12TH FENIX INFRASTRUCTURE WEBINAR

EBRAINS services deployed on ICEI services: Neuromorphic Computing front-end

Andrew Davison
Paris-Saclay Institute of Neuroscience
CNRS - Université Paris-Saclay

The ICEI and HBP projects have received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreements 800858 (ICEI), 720270 (HBP-SGA1), 785907 (HBP-SGA2) & 945539 (HBP-SGA3).

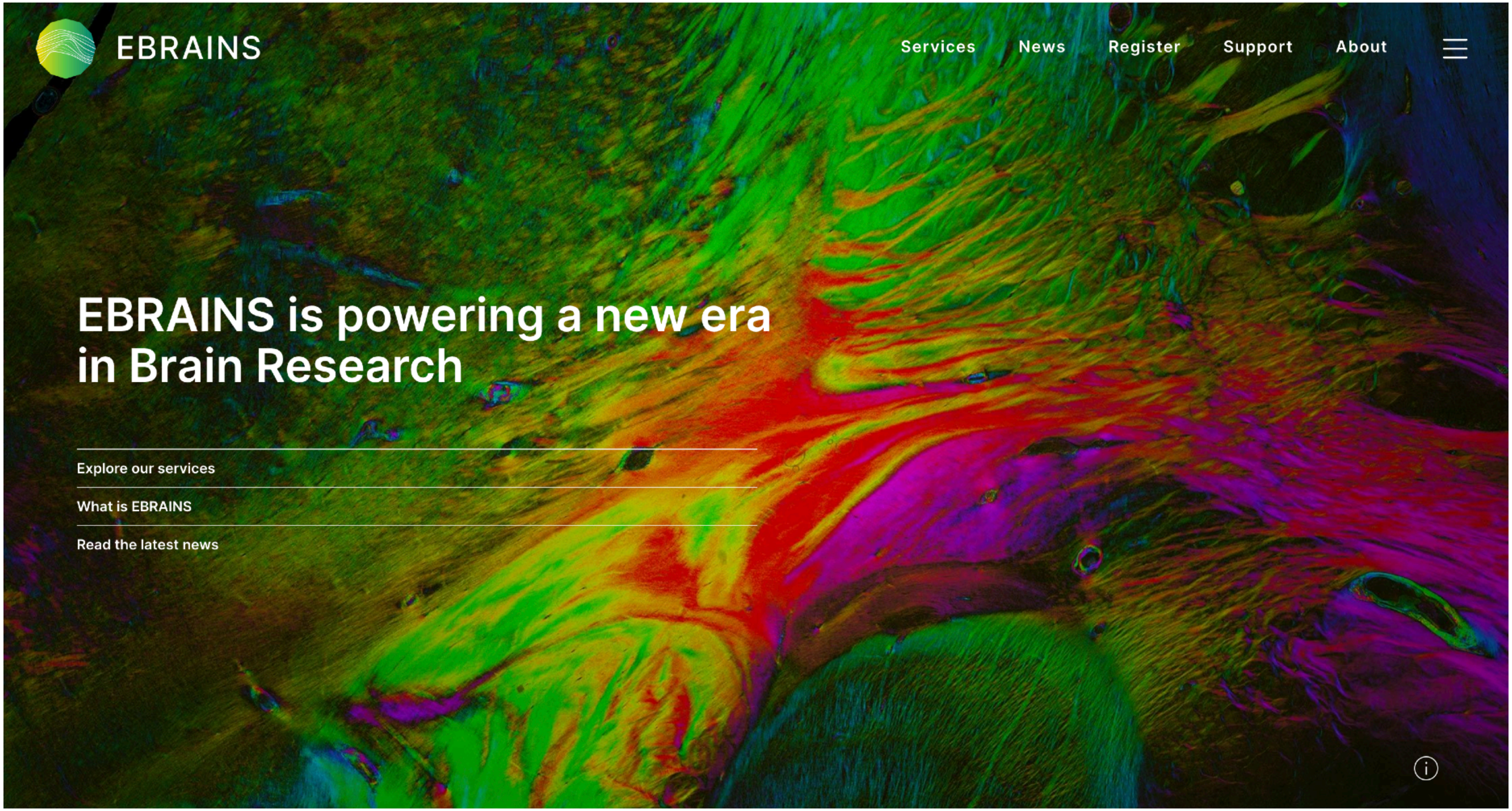
Outline

- Overview of the SpiNNaker and BrainScaleS neuromorphic systems
- How to use these neuromorphic systems
- How we're using the OpenStack infrastructure provided by ICEI to build and deploy our services
- Future plans
- Q&A

Target audience

- Neuroscientists
- HPC infrastructure users
- EBRAINS service developers
- Other platform service developers

