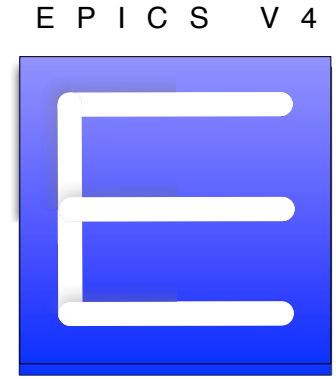
EPICS VERSION 4 AND SWISSFEL



http://epics-pvdata.sourceforge.net/

Gregory White, for EPICS V4 team, 23-Oct-2012, SLAC/PSI

EPICS Version 4 and SwissFEL Model

1. EPICS Version 4 Summary
2. Scientific Data Support
3. Model and other Data Services for SwissFEL
4. Working Group Organisation and Status

Gregory White, SLAC/PSI, for EPICS V4 team, 23-Oct-2012

The EPICS v4 Working group presently has the following members:

Name	Member Organisation	Status	Interests	Charter Deliverables	Scribe date
Gabriele Carcassi	BNL	Participant	General purpose services, client tools and their interoperability, such as PvManager, BOY, ChannelFinder, and data types	Directory Service specification, Directory Service implementation and pvlist tool, Interoperable Data Types specification, pvManager	7/Sep/2011
Benjamin Franksen	HZB	Observer			
Bob Dalesio	BNL	Participant, co-chair	Core architecture for control, administration	Money	14/Sep/2011, 22/Sep/2011, 7/Dec/2011
Michael Davidsaver	BNL	Observer			
David Hickin	Diamond	Participant			15/Feb/2012
Andrew Johnson	APS	Observer			
Timo Korhonen	PSI	Participant	Services for physics.		26/Oct/2011, 09/Nov/2011, 29/Feb/2012
Marty Kraimer	BNL	Participant	Core architecture, protocol standards and Java implementations of standards.	pvData Specification, IOC Pipeline Specification, pvAccess Implementations, pvData Implementations, pvIOC Implementations, EPICS v3 to EPICS v4 Interoperability report, Controls Application Developers Guide, Protocol Developers Guide	
Ralph Lange	HZB	Observer			30/Nov/2011, 4/Jan/2012, 8/Feb/2012
Nikolay Malitsky	BNL	Participant	Archiver, IOC, physics	pvIOC Implementations, Archive service	21/Dec/2011, 14/Mar/2012
James Rowland	Diamond	Participant	CSS/BOY client side for EPICS v4.	Lead editor of Nominal Architectures.	19/Oct/2011, 22/Nov/2011
Matej Sekornaja	Cosylab	Participant	Core architecture, protocol standards and C/C++ implementations of standards.	pvAccess Specification, pvAccess implementations, pvData implementations, pvIOC implementations	
Guobao Shen	BNL	Participant	Services for physics.	Performance Report	14/Dec/2011, 11/Jan/2012
Kunal Shroff	BNL	Observer	General purpose services, client tools and their interoperability, such as PvManager, ChannelFinder, data types.	Directory Service specification, Directory Service implementation and pvlist tool	
Greg White	PSI, SLAC	Participant, co-chair	Core architecture for services, Services architecture, model service	Interoperable Data Types specification, Services API Specification, Getting Started documentation	02/Nov/2011, 21/Mar/2012, 28-Mar-2012

Version 3 Supports Instrumentation

- Records represented either an input signal, an output signal or an operation to perform on a set of signals
  - Analog input, analog output, (multi-bit)binary input, (multi-bit) binary output, motor, event, PID, calc, etc…..
  - Agreeing on what a device is – is difficult. Is it a power supply or a magnet? Does a motor have an LVDT, an encoder, back lash?
- Records implement continuous control in an autonomous controller to perform DCS functionality.
- Many different types of research and industrial facilities successfully applied this to their plant for equipment control.
- Process Variables (PVs) are available across the network
  - Any field of any record can be a process variable.
  - Only functions on PVs are: get, put, monitor
  - Original EPICS was designed and implemented to be robust and fast (15K PVs per second to a client on a 100 MB network)
  - Channels always have a time stamp, alarm severity, and alarm status – the simple data type was not useful in most cases
  - Channels have metadata to describe display, control, and alarm information.
- MANY clients were developed on this interface in many languages on many operating systems implementing the full range of SCADA capabilities.
  - With two sites developing EPICS, there were two display managers.

