SpiNNaker Human Brain Project Portal Introduction

This document describes how to use the Human Brain Project (HBP) portal to run PyNN scripts on SpiNNaker machines.

1. Getting Started

This section describes the current state of affairs in relation to the SpiNNaker machine hosted by the HBP Portal and the software stacks limitations.

1.1 SpiNNaker Machine

Before even starting to run PyNN scripts on SpiNNaker machines via the portal, you should be aware that even though the total SpiNNaker machine capacity currently available by the HBP portal is a 5 cabinet machine, which accounts for 460,800 cores, as shown in Figure 1 & 2. This, unfortunately, is not available for use directly without special request due to the expectation that end user's models to not require this level of hardware support. The maximum size machine that is be directly available is a 2 cabinet machine, which represents 184,320 practical SpiNNaker cores, as shown in Figure 3 & 4.

The rest of the SpiNNaker machine (3 cabinets worth) is separated into a number of smaller machines, with differing resources, allowing the opportunity to support multiple jobs in parallel. These machine will consist of XXXXXXXXXXXXXXXXx.

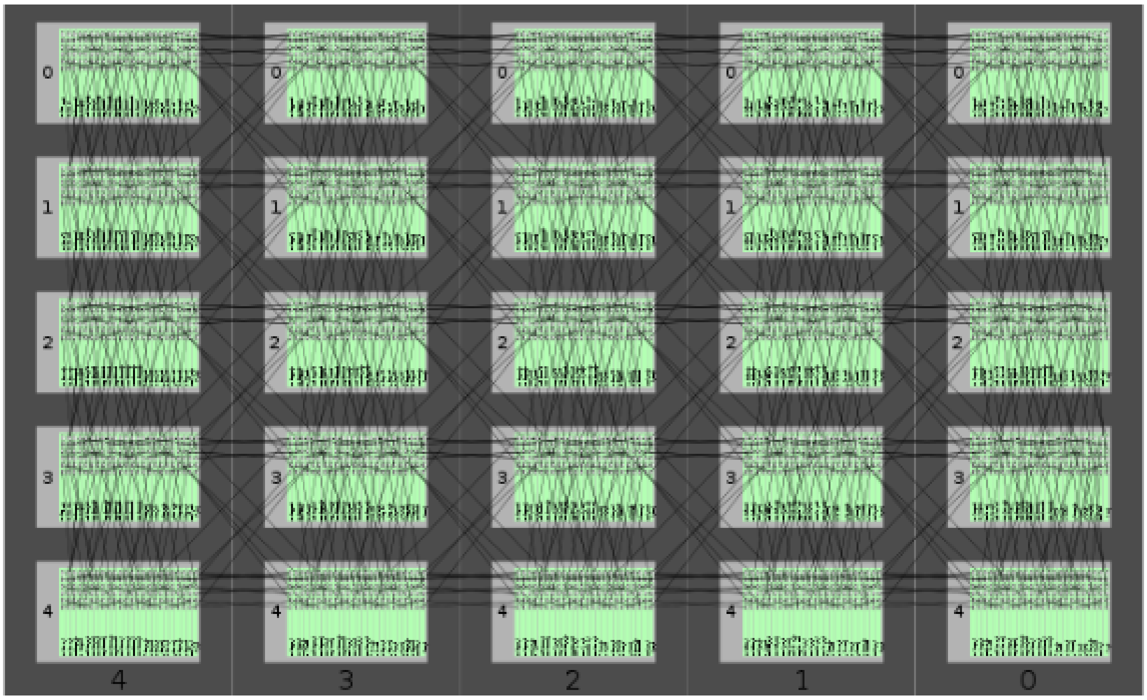


Figure 1: A 5 cabinet machine, wired together (abstract view).

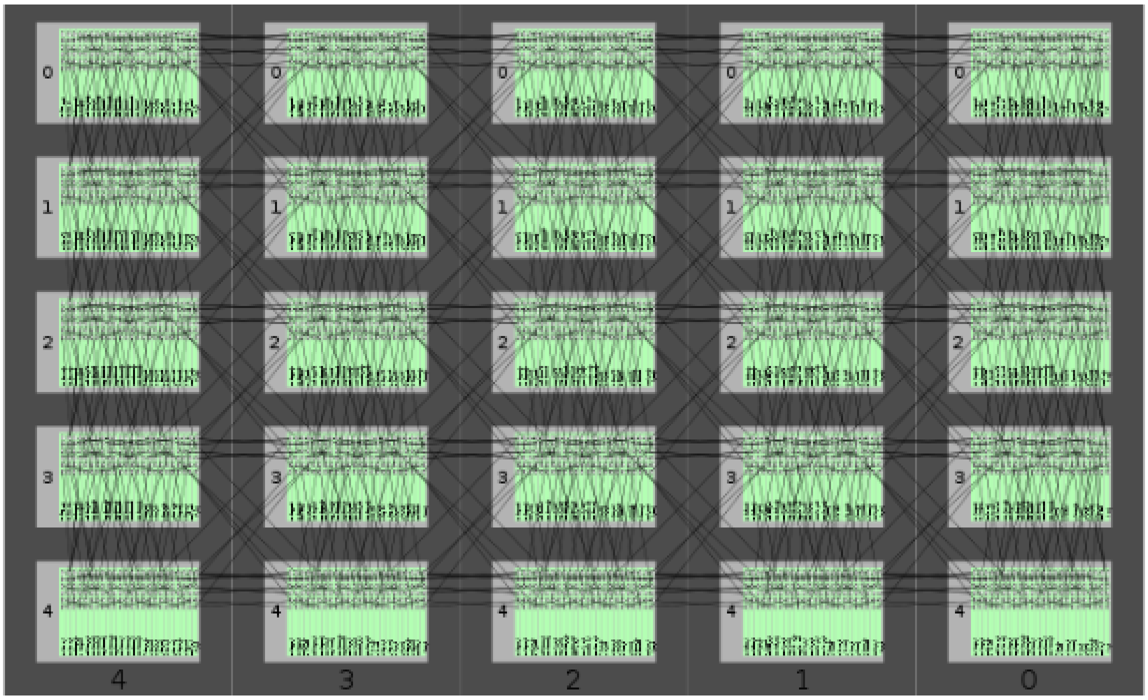


Figure 2: A 5 cabinet machine, wired together (real view).

