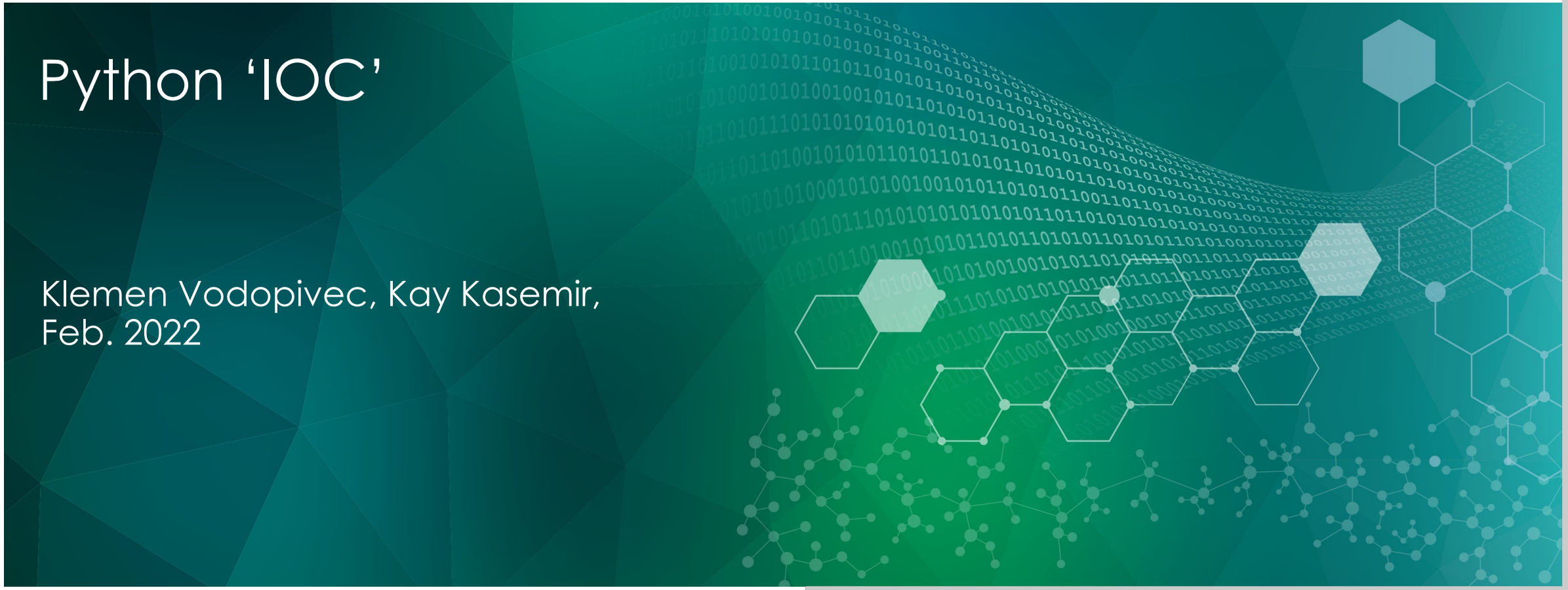
OAK RIDGE National Laboratory

Python 'IOC'