Version 3 Has Limited Support for Devices

- Records did not operate on things more complex than scalar signals.
  - No time domain, no frequency domain, no images.
  - No way to represent things more complicated than scalar signals and 1 dimensional arrays
- Process Variables available across the network could not support everything needed
  - No atomic command / response mechanism
  - No way to ask for a PV subject to parameters.
  - PVs metadata did not always fit properly for every field of a record – such as the display precision – what is the time stamp of this?
  - Typically a get is done on connection for display, alarm limits, and control metadata changes are not reflected.
  - Meta data was sent all of the time, so only time stamp and current alarm information is monitored.
- MANY clients added layers on top of V3 Process Variables to implement more complex data models

EPICS Version 4

EPICS V4 = EPICS V3 + New communications protocol
+ A platform for scientific data exchange and services
+ A platform for new IOC design

Version 4 Supports Complex Data Structures

pvData: Structured PVs are now available across the network

- Functions on PVs are: get, put, monitor, put/process/get (command/response)
- Structured data plus Directory services allow you to create whole data models like accelerators or telescopes

PVAccess: a new protocol, which supports middle layer High Performance RPC services, using the same protocol as control

pvIOC: In future the V4 IOC can potentially represent devices

- However, not yet.
- They will use V3 records at NSLS II.
- However, V4 serving device control data actually implemented by V3 works today.

"Normative" Types defined to provide metadata for more complex constructs: multi-channel array, table, N dimensional Array, Image.

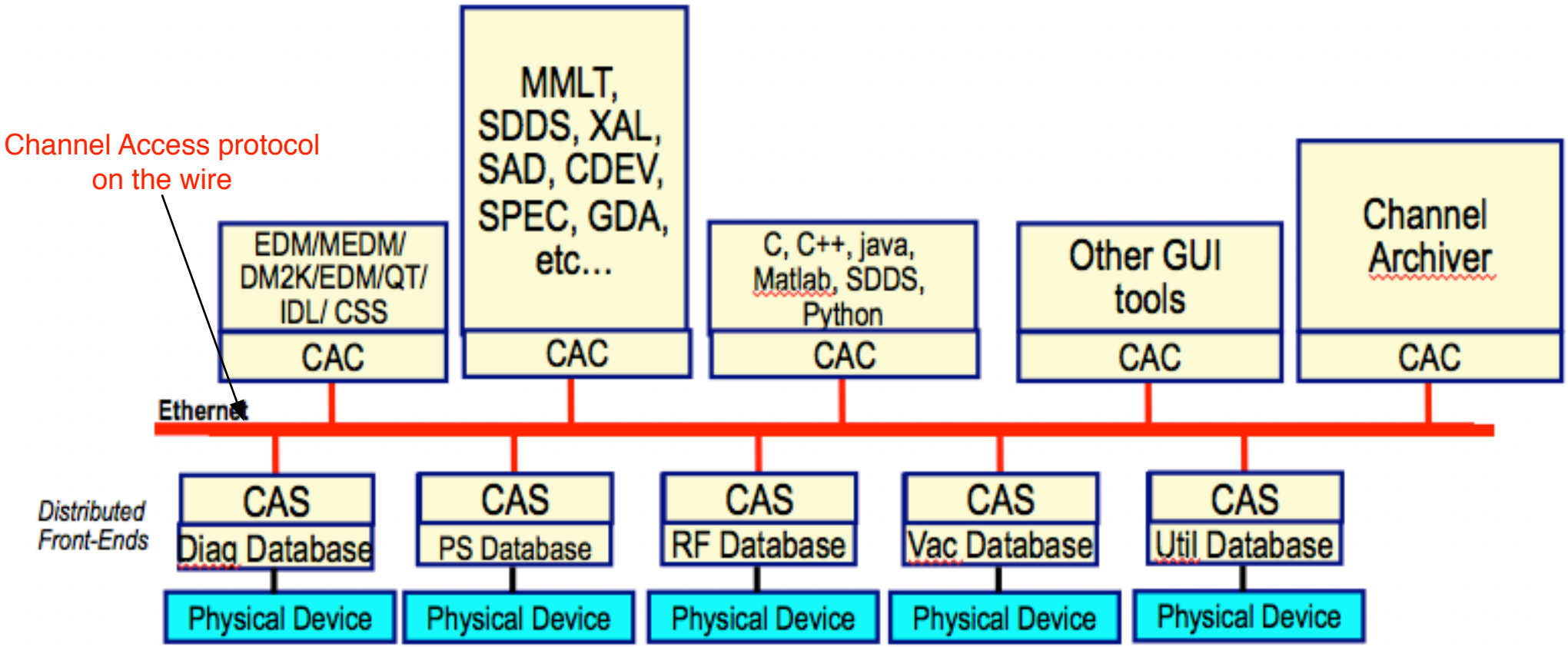
- PVData always has a time stamp, alarm severity, and alarm status
- Vectors have useful metadata and distinctions: time domain vector, frequency domain vector, histogram
- Operations can be performed on two PVs with the same normative types.

