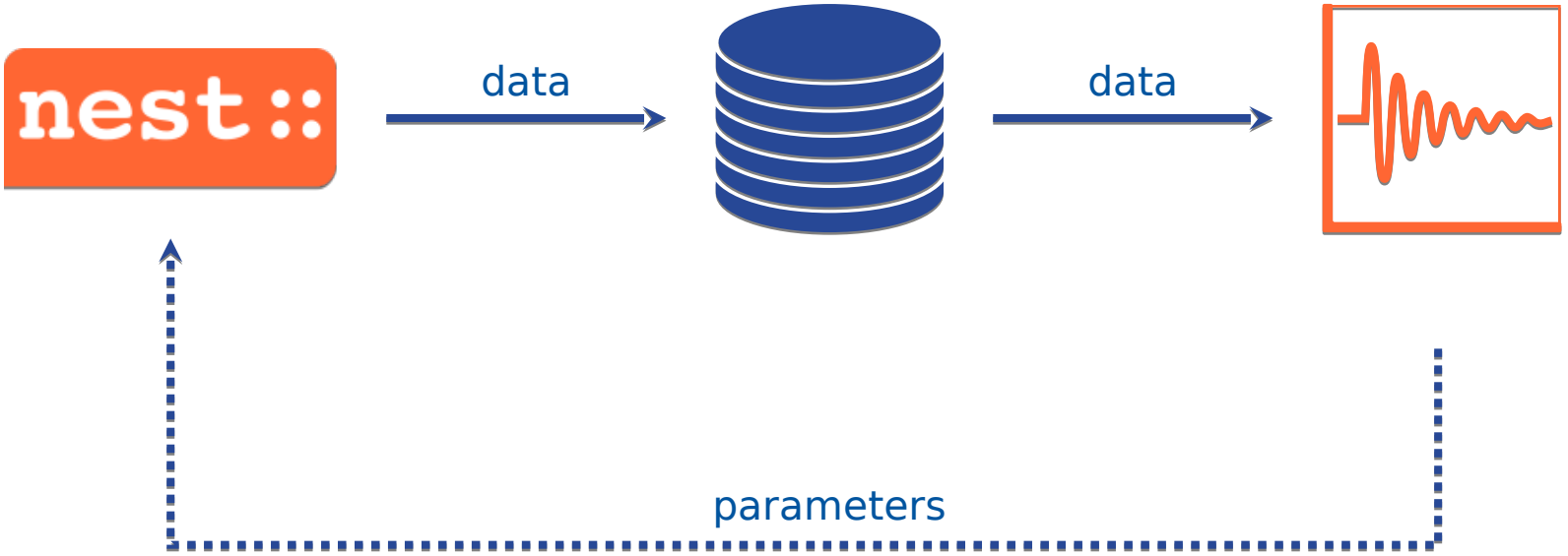
Streaming Live NEST Simulation Data Into Visualization and Analysis

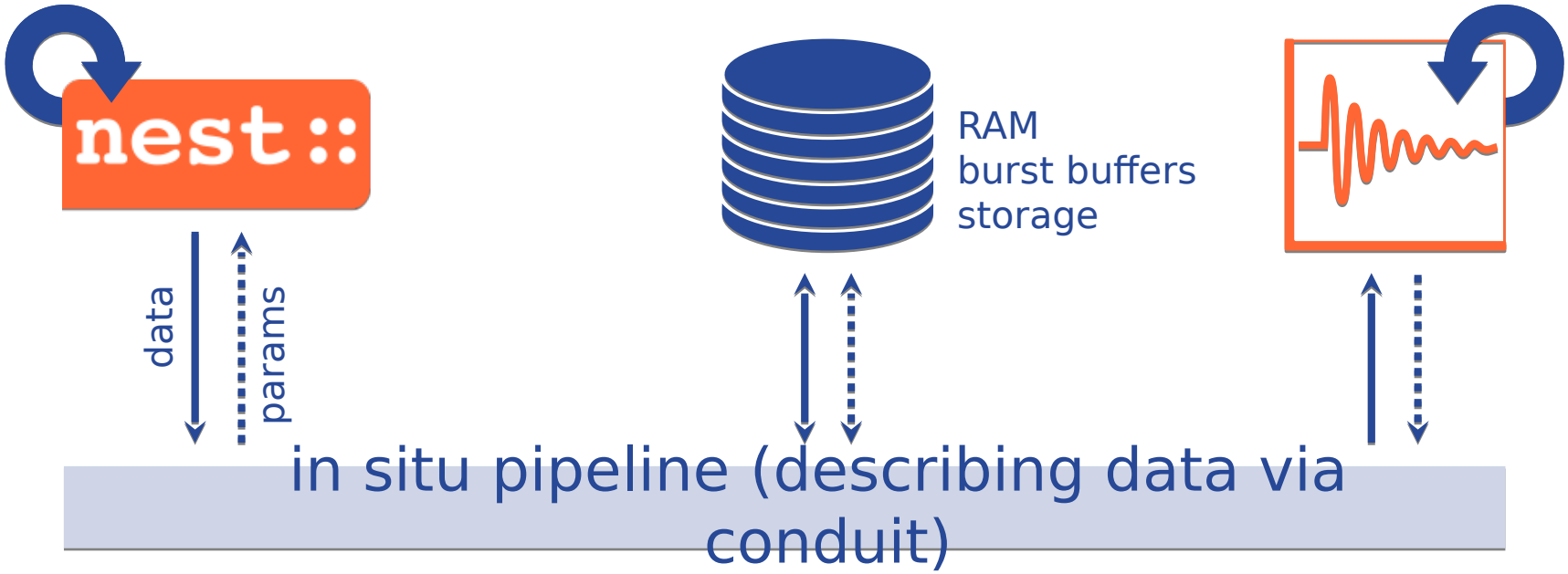
Simon Oehrl[1], Jan Müller[1], Jan Schnathmeier[1], Jochen Martin Eppler[2], Alexander Peyser[2], Hans Ekkehard Plesser[3], Benjamin Weyers[1], Bernd Hentschel[1], Torsten Kuhlen[1], Tom Vierjahn[1]

