



PMR2 Tutorial

Authors:

Dougal Cowan
Randall Britten

1 Contents

1	Contents	1
2	About this tutorial	2
3	PMR2 concepts.....	2
4	Working with the PMR2 web interface	3
4.1	Model listings	3
4.2	Searching the repository	4
5	Working with PMR2 using Mercurial.....	5
5.1	Registering an account and logging in	5
5.2	Cloning an existing workspace.....	5
5.3	Mercurial username configuration.....	7
5.4	Making changes to workspace contents	8
5.5	Committing changes	9
5.6	Pushing changes to the repository	10
6	Making an exposure using "roll-over"	11
7	Become a Mercurial power user	13
7.1	Pulling and merging	13
7.2	Sharing changes on a local network	14
8	Creating a new workspace	14
8.1	Adding files to the new workspace.....	15
8.2	Making an exposure from scratch	16

Auckland Bioengineering Institute PMR2 Tutorial

2 About this tutorial

The CellML repository is powered by software called Physiome Model Repository 2 (PMR2). PMR2 currently relies on the distributed version control system Mercurial (Hg), which allows the repository to maintain a complete history of all changes made to every file it contains. This tutorial demonstrates how to work with the repository using TortoiseHg, which provides a Windows explorer integrated system for working with Mercurial repositories.

Brief mention of the equivalent command line versions of the TortoiseHg actions will also be mentioned, so that these ideas can also be used without a graphical client, and on Linux and similar systems. These will be denoted by grayed boxes like this.

You will need to have TortoiseHg, OpenCell, and access to both a live and a non-live PMR2 model repository instance in order to go through this tutorial.

TortoiseHg: <http://tortoisehg.bitbucket.org/>

OpenCell: <http://www.cellml.org/>

3 PMR2 concepts

PMR2 and the CellML model repository use a certain amount of jargon - some is specific to the repository software, and some is related to distributed version control systems (DVCSs). Below are basic explanations of some of these terms as they apply to the repository.

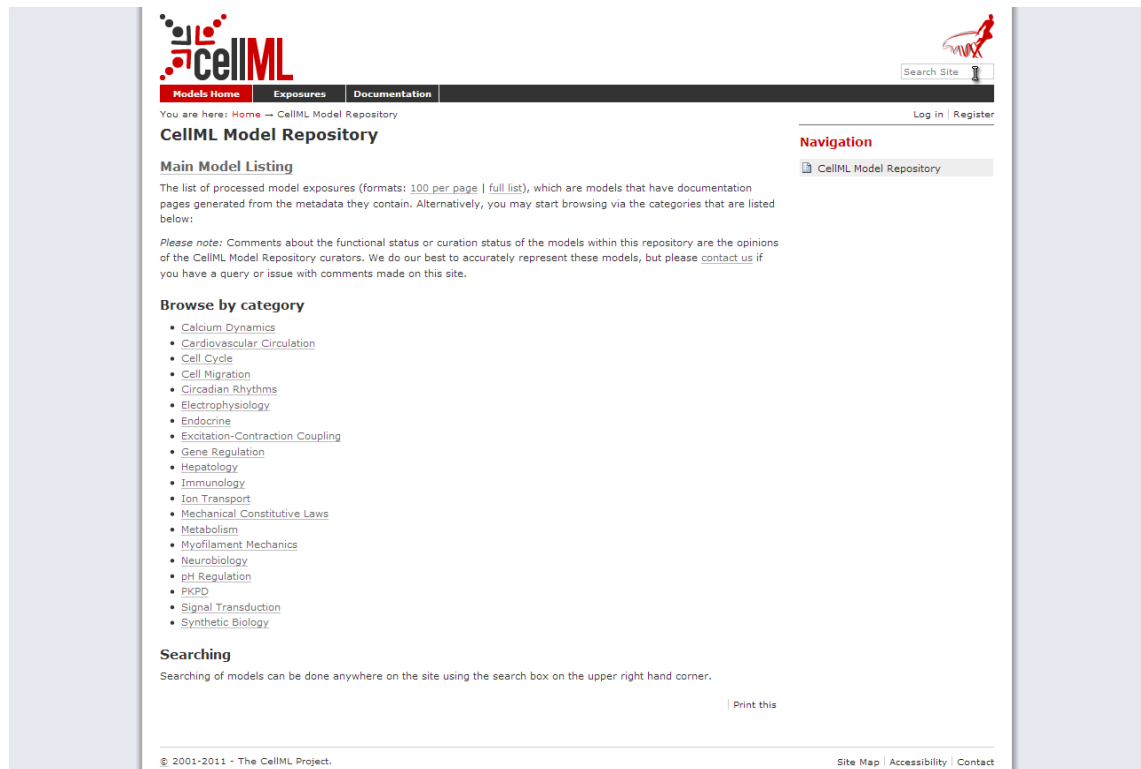
Workspace: A container (much like a folder or directory on your computer) to hold the files that make up a model, as well as any other files such as documentation or metadata, etc. In practical terms, each workspace is a Mercurial repository.

Exposure: An exposure is a publically viewable presentation of a particular revision of a model. An exposure can present one or many files from your workspace, along with documentation and other information about your model.

The Mercurial DVCS has a range of terms that are useful to know, and definitions of these terms can be found in the Mercurial glossary: <http://mercurial.selenic.com/wiki/Glossary>.