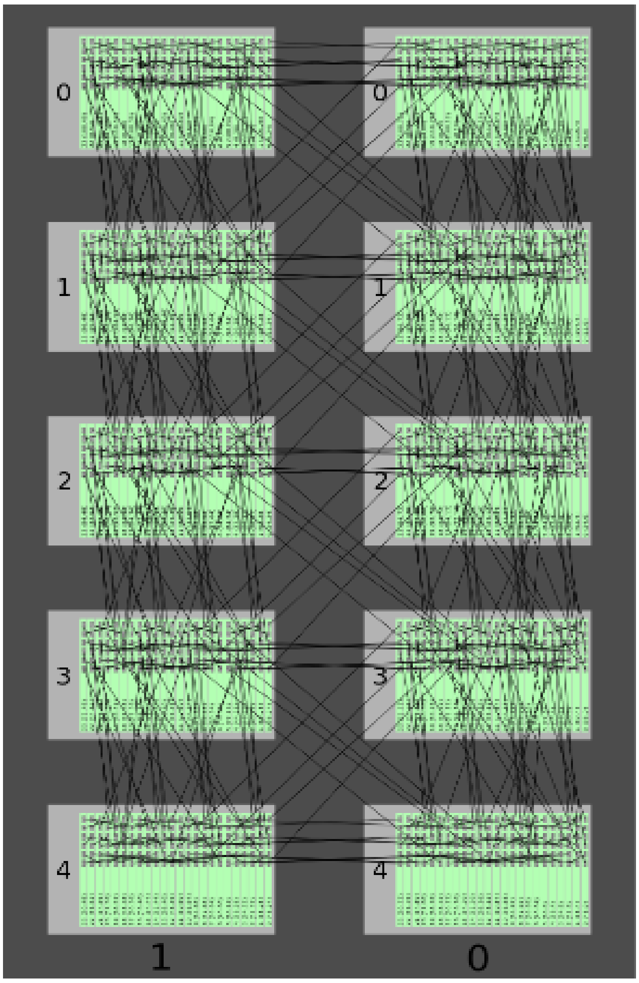


Figure 3: A 2 cabinet machine, wired together (abstract view).

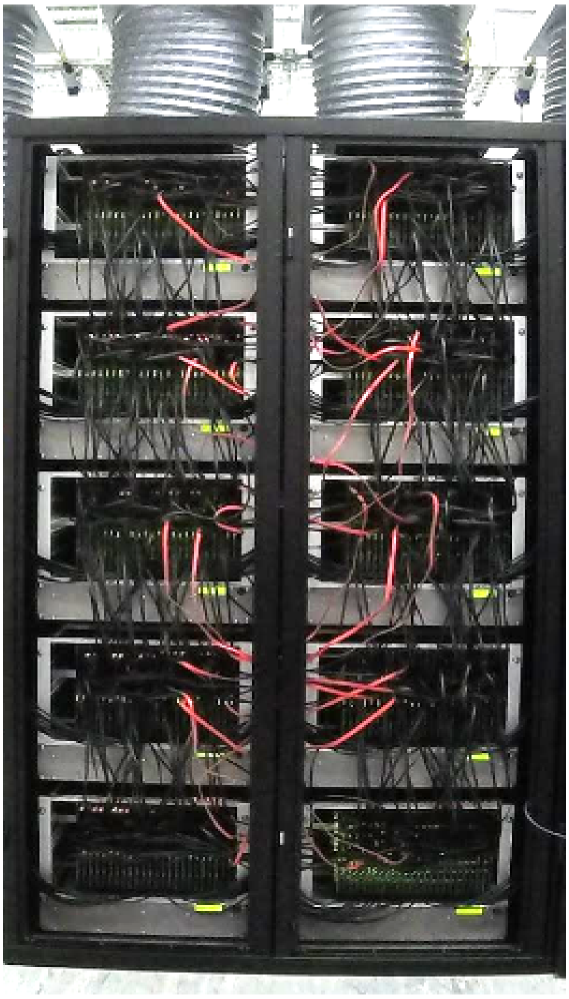


Figure 4: A 2 cabinet machine, wired together (real view).

1.2 Software Limitations

To understand how this relates to a PyNN script, you must first be aware that SpiNNaker stores all the data needed to execute a PyNN model within its on-board memory (SDRAM). This SDRAM is rather limited in size, as each SpiNNaker chip contains only 120 Meg of memory, which is split dynamically between the 16 cores that reside on the chip. This means on average, every core has available approximately 8 Meg to store the neuron parameters as well as the synaptic matrix of the atoms[1] it is executing. Most models to date are limited to 256 atoms per core[2], and therefore the connectivity between these 256 atoms and the rest of the simulation must fit within these rather tight constraints. Figure 5 shows an equation that shows how the amount of space needed by a PyNN model is calculated.