[1] RWTH Aachen University, Germany
[2] Forschungszentrum Jülich, Germany
[3] Norwegian University of Life Sciences, Ås

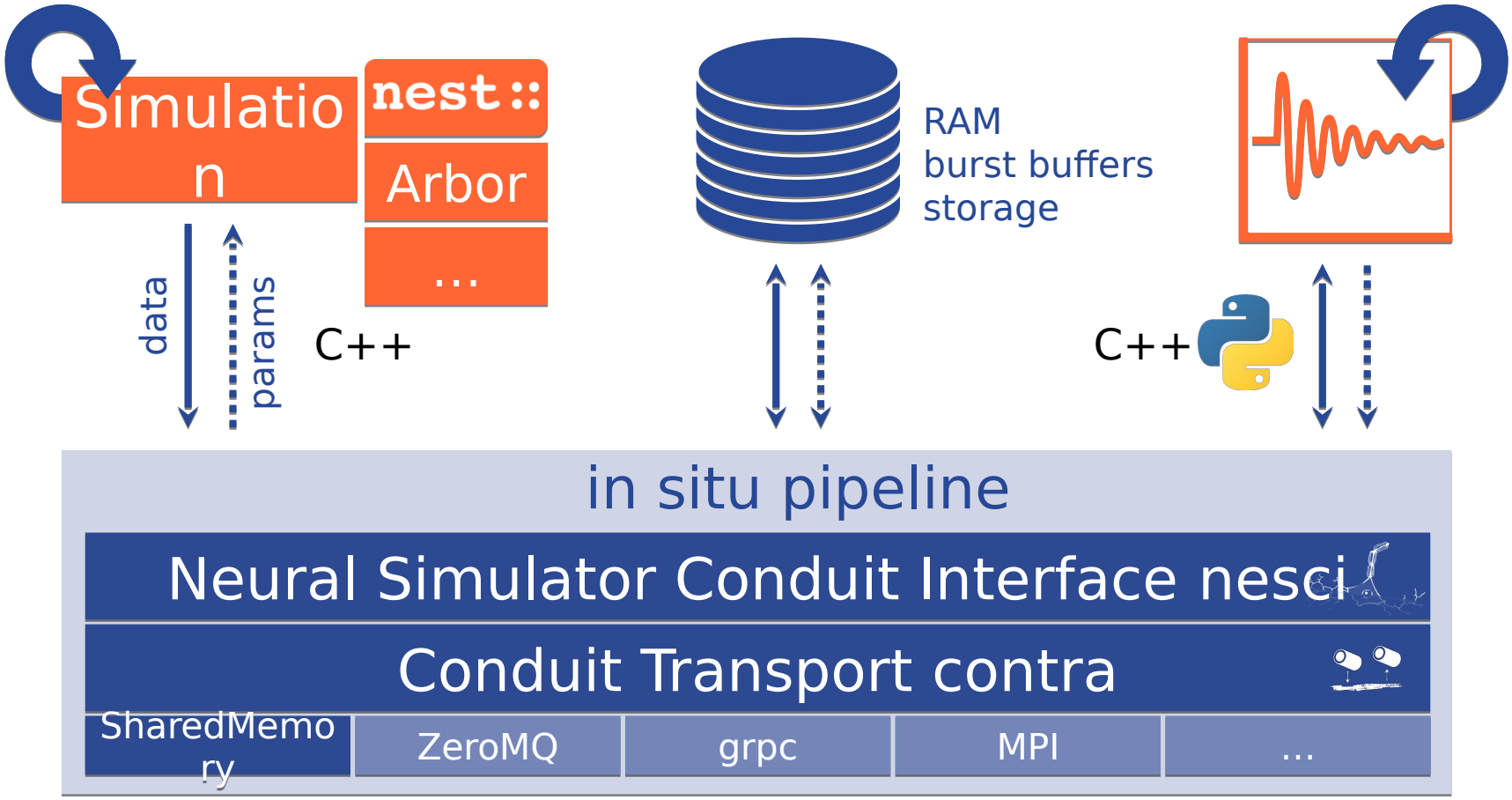
Motivation



In-situ Pipeline



In-situ Pipeline



Conduit – Simplified Data Exchange for HPC Simulations

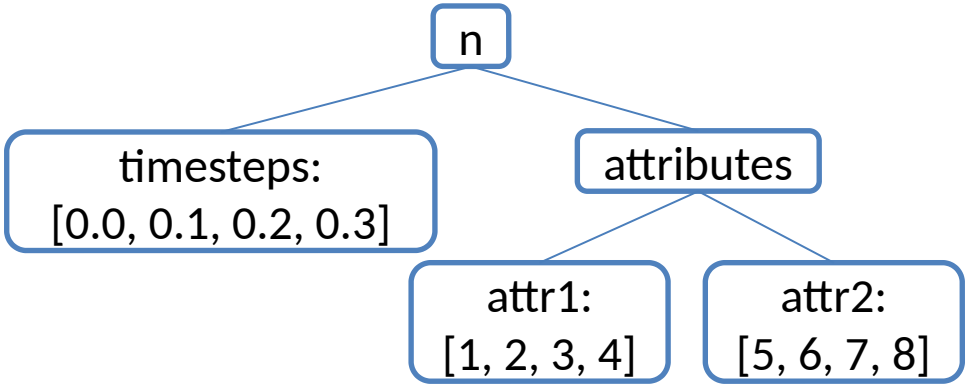