4 Working with the PMR2 web interface

This part of the tutorial will teach you how to find models in a live instance of PMR2, commonly called the "CellML repository" (<http://models.cellml.org>), how to view a range of information about those models, and how to download models. The first page in the repository consists of basic navigation, a link to the main model listing, a search box at the top right, and a list of model category links as shown below.

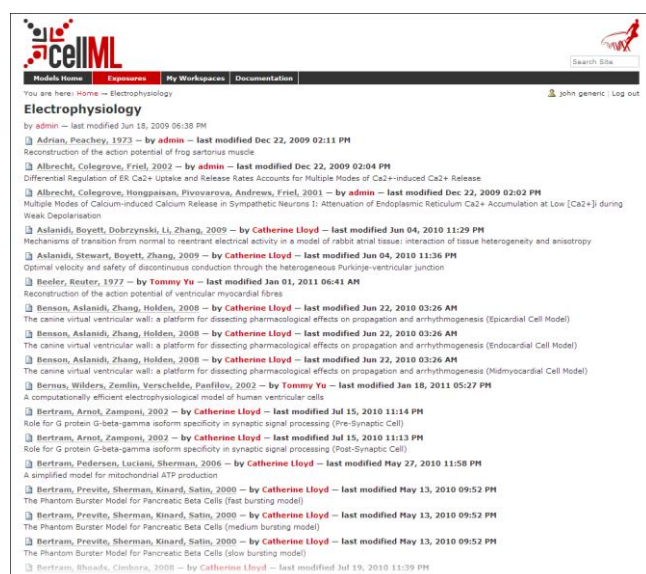


The screenshot shows the CellML Model Repository homepage. At the top left is the CellML logo. To its right is a search box labeled "Search Site". Below the logo is a navigation bar with links for "Models Home", "Exposures", and "Documentation". The main heading is "CellML Model Repository". Below this is a "Main Model Listing" section with a paragraph of text and a "Please note" section. To the right of the main content is a "Navigation" section with a link to "CellML Model Repository". At the bottom left is a copyright notice "© 2001-2011 - The CellML Project." and at the bottom right are links for "Site Map", "Accessibility", and "Contact".

4.1 Model listings

Clicking on the main model listing or any of the category listings will take you to a page displaying a list of exposed models in that category. Click on electrophysiology for example, and a list of over 100 exposed models in that category will be displayed, as shown here.

Clicking on an item in the list will take you to the exposure page for that model.



The screenshot shows the "Electrophysiology" category page in the CellML Model Repository. The page has a navigation bar with "Models Home", "Exposures", "My Workspaces", and "Documentation". Below the navigation bar is a list of model entries, each with a title, author(s), and date. The entries include:

- by admin - last modified Jun 15, 2009 05:38 PM
- Adrian, Peachley, 1973 - by admin - last modified Dec 22, 2009 02:11 PM
- Reconstruction of the action potential of frog sartorius muscle
- Albrecht, Colegrove, Friel, 2002 - by admin - last modified Dec 22, 2009 02:04 PM
- Differential Regulation of ER Ca²⁺ Uptake and Release Rates Accounts for Multiple Modes of Ca²⁺-induced Ca²⁺ Release
- Albrecht, Colegrove, Hongpakkan, Pivovarov, Andrews, Friel, 2001 - by admin - last modified Dec 22, 2009 02:02 PM
- Multiple Modes of Calcium-induced Calcium Release in Sympathetic Neurons I: Attenuation of Endoplasmic Reticulum Ca²⁺ Accumulation at Low [Ca²⁺]_i during Weak Depolarisation
- Aslanidi, Boyett, Dobrzynski, Li, Zhang, 2009 - by Catherine Lloyd - last modified Jun 04, 2010 11:29 PM
- Mechanisms of transition from normal to reentrant electrical activity in a model of rabbit atrial tissue: interaction of tissue heterogeneity and anisotropy
- Aslanidi, Stewart, Boyett, Zhang, 2009 - by Catherine Lloyd - last modified Jun 04, 2010 11:36 PM
- Optimal velocity and safety of discontinuous conduction through the heterogeneous Purkinje-ventricular junction
- Beeler, Reuter, 1977 - by Tommy Yu - last modified Jan 01, 2011 06:41 AM
- Reconstruction of the action potential of ventricular myocardial fibres
- Benson, Aslanidi, Zhang, Holden, 2008 - by Catherine Lloyd - last modified Jun 22, 2010 03:26 AM
- The canine virtual ventricular wall: a platform for dissecting pharmacological effects on propagation and arrhythmogenesis (Epicardial Cell Model)
- Benson, Aslanidi, Zhang, Holden, 2009 - by Catherine Lloyd - last modified Jun 22, 2010 03:26 AM
- The canine virtual ventricular wall: a platform for dissecting pharmacological effects on propagation and arrhythmogenesis (Endocardial Cell Model)
- Benson, Aslanidi, Zhang, Holden, 2008 - by Catherine Lloyd - last modified Jun 22, 2010 03:26 AM
- The canine virtual ventricular wall: a platform for dissecting pharmacological effects on propagation and arrhythmogenesis (Midmyocardial Cell Model)
- Bernus, Wilders, Zembla, Verschuelden, Panfilov, 2002 - by Tommy Yu - last modified Jan 18, 2011 05:27 PM
- A computationally efficient electrophysiological model of human ventricular cells
- Bertram, Arnot, Zampanti, 2002 - by Catherine Lloyd - last modified Jul 15, 2010 11:14 PM
- Role for G protein G-beta-gamma isoform specificity in synaptic signal processing (Pre-Synaptic Cell)
- Bertram, Arnot, Zampanti, 2002 - by Catherine Lloyd - last modified Jul 15, 2010 11:13 PM
- Role for G protein G-beta-gamma isoform specificity in synaptic signal processing (Post-Synaptic Cell)
- Bertram, Pedersen, Luciani, Sherman, 2006 - by Catherine Lloyd - last modified May 27, 2010 11:58 PM
- A simplified model for mitochondrial ATP production
- Bertram, Preville, Sherman, Kinard, Sattin, 2000 - by Catherine Lloyd - last modified May 13, 2010 09:52 PM
- The Phantom Burster Model for Pancreatic Beta Cells (Fast bursting model)
- Bertram, Preville, Sherman, Kinard, Sattin, 2000 - by Catherine Lloyd - last modified May 13, 2010 09:52 PM
- The Phantom Burster Model for Pancreatic Beta Cells (medium bursting model)
- Bertram, Preville, Sherman, Kinard, Sattin, 2000 - by Catherine Lloyd - last modified May 13, 2010 09:52 PM
- The Phantom Burster Model for Pancreatic Beta Cells (slow bursting model)
- Bertram, Rhoads, Cimbrora, 2008 - by Catherine Lloyd - last modified Jul 19, 2010 11:39 PM

