Management* screen.

The pre-configured nodes can be deleted and replaced with one or more nodes specific to your site.

2.3.4 Experiments

Experiments are grouped in *Projects* which have a designated client user, as well as optional project manager users.

2.3.5 Rule Generators

Rule generators control the addition of QC, blank and sweep components to experiment runs.

These components can be added at the beginning, end, or for every *n* samples.

If the *Apply Sweep Rule* option is ticked for the rule generator, the run builder will add a sweep component after each sample unless it is a blank or sweep.

Rule generators cannot be changed once in use, in order not to corrupt previously run experiments. You can however clone rule generators or create new versions. Unwanted rule generators can be disabled.

At least one rule generator must exist before a run can be created.

2.3.6 Samples

An experiment run requires information about the samples that will constitute that run. Samples are typically generated from sample classes or imported from CSV files.

2.3.7 Runs

Runs are created by selecting a set of samples and clicking *Add Selected Samples to Run*. Previously created runs can also be cloned into new runs.

Once you are happy with the run parameters and samples, you can start the run by clicking the *Generate Worklist* button.

This will display a CSV worklist in a new browser window. The worklist can then be copied and pasted into the lab instrument software.

Note: One important thing to note is that any web browser popup blocker must be disabled for the Mastr-MS web site.

Otherwise you will not be able to see CSV worklists, which are opened in a new window.

Most new web browsers have popup blockers enabled by default, but certain web addresses can be excluded from blocking.

After generating the worklist, the run is maked as “In Progress”.

2.3.8 Data Sync

The Mastr-MS data sync client periodically uploads new data found on the lab machine into the data management system. The data sync client is told by the Mastr-MS server which filenames to look for.

Once the samples data files are uploaded, the client marks that sample as complete. The percentage of complete samples is shown in the progress bar on the *Runs* page.

When all samples in a run are completed, the run is marked as complete.

2.3.9 Getting the data

Sample data can be seen from the *Files* page of the experiment. PBQCs, QCs and sweeps can be downloaded from the *Runs* page.

To download the data files, expand the *Files* tree and click the filename.

2.3.10 Importing metadata from file

Run and sample data can be imported from a CSV file. CSV files can be created with a text editor or spreadsheet program and usually end with the file extension `.csv`.

Sample CSV Import

To import samples, first create a CSV file which looks something like this:

```
Label,Weight,Comment
smplbla,1.234,Comment about the sample
smplblb,2.345,Another sample
```

Click the *Upload CSV file* button on the *Samples* page, then choose the file.

If you would like to update existing samples, first download the sample CSV file (which contains an `ID` column), edit the fields, then upload it again.

Completed Run Sample Data Import

If you have sample data on the lab machine which was not created through a worklist designed in Mastr-MS, and would like to add it to the data management system, you can import it from the *Runs* page.

Set up a CSV file with a single column `Filename` and list the filenames for the run.

Then click the *Capture Completed External Run* button and upload this file. You also need to specify a name for this run and which machine the data is on.

2.4 Release Notes

This page lists what changed in each Mastr-MS version. For instructions on how to upgrade the server, see *Upgrading to a new version*. For instructions on how to upgrade the datasync client, see *Program Updates*.

2.4.1 1.12.3 (16th June 2015)

Bug fix release.

- #4: Drop-down lists sometimes show a number instead of the field value
- #5: Times out when downloading large files
- #13: Browser times out when a request to zip up large files is sent

2.4.2 1.12.2 (6th March 2015)

Brown paper bag release.

- #12: RPM for 1.12.1 was faulty

2.4.3 1.12.1 (6th March 2015)

Bug fix release. Contains database migrations.

- #10: Disable cascading delete for nullable foreign key fields
- #11: Investigations carried over between projects

2.4.4 1.12.0 (5th March 2015)

Bug fix release. Contains database migrations.

Mastr-MS stops using JIRA and starts using the BitBucket issue tracker.