Conduit Provides:

- “A flexible way to describe hierarchal data”[1]
- “A sane API to access hierarchal data”[1]

C++

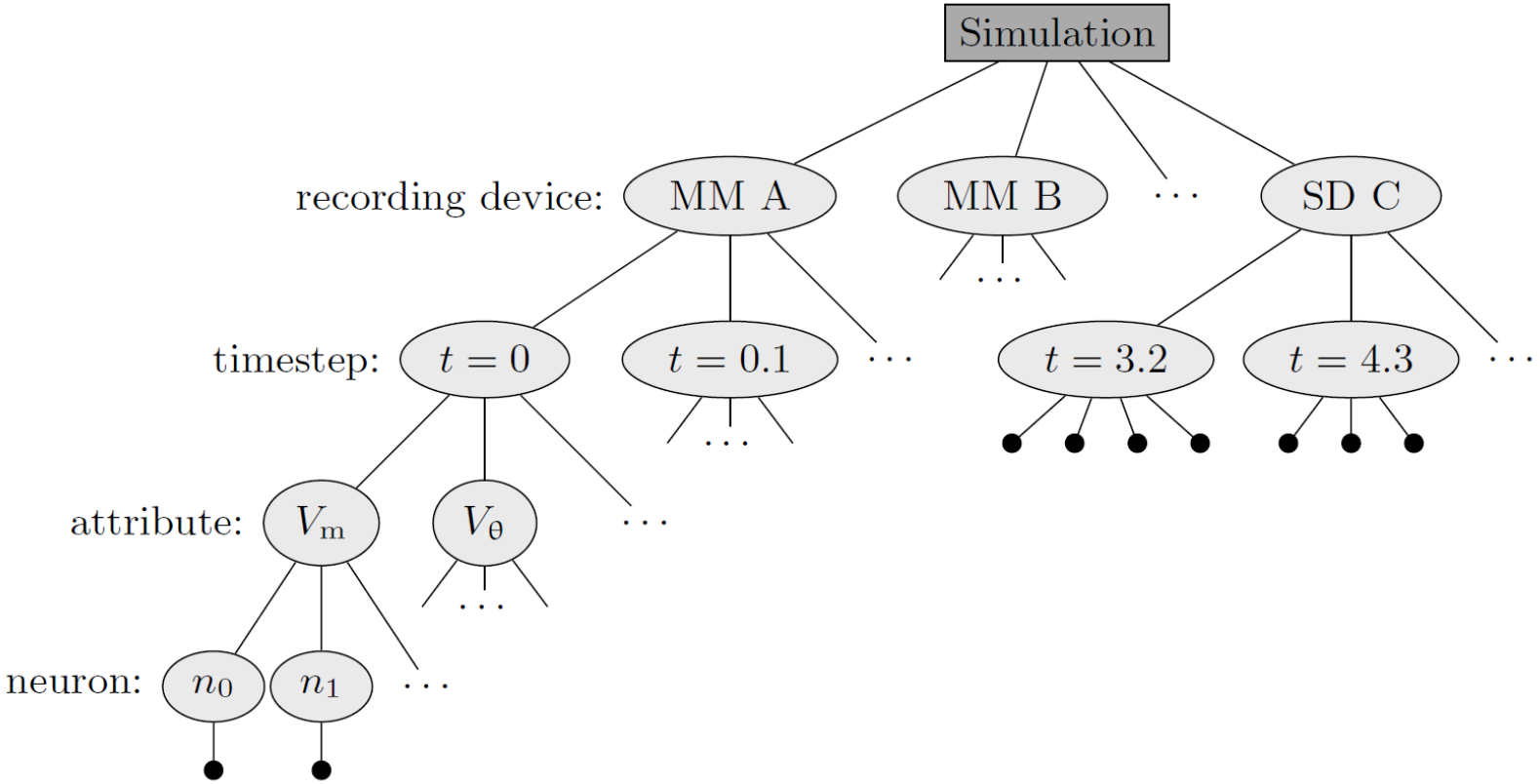
```
from conduit import Node

n = Node()
n['timesteps'] = [0.0, 0.1, 0.2,
0.3]
n['attributes/attr1'] = [1, 2, 3,
4]
n['attributes/attr2'] = [5, 6, 7,
8]
```