Neuromorphic Computing in EBRAINS



EBRAINS is powering a new era in Brain Research

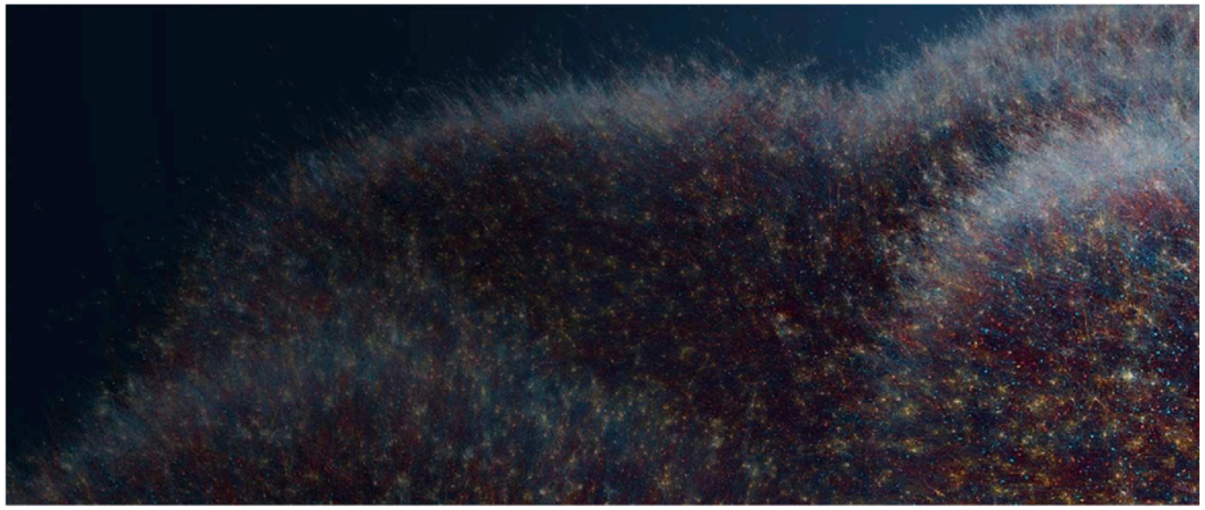
https://ebrains.eu

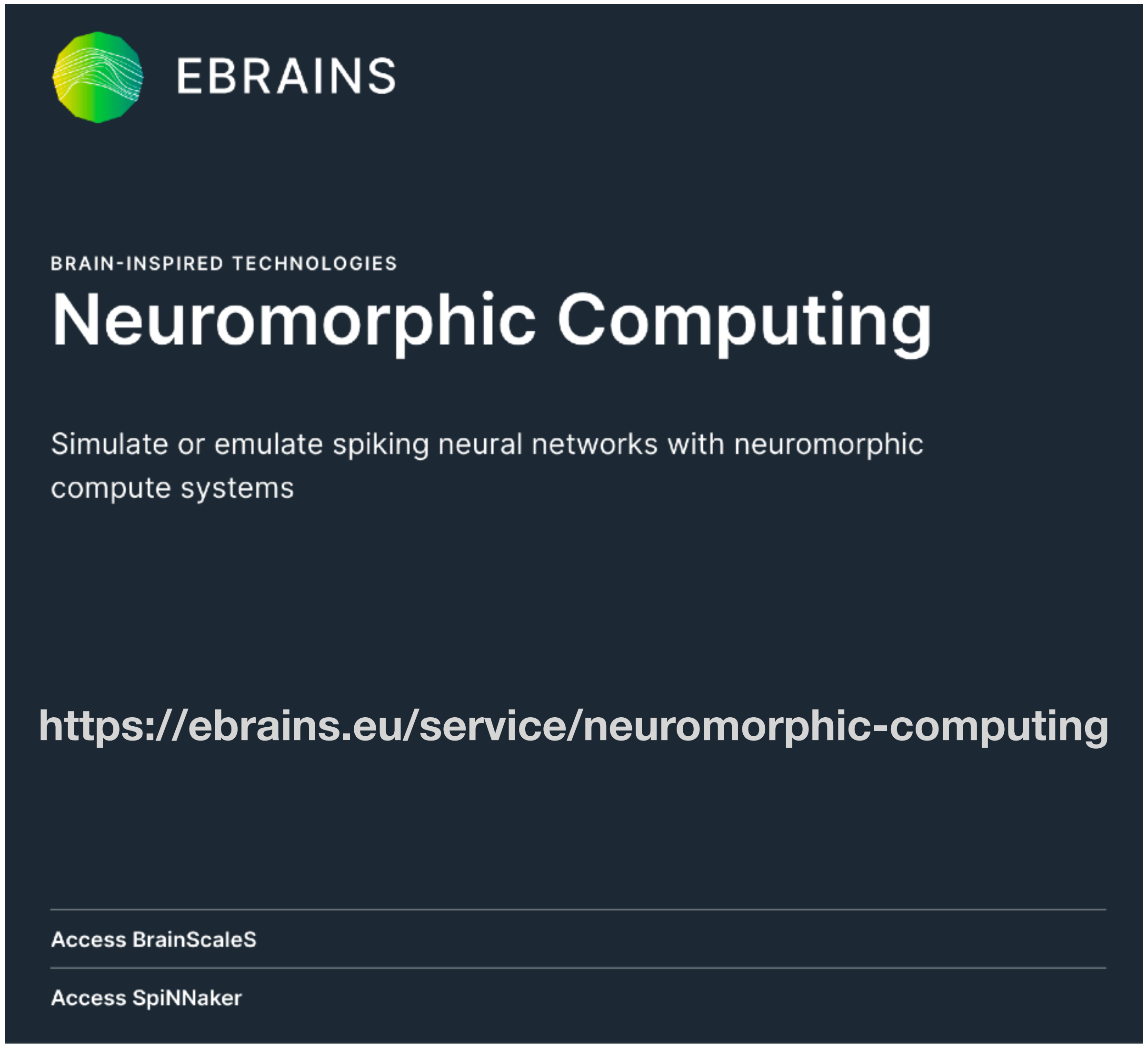
Neuromorphic Computing in EBRAINS

EBRAINS

Register for your EBRAINS account today

https://ebrains.eu/





EBRAINS

BRAIN-INSPIRED TECHNOLOGIES

Neuromorphic Computing

Simulate or emulate spiking neural networks with neuromorphic compute systems

https://ebrains.eu/service/neuromorphic-computing

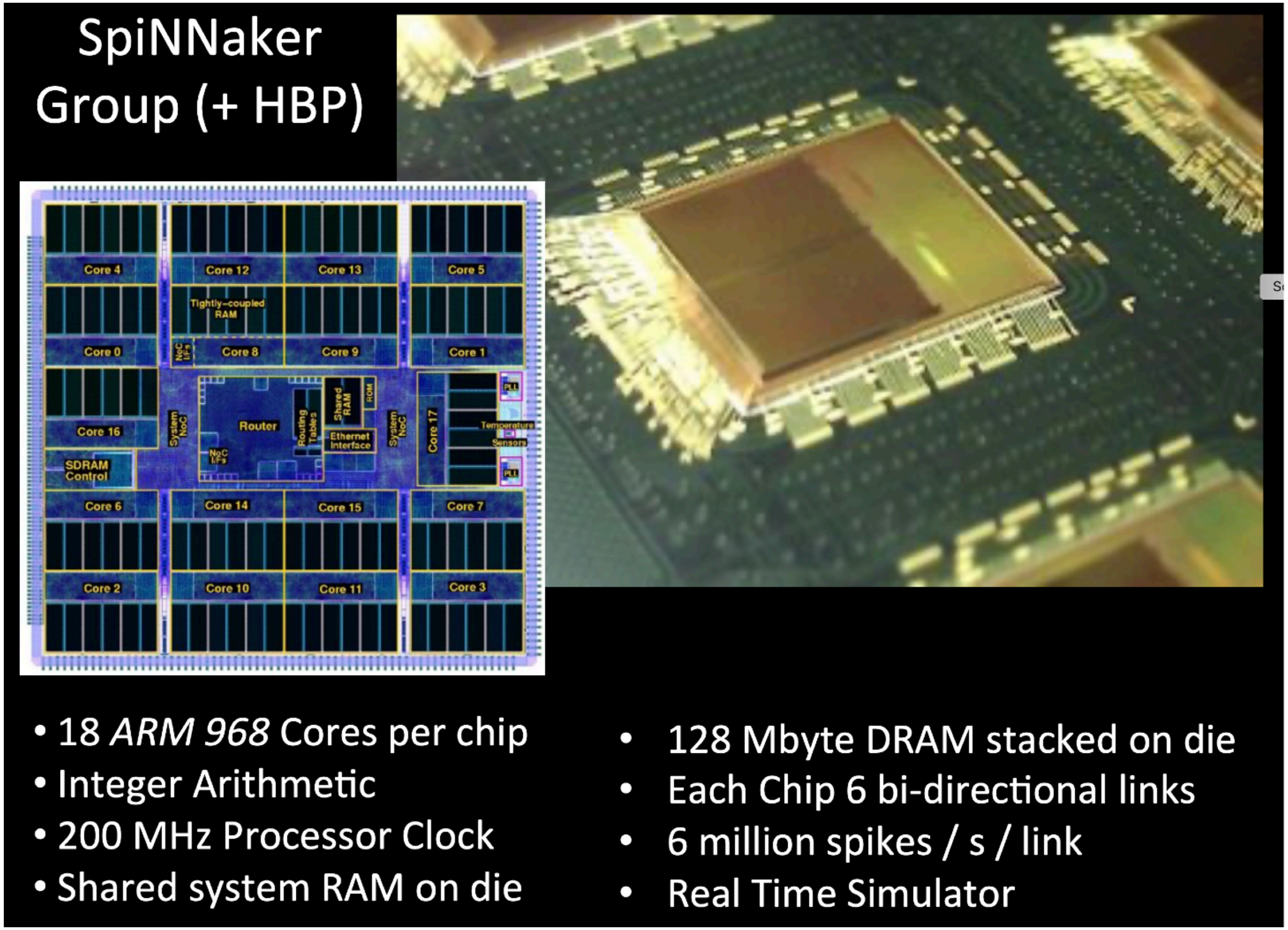
Access BrainScaleS

Access SpiNNaker

FENIX RI

SpiNNaker

- custom-designed digital chips optimised for spiking neural network models

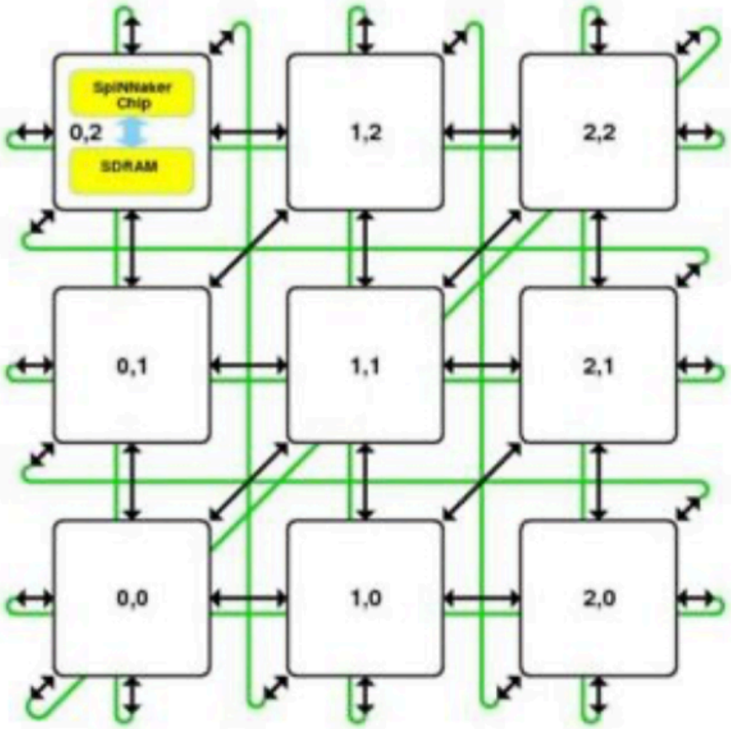


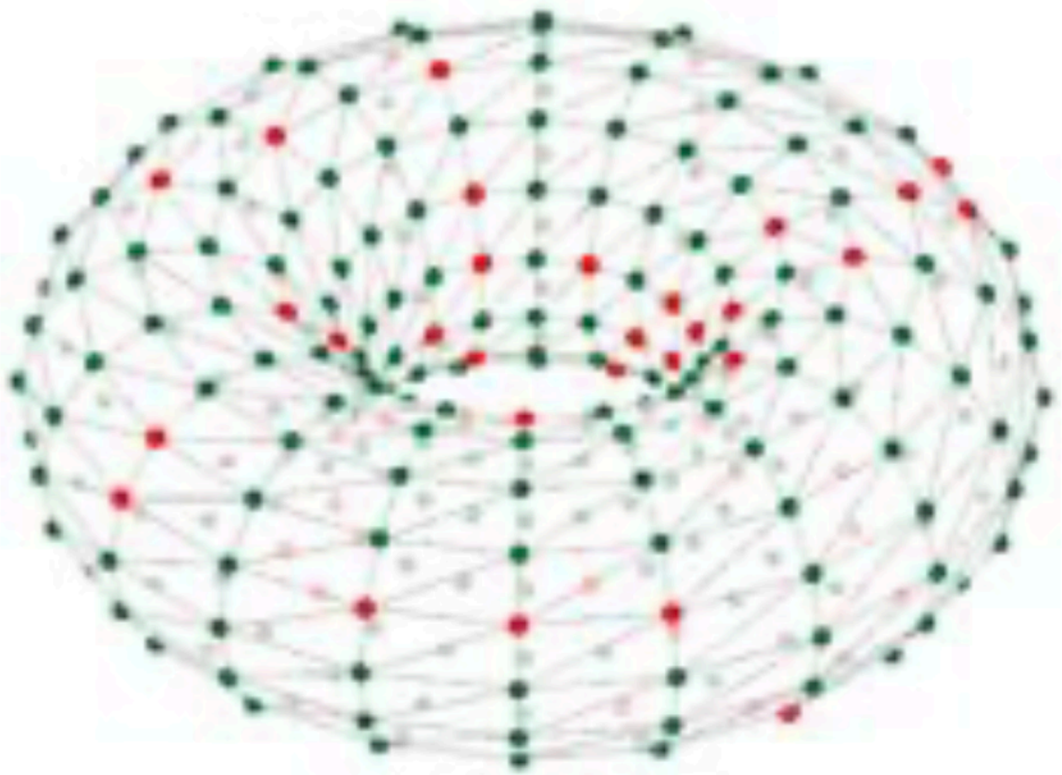
SpiNNaker Group (+ HBP)

- 18 *ARM 968* Cores per chip
- Integer Arithmetic
- 200 MHz Processor Clock
- Shared system RAM on die

- 128 Mbyte DRAM stacked on die
- Each Chip 6 bi-directional links
- 6 million spikes / s / link
- Real Time Simulator

SpiNNaker





Connectivity

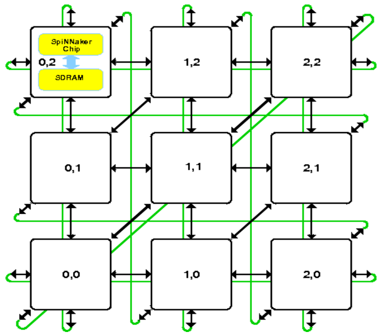
- Small packet based communication network
- Toroidal link geometry
- Maximum of 3 routings between any pair of nodes
- 6 Million spikes per second per bi-directional link

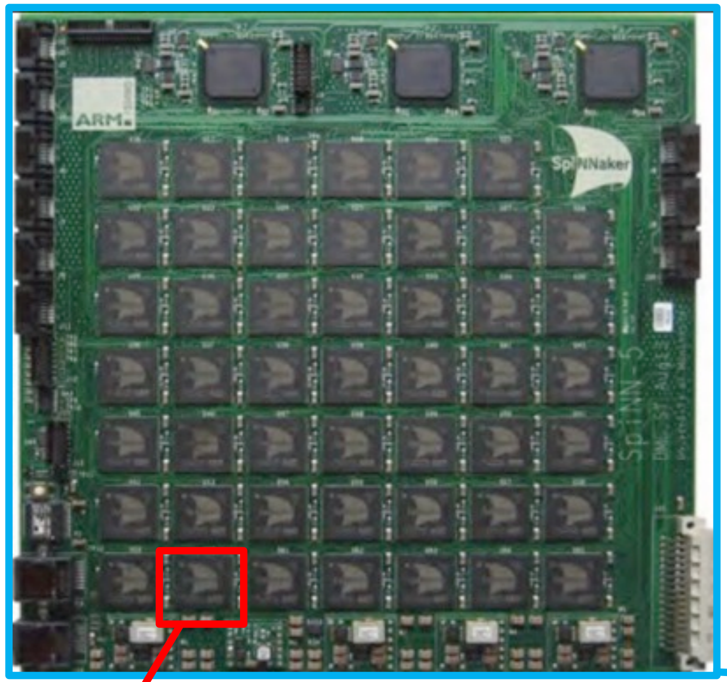
SpiNNaker

Biologically
Inspired
Massively
Parallel
Architectures

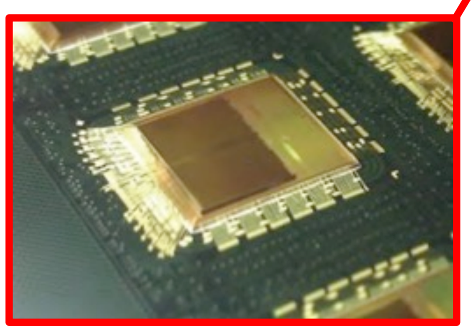
SpiNNaker

SpiNNaker board
(864 ARM cores)



