Clients Send to One Master Using "Send/Recv"

NOTE: a complete working example of this pattern (for a vonHamos spectrometer used in XCS and XPP) can be found here: https://github.com/chrisvam/psana_cpo/tree/master/vonhamos

In the previous parallelization examples, all the cores have been analyzing data identically. Another useful but somewhat more complex pattern is to have MPI clients intermittently sending updates to one master process that manages data visualization and storage. This pattern is typically used only when running in real-time from shared memory where it is difficult to make sure that all nodes call Gather or Reduce the same number of times (collective MPI operations like **gather/reduce cannot be used with shared memory**! deadlocks can easily arise). For real-time disk-based "smd" analysis Gather/Reduce can be used.

The idea here is to have two different pieces of code (client/master) that exchange data via MPI send/recv calls (used in such a way that clients can send to the master when desired). To see this example, copy the following four files from /reg/g/psdm/tutorials/examplePython/: client.py, master.py, mpidata.py, mpi_driver.py (parses arguments and decides to run master/client, as appropriate).

This code can be run with the following command, analyzing our usual tutorial data experiment with the detector named "cspad":

```
mpirun -n 2 python mpi_driver.py exp=xpptut15:run=54 cspad -n 10
```

The user should only modify client.py/master.py. The information to be transmitted from client to master is defined by these two lines in client.py:

```
md.addarray('img',img)
md.small.intensity = intensity
```

The first line adds an array to be transmitted to the master, the second line adds "small data", which can be any simple python object (if the data is large, you should convert it to a numpy array and use "addarray" instead). You can add as many lines like this as you like.

Note: MPI does not automatically propagate environment to remote nodes, so when you run on multiple mon nodes you must "**mpirun -n 2 -H daq-xpp-mon02,daq-xpp-mon03 <shellScript>**" where shellScript looks similar to this:

```
#!/bin/bash
source /reg/g/psdm/etc/psconda.sh
cd <appropriate directory>
python myAnalysis.py
```