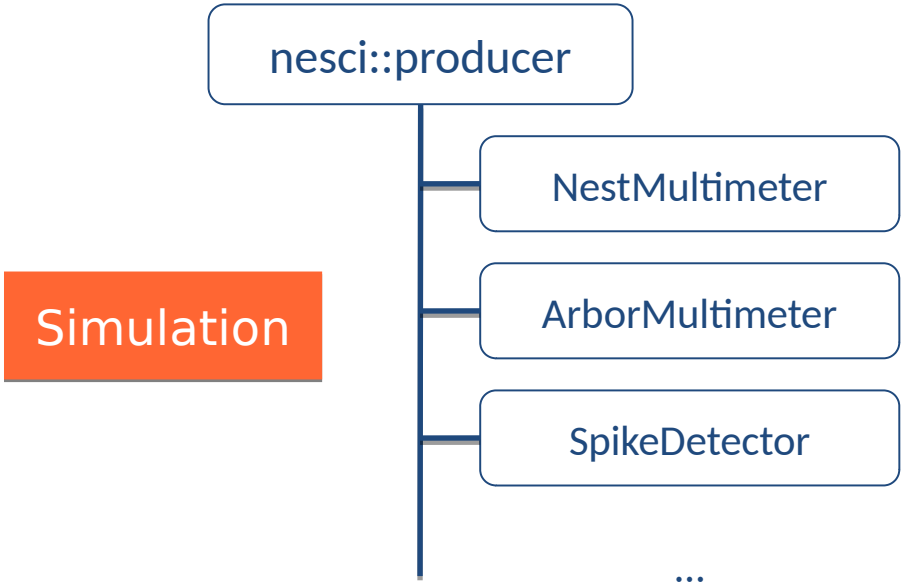


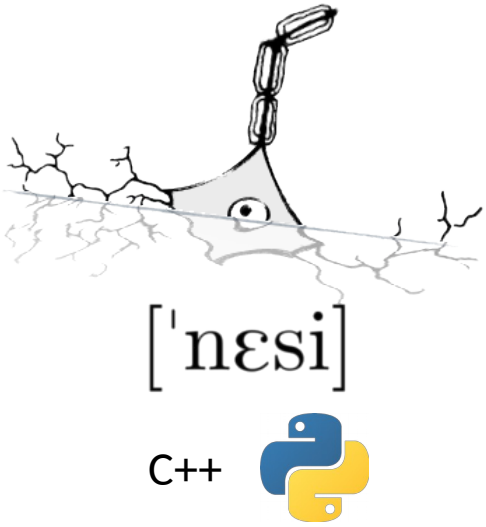
[1] http://llnl-conduit.readthedocs.io

Conduit – Simplified Data Exchange for HPC Simulations



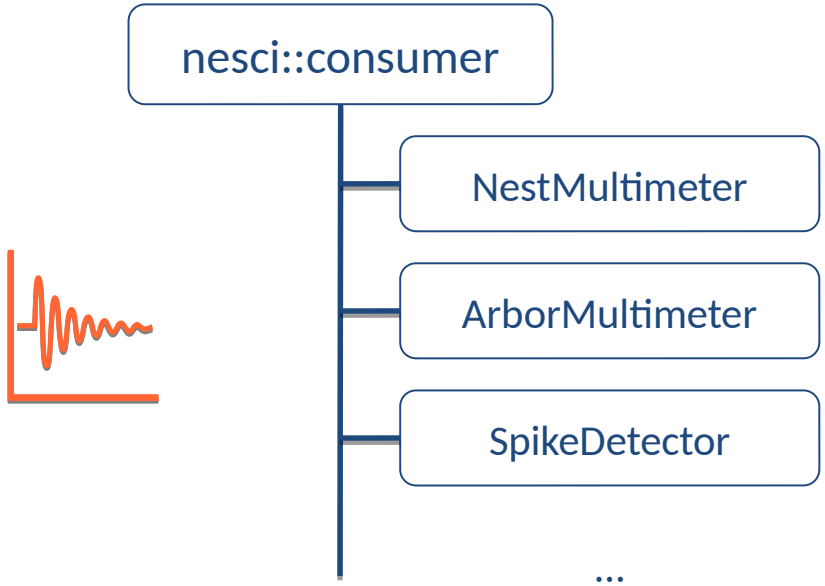
NESCI – Neuronal Simulator Conduit Interface

