
Osprey DCS

Comparison

- Channel Access
  - Multiplex requests
  - Server discovery
- DBR
  - Tag selects from predefined struct
- DBE
  - Small set of pre-defined
  - Not sent with update

- PV Access
  - Multiplex requests
  - Server discovery
- PVData
  - Self-describing
  - Hierarchical
  - Not recursive
- Bit mask
  - Per-field
  - Sent w/ updates

PVA/PVD new features

- All operations can cancel
- Per-monitor flow control
- Structured data
  - Can transport “anything”
  - Must decide on something!

Type components

- Kinds:
  - Primitive
    - Integer, Real, String
  - Composite
    - Structure, Union
  - Array
    - Array of Primitive or Composite
- Integer
  - 8, 16, 32, and 64 bit
  - Signed/unsigned
- Real
  - 32, 64 bit
- String
  - Variable length

Composite Types

- Have an ID string
- Structure
  - Ordered list of named fields
  - Like C struct
- Union
  - Discriminating (tagged)
    - Named possibilities
  - Variant
    - aka. Any

```
struct mytaggedunion {
    unsigned index;
    union {
        int32 A;
        float64 B;
        string C;
        structure X {
            int32 M,N;
        };
    };
};
```

Notation(s)

```
$ pvinfo cnt
CHANNEL  : cnt
STATE    : CONNECTED
ADDRESS  : 10.0.142.1:5075
epics:nt/NTScalar:1.0
    double value
    alarm_t alarm
        int severity
        int status
        string message
    time_t timeStamp
        long secondsPastEpoch
        int nanoseconds
        int userTag
...
```

```
struct NTScalar<double> {
    double value;
    alarm_t alarm;
    time_t timeStamp;
};
struct alarm_t {
    int severity;
    int status;
    string message;
};
...
```

Notation(s)

```
from p4p import Type

ntscalarF64 = Type([
   (‘value’, ‘d’),
   (‘timeStamp’, (‘S’
      ‘time_t’, [
         (‘severity’, ‘i’),
         (‘status’, ‘i’),
         (‘message’, ‘s’),
      ])),
], id=’epics:nt/NTScalar:1.0’)
```

```
struct NTScalar<double> {
    double value;
    alarm_t alarm;
    time_t timeStamp;
};
struct alarm_t {
    int severity;
    int status;
    string message;
};
...
```

https://mdavidsaver.github.io/p4p/values.html#type-definitions

Notation(s)

```
pvd::StructureConstPtr schema(pvd::getFieldCreate()->createFieldBuilder()
                              ->add("version", pvd::pvUInt)
                              ->addNestedStructureArray("clients")
                                ->add("name", pvd::pvString)
                                ->add("provider", pvd::pvString)
                                ->add("addrlist", pvd::pvString)
                                ->add("autoaddrlist", pvd::pvBoolean)
                                ->add("serverport", pvd::pvUShort)
                                ->add("bcastport", pvd::pvUShort)
                              ->endNested()
                              ->addNestedStructureArray("servers")
                                ->add("name", pvd::pvString)
                                ->addArray("clients", pvd::pvString)
                                ->add("interface", pvd::pvString)
                                ->add("serverport", pvd::pvUShort)
                                ->add("bcastport", pvd::pvUShort)
                                ->add("control_prefix", pvd::pvString)
                              ->endNested()
                              ->createStructure());
```

P4P Type/Value

- P4P
  - PVAccess for Python
- Type
  - Description of PVD Structure
- Value
  - Instance of Structure (aka PVStructure)
  - Also “changed” bit mask

https://mdavidsaver.github.io/p4p/values.html

Example

```
from p4p import Type, Value
from p4p.nt.common import alarm, timeStamp
mytype=Type([
    ('value','d'),
    ('timeStamp', timeStamp),
    ('alarm', alarm)
])
print(mytype.aspy())
```

https://mdavidsaver.github.io/p4p/values.html

Example (2)