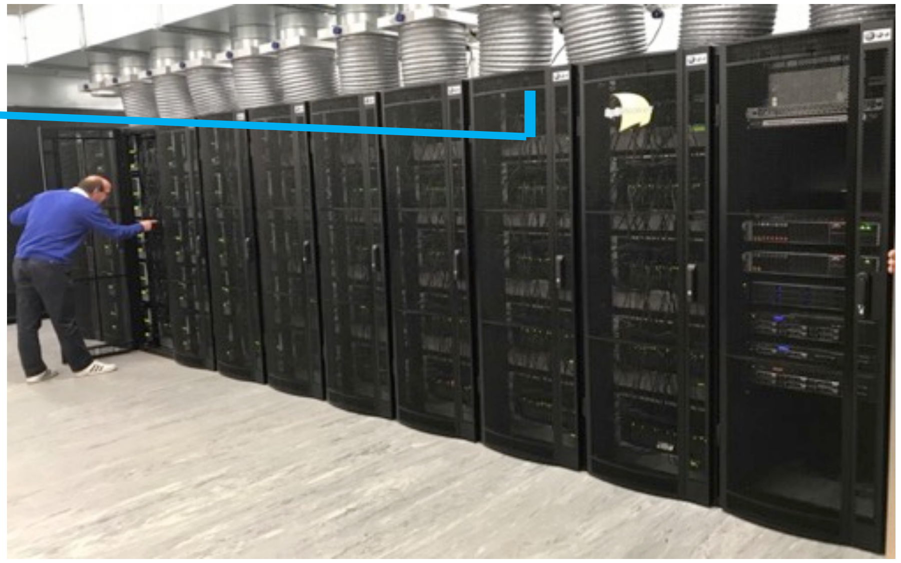


SpiNNaker chip
(18 ARM cores)



- HBP platform
  - 1M cores
  - 11 cabinets (including server)

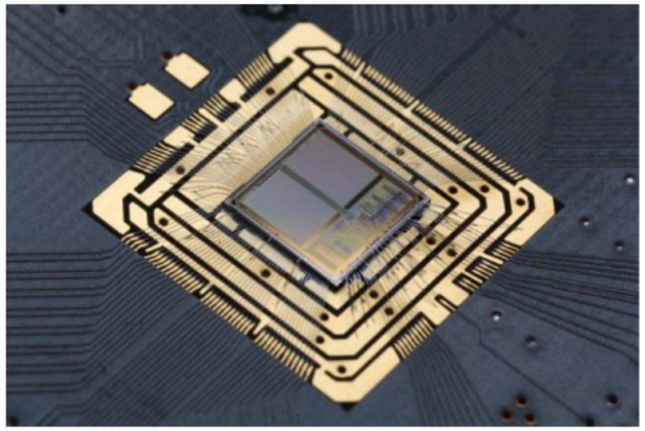
SpiNNaker racks
(1M ARM cores)

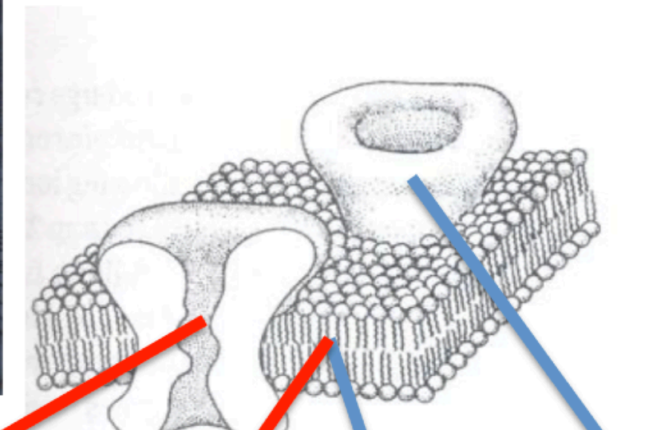


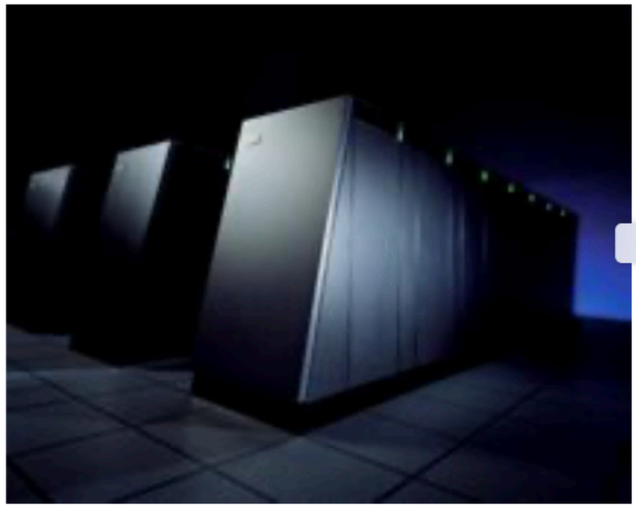
https://ebrains.eu/service/neuromorphic-computing#SpiNNaker

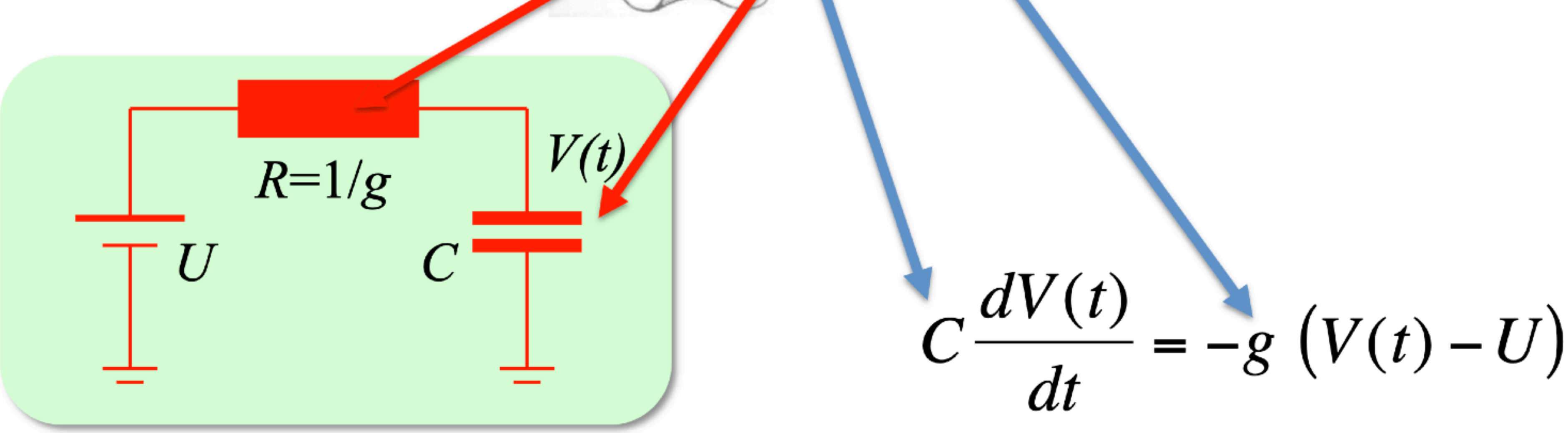
BrainScaleS

- custom-designed analog chips for emulating spiking neurons









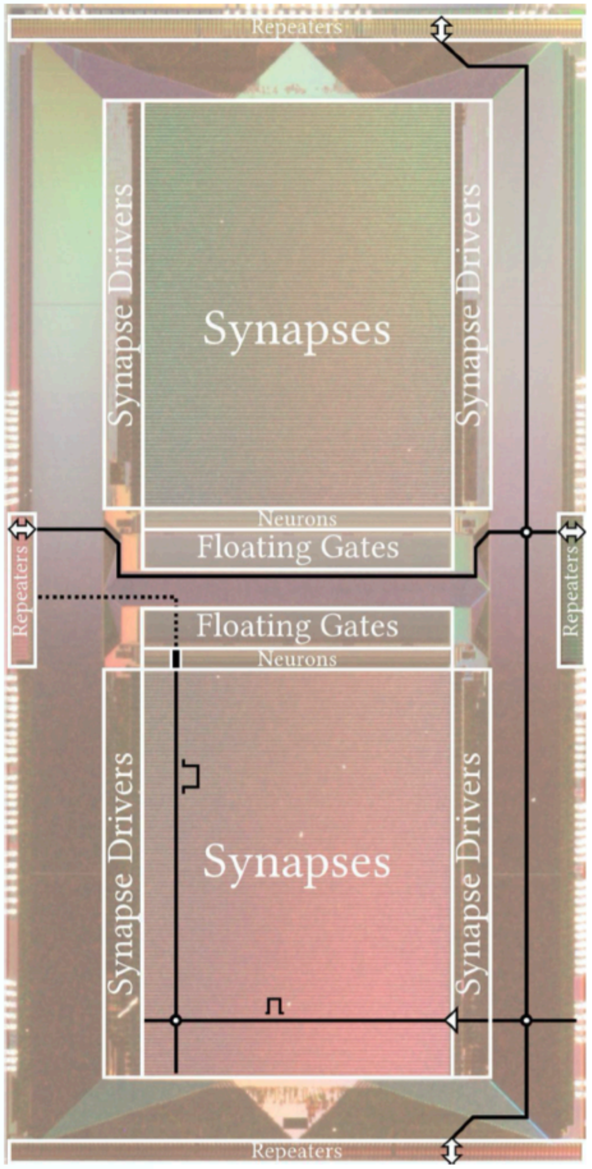
$$C\frac{dV(t)}{dt} = -g\left(V(t)-U\right)$$