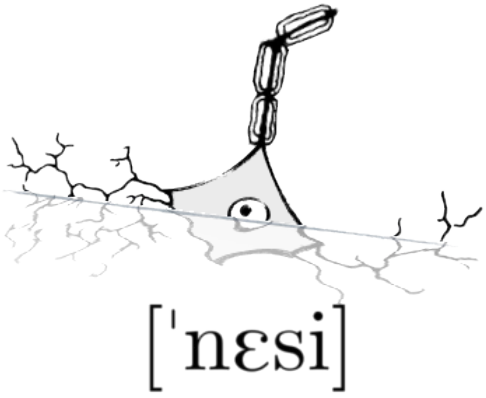





```
void Record(datum)
ConduitNode node()
```

NESCI – Neuronal Simulator Conduit Interface



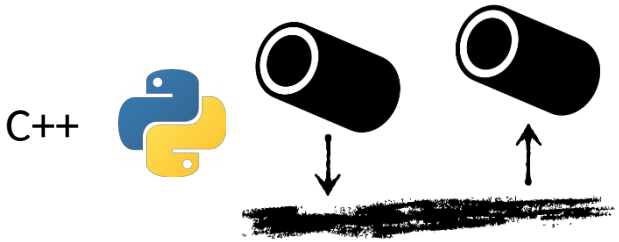


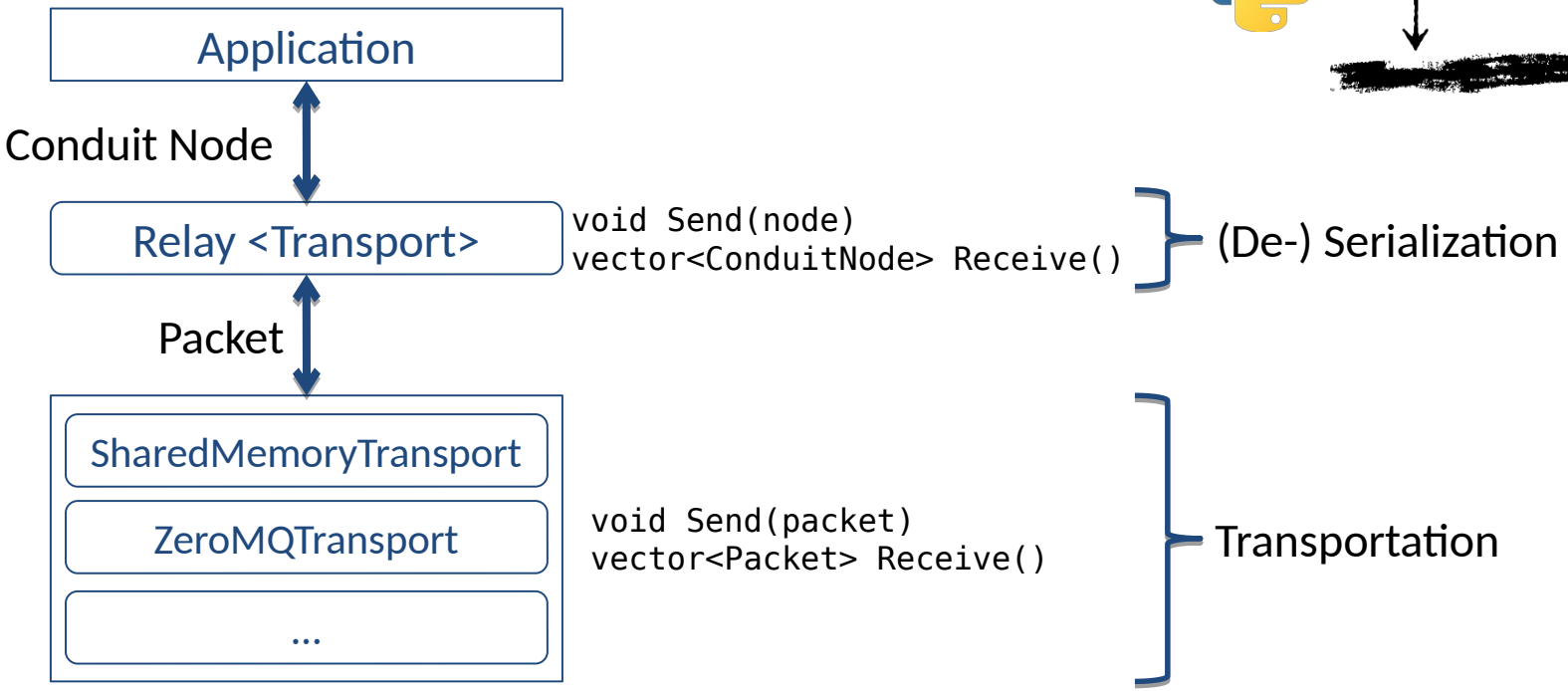
['nɛsi]

