PVAccess for Python (P4P)

Michael Davidsaver
Osprey DCS

EPICS

Two Python bindings already?

- P4P != pvaPy
- Different focus, different implementation
  - Breadth (pvaPy) vs. Depth (P4P)
- P4P has:
  - Supports get/put/rpc/monitor as client, only rpc as server
  - Fewer dependencies (no Boost Python)
  - “Better” documentation
    - https://mdavidsaver.github.io/p4p/
  - Easy access to structured data (w/ numpy)
  - “Pythonic” (follows conventions)
  - Automatic (un)packing (generic container ↔ python type)

Used by new MASAR micro-service
(642 lines of python code)

P4P: 868 lines python and 3321 C++

Examples

Client Operations

```
from p4p.client.thread import Context
ctxt = Context('pva')
print ctxt.get("pv:name")

ctxt.put("pv:name", 5)

def show(V):
    print "update", V
S = ctxt.monitor("pv:name", show)
# ...
S.close()
```

Automatic (un)Packing

```
from p4p.client.thread import Context
ctxt = Context('pva')
V = ctxt.get("some:scalar")

print V.value + 1
print V.alarm.severity
print V.timeStamp.secondsPastEpoch
print V.timeStamp.nanoseconds
```

```
from p4p.client.thread import Context
ctxt = Context('pva')
V = ctxt.get("some:scalar")

print V + 1
print V.severity
print V.timestamp # float
print V.raw_stamp # tuple
```

RPC Example

Client

```
from p4p.rpc import rpccall, rpcproxy
from p4p.client.thread import Context

@rpcproxy
class ExampleProxy(object):
    @rpccall("%sadd")
    def add(lhs='d', rhs='d'):
        pass

ctxt = Context('pva')
proxy = ExampleProxy(context=ctxt,
                     format="pv:prefix:")
print proxy.add(1, 1)
```

Server

```
from p4p.rpc import rpc, quickRPCServer
from p4p.nt import NTScalar

class MyExample(object):
    @rpc(NTScalar("d"))
    def add(self, lhs, rhs):
        return float(lhs) + float(rhs)

example = MyExample()

quickRPCServer(provider="Example",
               prefix="pv:prefix:",
               workers=2,
               target=example)
```