Figure 5: the equation for determining the size of SDRAM used by a PyNN model.

Due to these limitations, the maximum possible number of neurons that can be simulated by a PyNN script on the largest SpiNNaker machine available on the HBP portal is 47,185,920 neurons. This sounds rather large, till you factor in that each atom needs to have a maximum incoming fixed connections of XXXXXXX. Any more than this will result in a smaller number of atoms per core, and therefore a smaller total neurons being simulated.

To make matters worse, if the PyNN script has delays that are greater than 16 machine time steps [3] then each atom will have a delay model attached to it, which practically reduces your maximum number of atoms that are simulatable by half.

Finally, if the PyNN model includes plastic connections, these require more memory than fixed connections and therefore will reduce the number of atoms per core yet again. A summary of the rest of the SpiNNaker back end limitations can be found here:

http://spinnakermanchester.github.io/2015.006.AnotherFineProductFromTheNonsenseFactory/SPyNNakerLimitations.html

1.3 HBP Portal Limitations

The HBP portal executes its PyNN scripts in a batch mode process. This means certain capabilities which are supportable by the SpiNNaker software stack (sPyNNaker) are not currently by the HBP portal. These consist of:

1. Real time visualisation of an executing PyNN script to a standalone visualization kit.

[1] Atoms here represents the atomic element which each core models. In the case of PyNN, these are neurons from a given population.

[2] This limitation may be lifted in the future, but to date, this is due the way delay's are implemented on SpiNNaker to date.

[3] This is 16 milliseconds in a simulation that runs at 1ms time steps, or 1.6 milliseconds in a simulation running at 0.1 ms time steps

2. Closed loop simulation with real robotics.
3. Closed loop simulation with robots via the virtual robotic environment.

1.4 Script Limitations

This section describes some basic limitations that the PyNN script can experience when running on the HBP Portal.

1. Figures generated by the script must be stored in file format, no figures will be presented to the end user in an interactive mode.
2. A PyNN script running on the SpiNNaker backend needs to adhere to the subset of PyNN that is supported by sPyNNaker[4], a list of what is not implemented can be found here:

http://spinnakermanchester.github.io/2015.006.AnotherFineProductFromTheNonsenseFactory/SPyNNakerLimitations.html

1.5 Recommendations

This all means that we recommend that you test your simulation with the sPyNNaker front end in virtual mode before attempting to run it through the HBP portal. Instructions on how to install the sPyNNaker front end can be found here:

http://spinnakermanchester.github.io/2015.006.AnotherFineProductFromTheNonsenseFactory/PyNNOnSpinnakerInstall.html

and instructions on how to operate the tool chain in virtual mode can be found here:

http://spinnakermanchester.github.io/2015.006.AnotherFineProductFromTheNonsenseFactory/VirtualMode.html

[4] This is the spinnaker software stacks front end for simulating PyNN neuron models.

2. Running A PyNN script on the HBP portal

This section describes how a end user goes about simulating

2.1 Getting a HBP account

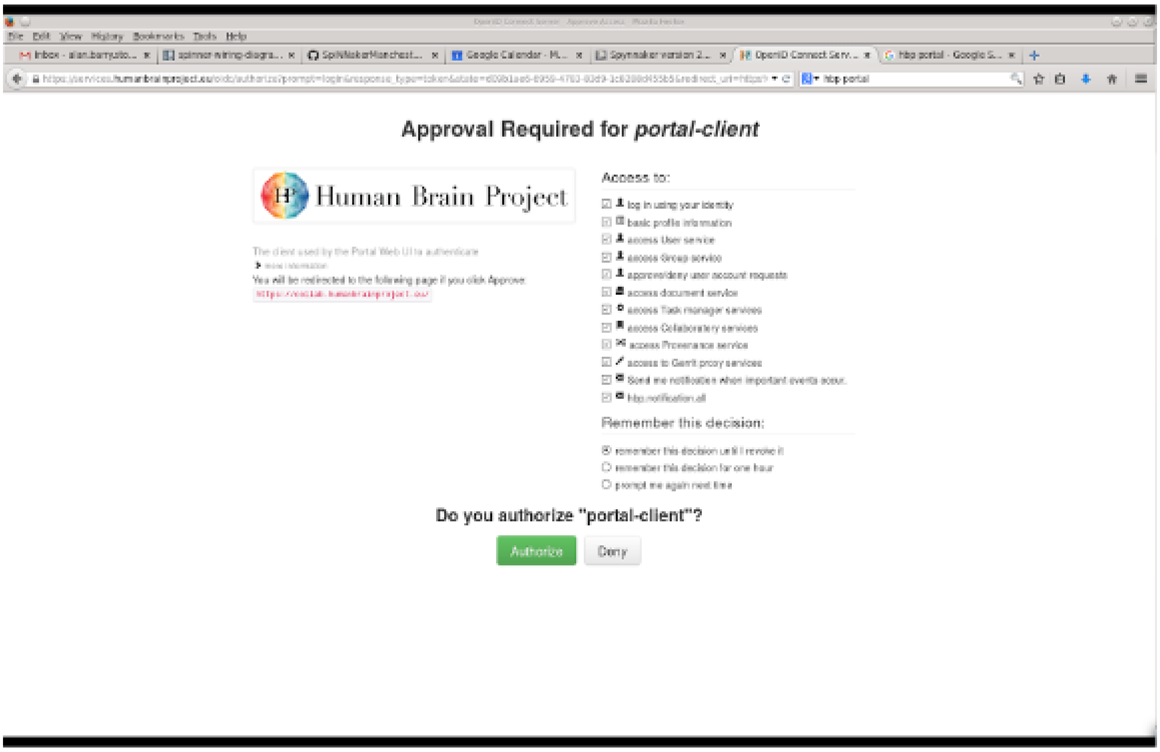
To acquire a HBP account, you must communicate with the head of your HBP SP which will then organise getting yourself said account details.