C++

```
void SetNode(node)
vector<string> GetTimesteps()
vector<string> GetAttributes(time)
vector<string> GetNeuronIds(time, attributes)
vector<double> GetTimestepData(time, attribute)
vector<double> GetTimeSeriesData(attribute, neuron_id)
double GetDatum(time, attribute, neuron_id)
```

CONTRA – Conduit Transport





NEST Recording Backend

```
contra::Relay<contra::SharedMemoryTransport> relay;
map<nest::RecordingDevice, nesci::producer::Device> recorders;
```

```
void write(device, event, values)
{
    datum = CreateNestMultimeterDatum(event, values);
    recorders[device].Record(datum);
}


void synchronize()
{
    for (recorder : recorders)
    {
        relay.Send(recorder.node);
        recorder->Clear();
    }
}
```

2D Visualization

```
def SetupStreaming(self):
  self.node = pycontra.Node()
  self.relay = pycontra.SharedMemoryTransportRelay()
  self.multimeter = pynesci.consumer.NestMultimeter('...')
  self.multimeter.SetNode(self.node)
```

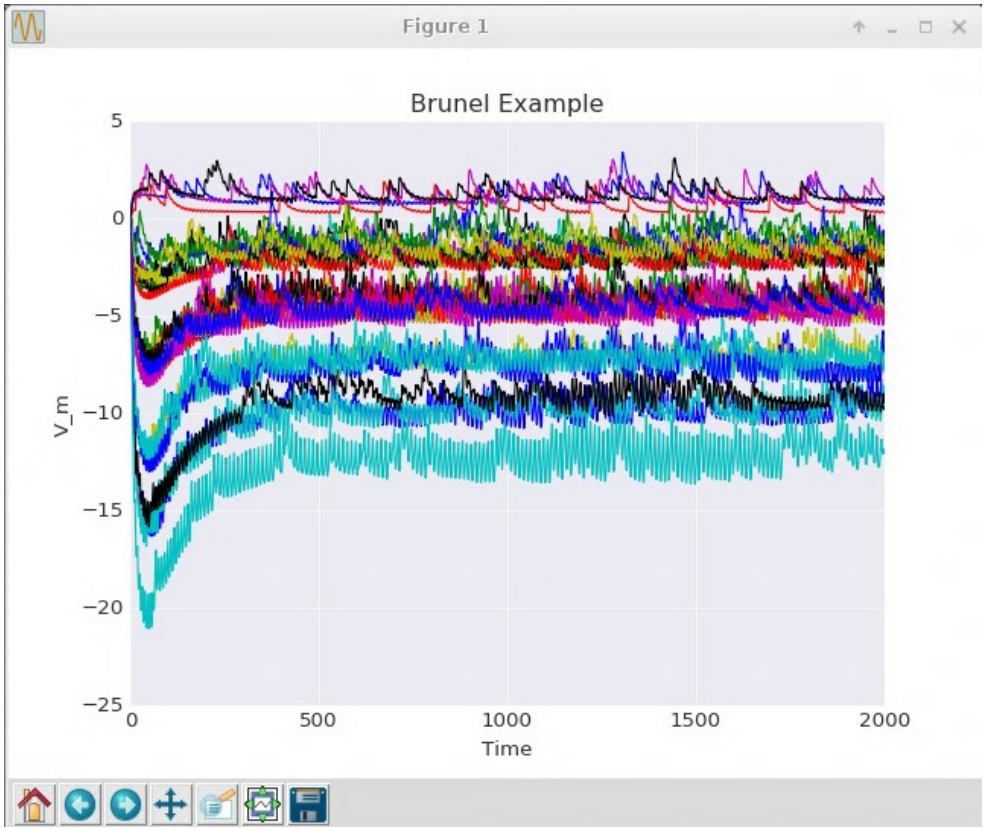
2D Visualization

```
def Poll(self):
  nodes = self.relay.Receive()
  for node in nodes:
    self.node.Update(node)
  self.Plot()
```