MANY servers are being developed on this interface to implement middle layer through a collaboration for physics applications. A second collaboration is being established for beam line control, data acquisition, and data analysis.

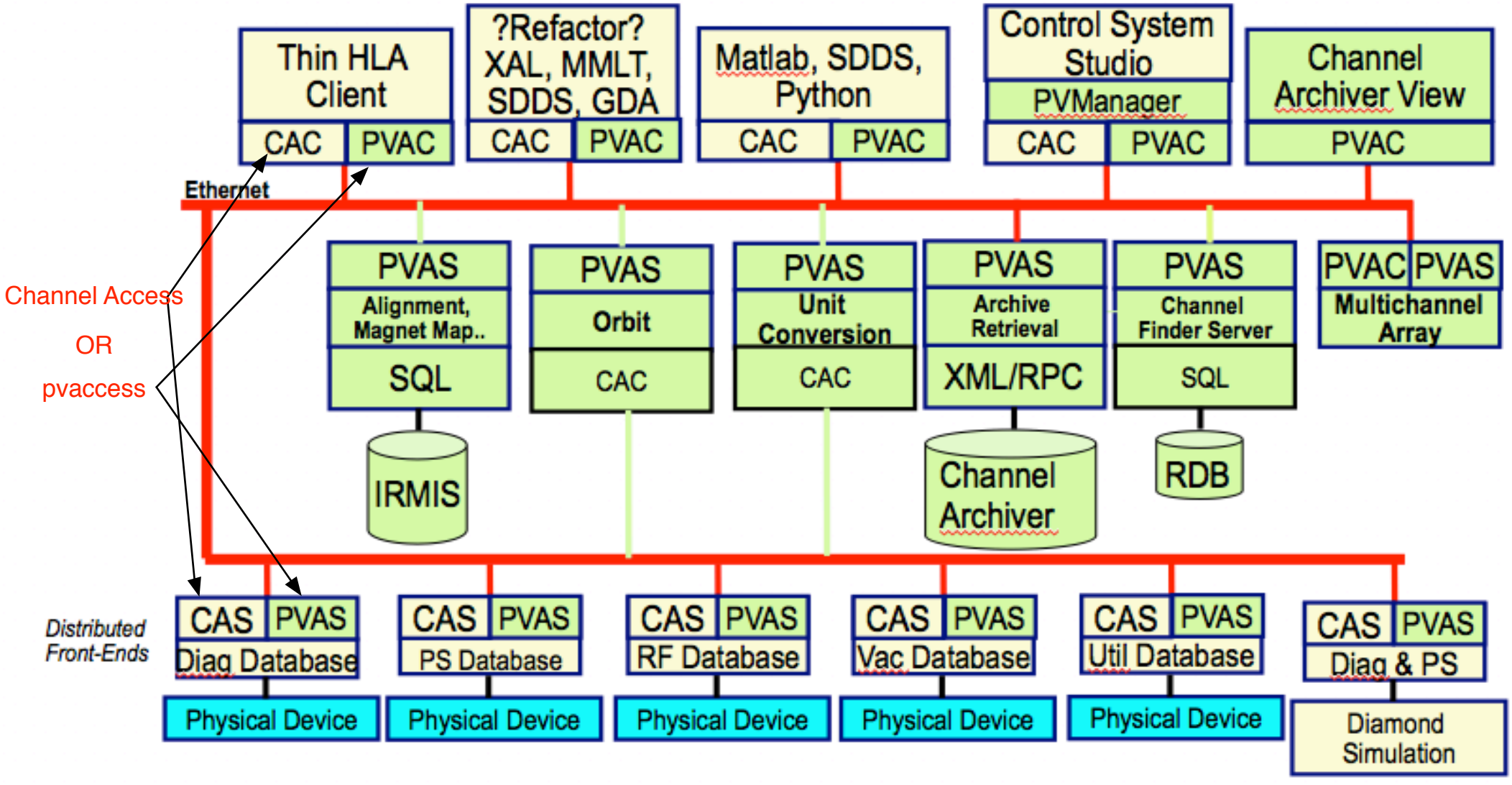
BOB DALESIO GREG WHITE

BOB DALESIO

EPICS Version 3 Architecture



Applying Version 4 to Machine Control



EPICS V4 Principal Additions

New Functionality	Provided by in EPICS V4
CA => pvAccess : A Standardized protocol specification	
Structured Data Exchange and PV Records	pvAccess
Arguments	
Send only deltas	
Full Asynchronous Error and Message passing	pvData
Unsigned Int directly supported	
High Performance RPC Data Service Software Platform	
New IOC to support above	pvIOC
XML defined EPICS DB	

EPICS V4 Principal Science Support Additions

New Functionality	Provided in EPICS V4 by
Scientific Data Services	channelRPC
Standardized High Level Data Types	Normative Types
Data Acquisition Management Tools	pvManager, Gather platform
Directory Service	ChannelFinder EPICS V4 service
Direct Matlab and Python support	C++, Java and Python bindings

EXAMPLE 1.

Example 1: Archiver Data Service.