4.2 Searching the repository

You can search for the model that you wish to work on by entering a search term in the box at the top right of the page. Many of the models in the repository are named by the first author and publication date of the paper, so good search query might be something like "goldbeter 1991". A list of the results of your search will probably contain both *workspaces* and *exposures* - you will need to click on the workspace of the model you wish to work on. Workspaces can be identified because their links are pale blue and have no details line following the clickable link. In the following screenshot, the first two results are workspaces, and the remainder are exposures.

The screenshot shows a web interface with a navigation bar at the top containing 'Models Home', 'Exposures', 'My Workspaces', and 'Documentation'. Below the navigation bar, there is a search bar with the text 'goldbeter 1991' and a 'search' button. A message below the search bar asks if the user did not find what they were looking for and suggests using 'Advanced Search'. The main heading is 'Search results — 8 items matching your search terms', followed by a link to 'Subscribe to an always-updated feed of these search terms'. The search results are listed as follows:

- Goldbeter, 1991** (workspace link) by admin — last modified Jul 05, 2010 01:08 PM — Relevance: 100%
- Dupont Berridge Goldbeter 1991** (workspace link) by cmlloyd — last modified Jan 20, 2011 04:16 PM — Relevance: 99%
- A theory for controlling cell cycle dynamics using a reversibly binding inhibitor** (exposure link) A theory for controlling cell cycle dynamics using a reversibly binding inhibitor by cmlloyd — last modified Jul 07, 2010 04:04 AM — Relevance: 70%
- Signal-induced Ca2+ oscillations: Properties of a model based on (Ca2+)-induced Ca2+ release (Model B)** (exposure link) Signal-induced Ca2+ oscillations: Properties of a model based on (Ca2+)-induced Ca2+ release by cmlloyd — last modified Aug 26, 2010 10:38 PM — CellML Model — Relevance: 66%
- Signal-induced Ca2+ oscillations: Properties of a model based on (Ca2+)-induced Ca2+ release (Model A)** (exposure link) Signal-induced Ca2+ oscillations: Properties of a model based on (Ca2+)-induced Ca2+ release by cmlloyd — last modified Aug 26, 2010 10:38 PM — CellML Model — Relevance: 64%
- A minimal cascade model for the mitotic oscillator involving cyclin and cdc2 kinase (Model without MIRIAM annotations)** (exposure link) A minimal cascade model for the mitotic oscillator involving cyclin and cdc2 kinase by cmlloyd — last modified Jul 05, 2010 10:38 PM — Relevance: 53%
- A minimal cascade model for the mitotic oscillator involving cyclin and cdc2 kinase (Model with MIRIAM annotations)** (exposure link) A minimal cascade model for the mitotic oscillator involving cyclin and cdc2 kinase by cmlloyd — last modified Jul 05, 2010 10:38 PM — Relevance: 53%

Click on an exposure result to view information about the model and to get links for downloading or simulating the model. Click on workspaces to see the contents of the model workspace and the revision history of the model.

5 Working with PMR2 using Mercurial

This part of the tutorial will teach you how to clone a workspace from the model repository using a Mercurial client, create your own workspace, and then push the cloned workspace into your new workspace in the repository. We will be using a non-live temporary instance of the repository for this part of the tutorial, so there is no need to worry about messing up the contents of the real repository.

5.1 Registering an account and logging in

First, navigate to the temporary model repository at <http://bioeng77.bioeng.auckland.ac.nz:7380/pmr>.

The home page of the temporary repository is different to the main CellML model repository - it has no category listings, and instead it repeatedly lists the same exposure.

In order to make changes to models in the CellML repository, you must first register for an account. The "Log in" and "Register" links can be found near the top right corner of the page. Your account will have the appropriate access privileges so that you can push any changes you have made to a model back into the repository.