2.2 Creating a Job on the HBP portal/collaboratory.

1. You first must log onto the HBP portal/collaboratory via the link below:

https://collab.humanbrainproject.eu/

2. If you have not logged on before, you may be presented with an Approval request by the portal-client. As shown in Figure 6. If you encounter this, we recommend Authorizing this request, as otherwise ?????????????????



Approval Required for *portal-client*

The client used by the Portal Web UI to authenticate

more information

You will be redirected to the following page if you click Approve:

https://collab.humanbrainproject.eu/

Access to:

☑ log in using your identity
☑ basic profile information
☑ access User service
☑ access Group service
☑ approve/deny user account requests
☑ access document service
☑ access Task manager services
☑ access Collaboratory services
☑ access Provenance service
☑ access to Gerrit proxy services
☑ Send me notification when important events occur.
☑ hbp.notification.all

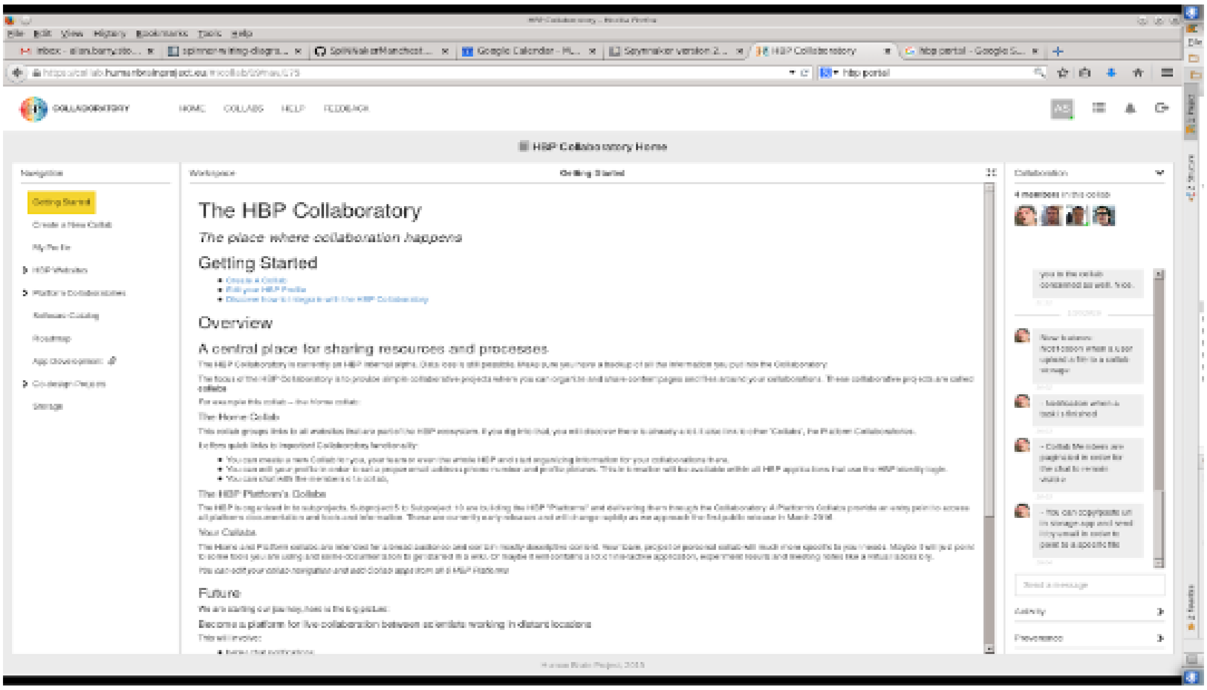
Remember this decision:

● remember this decision until I revoke it
○ remember this decision for one hour
○ prompt me again next time

Do you authorize "portal-client"?

