A "canonical" experimental recipe (scanning):

\[
\text{while not done:} \\
\text{exp\_parameters\_set}(\lambda; T, P, B) \\
\text{accumulate\_statistics}(t, N)
\]
Thermoelectric materials

power space probes

refrigeration

capture “waste heat”

SNS Data Acquisition System

- Control & Events
- Real Time
- Slow Controls

ORNL.GOV

Data Analysis

INST-DFS

RTDL Event Link 192.168.71.XXX

Detector Electronics

Detectors

Detector Systems 192.168.70.XXX

Control Systems 192.168.62.XXX

Satellite HW Subnet 192.168.63.XXX

Satellite HW Subnet 192.168.63.XXX

remoteservice

(Satellite Computers) sampleenv, chopper, ancillary, motors

media box

Ethernet 2 Serial

RS-232 Cable

Serial Controller

Ethernet 2 GPIB

GPIB Cable

GPIB Controller

Remote Service

HFIR

Neutron Sciences

OAK RIDGE NATIONAL LABORATORY
U.S. DEPARTMENT OF ENERGY
Communication Overview

Python for Data Acquisition System = PyDas

Control Computer
- DcomClient
- Motors Control Application
- ...
- Sample Environment Control Application

Satellite Computers
- Data Preprocessor
- Motors Satellite Application
- ...
- Sample Environment Satellite Application

Hardware
- Detector Electronics
- Motors
- ...
- Sample Environment Hardware
What do the scientists want?
What do the scientists want?
## PyDas Architecture

### PyDas Main Application

<table>
<thead>
<tr>
<th>IPyDas shell</th>
<th>PyDas GUI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Abstraction: pydas.devices</td>
<td>Experiment Abstraction: pydas.scan, pydas.lib</td>
</tr>
<tr>
<td>Core Packages: pydas.core</td>
<td>REST Client: pydas.web</td>
</tr>
<tr>
<td>Memory Maps: dasmapcq, memmap</td>
<td>GUI Widgets</td>
</tr>
<tr>
<td>Win32 API (PyWin32)</td>
<td>ipython, numpy, scipy, matplotlib</td>
</tr>
<tr>
<td>GUI Base: wxPython</td>
<td></td>
</tr>
</tbody>
</table>
PyDas GUI Example: Scanning
PyDas GUI Example: Status Page

[Image of the PyDas GUI interface showing various statuses and settings]

SciPy Conference 7/14/2011
PyDas Shell (IPyDas)

Welcome to IPyDas Shell.

Python 2.5.4 (r254:67916, Dec 23 2008, 15:10:54) [MSC v.1310 32 bit (Intel)]
Type "copyright", "credits" or "license" for more information.

IPython 0.9.1 -- An enhanced Interactive Python.
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', ?object also works, ?? prints more.
!command -> Execute command in shell
TAB -> Autocompletion
In [1]: das.sleep(1.0)
Out[1]: True
In [2]: das.sleep(1.0)
Out[2]: True
In [3]: das.sleep(1.0)
Out[3]: True
In [4]:

Log:
[INFO] 10/09/12 18:04:02: DAS waiting: sleeping for 1 seconds; done!
[INFO] 10/09/12 18:04:02: DAS waiting: PCharge -> 380000000000.0
[INFO] 10/09/12 18:04:28: DAS waiting: sleeping for 1 seconds
[INFO] 10/09/12 18:04:38: DAS waiting: sleeping for 1 seconds; done!
[INFO] 10/09/12 18:04:42: new value assigned to SRemote2Request: 0.5
Examples: Scanning in IPyDas

```python
s = scan('Motor1', arange(10.0, 90.0, 0.5), runtime=30.0, plot='counts')
s.fit('gauss+poly2')
```

```python
for SampleTemp.value in [273.0, 293.0, 303.0, 323.0]:
    scan('Energy', [25., 35., 50], pcharge=3e12,
         title='Nobel Prize Data at T=%s K' % SampleTemp.value)
```
while not done:
    exp_parameters_set(λ;T,P,B)
    accumulate_statistics(t,N)

Beam time is expensive!

1) How to choose the parameters in the most optimal way?

2) How long to measure in each setting?
A Hypothetical Experiment

![Graph showing a hypothetical experiment result.](image-url)
VULCAN – Engineering Diffraction
Sample Schema

Experiment Script

Strain Mapping

unexpected area of interest

crack tip
PyDAS REST Interface

- **NEW Experiment (PUT)**
  - http://host/NEWEXP?EXPID=weld&XMIN=0.0&XMAX=10.0&MAXPTS=20

- **Current Point (GET)**
  - http://host/CURR_PT

- **Quality of Data - are we there yet? (GET)**
  - http://host/QOD
Example Implementation

- **PyDas (REST Client)**
- **Instrument**
- **REST Server**
- **StatusViewer**
- **Optimizer**
  - Where to measure?
- **Fitting**
  - When to stop?

Connections:
- HTTP
- NEUTRON DATA
RESTful Web Interface

das = get_das()
cli = DasWebClient('http://localhost:8080')
qod = DasWebClient('http://localhost:8080/QOD')
cli.put(r'/NEWEXP?EXPID=weld&XMIN=0.0&XMAX=10.0&MAXPTS=20')
while True:
    new_x = cli.get(r'/CURR_PT')
    if new_x is None: break
    das.start()
    das.waiton(qod)
das.stop()
the latest histogram

the latest fitting results

historical fitting results

Fitting Result

fitting error

Peak Position

i^{th} Fitting Result
Summary & Outlook

- PyDas “glues” various DAS components together
- We’ve added and prototyped REST interface
- We will look into new IPython/ZMQ more closely

Thanks to all engineering staff from SNS Data Acquisition Group and especially to Lloyd Clonts, Gayle Greene, Steve Hicks, Andre Parizzi, Richard Riedel, Mariano Ruiz-Rodriguez and Madhan Sundaram
The End.