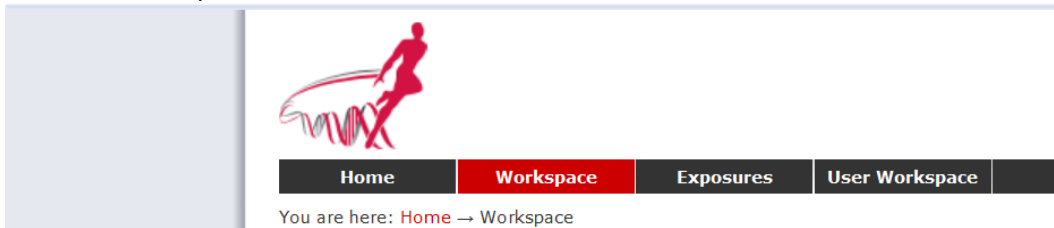
Click on the *Register* link near the top right, and fill in the registration form. Enter your username and desired password. You can now log in to the repository. This username and password are also the credentials you use to interact with the repository via Mercurial. Note that compared to registering for the real CellML repository, this process is somewhat streamlined for the purposes of this tutorial.

Once logged in to the repository, you will notice that there is a new link in the navigation bar, *User Workspace*. This is where all the workspaces you create later on will be listed. The *Log in* and *Register* links are also replaced by your username and a *Log out* link.

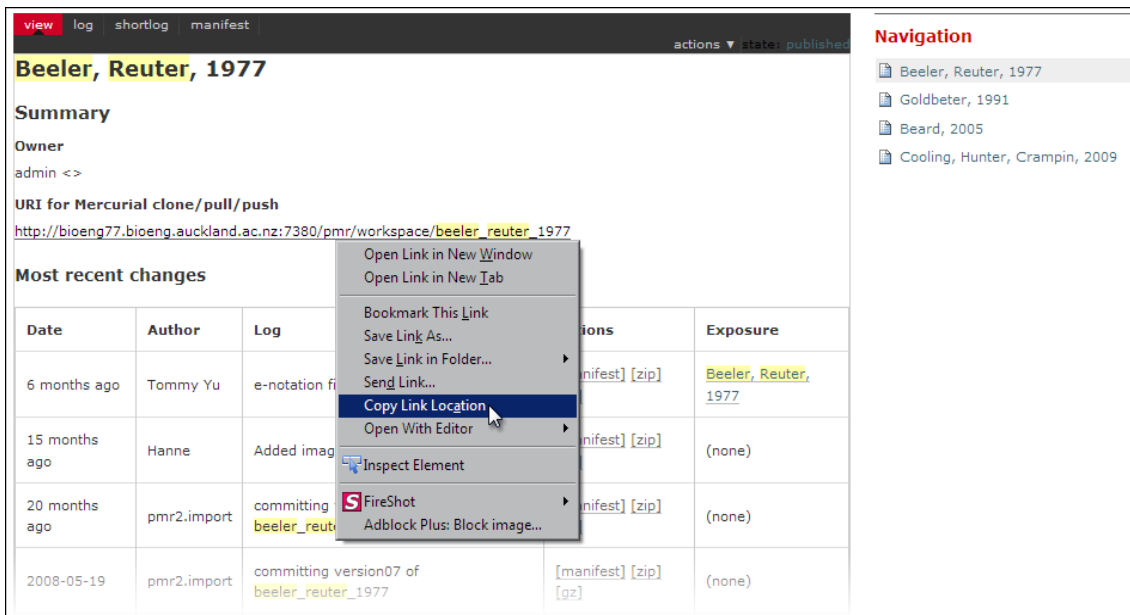
5.2 Cloning an existing workspace

It is essential to use a Mercurial client to obtain models from the repository for editing. The Mercurial client is not only able to keep track of all the changes you make (allowing you to back-track if you make any errors), but using a Mercurial client is the only way to add any changes you have made back into the repository.

You will be allocated your own workspace for the tutorial. It already has a revision history and some exposures. To see a list of all the workspaces, click on "Workspace". Then click on the workspace that has been allocated to you for the tutorial.

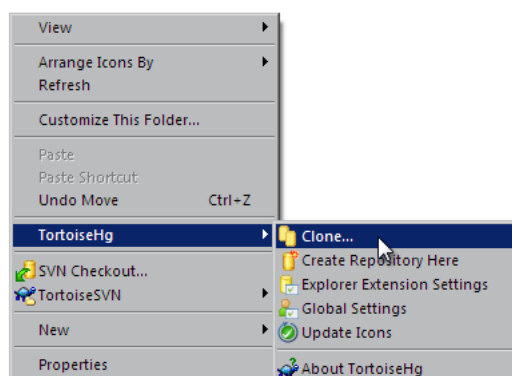


Now copy the URI for Mercurial clone/pull/push. This can be done in Firefox by right-clicking on the link and selecting *Copy Link Location* from the context menu.

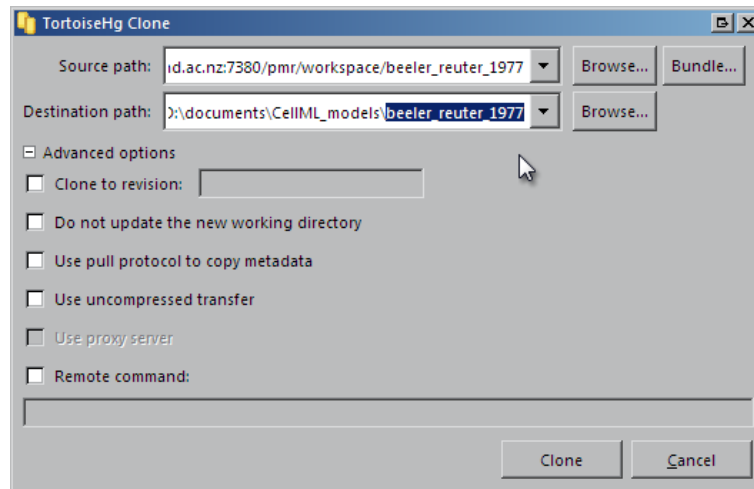


Next, use your Mercurial client to *clone* the workspace using the URI you just copied. Cloning the workspace will create a local repository containing a copy of the entire contents and change history of the cloned workspace in a folder on your computer.