Klemen Vodopivec, Kay Kasemir,
Feb. 2022



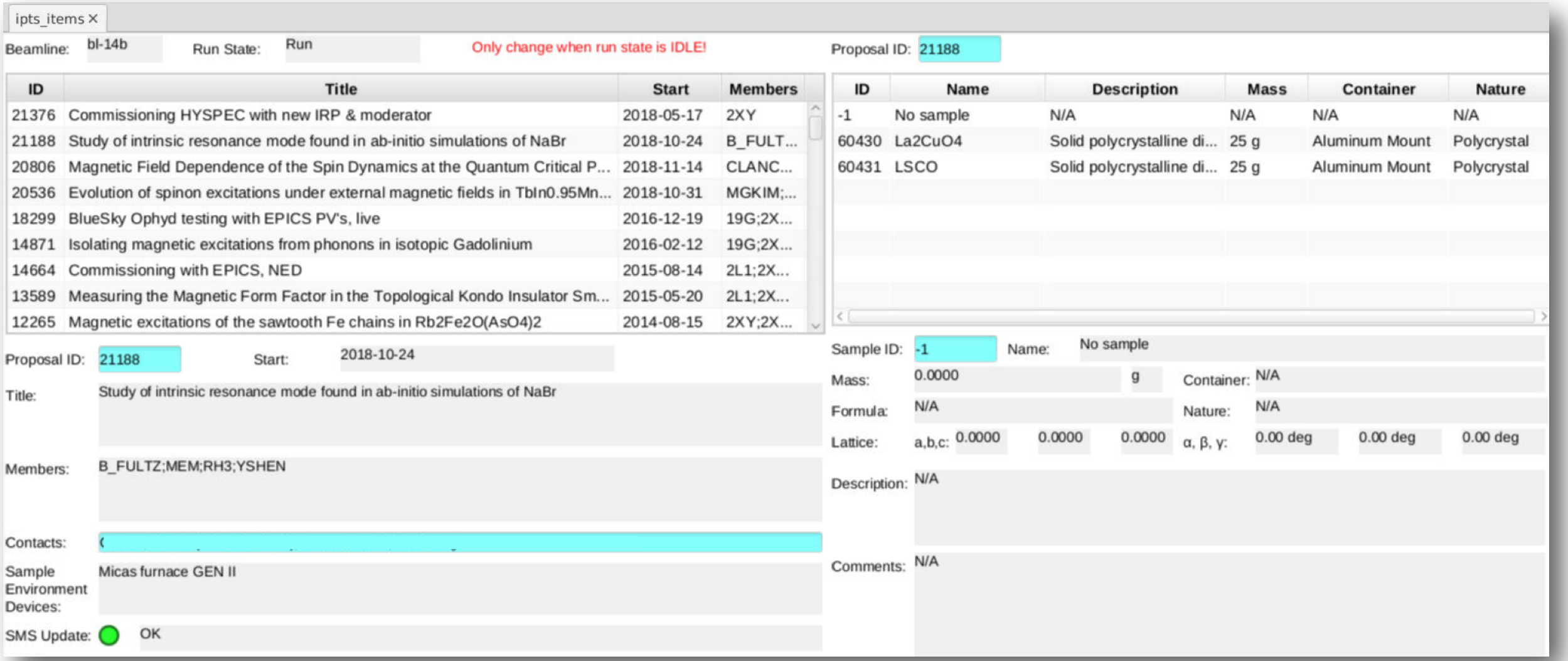
ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

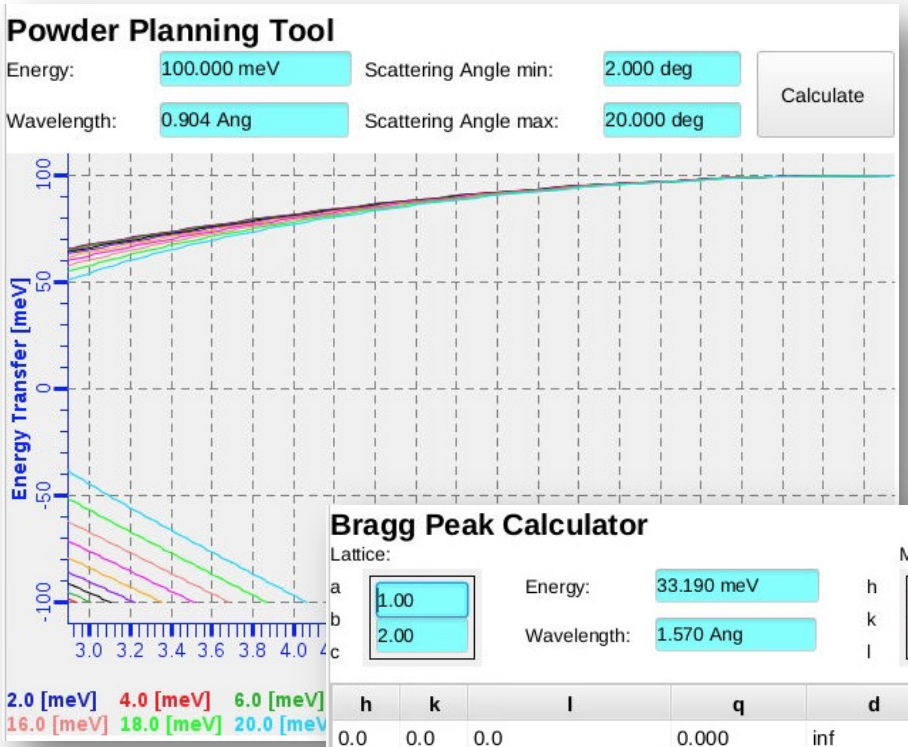
Why Python? Why Python IOC?