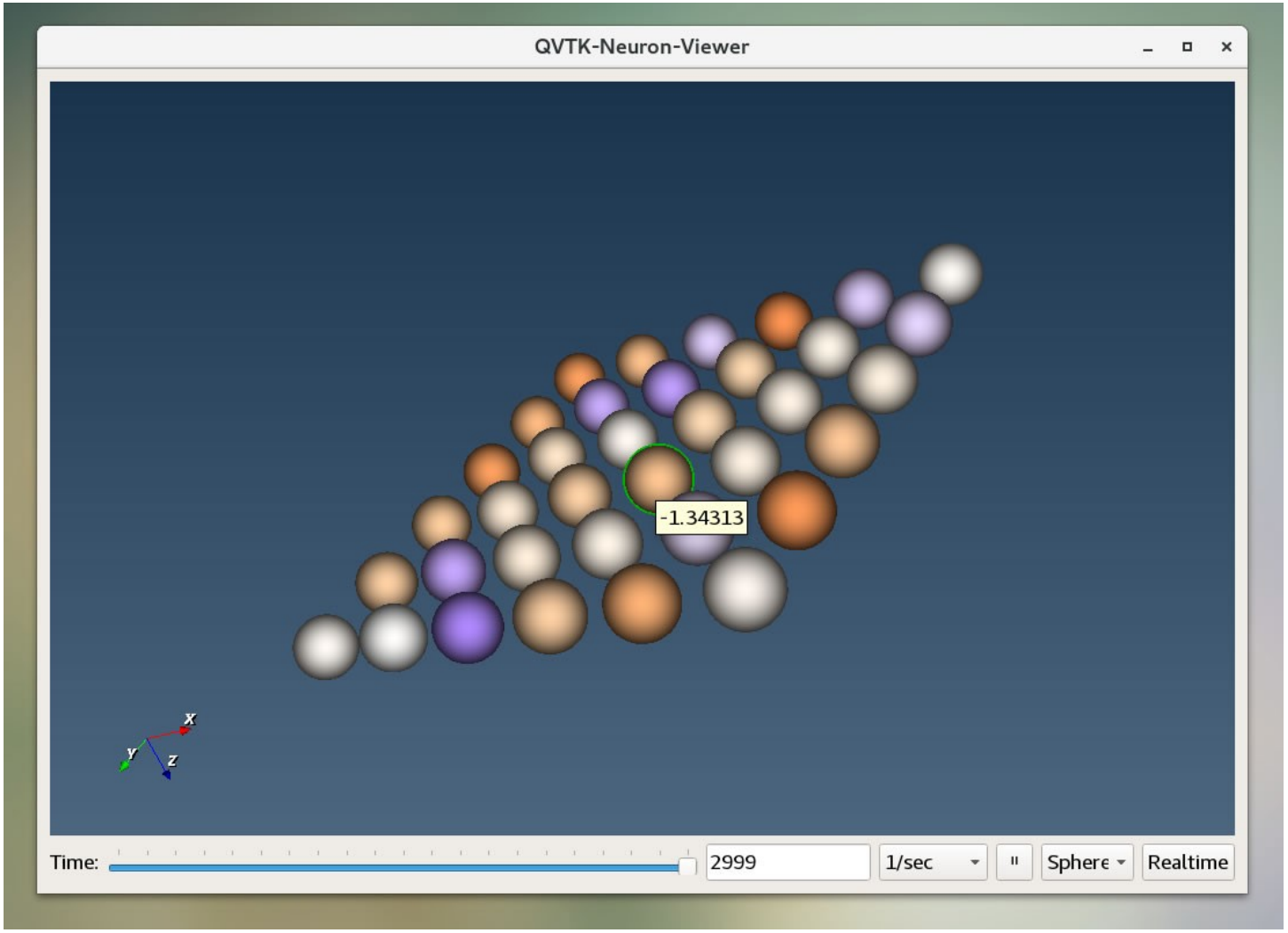
2D Visualization

```
def Plot(self):
  timesteps = self.multimeter.GetTimesteps()
  neuron_ids = self.multimeter.GetNeuronIds(timesteps[0],
'V_m')
  times = [float(t) for t in timesteps]
  for neuron_id in neuron_ids:
    values = self.multimeter.GetTimeSeriesData(attribute,
                                               neuron_id)
    self.axis.plot(times, values)
```