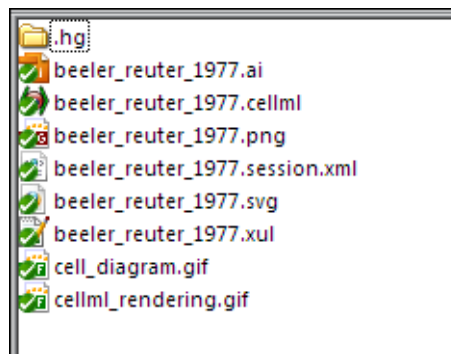
In Windows explorer, find the folder where you want to create the clone of the workspace. Then right click to bring up the context menu, and select *TortoiseHG > Clone...* as shown below:



Paste the copied URL into the *Source path*: area, and enter the directory name of the workspace (in this case `beeler_reuter_1977`) onto the end of the directory path where you chose to clone the workspace.



Click the *Clone* button. This will create a clone of the workspace onto your local machine - the files will look something like the screenshot below, with overlays on the icons showing the Mercurial status of the files.



Command line equivalent: `hg clone URL`

5.3 Mercurial username configuration

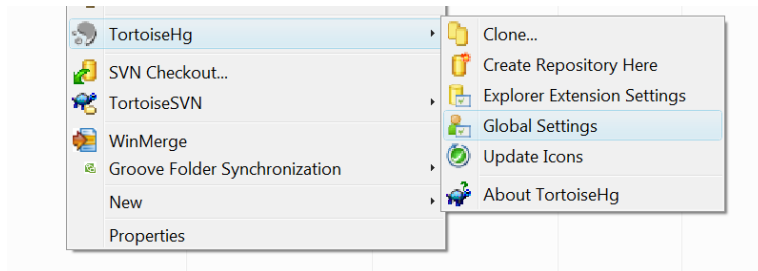
IMPORTANT: Username setup for Mercurial

Since you are about to make changes, your name needs to be recorded as part of the workspace revision history. When commit your changes using Mercurial, it is initially "offline" and independent of the central PMR2 instance. This means that you have to set-up your username for the Mercurial client software, even though you have registered a username on the PMR2 site.

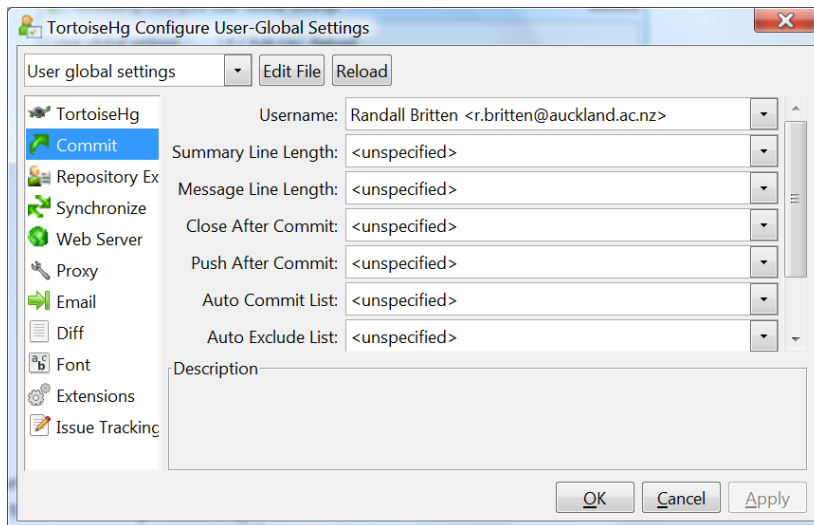
You only need to do this once.

Steps:

- For TortoiseHg, right click on any file or folder in Windows Explorer, and select "Global Settings".



- Select "Commit" and then enter your name followed by your e-mail address in "angle brackets" (i.e. less-than "<" and greater-than ">"). Actually, you can enter anything you want here, but this is the accepted best practice. Note: this information becomes visible publicly if the PMR2 instance that you push you changes to is public.



Command Line equivalent: edit config text file:

Config file location:

Per repository: "<repo>\.hg\hgrc"

System wide:

Linux: "%USERPROFILE%\hgrc"

Windows: "%USERPROFILE%\mercurial.ini"

Entry:

```
[ui]
username = Firstname Lastname <firstname.lastname@example.net>
```

Replace underlined text with your details.