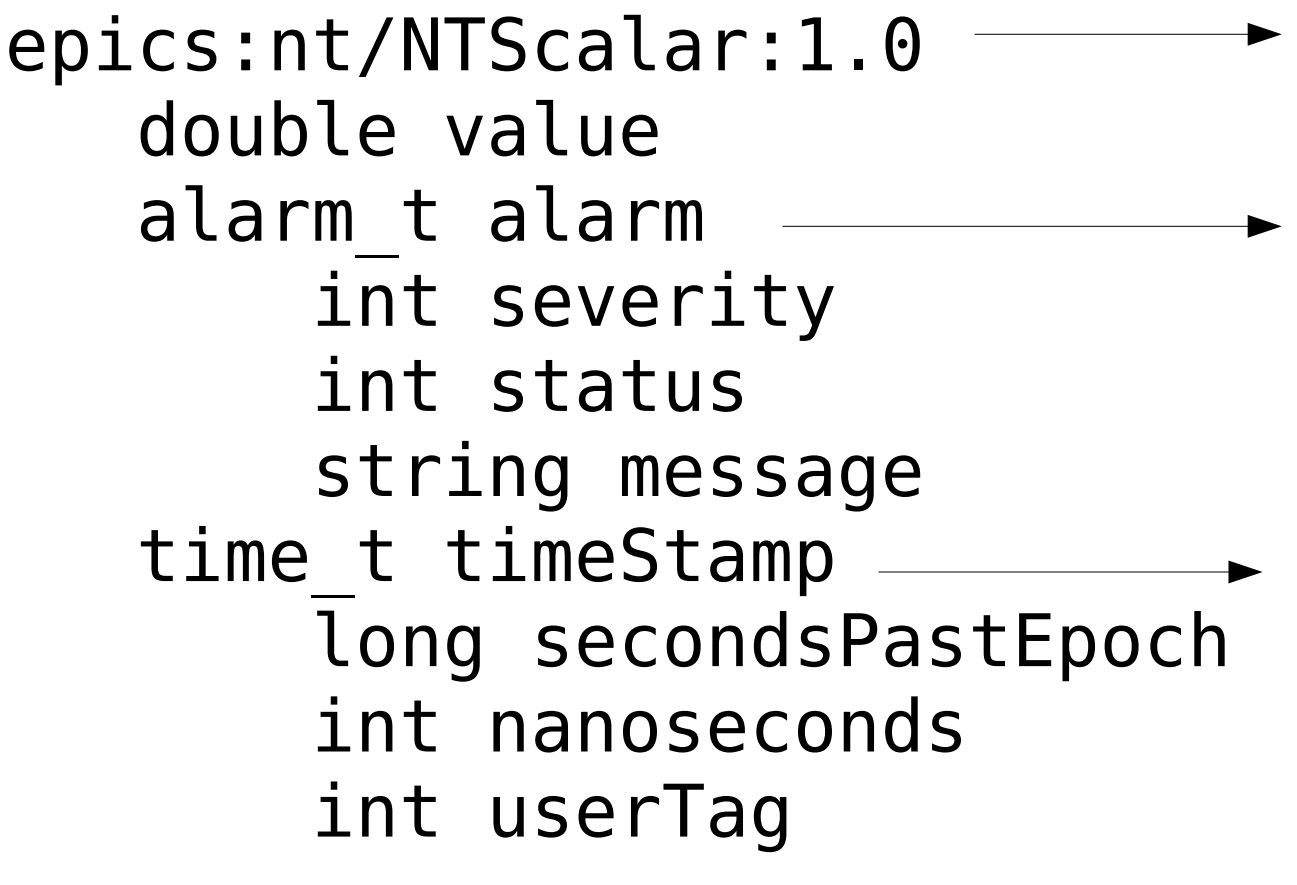
```
V=Value(mytype, {'value':42})
print(V.value)
print(V.alarm.severity)
print(V['alarm.severity'])
print(V['alarm']['severity'])


V.value = 43
```

field bit masks

- One bit per Structure field
  - One for entire Union or Array
- One bit “shorthand” for entire Structure
- changed bit mask
  - Get/put/monitor
- Overrun bit mask
  - monitor