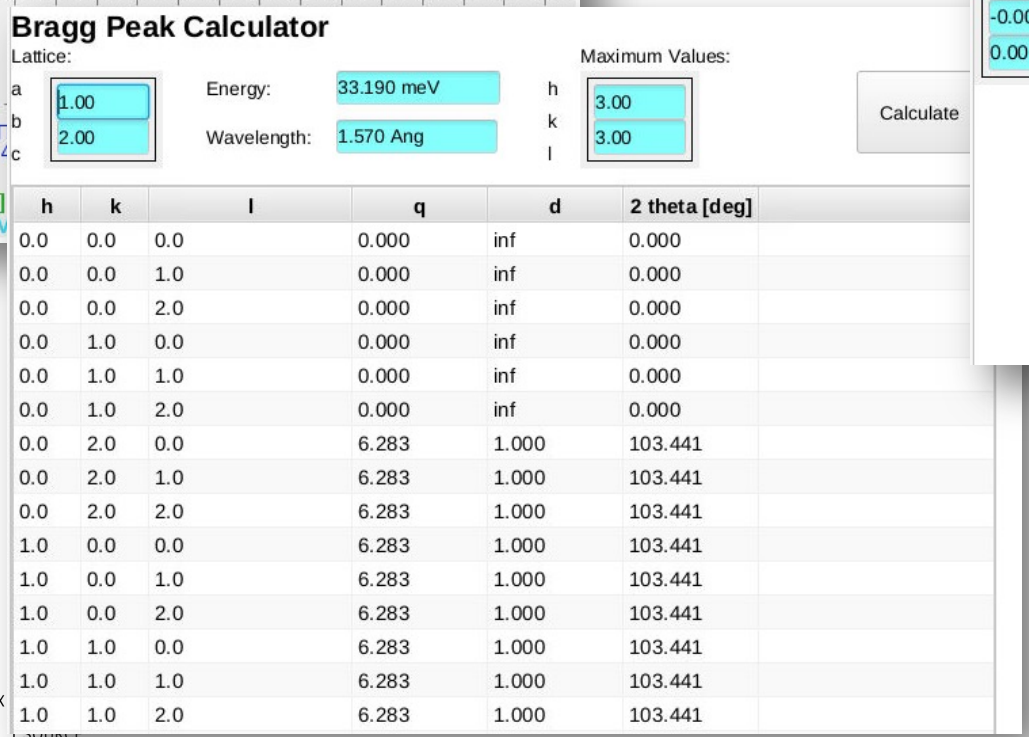
- Python widely adopted language, easy to learn, develop, test
- Large set of readily available high-level functions
  - Crunching data (numpy, scipy, etc.)
  - Interfacing external sources, ie. databases, web services
- When existing EPICS functionality doesn't cover the new use case…
- …but want to use robust EPICS infrastructure
  - IOC shell (interactive console, commands, logging)
  - Reliable CA & PVA communication
  - Archives, alarms, autosave
  - GUI tools