- #3: Project manager drop-down appears empty
- #7: Sends out e-mails with "Madas"
- #9: Deleting investigation also deletes experiment

2.4.5 1.11.4 (8th December 2014)

Bug fix release.

- [MAS-76] - Migration from empty database broke

2.4.6 1.11.3 (19th November 2014)

Bug fix release.

- [MAS-74] - Put investigation name in ISA-Tab files

2.4.7 Client 0.4.9 (22nd October 2014)

Datasync client bug fix release.

- [MAS-75] - Datasync client doesn't support quoting in rsync config

2.4.8 1.11.2 (22nd October 2014)

New feature release.

- [MAS-73] - Provide a direct URL to the registration screen

2.4.9 1.11.1 (1st September 2014)

Bug fix release.

- [MAS-72] - Always output JSON format from API

2.4.10 1.11.0 (28th August 2014)

New feature release. Contains database migrations.

- [MAS-69] - ISA-TAB export: Put experiment samples in the same investigation if they are the same samples
- [MAS-71] - Update Django to 1.6.6

2.4.11 1.10.1 (7th August 2014)

Bug fix release. Contains database migrations.

- [MAS-68] - Handle migration of users with no e-mail address

2.4.12 1.10.0 (7th August 2014)

New feature release.

- [MAS-66] - Increase length of usernames
- [MAS-67] - Enable user logging in production again

This release contains database migrations which need to be run after upgrading the RPM. User e-mail addresses must be unique now. The migration process will change duplicate e-mail addresses. If any e-mail addresses were changed, it will say so. You must then clean up those users from the admin page.

If the database migration fails due to an error with the `userlog_*` tables, just drop them and try the migration again:

```
# mastrms dbshell
DROP TABLE "userlog_loginlog";
DROP TABLE "userlog_failedloginlog";
```

2.4.13 1.9.4 (23rd June 2014)

Bug fix release.

- [MAS-65] - Change title MA LIMS to MASTR-MS

2.4.14 1.9.3 (5th June 2014)

Bug fix release.

- [MAS-64] - Make ISA-Tab output validate with isatools validator

2.4.15 1.9.2 (29th May 2014)

New feature release.

- [MAS-61] - Produce ISA-Tab study and assay files
- [MAS-62] - Update Django to 1.6.4
- [MAS-63] - Improve environment variable config code

2.4.16 1.9.1 (1st May 2014)

New feature release.

- [MAS-61] - Add ISA-Tab fields for study and assay

2.4.17 1.9.0 (13th Mar 2014)

New feature release.

- [MAS-59] - ISA-TAB format export

2.4.18 1.8.2 (20th Feb 2014)

Bug fix release. You can now put multiple space-separated values for `allowed_hosts` and `memcache` in `/etc/mastrms/mastrms.conf`.

- [MAS-55] - Missing samples labels etc when cloning experiments
- [MAS-56] - CSV upload broke with python27-mod_wsgi
- [MAS-57] - Client code using extjs grid is saving null sample weights
- [MAS-60] - Settings: multiple memcache servers and allowed hosts

2.4.19 1.8.1 (31st Jan 2014)

Bug fix release. More options were added to the default config files.

- [MAS-54] - Add wider menu of settings in mastrms.conf

2.4.20 1.8.0 (30th Jan 2014)

New feature and bug fix release.

Mastr-MS now requires the IUS repo. It can be added according to the instructions in *Yum repository setup*. If you get dependency errors on installation, it is probably because the ius-release RPM isn't installed.

Note: In this version the format of the config file has changed. You will need to manually update the settings.

The settings are no longer stored in `/etc/ccgapps/appsettings`. They are now in `/etc/mastrms`. After installing the RPM, edit `/etc/mastrms/mastrms.conf` and copy in just the listed settings from `/etc/ccgapps/appsettings/mastrms.py`.

After restarting the web server and checking that it works, the old settings file can be moved into a backup location.