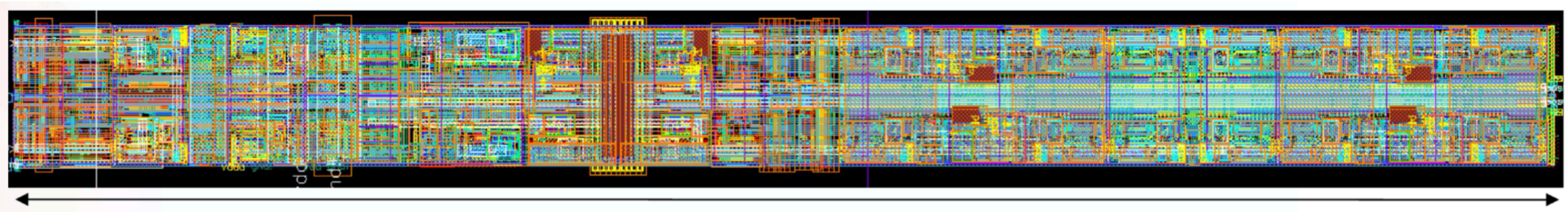
Physical Model

Mathematical Model

BrainScaleS

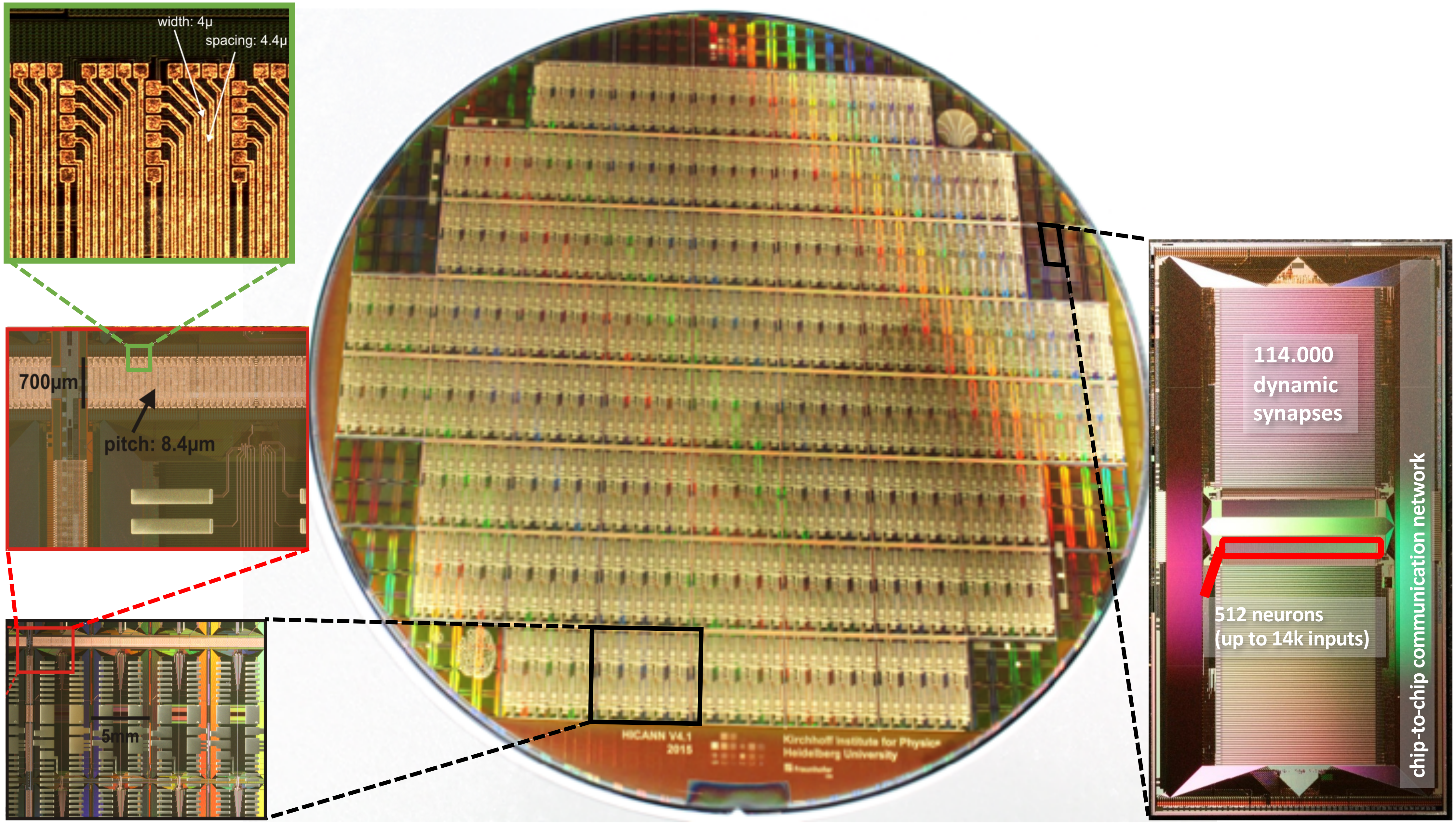
- Adaptive Exponential Integrate and Fire (AdEx) Model
- Accelerated dynamics compared to biological real-time (1000-10000 x)
- 512 analog neurons,110000 plastic synapses
- Digital communication → mixed-signal system
- Sparse crossbar switches connecting buses→ programmable network connections
- Analog parameter storage (floating gates)

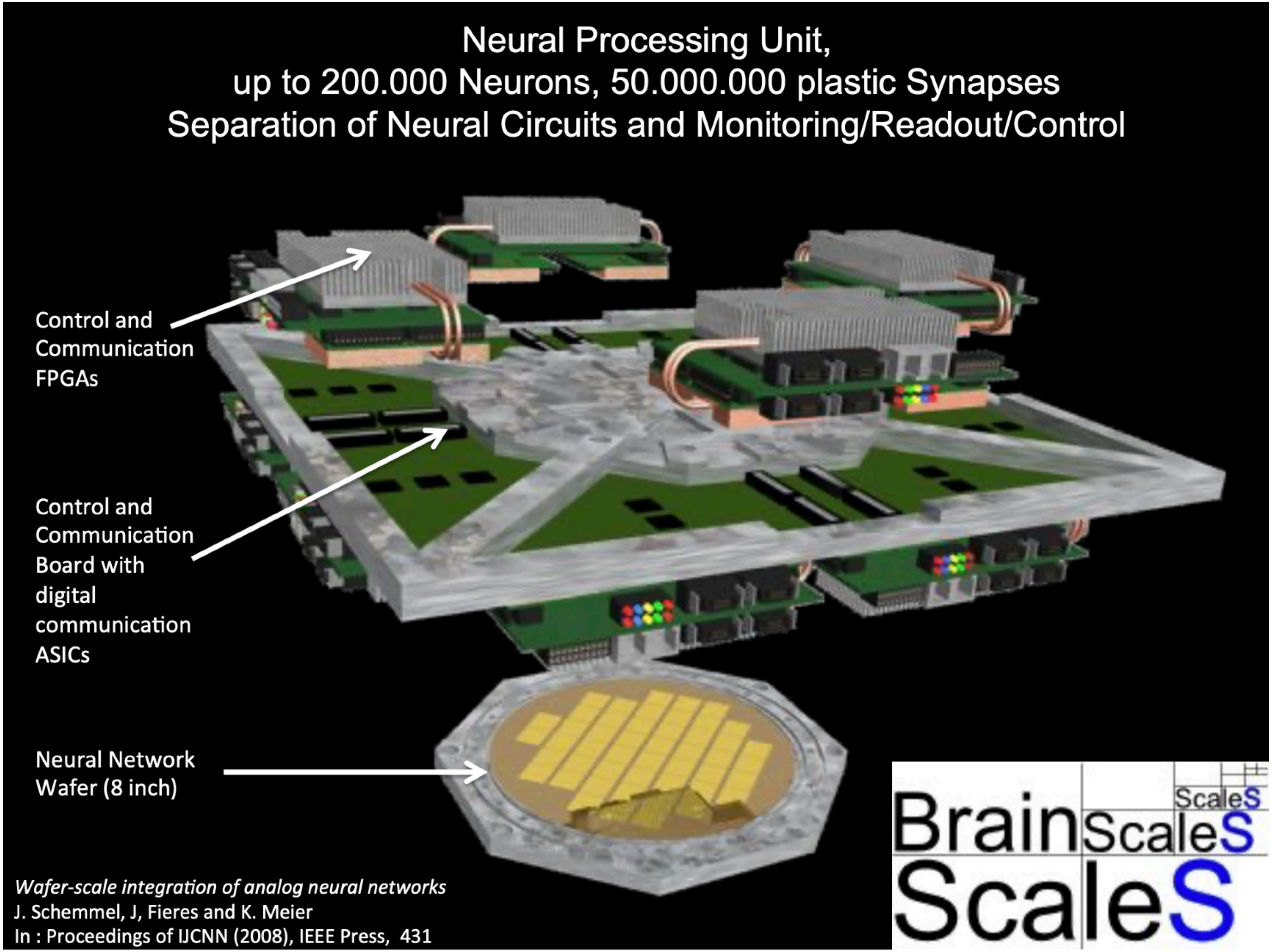




layout drawing of two neurons: 150x20 µm²

BrainScaleS





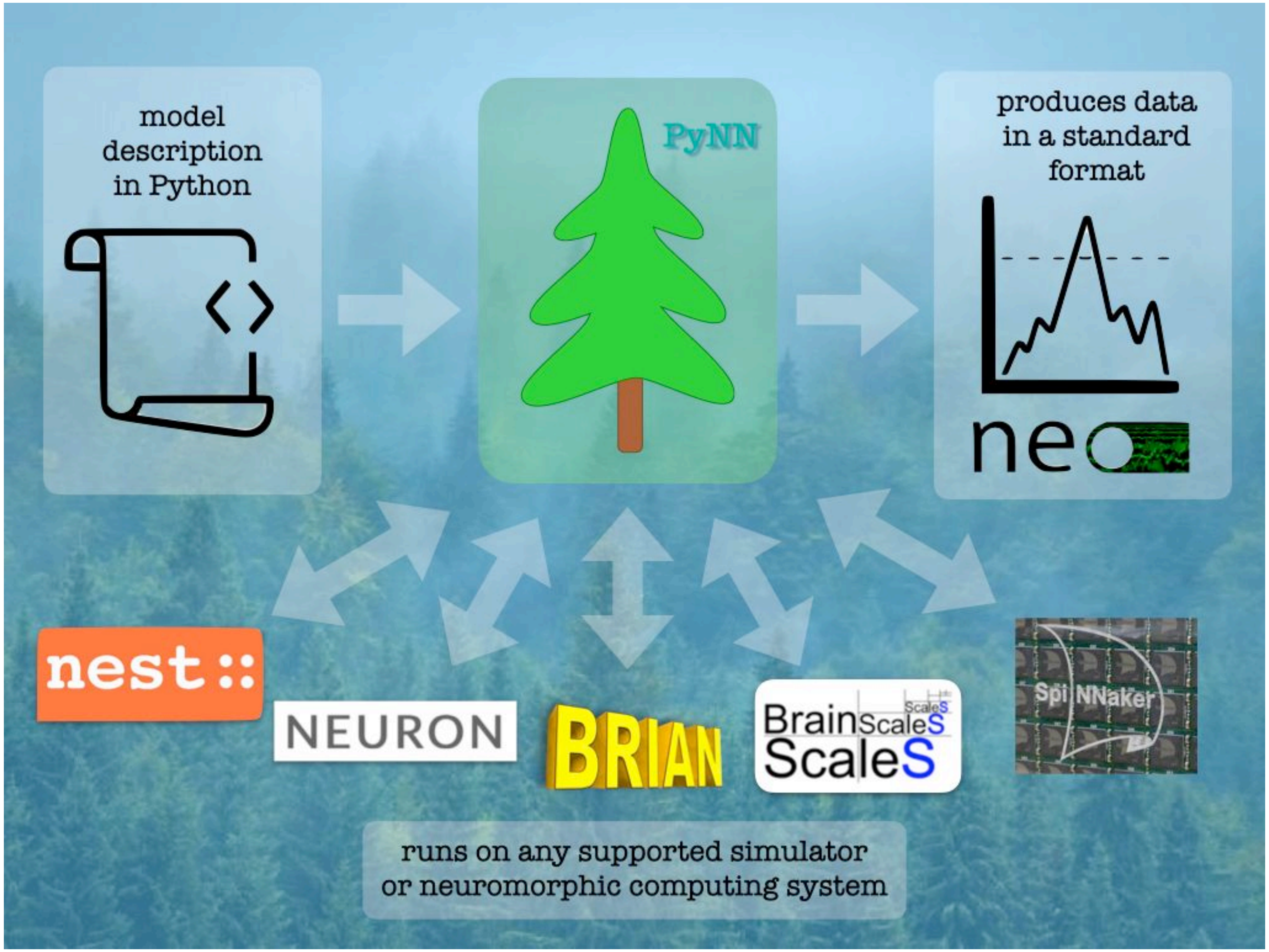
Neural Processing Unit,
up to 200.000 Neurons, 50.000.000 plastic Synapses
Separation of Neural Circuits and Monitoring/Readout/Control