Example: Proposal Database

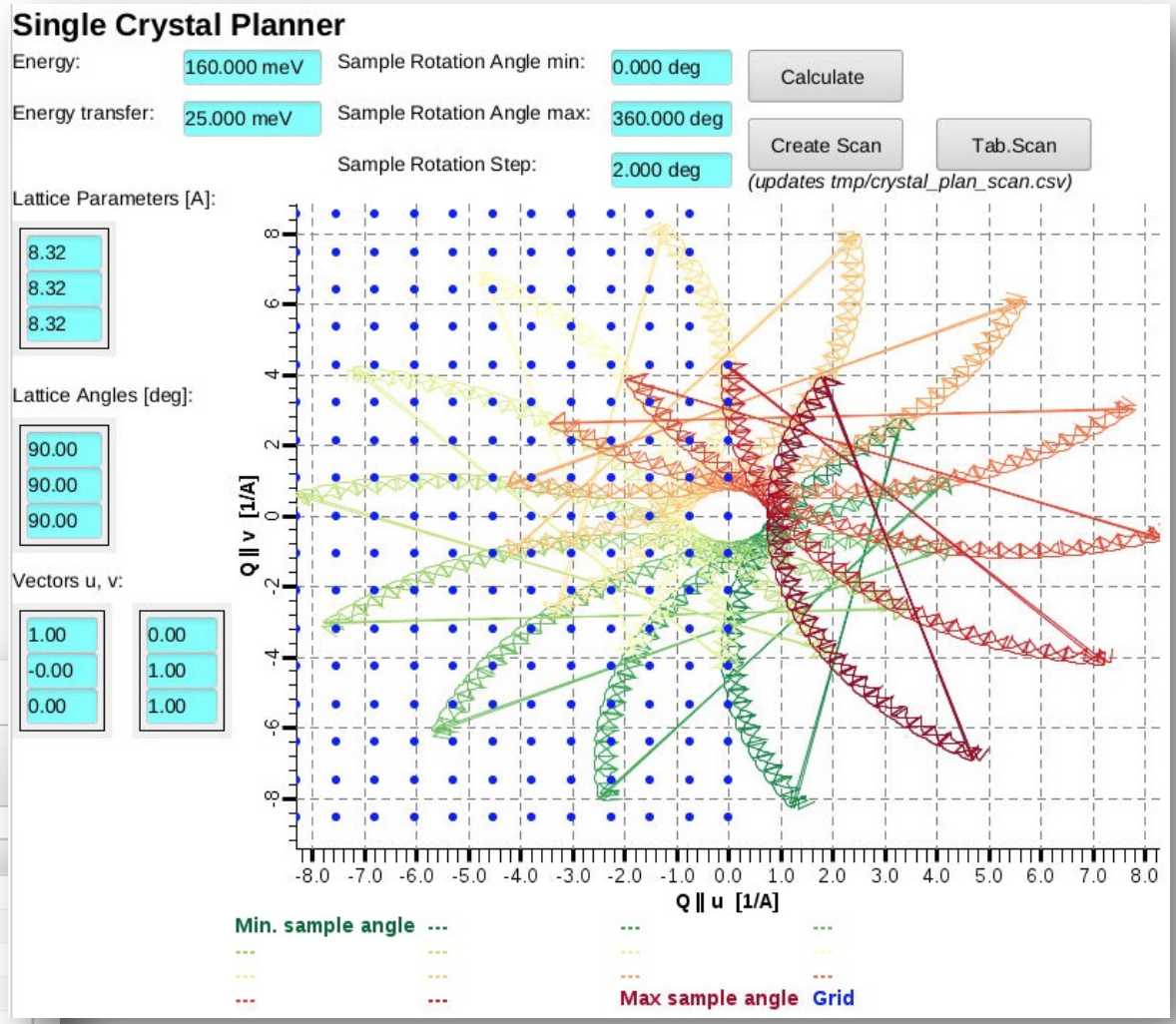


Example: Computations

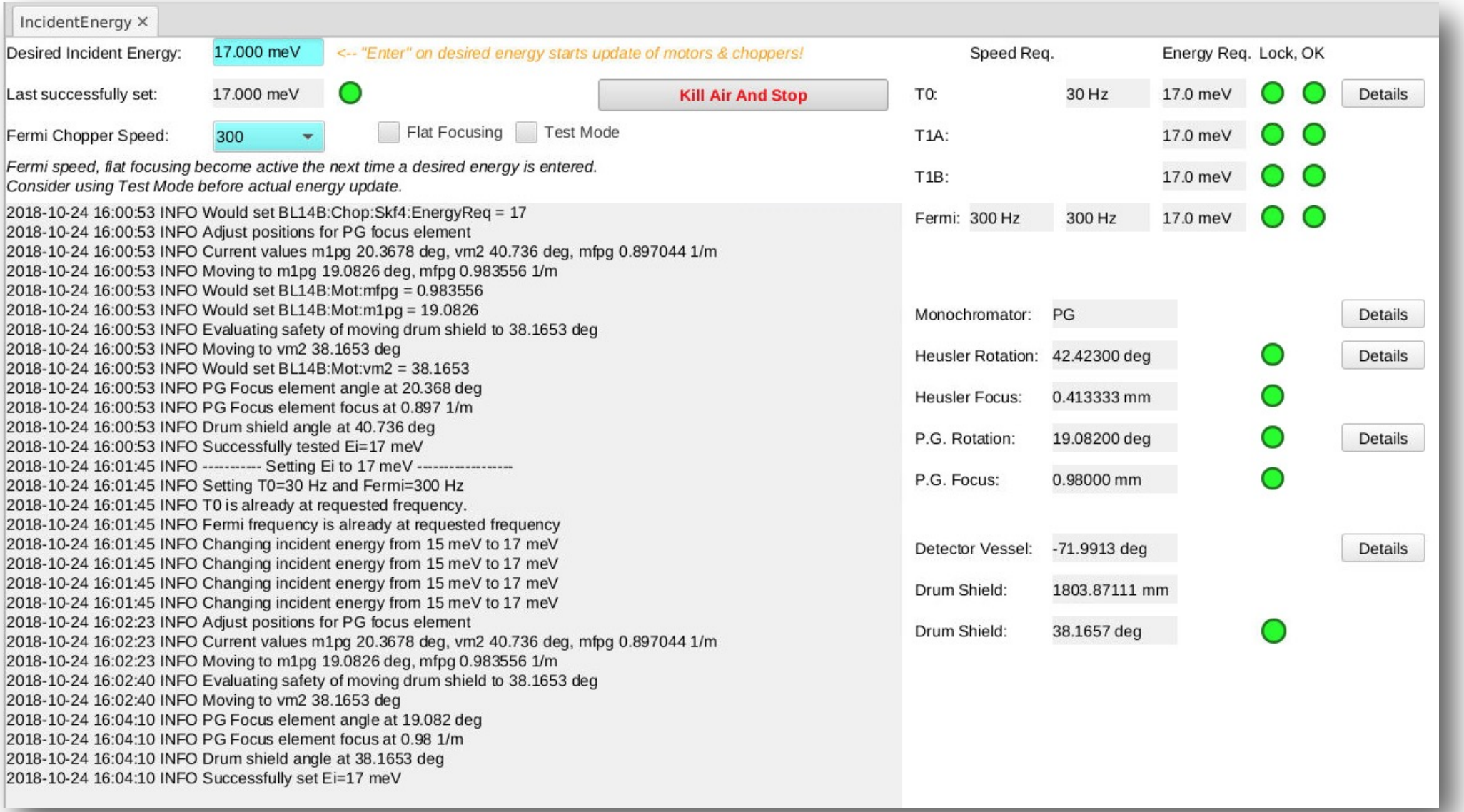




h	k	l	q	d	2 theta [deg]
0.0	0.0	0.0	0.000	inf	0.000
0.0	0.0	1.0	0.000	inf	0.000
0.0	0.0	2.0	0.000	inf	0.000
0.0	1.0	0.0	0.000	inf	0.000
0.0	1.0	1.0	0.000	inf	0.000
0.0	1.0	2.0	0.000	inf	0.000
0.0	2.0	0.0	6.283	1.000	103.441
0.0	2.0	1.0	6.283	1.000	103.441
0.0	2.0	2.0	6.283	1.000	103.441
1.0	0.0	0.0	6.283	1.000	103.441
1.0	0.0	1.0	6.283	1.000	103.441
1.0	0.0	2.0	6.283	1.000	103.441
1.0	1.0	0.0	6.283	1.000	103.441
1.0	1.0	1.0	6.283	1.000	103.441
1.0	1.0	2.0	6.283	1.000	103.441



Example: Energy Adjustment



- Could use sequencer, but had existing python code

Available Python IOC solutions

pcaspy

https://github.com/paulscherrerinstitute/pcaspy

- Most mature, started in 2011
- Purely in Python code
- But missing EPICS goodies
  - autosave, IOC commands
- Custom records implementation
  - No RTYP, EGU, SCAN etc. fields
  - Affects GUI tools

pyDevSup

http://mdavidsaver.github.io/pyDevSup/

- Plugs into standard EPICS IOC
- Invoked from EPICS records
- Invoked Python code must extend provided class
- Most flexibility to update records and fields