5.4 Making changes to workspace contents

Your cloned workspace is now ready for you to edit the model file and make a commit each time you want to save the changes you have made. As an example, open the model file in Notepad++ and remove the paragraph which describes validation errors from the documentation section, as shown below:

```

artificial stimulus component has been added this model to allow it
to reproduce the action potential simulation shown in Figure 4 of the
publication. The model is known to run and integrate in the PCEnv and
COR CellML environments. A PCEnv session file is also associated with
this model.
34 .....
35 </para>
36 <para>
37 ValidateCellML detects unit inconsistency within this model.
38 </para>
39 .....
40 <para>
41 .....
42 </para>
43 </section>
44 <sect1 id="sec_structure">

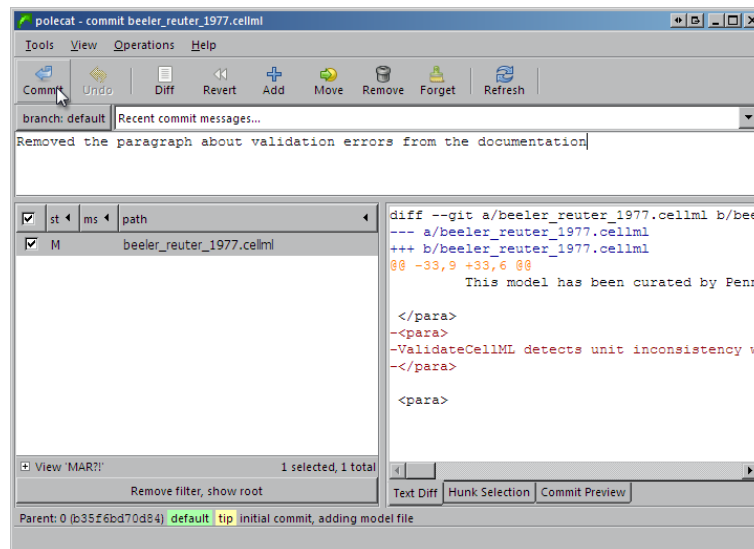
```

Save the file, and notice that the icon has changed to a red cross, indicating that the file now has uncommitted changes.

5.5 Committing changes

Using the shell menu for the added file, select *Hg Commit...* A window will appear showing details of the changes you are about to commit, and prompting for a commit message. Every time you commit changes, you should enter a useful commit message with information about what changes have been made. In this instance, something like *"Removed the paragraph about validation errors from the documentation"* is appropriate.

Click on the Commit button at the far left of the toolbar. The icon overlay for the file will now change to a green tick, indicating that changes to the file have been committed.

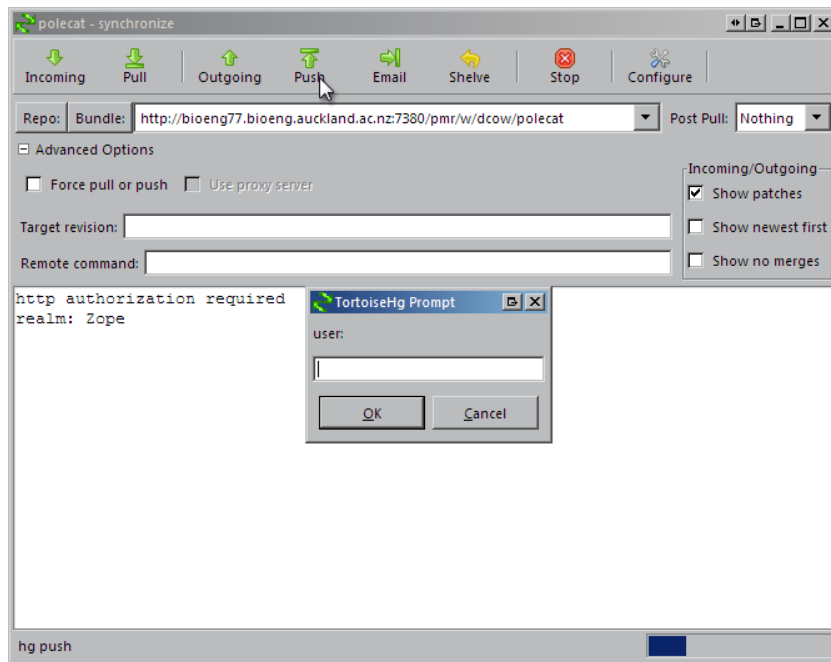


Command Line equivalent: hg commit -m "Removed the paragraph about validation errors from the documentation"

5.6 Pushing changes to the repository

Your cloned workspace on your local machine now has a small history of changes which you wish to *push* into the repository.

Right click on your workspace folder in Windows explorer, and select *Hg Synchronize* from the shell menu. This will bring up a window from which you can manage changes to the workspace in PMR2. Click on the Push button in the toolbar, and enter your username and password when prompted.



Command line equivalent: `hg push`

Now navigate to your workspace on the temporary repository, or refresh the page if you still have it open in your browser. The workspace will now show entries under the Most recent changes, complete with the commit messages you entered for each commit, as shown below:

view edit log shortlog manifest exposure rollover sharing layout actions state: private

polecat

Summary

Owner
dougal cowan <dj.cowan@auckland.ac.nz>

URI for Mercurial clone/pull/push
<http://bioeng77.bioeng.auckland.ac.nz:7380/pmr/w/dcow/polecat>

Most recent changes

Date	Author	Log	Options	Exposure
8 minutes ago	Dougal Cowan	Removed the paragraph about validation errors from the documentation	[manifest] [zip] [gz]	(none)
18 minutes ago	Dougal Cowan	initial commit, adding model file	[manifest] [zip] [gz]	(none)

6 Making an exposure using "roll-over"

As explained earlier, an exposure aims to bring a particular revision to the attention of users who are browsing and searching the repository.