Control and Communication FPGAs
Control and Communication Board with digital communication ASICs
Neural Network Wafer (8 inch)
BrainScaleS
ScaleS

Wafer-scale integration of analog neural networks
J. Schemmel, J, Fieres and K. Meier
In : Proceedings of IJCNN (2008), IEEE Press, 431

https://ebrains.eu/service/neuromorphic-computing#BrainScaleS

Programming neuromorphic computers


model description in Python
PyNN
produces data in a standard format
neo
nest::
NEURON
BRIAN
BrainScaleS
SpiNNaker
runs on any supported simulator or neuromorphic computing system

https://ebrains.eu/service/PyNN

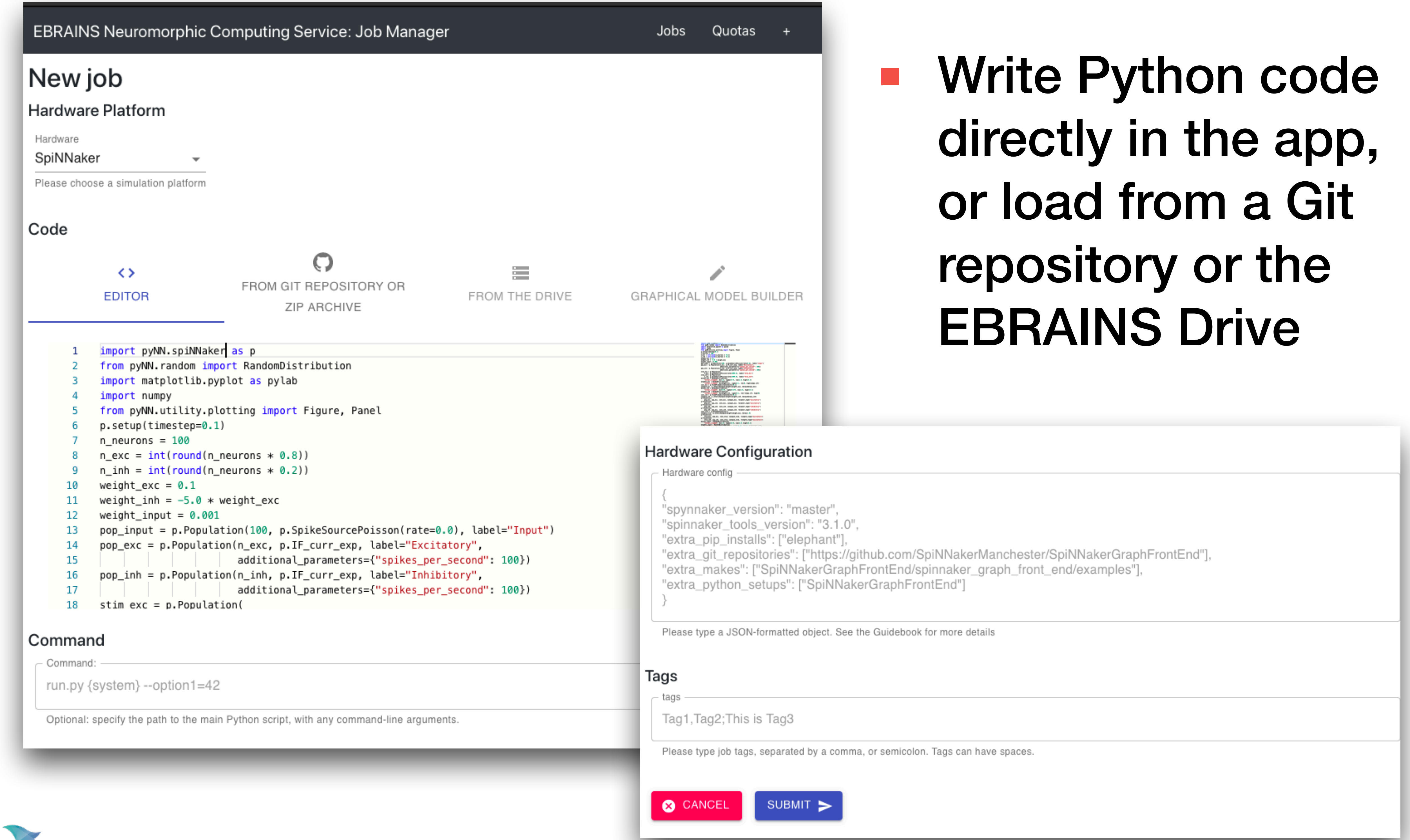
Programming neuromorphic computers

```
import pyNN.nest as simulator
import pyNN.neuron as simulator
import pyNN.brainscales as simulator
import pyNN.spiNNaker as simulator

cell_type = …

p1 = simulator.Population(size1, cell_type, structure)
p2 = simulator.Population(size2, another_cell_type,
                          structure)

all = p1 + p2
all.record([‘spikes’, ‘v’])

connections = simulator.Projection(
    p1, p2,
    connection_rule,
    synapse_type)

simulator.run(1000.0)
```

How to access SpiNNaker and BrainScaleS

- Batch mode
  - Web app
  - Python client (Jupyter notebooks)
  - Cmdline client
  - UNICORE
- Interactive mode
  - Jupyter notebooks

Job Manager app



EBRAINS Neuromorphic Computing Service: Job Manager
Jobs
Quotas
+

New job

Hardware Platform

Hardware: SpiNNaker

Please choose a simulation platform

Code

EDITOR
FROM GIT REPOSITORY OR ZIP ARCHIVE
FROM THE DRIVE
GRAPHICAL MODEL BUILDER

```python
import pyNN.spiNNaker as p
from pyNN.random import RandomDistribution
import matplotlib.pyplot as pylab
import numpy
from pyNN.utility.plotting import Figure, Panel
p.setup(timestep=0.1)
n_neurons = 100
n_exc = int(round(n_neurons * 0.8))
n_inh = int(round(n_neurons * 0.2))
weight_exc = 0.1
weight_inh = -5.0 * weight_exc
weight_input = 0.001
pop_input = p.Population(100, p.SpikeSourcePoisson(rate=0.0), label="Input")
pop_exc = p.Population(n_exc, p.IF_curr_exp, label="Excitatory",
                       additional_parameters={"spikes_per_second": 100})
pop_inh = p.Population(n_inh, p.IF_curr_exp, label="Inhibitory",
                       additional_parameters={"spikes_per_second": 100})
stim_exc = p.Population(
```

Command

Command: run.py {system} --option1=42

Optional: specify the path to the main Python script, with any command-line arguments.

Hardware Configuration

Hardware config

```
{
"spynnaker_version": "master",
"spinnaker_tools_version": "3.1.0",
"extra_pip_installs": ["elephant"],
"extra_git_repositories": ["https://github.com/SpiNNakerManchester/SpiNNakerGraphFrontEnd"],
"extra_makes": ["SpiNNakerGraphFrontEnd/spinnaker_graph_front_end/examples"],
"extra_python_setups": ["SpiNNakerGraphFrontEnd"]
}
```

Please type a JSON-formatted object. See the Guidebook for more details

Tags

tags: Tag1,Tag2;This is Tag3

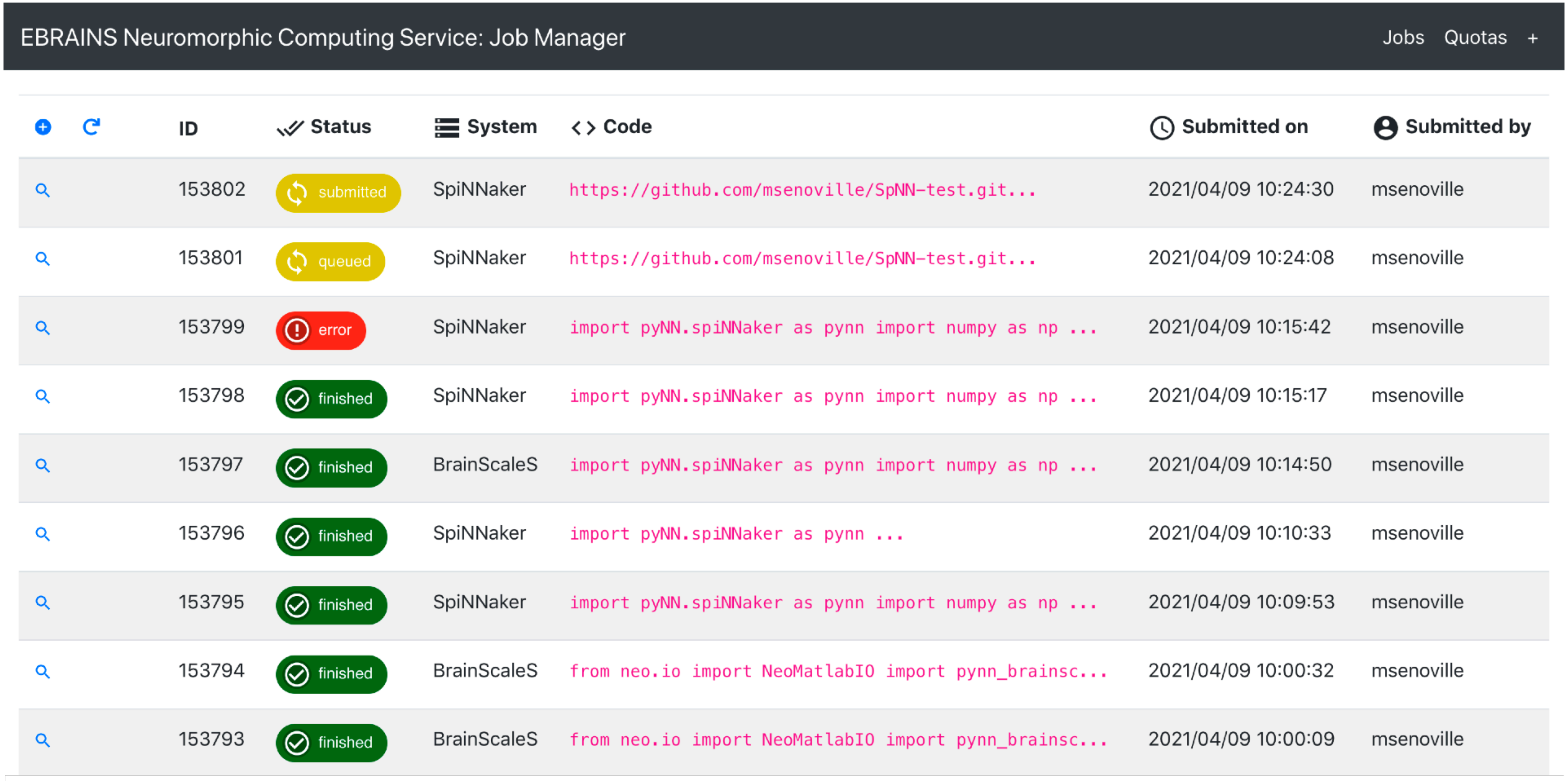
Please type job tags, separated by a comma, or semicolon. Tags can have spaces.