Figure 6: Approval Request for portal-client

3. Once you log onto the HBP portal, you should see a screen such as in Figure 7.



HBP Collaboratory Home

The HBP Collaboratory

The place where collaboration happens

Getting Started

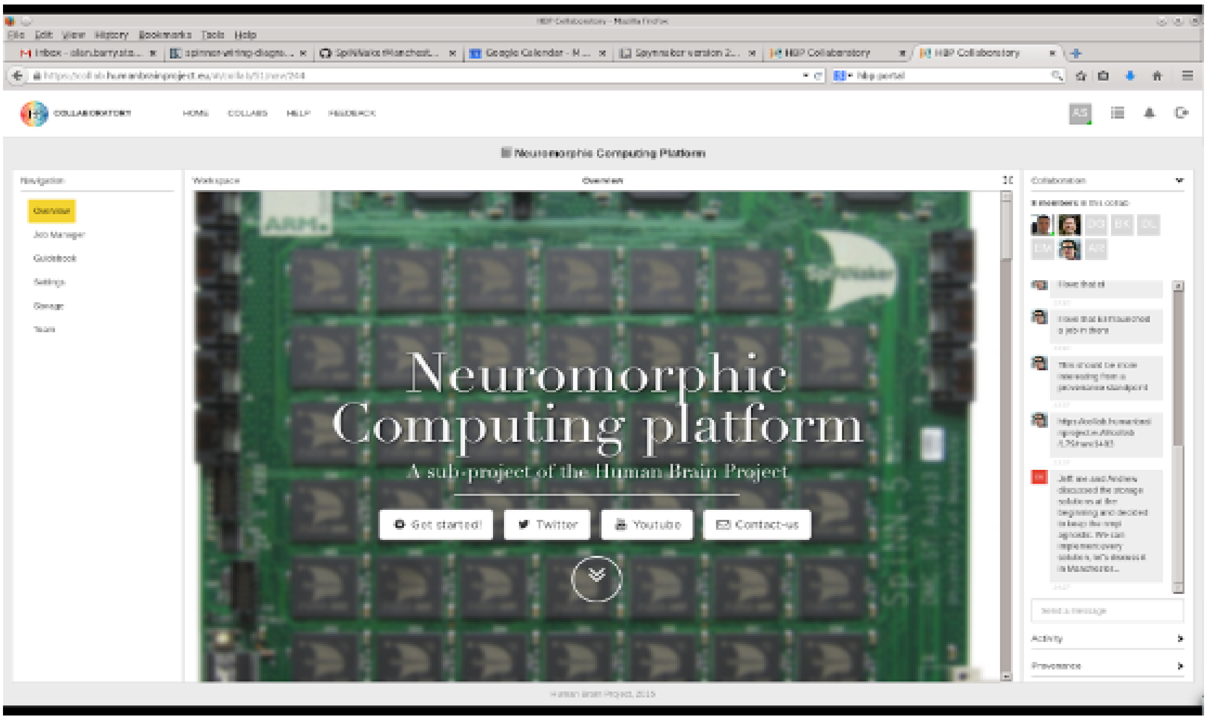
Overview

A central place for sharing resources and processes

Future

Figure 7: HBP portal/Collaboratory Home Page

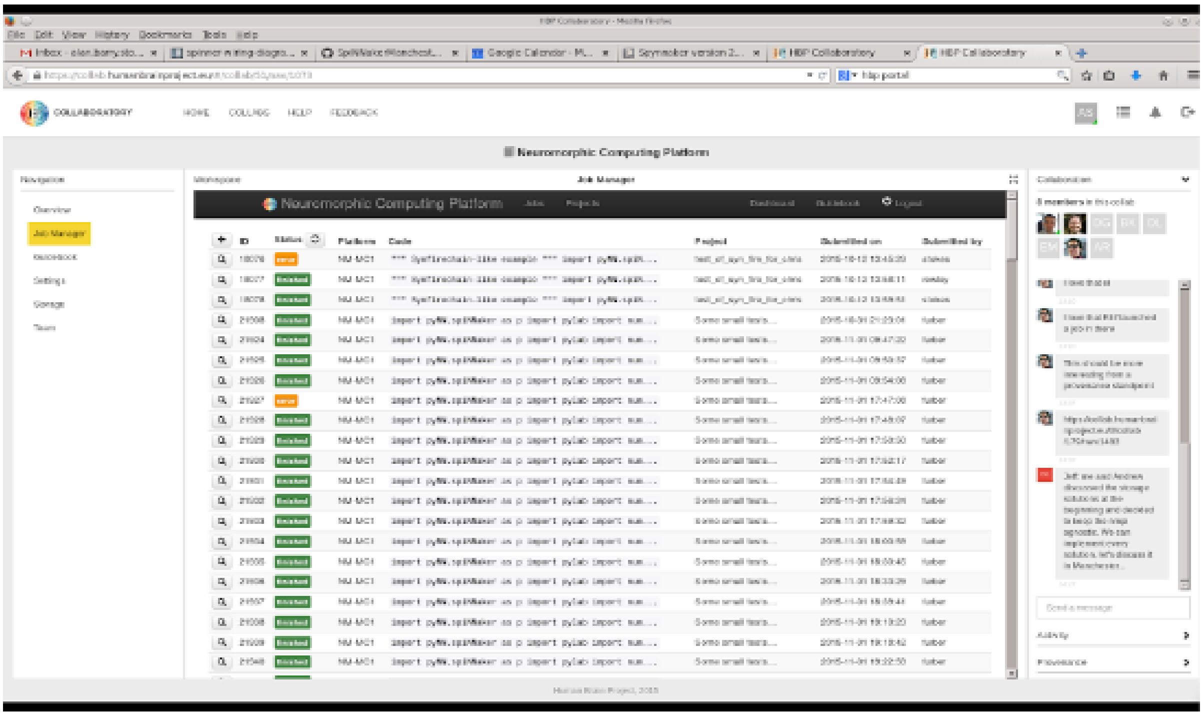
4. On the left sidebar, open up the tab called “Platform Collaboratories” (5th one down) and click on “Neuromorphic” (5th one down). You should then see the page shown in Figure 8.



Neuromorphic Computing Platform

Figure 8: Neuromorphic Computing Platform Home page

5. From the sidebar, click on Job Manager (second one down). You should then see a page such as Figure 9. This lists all the jobs that have been submitted to the HBP portal to which you have approval to view.