There are two ways of making an exposure. "Rolling over" an exposure is the method used when a workspace already has an existing exposure, and the updates to the workspace have not fundamentally changed the structure of the workspace. This means that all the information used in making the previous exposure is still valid for making a new exposure of a more recent revision of the workspace¹.

Steps:

- From the view page of your workspace, select "exposure rollover".



- Next to each of the exposures listed in the "Exposure column" is an "Option Button" (i.e. one of these:). Select the option button for the latest exposure. This is the exposure that you are going to roll-over, in other words, it serves as the "template" for the new exposure you are about to create.

¹ Strictly speaking, an exposure can be rolled over to an older revision as well, but this is not the usual usage.

Exposure Rollover

Changeset	Date	Author	Log	Options	Exposure
<input checked="" type="radio"/> bb30e3485899	2 hours ago	Randall Britten	Test.	[manifest] [zip] [gz]	(none)
<input type="radio"/> 266541f690f7	6 months ago	Tommy Yu	e-notation fix	[manifest] [zip] [gz]	<input checked="" type="radio"/> Beeler, Reuter, 1977
<input type="radio"/> a5dfb07efdd3	15 months ago	Hanne	Added images in ai and svg format	[manifest] [zip] [gz]	(none)
<input type="radio"/> d4ac7e982034	20 months ago	pmr2.import	committing version08 of beeler_reuter_1977	[manifest] [zip] [gz]	<input type="radio"/> Beeler, Reuter, 1977
<input type="radio"/> fbca003b1306	2008-05-19	pmr2.import	committing version07 of beeler_reuter_1977	[manifest] [zip] [gz]	(none)

- Next to each of the changeset numbers under the "Changeset" column, there is also an option button. Select the option button for the latest revision. This is the revision that you are going to expose.
- At the bottom of the listing, select the "Migrate" button.
- The new exposure page will be displayed visible. However ...
- Near the top right, you will see "state: private" displayed. Select this, and then a drop-down menu appears. Select "submit for publication".

Home Workspace **Exposures** User Workspace

You are here: Home → Exposures → Beeler, Reuter, 1977

contents **view** edit curation builder documentation rollover sharing layout

actions ▼ display ▼ add new... ▼ state: **private** ▼

Beeler, Reuter, 1977

submit for publication

advanced...

by **Randall Britten** — last modified Mar 07, 2011 07:35 PM

- The state will change to "pending review".
- The administrator of the PMR2 instance will then review and publish the exposure, making the exposure visible to all users of the site.

7 Become a Mercurial power user

7.1 Pulling and merging

If you are collaborating with others, you might find that they have pushed changes to the repository. To get updates, you "pull". You can also clone your cloned workspace, and try out two different approaches, and capture the revision history for both, and then merge them back if you ever want to.

Right click on the top level folder of your cloned workspace.

Click on Synchronize from the TortoiseHg menu.

Click "Pull".

You now have a copy of the revision history. However, the files that you work with are not up-to-date with the latest version in that revision history. You need to do an "update" as well.

Right click again on the top level folder of your cloned workspace and choose TortoiseHg->update.

Command line equivalent:

```
hg pull
hg update
```

If both you and others have made changes, you might have to do a merge.

To demonstrate this, make a clone of your workspace in an adjacent directory. This is done the same way as the clone you made earlier, but rather than the URL from PMR2, just use the path on your computer's local storage to the workspace.

Make some dummy changes and commits separately in the original workspace and in the second cloned workspace.

Now, from the second cloned workspace, do a pull.

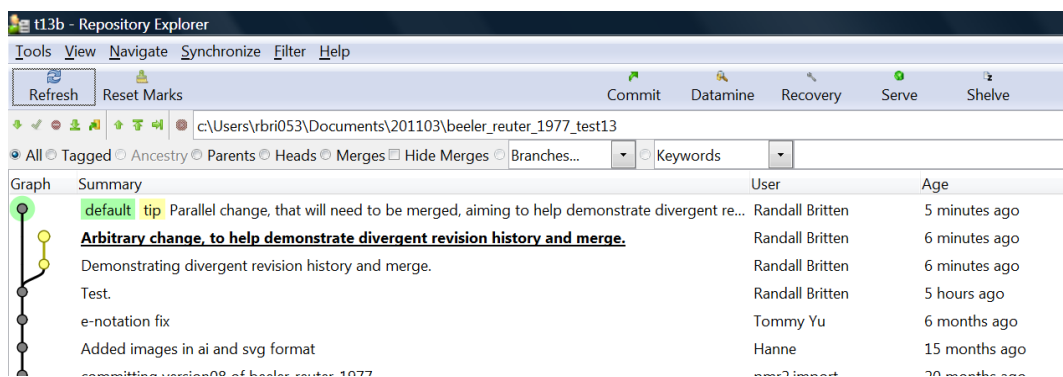
You might see something like this in the log output from a pull:

```
added 1 changesets with 1 changes to 1 files (+1 heads)
(run 'hg heads' to see heads, 'hg merge' to merge)
```

The "+1 heads" means that there are now more head revisions (i.e. latest revisions).

To do a merge:

To do this, choose TortoiseHg->Repository Explorer.



The entry in bold text is the current working directory revision.

Right click on the other branch of the history, choose "Merge with".

If there are no conflicts, the Merge will be automatic, otherwise you will be prompted to edit text files and choose which versions of the binary files to use.