Data are served by a V4 service, over pvAccess. That is, entirely EPICS V4 core, no extension

```
$ pvget -a starttime=21-Jun-2012T17:50:00 -e endtime=now QUAD34_Bfield;history

##QUAD34_Bfield
#timePastEpoch(s)    #value                #Date                       #Alarm
496169397.856321000   7.355487346649e-02  Wed Jun 21 17:49:57 2012  NO ALARM
496169401.996447000   1.682446300983e-01  Wed Jun 21 17:50:01 2012  NO ALARM
496169410.052636000   2.558367252350e-01  Wed Jun 21 17:50:10 2012  NO ALARM
496169420.109690000   3.173123300076e-01  Wed Jun 21 17:50:20 2012  NO ALARM
496169430.100015000   2.159405648708e-01  Wed Jun 21 17:50:30 2012  NO ALARM
496169440.081932000   4.953919649124e-01  Wed Jun 21 17:50:40 2012  NO ALARM
496169450.089935000   3.187555372715e-01  Wed Jun 21 17:50:50 2012  NO ALARM
496169450.699760000   0.000000000000e+00  Wed Jun 21 17:50:50 2012  Disconnected
496169450.699760000   0.000000000000e+00  Wed Jun 21 17:50:50 2012  Archive_Off
496169537.905713000   0.000000000000e+00  Wed Jun 21 17:52:17 2012  Disconnected
```

EXAMPLE 2.

Example using the general purpose EPICS V4 client (caget) to get a quadrupole's R-matrix from an EPICS V4 implemented model service.