- **Write Python code directly in the app, or load from a Git repository or the EBRAINS Drive**

FENIX RI

https://job-manager.hbpneuromorphic.eu/

Job Manager app

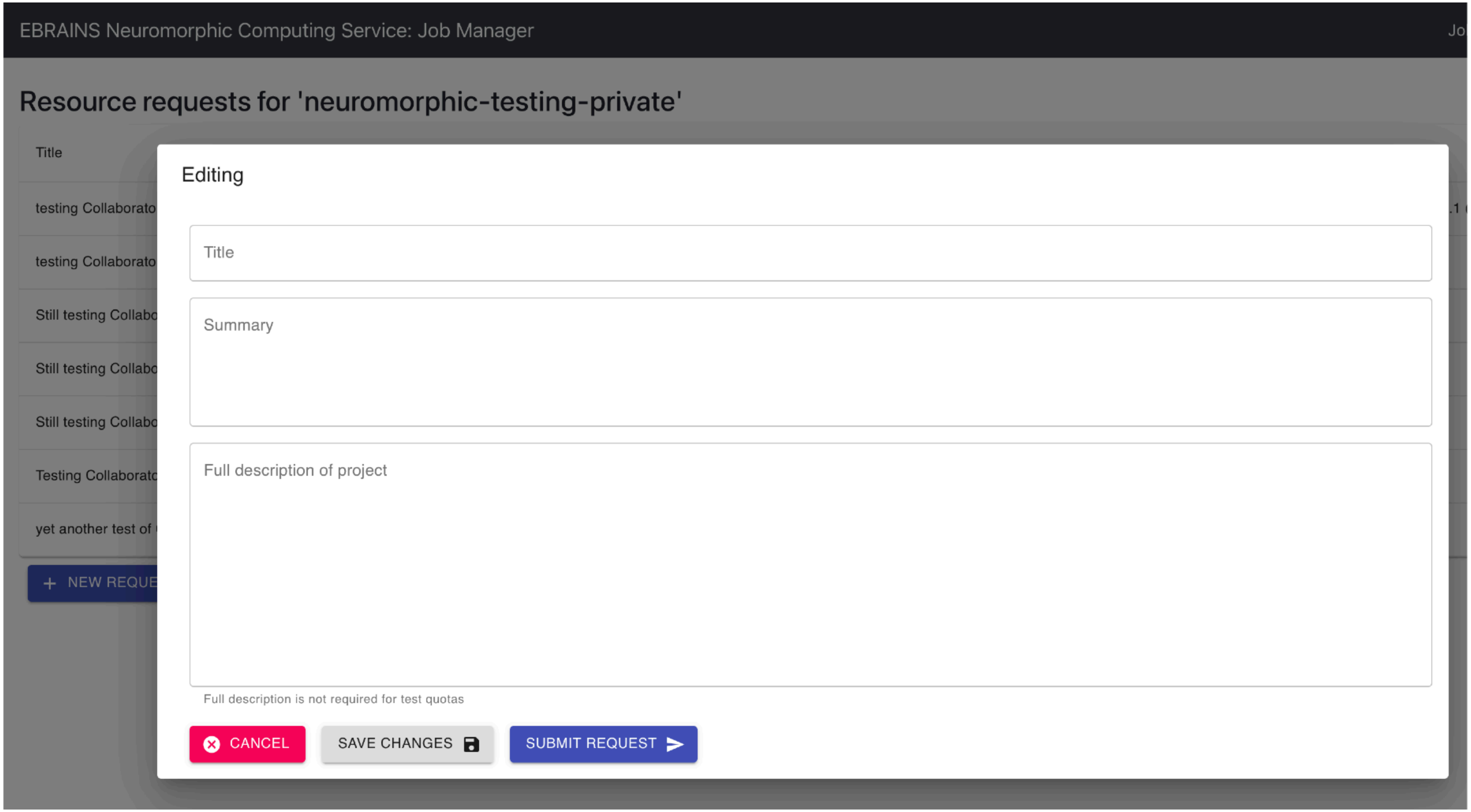


EBRAINS Neuromorphic Computing Service: Job Manager

Jobs Quotas +

⊕ ↻	ID	Status	System	Code	Submitted on	Submitted by
🔍	153802	submitted	SpiNNaker	`https://github.com/msenoville/SpNN-test.git...`	2021/04/09 10:24:30	msenoville
🔍	153801	queued	SpiNNaker	`https://github.com/msenoville/SpNN-test.git...`	2021/04/09 10:24:08	msenoville
🔍	153799	error	SpiNNaker	`import pyNN.spiNNaker as pynn import numpy as np ...`	2021/04/09 10:15:42	msenoville
🔍	153798	finished	SpiNNaker	`import pyNN.spiNNaker as pynn import numpy as np ...`	2021/04/09 10:15:17	msenoville
🔍	153797	finished	BrainScaleS	`import pyNN.spiNNaker as pynn import numpy as np ...`	2021/04/09 10:14:50	msenoville
🔍	153796	finished	SpiNNaker	`import pyNN.spiNNaker as pynn ...`	2021/04/09 10:10:33	msenoville
🔍	153795	finished	SpiNNaker	`import pyNN.spiNNaker as pynn import numpy as np ...`	2021/04/09 10:09:53	msenoville
🔍	153794	finished	BrainScaleS	`from neo.io import NeoMatlabIO import pynn_brainsc...`	2021/04/09 10:00:32	msenoville
🔍	153793	finished	BrainScaleS	`from neo.io import NeoMatlabIO import pynn_brainsc...`	2021/04/09 10:00:09	msenoville

Job Manager app



EBRAINS Neuromorphic Computing Service: Job Manager

Resource requests for 'neuromorphic-testing-private'

Editing

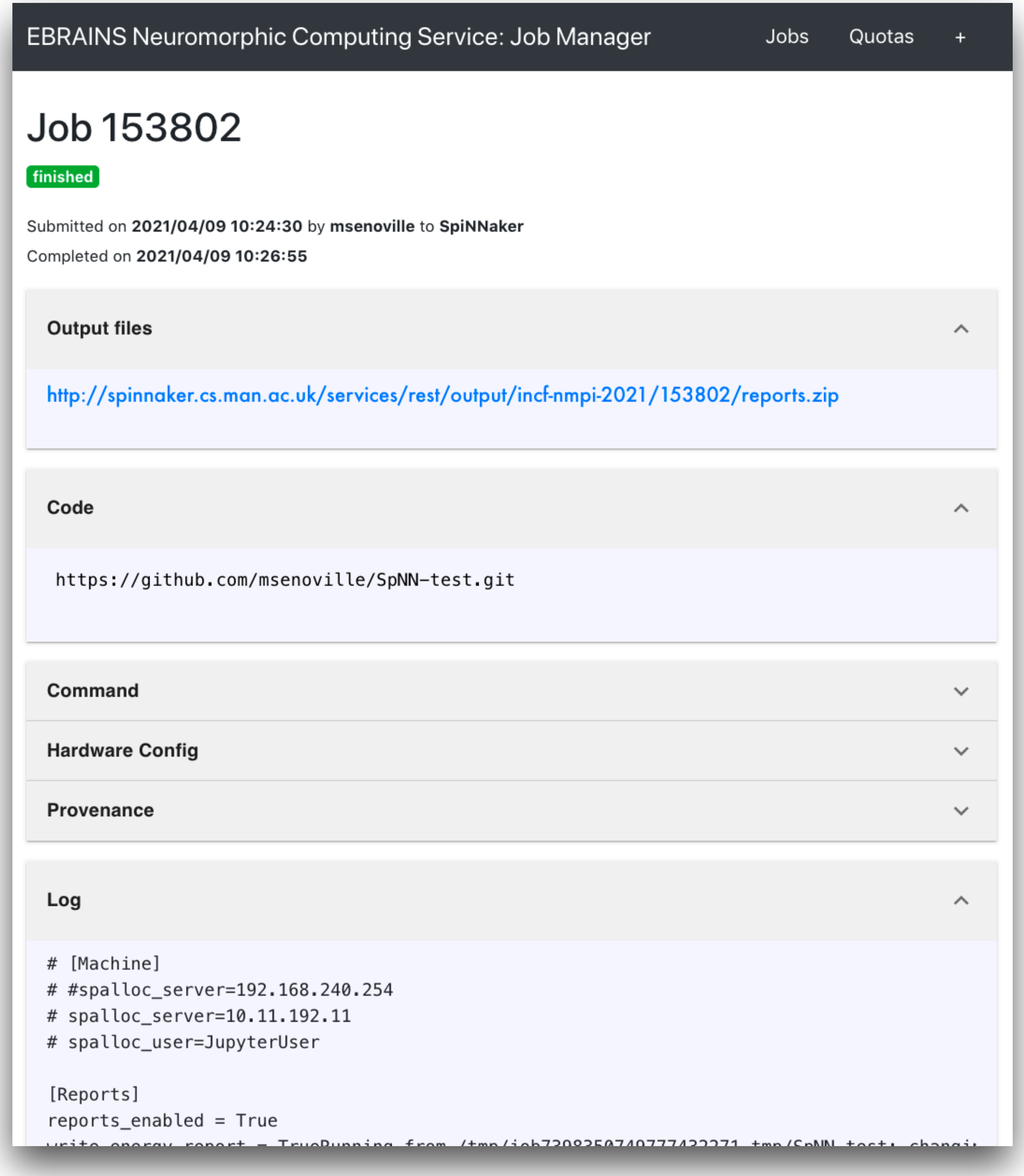
Title

Summary

Full description of project

Full description is not required for test quotas

Job Manager app



Job 153802

Submitted on **2021/04/09 10:24:30** by **msenoville** to **SpiNNaker**
Completed on **2021/04/09 10:26:55**

Output files

http://spinnaker.cs.man.ac.uk/services/rest/output/incf-nmpi-2021/153802/reports.zip