PVA field bit masks (2)



Osprey DCS

Example (3)

```
print(V.asSet())
print(V.changed())
print(V.changed('value'))
print(V.changed('alarm.severity'))
V.alarm.severity = 0
print(V.changed('alarm.severity'))
```

Osprey DCS

Normative Types

- Standard type definitions
- NTScalar, NTScalarArray
  - Equivalent to DBR_*
- NTEnum
  - DBR_ENUM
- NTTable
  - struct of arrays
- NTURI
  - RPC calling convention
- NTNDArray
  - areaDetector
- NTMultiChannel
  - MASAR
  - save/restore

PVA Configuration

- EPICS_PVA_ADDR_LIST
- EPICS_PVA_AUTO_ADDR_LIST
- EPICS_PVA_BROADCAST_PORT
- EPICS_PVA_SERVER_PORT

PVA Operations

- Get
  - pvget
- Put
  - pvput
- Monitor
  - pvget -m
- GetField
  - pvinfo
- RPC
  - eget -s
- List
  - pvlist
- PutGet
- Array

Exercise

- Start IOC

  softIocPVA -m TST: -d demo.db
- Use CLI tools

  pvinfo/pvget/pvput/eget

- pvget -v
  - Show entire Structure
- pvget -d
  - Show low level debug

- demo.db contains
  - TST:phase
  - TST:sine
  - TST:out
    - TST:in
  - TST:sel
  - TST:slow
    - .ODLY

Example

- Run ipython

```
from p4p.client.thread import Context

ctxt=Context('pva')
print(ctxt.get('TST:phase'))
Sat Mar  3 19:55:24 2018 59.0
help(ctxt)
```

https://mdavidsaver.github.io/p4p/client.html

pvRequest

- Extra information
- Operation creation
- Mask/select structure fields
- Key/value modifying Operation
  - standard-ish
- Convenience syntax
- “field()”
- “field()record[]”
- “record[block=true]”

pvRequest (2)

- standardize
  - block=true|false
    - Put wait for processing. Default is false
  - process=true|false|passive
    - Trigger remote processing. Default is passive
  - queueSize=#
    - Monitor queue size.
  - pipeline=true|false
    - Monitor flow control.
- QSRV specific
  - atomic=true|false
    - Group use atomic (multi-lock) access

pvRequest (3)

- pvput pv:name value
- pvput -r ‘record[block=true]’ pv:name value
  - aka. Put w/ callback
- pvget -m pv:name
- pvget -m -r ‘record[queueSize=10,pipeline=true]’ pv:name

Example

- CLI

  time pvput TST:slow 4

  time pvput -r ‘record[block=true]’ TST:slow 5

- P4P

```
ctxt.put('TST:slow', 5)
ctxt.put('TST:slow', 5, wait=True)
ctxt.put('TST:slow', 5, request='record[block=true]')
```

Get/Monitor Operation

- Inputs
  - PV name
  - pvRequest
- Outputs
  - Status
  - Result value
  - Changed mask
  - Overrun mask (Monitor only)

Example (2)

```
from p4p.client.thread import Context
ctxt=Context('pva')
def show(V):
    print(V)
S=ctxt.monitor('TST:sine', show)
# ...
S.close()
```

Put Operation

- Inputs
  - PV name
  - pvRequest
  - Input value
  - Put mask
- Outputs
  - Status

Exercise 1

- Monitor for meta-data change
- Use Context.monitor()
- Subscribe to ‘TST:sine’
- Print when field ‘value’ *doesn’t* change
- Change eg. ‘TST:sine.HOPR’

https://github.com/mdavidsaver/p4p/blob/master/example/monitor_client.py
~/build-epics/p4p/example/monitor_client.py