ID	Status	Platform	Code	Project	Submitted on	Submitted by
18076	error	NM-MC1	""" Synfirechain-like example """ import pyNN.spiN...	test_of_syn_fire_for_chris	2015-10-12 13:45:20	astokes
18077	finished	NM-MC1	""" Synfirechain-like example """ import pyNN.spiN...	test_of_syn_fire_for_chris	2015-10-12 13:48:11	rowley
18078	finished	NM-MC1	""" Synfirechain-like example """ import pyNN.spiN...	test_of_syn_fire_for_chris	2015-10-12 13:49:43	astokes
21508	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-10-31 21:23:04	faber
21524	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 08:47:32	faber
21525	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 08:50:37	faber
21526	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 08:54:06	faber
21527	error	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 17:47:06	faber
21528	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 17:48:07	faber
21529	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 17:50:30	faber
21530	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 17:52:17	faber
21531	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 17:54:49	faber
21532	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 17:58:26	faber
21533	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 17:59:32	faber
21534	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 18:00:59	faber
21535	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 18:03:45	faber
21536	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 18:33:29	faber
21537	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 18:35:41	faber
21538	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 19:13:20	faber
21539	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 19:18:42	faber
21540	finished	NM-MC1	import pyNN.spiNNaker as p import pylab import num...	Some small tests...	2015-11-01 19:22:50	faber

8 members in this collab

I love that it

I see that [illegible] launched a job in there

This should be more interesting from a provenance standpoint

https://collab.humanbrainproject.eu/#/collab/1403

Jeff, we and Andrew discussed the storage solutions at the beginning and decided to keep the Image agnostic. We can implement every solution, let's discuss it in Manchester...

Figure 9: Previous submitted Jobs on the HBP portal.

6. If you already have a project created, then go to point 13, otherwise carry on reading.
7. If you have not created a project before, above the “Create Job” title is a header, which contains a “project” tab, as shown in Figure 10. Please click on the Projects tab. This will take you to a page as shown in Figure 11.

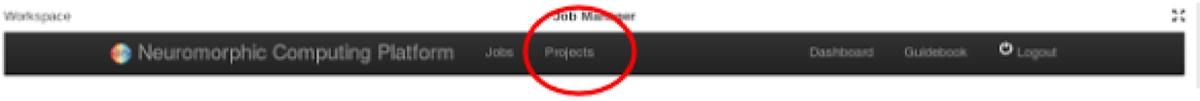


Figure 10: projects tab

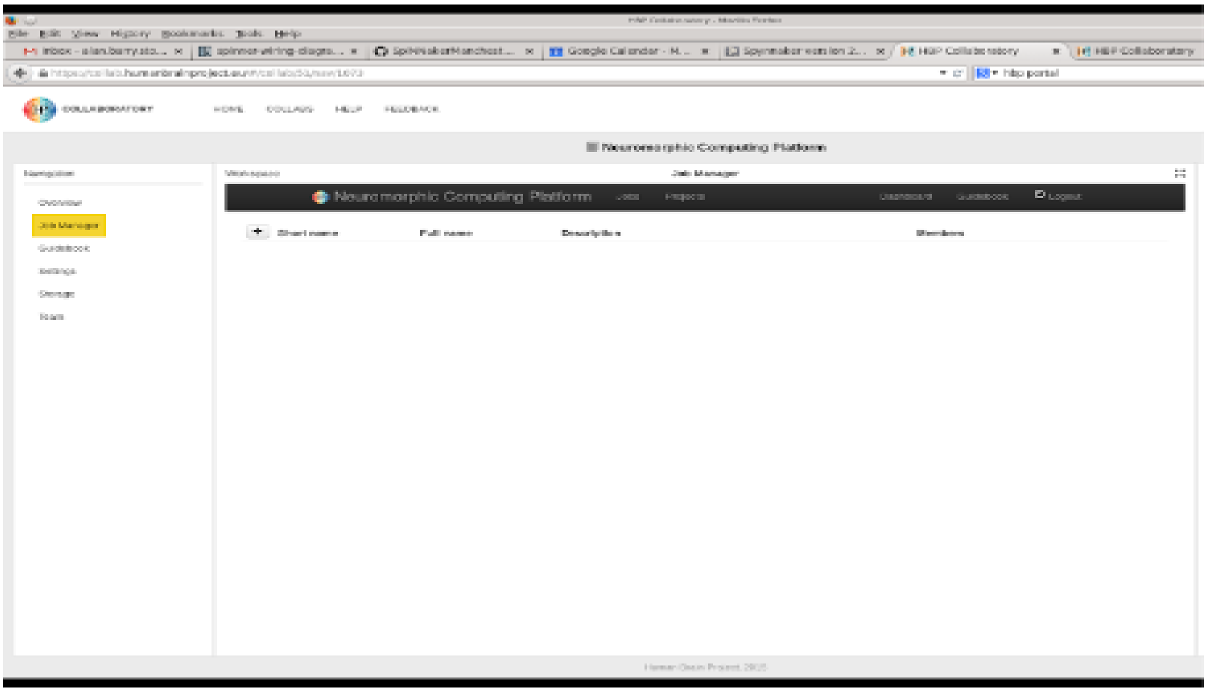
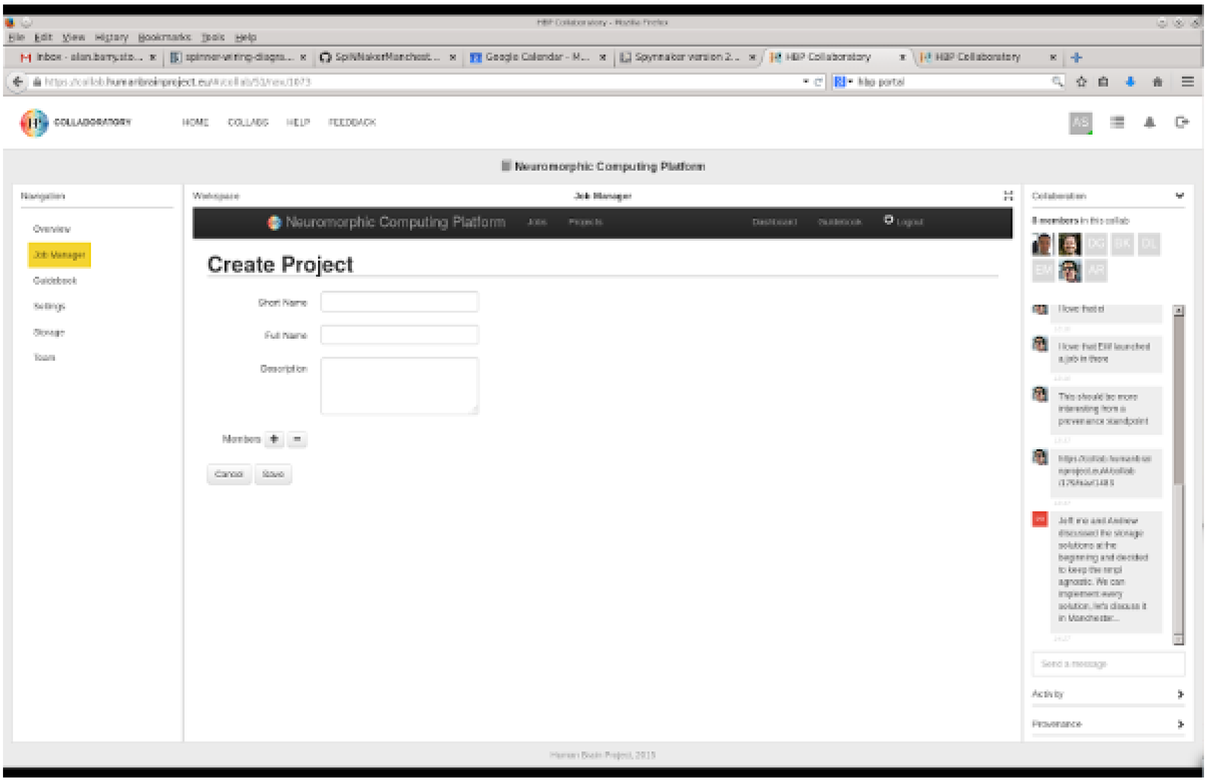