Code

```
https://github.com/msenoville/SpNN-test.git
```

Command

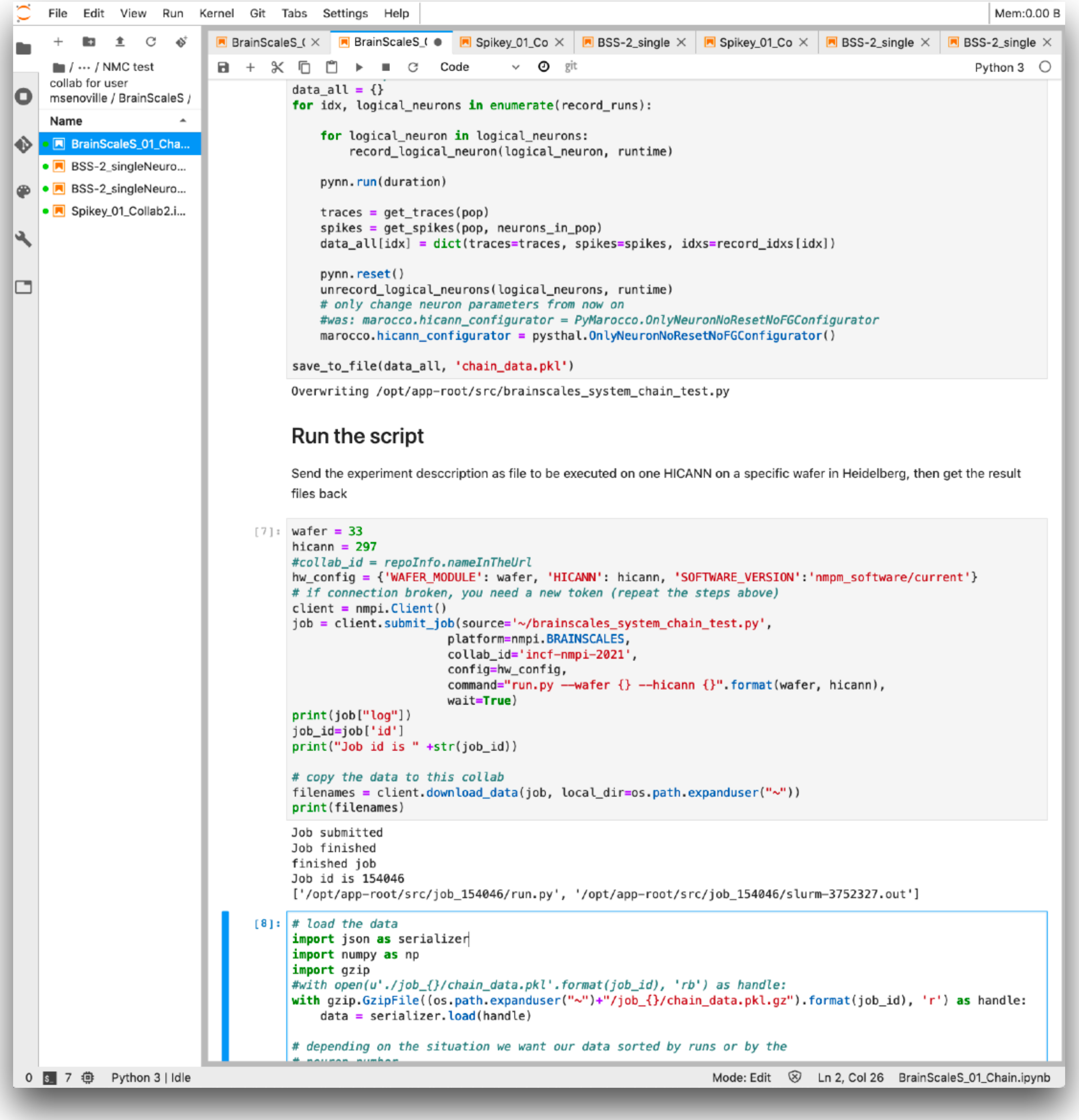
Hardware Config

Provenance

Log

```
# [Machine]
# #spalloc_server=192.168.240.254
# spalloc_server=10.11.192.11
# spalloc_user=JupyterUser

[Reports]
reports_enabled = True
write_energy_report = TrueRunning from /tmp/job7398350749777432271.tmp/SpNN-test; changi
```

Python client



```python
data_all = {}
for idx, logical_neurons in enumerate(record_runs):

    for logical_neuron in logical_neurons:
        record_logical_neuron(logical_neuron, runtime)

    pynn.run(duration)

    traces = get_traces(pop)
    spikes = get_spikes(pop, neurons_in_pop)
    data_all[idx] = dict(traces=traces, spikes=spikes, idxs=record_idxs[idx])

    pynn.reset()
    unrecord_logical_neurons(logical_neurons, runtime)
    # only change neuron parameters from now on
    #was: marocco.hicann_configurator = PyMarocco.OnlyNeuronNoResetNoFGConfigurator
    marocco.hicann_configurator = pysthal.OnlyNeuronNoResetNoFGConfigurator()

save_to_file(data_all, 'chain_data.pkl')
```

```
Overwriting /opt/app-root/src/brainscales_system_chain_test.py
```

Run the script

Send the experiment desccription as file to be executed on one HICANN on a specific wafer in Heidelberg, then get the result files back

```python
[7]: wafer = 33
hicann = 297
#collab_id = repoInfo.nameInTheUrl
hw_config = {'WAFER_MODULE': wafer, 'HICANN': hicann, 'SOFTWARE_VERSION':'nmpm_software/current'}
# if connection broken, you need a new token (repeat the steps above)
client = nmpi.Client()
job = client.submit_job(source='~/brainscales_system_chain_test.py',
                      platform=nmpi.BRAINSCALES,
                      collab_id='incf-nmpi-2021',
                      config=hw_config,
                      command="run.py --wafer {} --hicann {}".format(wafer, hicann),
                      wait=True)
print(job["log"])
job_id=job['id']
print("Job id is " +str(job_id))

# copy the data to this collab
filenames = client.download_data(job, local_dir=os.path.expanduser("~"))
print(filenames)
```

```
Job submitted
Job finished
finished job
Job id is 154046
['/opt/app-root/src/job_154046/run.py', '/opt/app-root/src/job_154046/slurm-3752327.out']
```

```python
[8]: # load the data
import json as serializer
import numpy as np
import gzip
#with open(u'./job_{}/chain_data.pkl'.format(job_id), 'rb') as handle:
with gzip.GzipFile((os.path.expanduser("~")+"/job_{}/chain_data.pkl.gz").format(job_id), 'r') as handle:
    data = serializer.load(handle)

# depending on the situation we want our data sorted by runs or by the
# neuron number
```

```
pip install hbp-neuromorphic-platform
```

Command-line client

- provided with the Python client
- develop code locally, submit to remote NMC system, results downloaded to local directory

```
$ nmpi run --help
Usage: nmpi run [OPTIONS] SCRIPT

  Run a simulation/emulation

Options:
  -p, --platform TEXT    SpiNNaker, BrainScaleS, BrainScaleS-2,
                         or Spikey
  -t, --tag TEXT         Add a tag to the job
  -b, --batch            Submit job then return immediately
  -o, --output-dir TEXT  Output directory
  -e, --server-env TEXT
  --help                 Show this message and exit.
```

UNICORE

- in development for BrainScaleS
- submit a neuromorphic computing job in the same way as any other HPC job

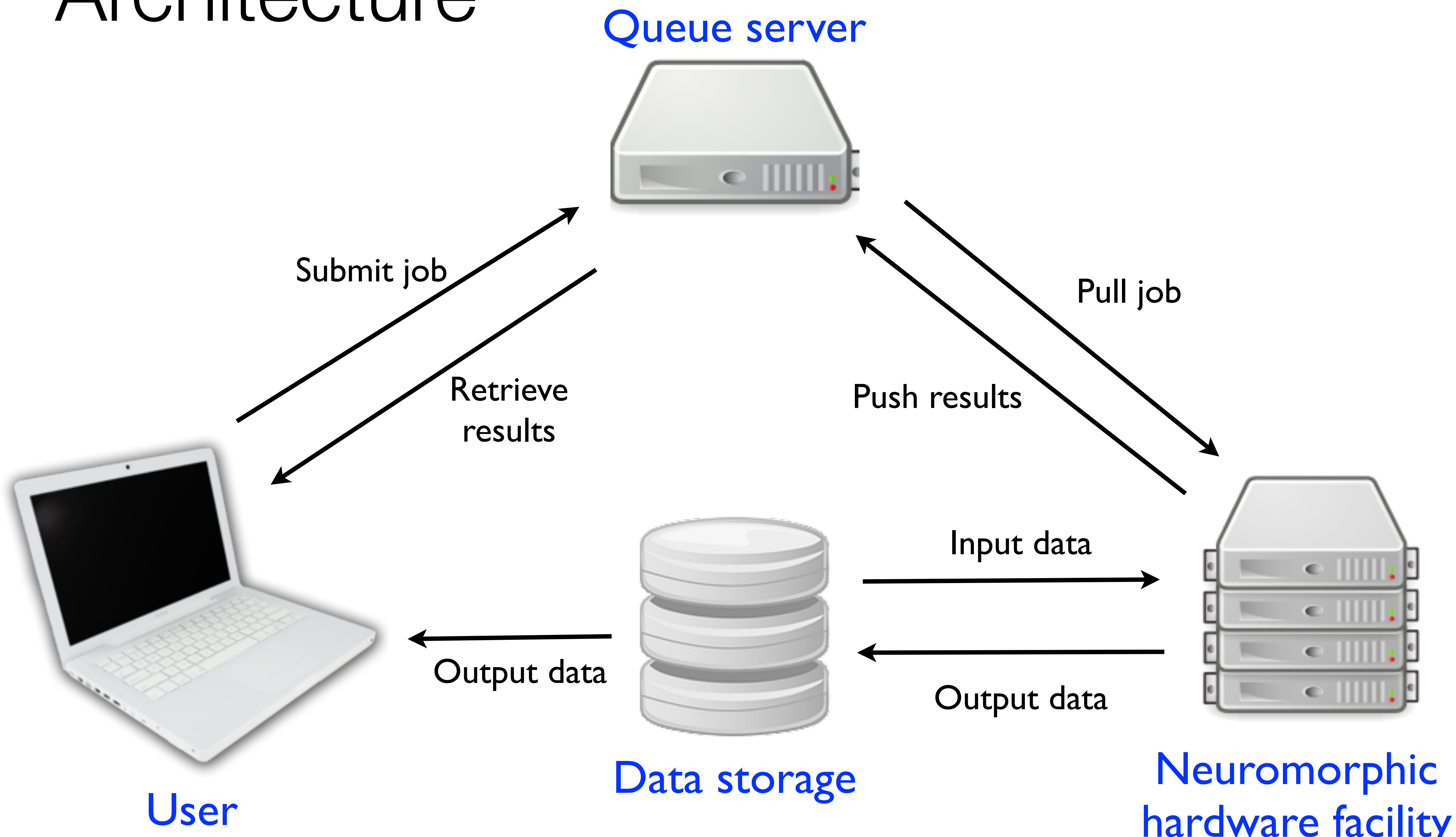
https://sourceforge.net/p/unicore/

Interactive access

- Jupyter notebooks running live in Manchester, giving interactive access to SpiNNaker
- PyNN commands are executed directly, allowing exploration and iterative development (*cf* submitting an entire PyNN script as a batch job)