- [MAS-52] - Switch RPM to new build method
- [MAS-53] - Fix file extension in worklist

2.4.21 1.7.0 (19th Dec 2013)

New feature release

- [MAS-49] - General File Extension (Issue 132)
- [MAS-50] - Renaming files in file manager

2.4.22 1.6.2 (26th Nov 2013)

Bug fix release

- [MAS-45] - Put run QC data as a subfolder of experiment data

2.4.23 1.6.0 (25th Nov 2013)

New feature release

Bug fixes

- [MAS-48] - CSV import – should ignore empty weight values

Improvements

- [MAS-45] - Put run QC data as a subfolder of experiment data
- [MAS-47] - Allow creation of own folders within experiment files

2.5 Credits

2.5.1 ABF Team

Project Director Prof. Matthew Bellgard

Project Leader Adam Hunter

Software Developers

Rodney Lorrimar, Grahame Bowland, Maciej Radochonski, Brad Power, Tamas Szabo, Nick Takayama, Andrew Macgregor

System Administrator David Schibeci

2.5.2 MA Team

MA Informatics Group Leader Dr. Saravanan Dayalan

Analytical Lead Mr. David De Souza

IT Manager Mr. Thu Nguyen

2.6 Mastr-MS Development

This document is intended for developers who wish to change Mastr-MS or see how it works.

Mastr-MS is Django web application and works with both PostgreSQL and MySQL.

2.6.1 Source Code

The Mastr-MS code is hosted at BitBucket.

https://bitbucket.org/ccgmurdoch/mastr-ms

Git is required to check out the code.

2.6.2 Issue Tracking

All new bugs/feature requests should be submitted to the issue tracker:

https://bitbucket.org/ccgmurdoch/mastr-ms/issues/new

If all significant changes have a ticket associated, then release notes can be accurately generated.

Mastr-MS was originally hosted on Google Code, where there still remains a small bug list:

http://code.google.com/p/mastr-ms/issues/list

2.6.3 Development Environment

We develop on Ubuntu. Installing the following packages is recommended:

```
sudo apt-get install postgresql-server \
    ipython python-pip python-sphinx \
    python-wxgtk2.8 xvfb xserver-xephyr
```

Packages helpful but not really required:

```
python-django \
python-werkzeug python-django-extensions \
python-selenium chromium-chromedriver \
```

2.6.4 Building a CentOS RPM

The RPM build is unlikely to work unless done under CentOS. Assuming the Mastr-MS source code is checked out at `/usr/local/src`, you can build an RPM in more-or-less the normal way by running these commands:

```
CCGSOURCEDIR=/usr/local/src
export CCGSOURCEDIR
cd $CCGSOURCEDIR && rpmbuild -bb centos/mastrms.spec
```

The spec file requires `CCGSOURCEDIR` to be set. It will download all the python dependencies with `pip`, create the RPM, and output it to `~/rpmbuild/RPMS` (or the location you have configured in `~/.rpmrc`).

2.6.5 How to build the sync client

The support libraries and binaries are in the `mdatasync_client/client/supportwin32` directory.

You should be able to build the client on a 32-bit Windows XP box, by installing these resources as described below.

Initial Setup of build environment

1. Copy client and supportwin32 over to the windows machine.
2. Install Python27 (utf8 encoding for json is broken in earlier versions)
3. Install wx for python 2.7
4. Install py2exe for python 2.7
5. Install NSIS
6. Install 7Zip
7. Extract the `EnvVarUpdate` extension
8. Copy the extension into `c:\program files\nsis\include`
9. Open a shell
10. Change dir to the `client` dir

Building the code

First, make sure you have updated the version number in `version.py` to be unique, and sequentially higher than previous ones.

On Windows box, get the latest client dir. Run:

```
c:\Python27\python.exe setup.py bdist_esky
```

Then unzip the build you just did (in `dist/`) so that the files are available to the installer.

If you then ran the `nsi` file (right click on the file and choose 'Compile with NSIS'), it would make an installer, using the build you just did (in `/dist`) as a source. So that is how you make an installer.