Figure 11: Project Listing screen

8. To create a new project look for the + sign on the left hand side of the column “short name”. Click on this button will result in the screen in Figure 12.



I love that it

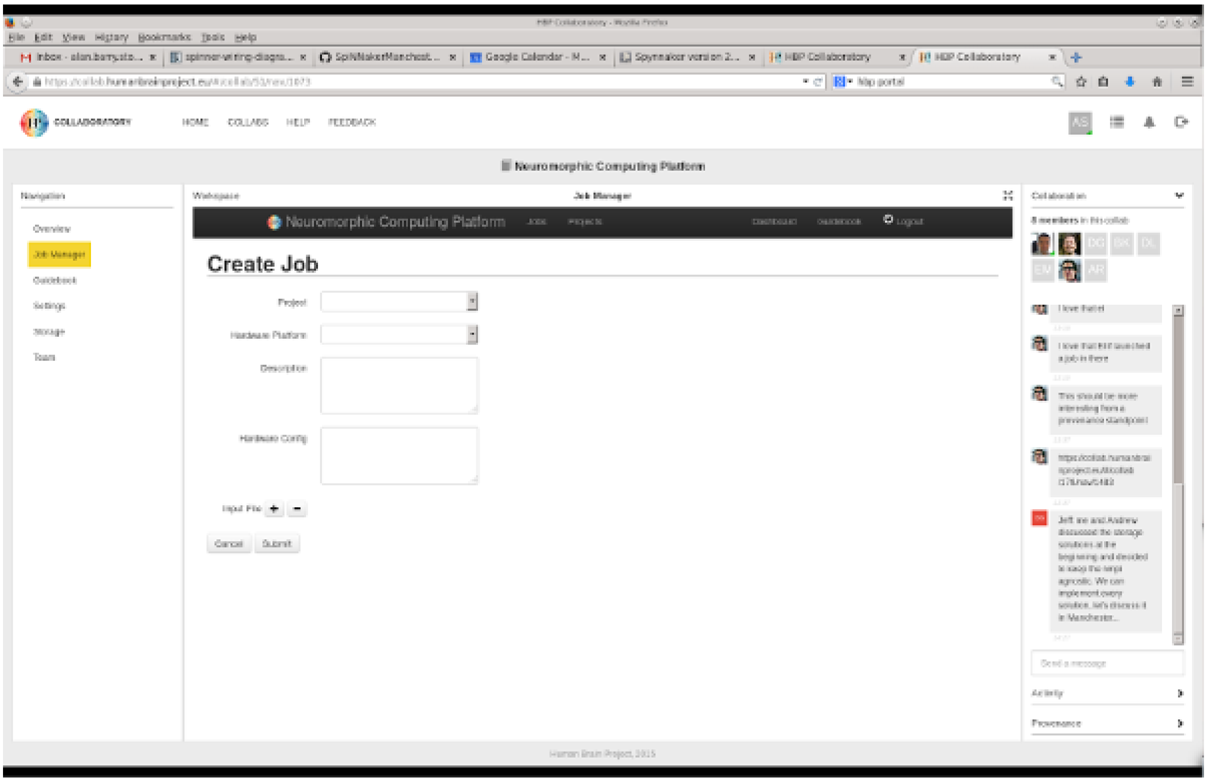
I love that EW launched a job in there

This should be more interesting from a provenance standpoint

Jeff me and Andrew discussed the storage solutions at the beginning and decided to keep the impl agnostic. We can implement every solution, let's discuss it in Manchester...