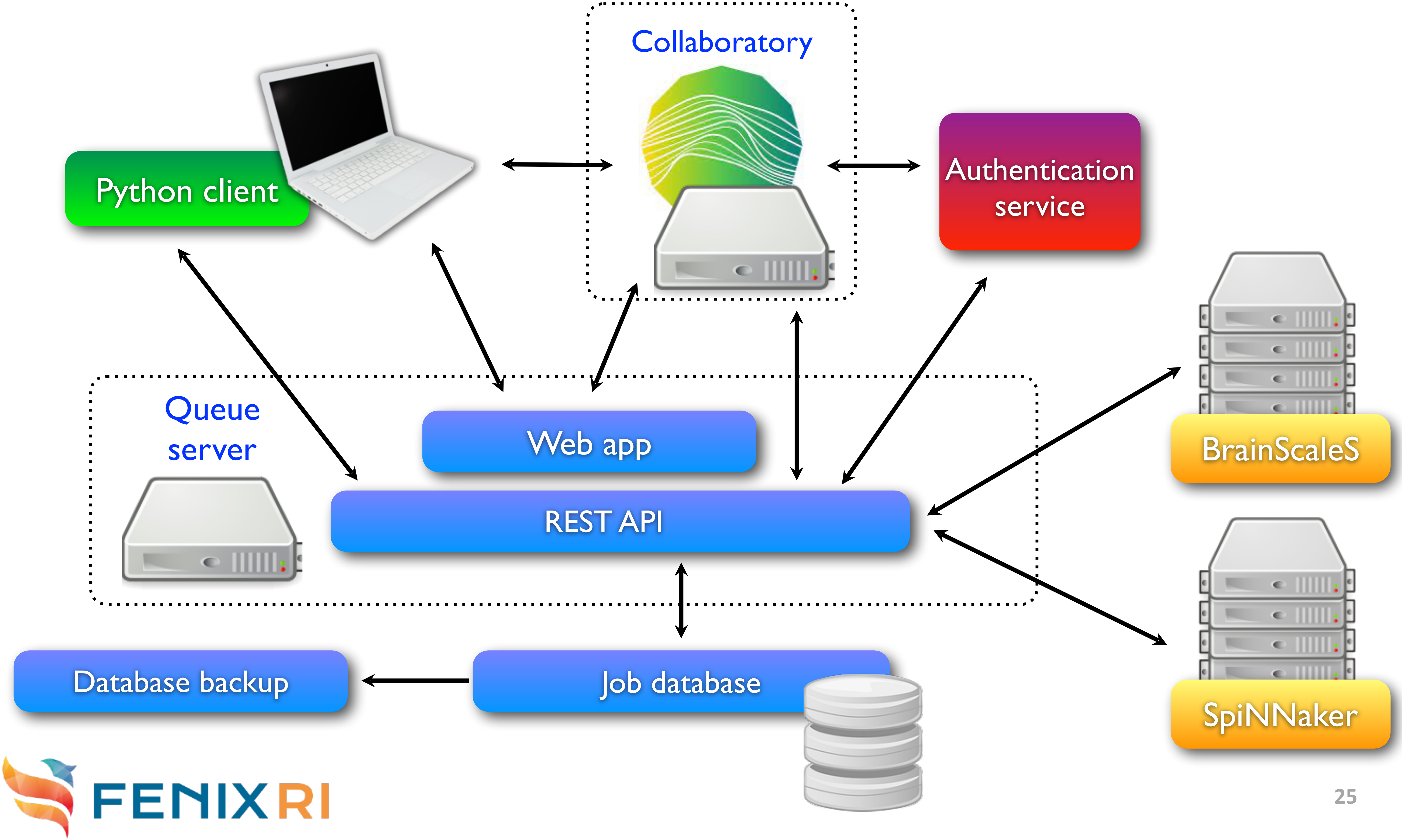
https://spinn-20.cs.man.ac.uk/

Architecture

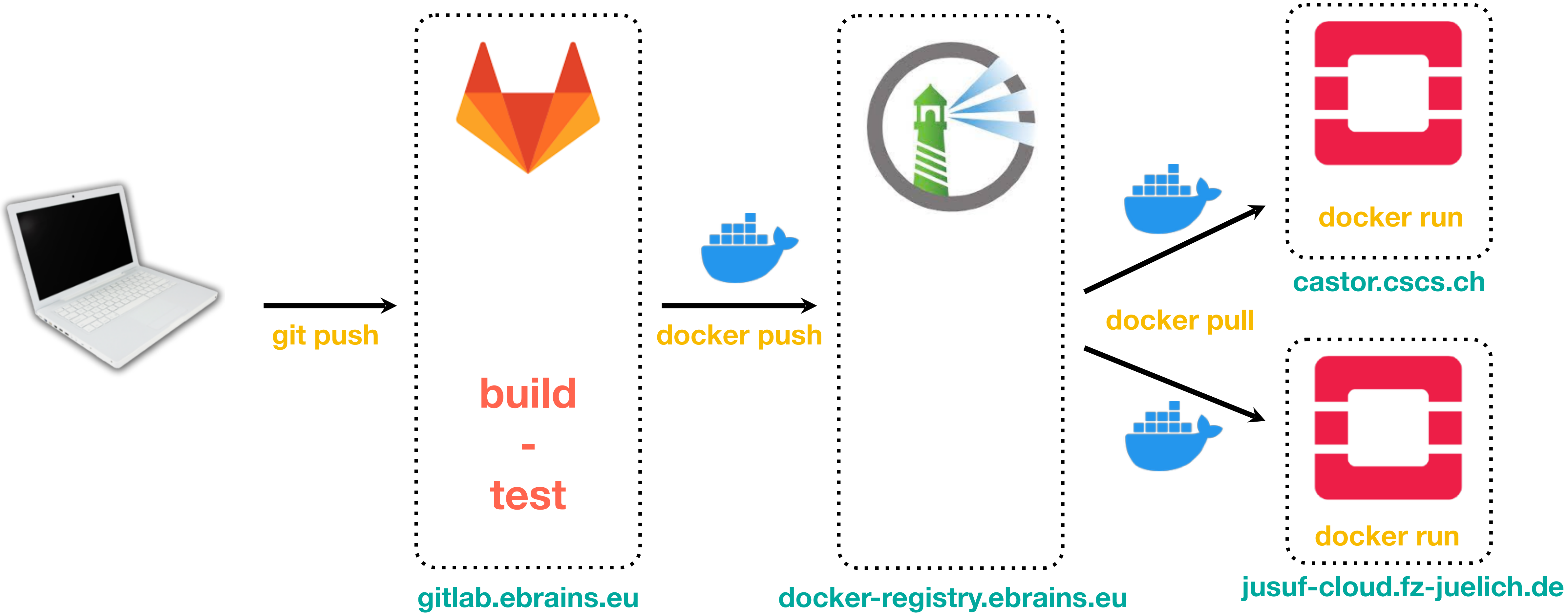


FENIX RI

Architecture



Deployment



git push

build

-

test

gitlab.ebrains.eu

docker push

docker-registry.ebrains.eu

docker pull

docker run

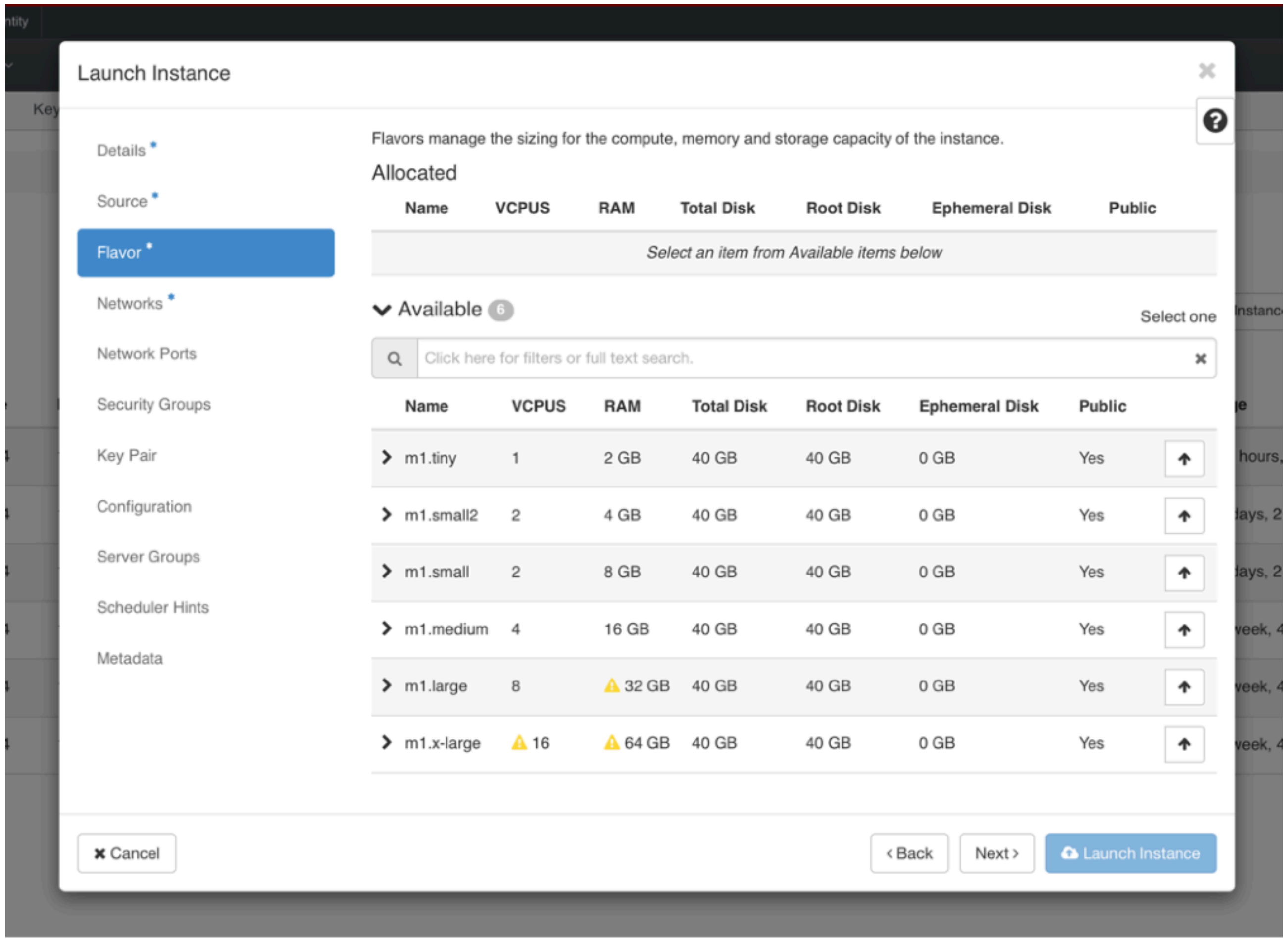
castor.cscs.ch

docker run

jusuf-cloud.fz-juelich.de

Castor, Pollux and Jusuf Cloud

- Production and staging servers running on Fenix VM Services at CSCS (Castor/ Pollux)
- Backup production servers running on Fenix VM Services at Jülich (JUSUF Cloud)
- Resources obtained via ICEI project
- VMs and volume storage are launched and configured manually via OpenStack dashboard and ssh



Launch Instance

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

Allocated

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
Select an item from Available items below						

Available 6

Select one

Click here for filters or full text search.

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
m1.tiny	1	2 GB	40 GB	40 GB	0 GB	Yes
m1.small2	2	4 GB	40 GB	40 GB	0 GB	Yes
m1.small	2	8 GB	40 GB	40 GB	0 GB	Yes
m1.medium	4	16 GB	40 GB	40 GB	0 GB	Yes
m1.large	8	⚠ 32 GB	40 GB	40 GB	0 GB	Yes
m1.x-large	⚠ 16	⚠ 64 GB	40 GB	40 GB	0 GB	Yes

Future plans / in progress

- increase automation of VM creation and Docker deployment using OpenStack API/ command line and Ansible
- automated failover between Castor and JUSUF
- centralised logging / monitoring

Contact me:

- andrew.davison@cnrs.fr
- Twitter: @apdavison