Exercise 2

- Step scan of sim motor
  - Use `request='record[block=true]'`

Exercise 3

- Get and render image
  - From IOC iocimagedemo
  - $ ipython -pylab
  - Use numpy.reshape()
  - Use imshow()

Exercise 4

- RPC list concatenation service

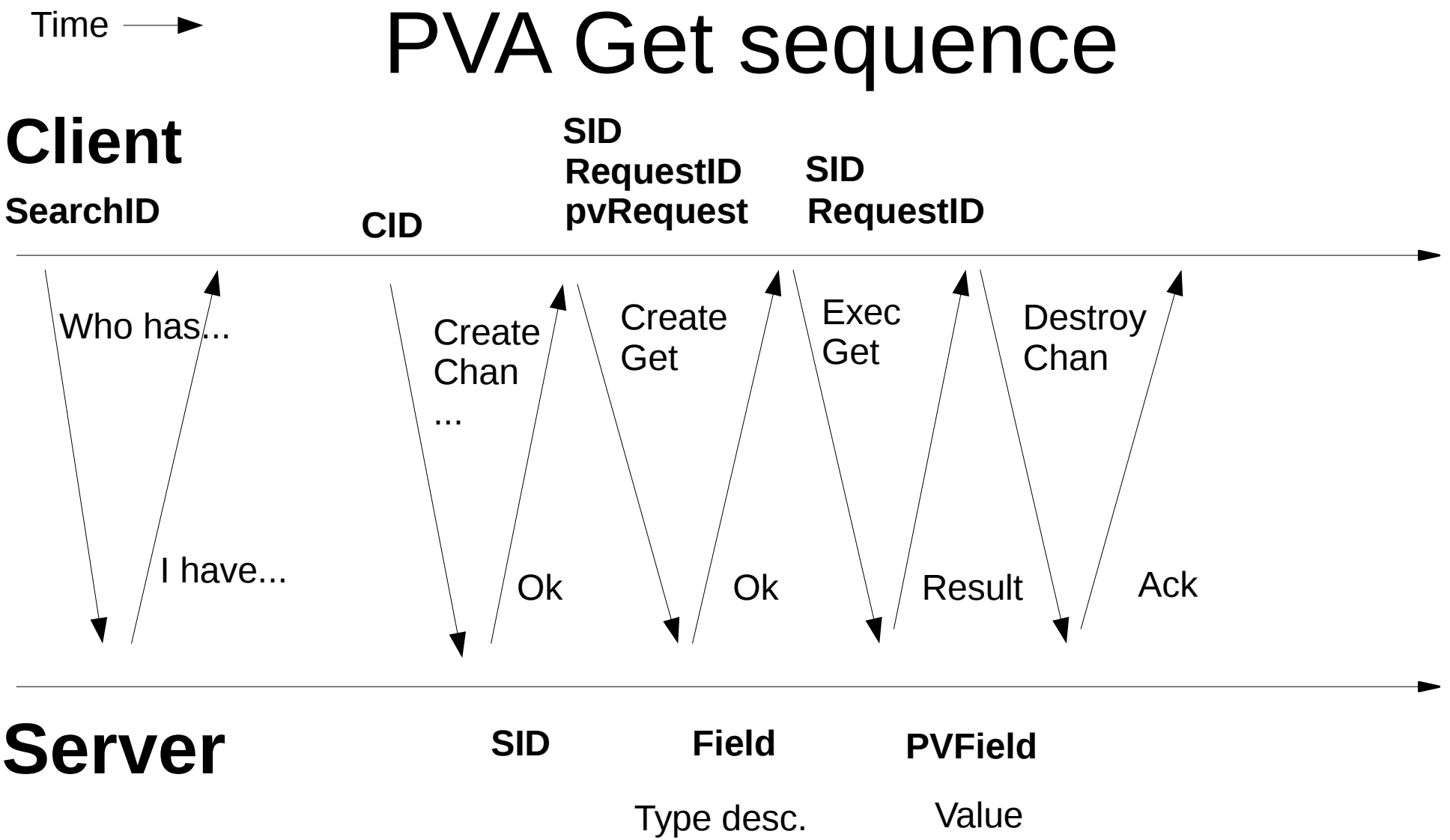
Protocol State

- CA
- SearchID
- SID+CID
  - Channel
- SUBID
  - Subscription

- PVA
- SearchID
- SID+CID
  - Channel
- RequestID
  - Get/put/monitor/…
  - Type description
    - Except RPC
  - Extra round trip