Figure 12: Create Project page

9. Fill in the three boxes accordingly.
10. On the members button, click + and add "**uman**" as one of the members, as well as yourself. Click save and you should be sent back to a screen such as the one in Figure 11.
11. You should see that the screen has changed slightly, in that there is now a notification saying "Your project has been created" and the listing should now include a entry which is your project.
12. To get back to the page where you can add new jobs, you need to refresh the job manager page. The easiest way to do this is to click on the overview tab on the left side of the page, and then click on the Job Manager page again. You should get back to the screen as shown in Figure 9.
13. To submit a new job, look for a + sign button (will be located at the top of the list of previously ran jobs) and click it. This will take you into a window such as the one shown in Figure 13.



I love that!

I love that KEF launched a job in there

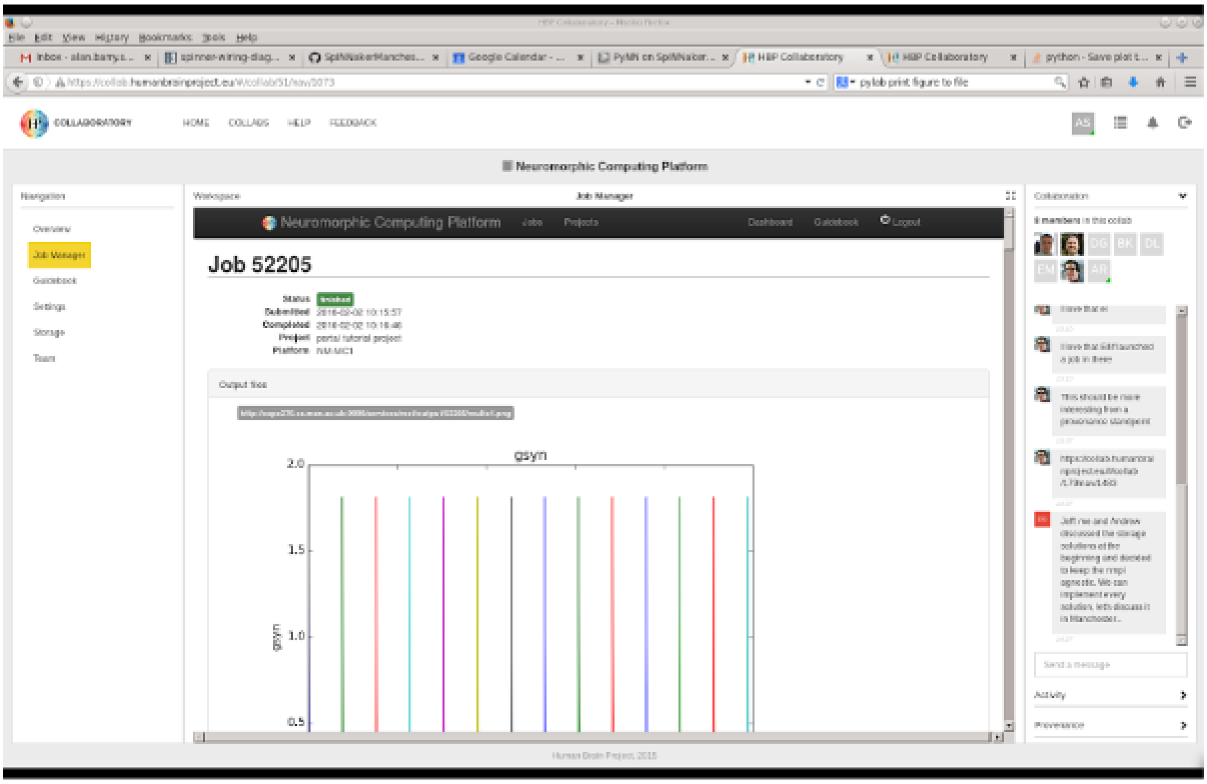
This should be more interesting from a provenance standpoint

Jeff, me and Andrew discussed the storage solutions at the beginning and decided to keep the [illegible] agnostic. We can implement every solution, let's discuss it in Manchester...

Figure 13: Job creation page

14. Firstly, open the drop down for the Project and select your projects "long title".
15. To create a job that runs on the SpiNNaker platform you must select NM-MC1 from the "Hardware Platform" field.
16. "Description" can be one of 2 options. These are:
    - A PyNN script that you're going to run.
    - A URL to a github repository which requires to have a script called run.py within in.
17. "Hardware Config" does not need to be filled in to use the SpiNNaker backend.

18. Click Submit. This has now submitted your job to the HBP portal, and you'll be moved back to the page displayed in Figure 9 but with the notification "Your Job has been submitted. You will receive further updates by email"
19. To get access to the results. Scroll down the page to the bottom, where the most recent jobs have been placed.
20. Your Job will start in the "submitted" stage, then "running" stage, then either "finished" or "error".
21. To get the output from the script, once the script has reached "finished" state. Click on the magnifying glass symbol next to your job. You should go to a page like the one shown in Figure 14.



Job 52205

Status: finished
Submitted: 2016-02-02 10:15:57
Completed: 2016-02-02 10:16:46
Project: portal tutorial project
Platform: NM-MC1

This should be more interesting from a provenance standpoint

Jeff me and Andrew discussed the storage solutions at the beginning and decided to keep the nmpi agnostic. We can implement every solution, let's discuss it in Manchester...

Figure 14: Script Output Page

22. A more detailed description of this behaviour for all platforms, and how to use the HBP portal in scripted mode can be found here:

https://collab.humanbrainproject.eu/#/collab/51/nav/1069

Congratulations, you have now ran a PyNN script job via the HBP portal.