PyDevice

https://github.com/klemenv/PyDevice

- Plugs into existing EPICS IOCs, simplest to use
- Invoked from EPICS records
- Arbitrary Python code invoked from EPICS records
- Can't change record fields directly
- EPICS-independent Python code
  - Can be reused or tested outside EPICS environment

pcaspy in more details

- Can be used as an EPICS IOC or a client
- All code is in Python
  - Defining database records
  - Serving the connections
  - Handling requests
  - Interfacing with other IOCs
- Needs to reimplement common EPICS functionality
  - autosave
  - Access Security
  - Data types

PyDevice in more details

- Call any Python code directly from record
- Supports most popular EPICS records
  - ai, ao, longin, longout bi, bo, mbbo, mbbi, stringin, stringout, waveform
- Pass record fields to Python code
  - Values evaluated when record processes
  - Return value pushed to record VAL field
  - Types automatically converted
- Python exceptions translate into SEVR&STAT

```
record(ai, "Example:AbsValue") {
    field(DTYP, "pydev")
    field(INP, "@abs(VAL)")
}
```

```
caput Example:AbsValue -2.2
```

```
record(ai, "Example:Delay") {
    field(DTYP, "pydev")
    field(INP, "@time.sleep(VAL)")
}
```

```
caput -c Example:Delay 5
```

pcaspy vs PyDevice

/ics/examples/python/demo_pcaspy_ioc.py

```
from pcaspy import SimpleServer, Driver
import demo
pvdb = { 'random' : {
    # 'desc': 'return value between 0 and hilim'
    'type'  : 'float',
    'hilim' : 10 }
}
class DemoDriver(Driver):
    def __init__(self):
        super(demoDriver, self).__init__()
    def read(self, reason):
        if reason == 'random':
            limit = self.getParamInfo(reason)['hilim']
            value = demo.getRandom(limit)
        else:
            value = self.getParam(reason)
        return value
if __name__ == '__main__':
    server = SimpleServer()
    server.createPV('training:pcaspy:', pvdb)
    driver = DemoDriver()
    while True:
        server.process(0.1)
```

/ics/examples/db/demo_pyioc.db

```
record(ai, "$(user):random")
{
        field(DESC, "Random value in range [0,HOPR]")
        field(HOPR, "10")
        field(DTYP, "pydev")
        field(INP,  "@demo.getRandom(HOPR)")
}
```

/ics/examples/iocBoot/ioc_pydevice/st.cmd

```
#!../../bin/linux-x86_64/pydevice
< envPaths
epicsEnvSet("PYTHONPATH","$(TOP)/python")
cd "${TOP}"
dbLoadDatabase "dbd/demo_pyioc.dbd"
pydevice_registerRecordDeviceDriver pdbbase
pydev "import demo"
dbLoadRecords "db/pydevice.db","user=training:pydevice"
iocInit
```

/ics/examples/python/demo.py

```
import random
def getRandom(limit=1):
    # random() returns value in range [0,1]
    return random.random() * limit
```