https://epics.anl.gov/base/R3-16/0-docs/CAproto/index.html

PVA Get sequence



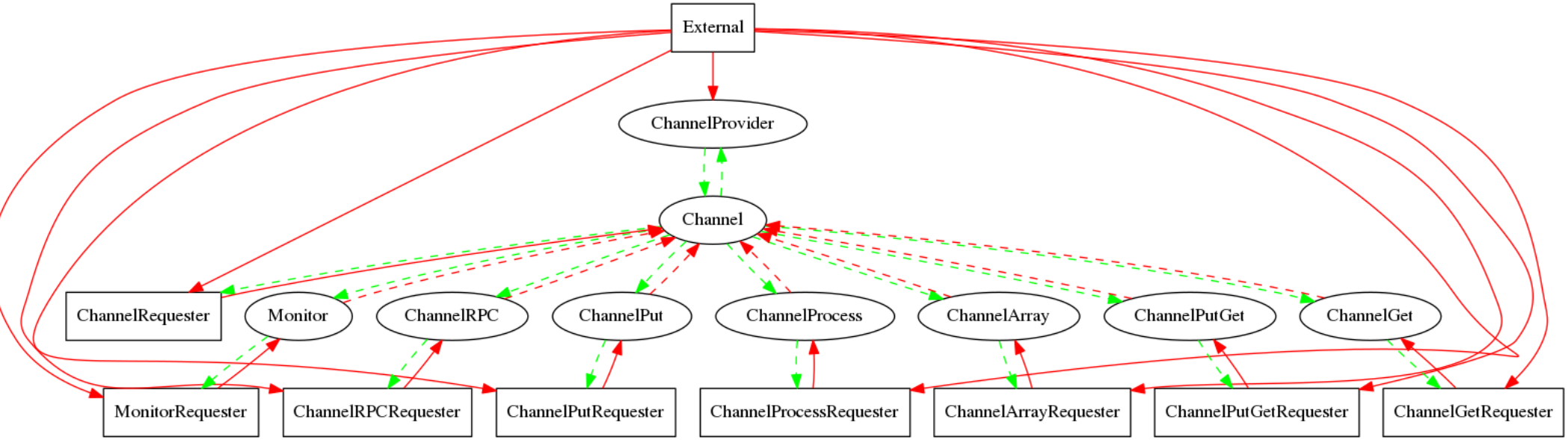
https://github.com/mdavidsaver/cashark

wireshark -X lua_script:pva.lua test/pva-ops.pcapng

Osprey DCS

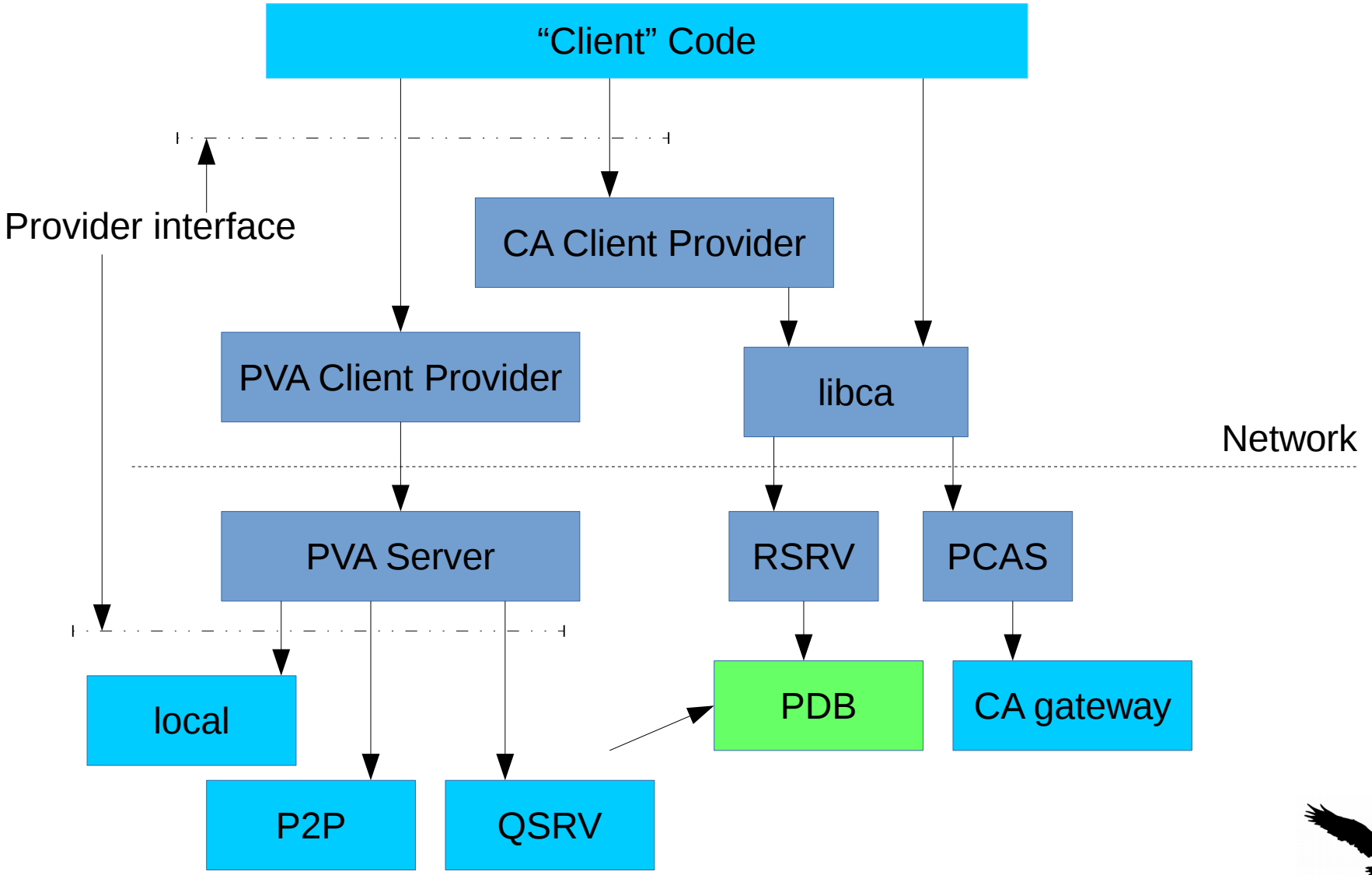
Roles

- Operation method
  - Client calls
  - Server implements
- Requester
  - Client implements
  - Server calls


External
ChannelProvider
Channel
ChannelRequester
Monitor
ChannelRPC
ChannelPut
ChannelProcess
ChannelArray
ChannelPutGet
ChannelGet
MonitorRequester
ChannelRPCRequester
ChannelPutRequester
ChannelProcessRequester
ChannelArrayRequester
ChannelPutGetRequester
ChannelGetRequester

Osprey DCS

PV* Nomenclature



Osprey DCS