Command line equivalent:

```
hg merge
```

7.2 Sharing changes on a local network

You can share your changes with others directly, even before you push them to the main repository. Find a partner who is doing the tutorial. One of you activates a temporary web server for your workspace, and the other can clone changes from it. This is a "pull" model, where your collaborators have to actively retrieve your changes from your temporary web server.

Right click on a cloned workspace folder, and choose "TortoiseHg->Web Server", then select "Start".

A URL will be displayed. Ask your partner to clone from this URL.

They should then run the web server in the same way, and you can pull changes back into your workspace from them. This allows collaboration independent of the central system, and is one of the key reasons for using a DVCS. You can also mail revision histories. See the Mercurial documentation for details.

Command line equivalent: `hg serve`

8 Creating a new workspace

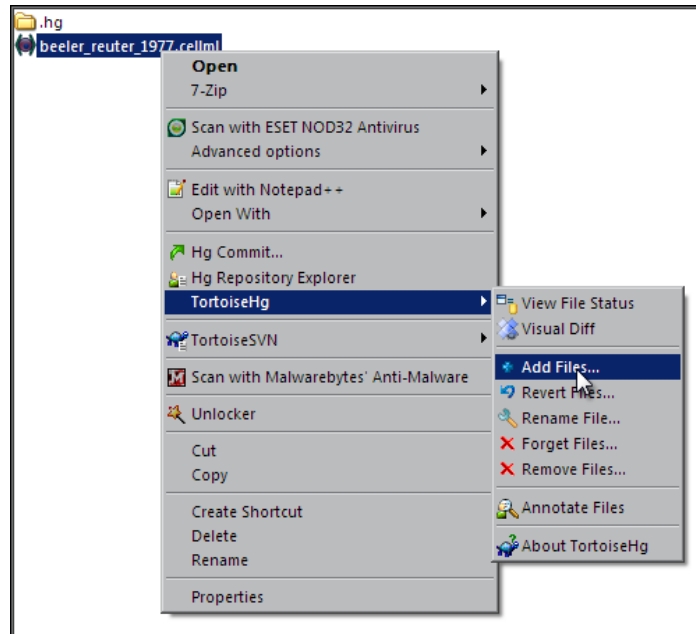
To create a new workspace for your own files, first click on the User Workspace navigation bar button, and then click on your username folder link. You will now have a series of menus running along below the navigation bar - *actions*, *display*, *add new*, and *state*. Click on the *add new* menu and select *pmr2 workspace*. You will be presented with a form requesting that you enter an ID, title, and description for your new workspace. Only ID is required.

For workspaces based on published papers, the conventional ID scheme is first author, year, separated by an underscore - for example, *goldbeter_1991*. You may give your workspace whatever ID you wish - then click on the *Add* button.

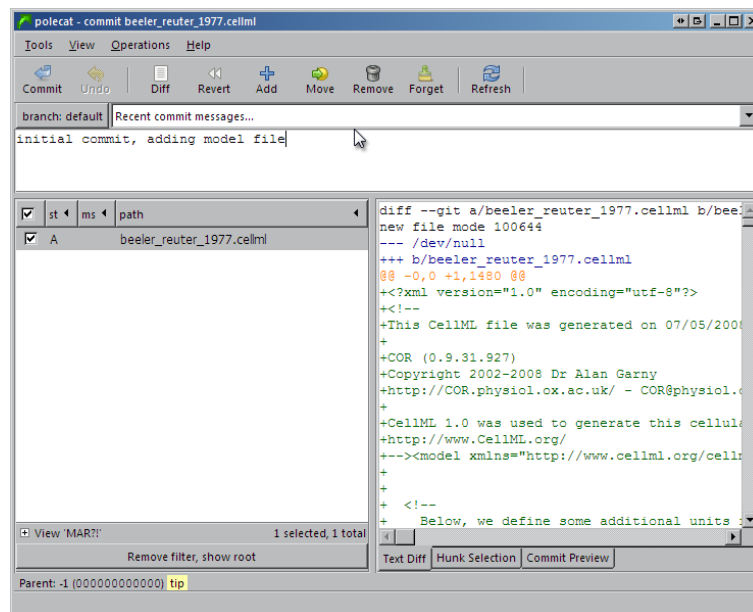
Once you have added your new workspace, you will be presented with the standard view of the workspace. This will show the summary information, the URI for Mercurial clone/pull/push, and the revision history (*Most recent changes*) which will be empty.

8.1 Adding files to the new workspace

First, clone your new workspace just like you did earlier, using the URL for this new workspace. For this tutorial, you can simply copy the model file from elsewhere, e.g. the `beeler_reuter_1977` workspace into your new workspace using Windows explorer. Then right click on the file you copied into your workspace, and select *TortoiseHg > Add Files...* from the shell menu.



A window will appear showing a list of the files that can be added. Simply click the *Add* button to add the model file to your repository. The icon overlay for the file will change to a blue cross, indicating that the file has been added. Adding a file is a change to the repository, and you must now *commit* this change, using the process described earlier, but with a useful commit comment.



You can now also push this change to the repository, again, using the same process as described earlier.

8.2 Making an exposure from scratch

Making an exposure is more complicated for this workspace, since it is a brand new workspace, and there is not a previous exposure that can simply be rolled over.

Basically, the type of exposure is chosen, and then a form is completed selecting contents from the work workspace for each part of the exposure, and filling out some details regarding the exposure.

This part of the tutorial has still to be written up, but will be demonstrated by one of the instructors.