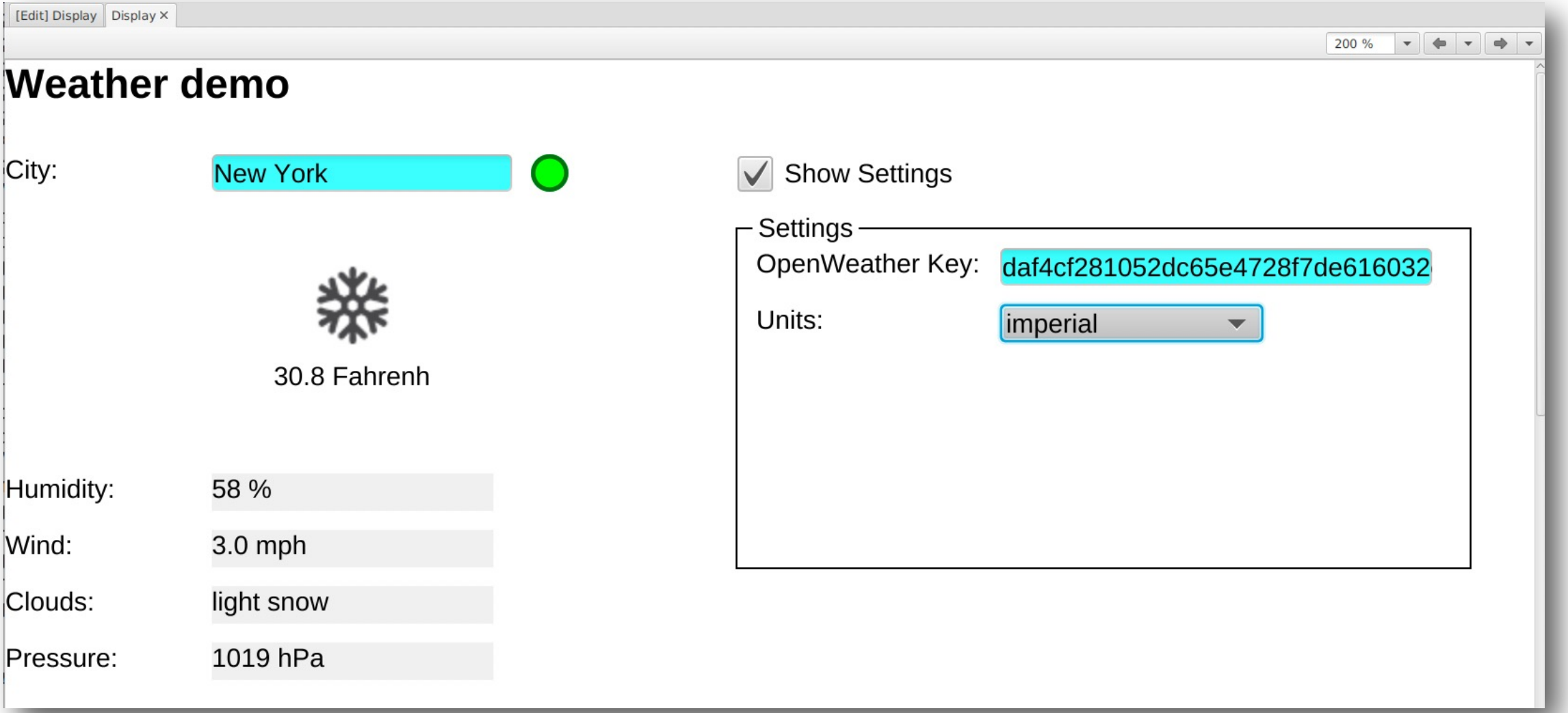
PyDevice latest: pycalcRecord

- New record type to support passing multiple parameters
  - Similar to aSub, genSub, calc records
  - But evaluate Python code

```
record(pycalc, "Example:gcd") {
    field(INPA, "training:pyioc:X1 CP")
    field(INPB, "training:pyioc :Y1 CP")
    field(INPC, "training:pyioc :X2 CP")
    field(INPD, "training:pyioc :Y2 CP")
    field(CALC, "gcd(A,B,C,D)")
}
```

```
record(pycalc, "Example:GetHtml") {
    field(INPA, "Example:Proto CP")
    field(INPB, "Example:Hostname CP")
    field(INPC, "Example:Port CP")
    field(CALC, "mywebClientFetch('A', 'B', C)")
}
```

Weather demo



Handling Table, Structure

h	k	l	q	d	2 theta [deg]
0.0	0.0	0.0	0.000	inf	0.000
0.0	0.0	1.0	0.000	inf	0.000
0.0	0.0	2.0	0.000	inf	0.000
0.0	1.0	0.0	0.000	inf	0.000
0.0	1.0	1.0	0.000	inf	0.000
0.0	1.0	2.0	0.000	inf	0.000
0.0	2.0	0.0	6.283	1.000	103.441
0.0	2.0	1.0	6.283	1.000	103.441
0.0	2.0	2.0	6.283	1.000	103.441

- Yes, pickled byte waveform is a hack
  - Requires scripts
  - Only works with python as server & client
- pvAccess can handle custom structures
  - Better for server side
  - Client likely needs more than dump of structure; Will still require script for user-friendly display.
- pvAccess IOC Python libraries
  - pvaPy
  - p2p

Summary

Python with CA server & client libs can act as IOC

- Great tool to have
- Doesn't replace all IOCs