If you want to publish just the update, you take the `.zip` that is generated in the `dist` directory, and `scp` it to the distribution URL. Currently, this is on S3, under http://repo.ccgapps.com.au/ma/

As long as the `version.py` version was incremented, a new version will be available to esky.

Errors

When building with Esky, if you get the message `cannot access ../main.py. The file is in use by another process/`, this probably means your python code fails a syntax check or has some other runtime error.

2.6.6 Running Tests

Testing requirements:

- wxpython
- Xvfb

- selenium
- chrome/firefox webdriver
- splinter
- dingus
- python xvfbwrapper

Command to run:

```
./manage.py test --exclude=yaphc --exclude=esky --exclude=httplib2
```

2.6.7 Testing

Mastr-MS contains some system tests as well as unit tests. The current test classes are:

- `mastrms.mdatasync_client.client.test.tests.BasicClientTests`
- `mastrms.mdatasync_client.client.test.tests.DataSyncServerTests`
- `mastrms.mdatasync_server.tests.SyncTests`
- `mastrms.mdatasync_server.tests.AdminTests`

The WxPython client is tested using the `TestClient` class. `TestClient` runs the client in a thread and allows "clicking" on GUI buttons by calling the associated event handlers.

To generate sample data files like the lab instrument software would create, use the `Simulator` class. Simulator can be run as a standalone WxPython GUI, or operated programmatically through `Worklist`.

For testing you usually don't want to do actual file transfers, but you do want to check that `rsync` was called. For this, we put a mock command into the `PATH` which can be queried from the test cases.

2.6.8 Documentation

The documentation is in Sphinx format under the `docs` subdirectory of the source. To build it, simply run:

```
make html
```

2.6.9 Product Overview

This list, written by Brad, may be useful in understanding the system.

Goals of MASTR

- To provide a web based tool for experimental design, sample metadata configuration, and sample data acquisition.
- To enable researchers and scientists from geographically separate institutions to work together on experiments, analysis, and to be able to share results and outcomes.
- To enable institutions to provide quotes for analysis work to third-parties, with automatic linkage through to the relevant projects and experiments.

Features of MASTR

- User / Group administration
- **Experimental design, catering for:**
  - User roles and access control
  - Sample origin metadata
  - Sample timeline and treatment metadata
  - Sample tracking
  - Sample information import / export via CSV
  - Standard Operating Procedure upload
  - Run creation, generating worklists for the purposes of instrument automation.
  - Fully customisable rules system for worklist generation
  - Sample blocks, order, randomisation, and solvents/blanks can be specified as a programmable template.
  - Worklist rulesets can be rolled out per individual, or shared with groups or the entire institution
  - Rulesets can be branched and cloned
  - Runs and Experiments can also be cloned for convenience.
- **Data acquisition, consisting of:**
  - A program which runs on the computer connected to the instrument which processes MASTR worklists
  - The program will check periodically for filesets related to experiment runs being performed for MASTR
  - The sample data is compressed and uploaded to the MASTR-connected storage, and optionally archived on the client machine.
  - The sample data is then securely available to relevant users through the MASTR web interface for viewing or download.
  - Full end-to-end data acquisition, from experimental design to sample file access.
- **Quote requests and tracking**
  - A system for third parties to request quotes for analysis work of any of the institutions in MASTR
  - Institution Administrators or Node Representatives can review the requests and service replies, with optional PDF attachments.
  - Full quote event history is maintained.
  - Quotes in the system can be linked to Projects / Experiments
- **Modern technologies**
  - Able to be accessed in all major web browsers
  - Lightweight and powerful UI
  - Open data formats and transports used (rsync, json)
  - Open Source code repository

2.6.10 Auto-generated Documentation

Tests

This is a list of testing classes. For complete information, please consult the source code.

If this page is empty, it means that the `sphinx.ext.autodoc` generated documentation failed to build.