2D Visualization



3D Visualization



Overhead

Brunel Model

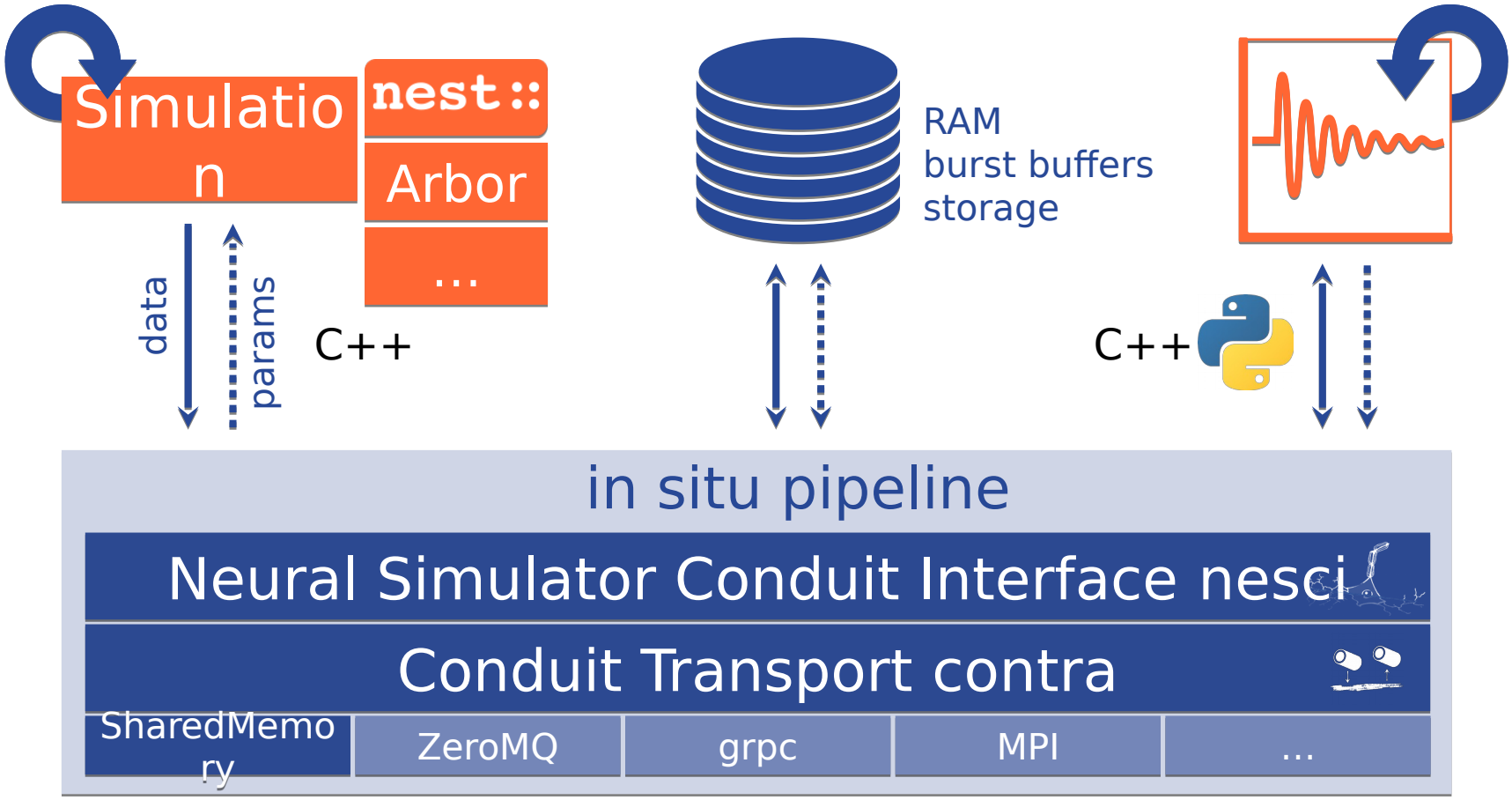
- 408 neurons
- 1000ms simulation step
- SharedMemoryTransport

Baseline	ASCII	NESCI & CONTRA
0.25s	1.18s	5.16s

Future Work

- Improve Performance
  - Memory Management
  - Threading
- More Transport Protocols
  - ZeroMQ
  - gprc
  - MPI
  - …
- Glue library for NESCI & CONTRA

Summary


Simulation
nest::
Arbor
...
data
params
C++
RAM
burst buffers
storage
C++
in situ pipeline
Neural Simulator Conduit Interface nesci
Conduit Transport contra
SharedMemory
ZeroMQ
grpc
MPI
...