Note: Arguments

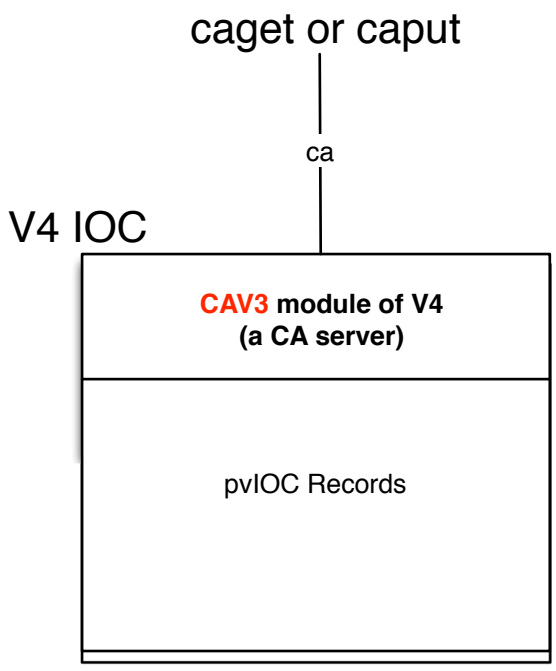
```
$ pvget QUAD:LI21:271:R -a TYPE=DESIGN -a POS=MID -a RUN=LATEST
       0.23     0.1234  0.0      0.0      0.067562 0.001167
      -0.34520  0.0923  0.0      0.0      0.046981 0.001514
         0.0      0.0    1.881007  4.857304 0.0      0.0
         0.0      0.0   -1.50064  -3.862346 0.0      0.0
      -0.00132 -0.001129 0.0      0.0      0.224701 0.003894
       0.162595 0.10285  0.0      0.0    -19.603   -0.233109
```

Note: prints as a matrix

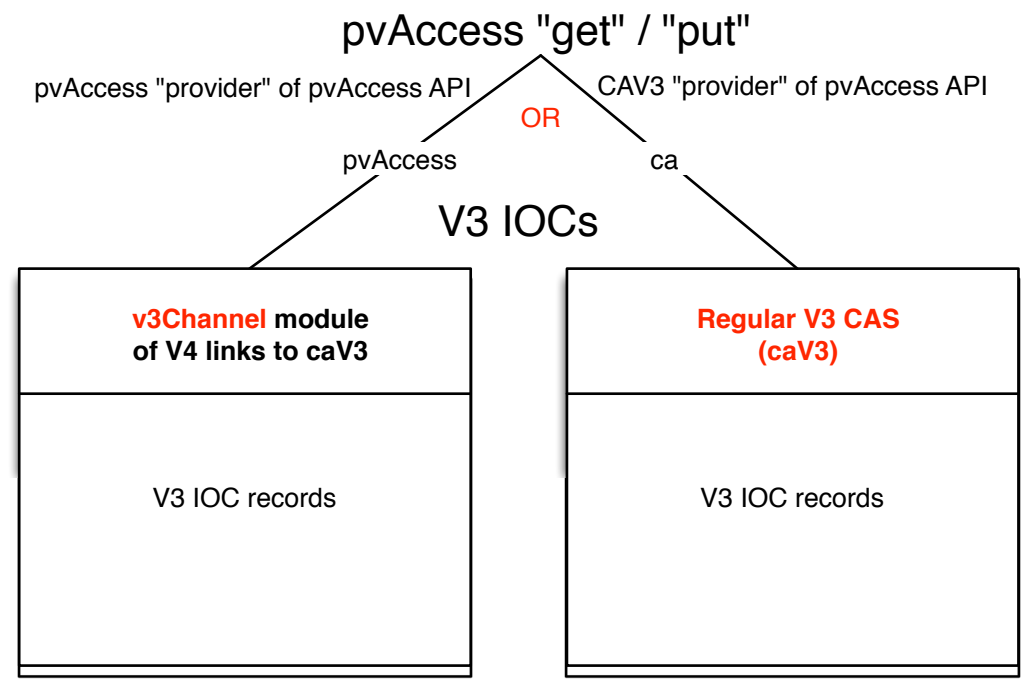
EPICS V3-V4 INTEROPERATION

Interop is via V3's "CAV3" and V4 pvIOC subsystem "V3Channel"

V3 client ←→ V4 server

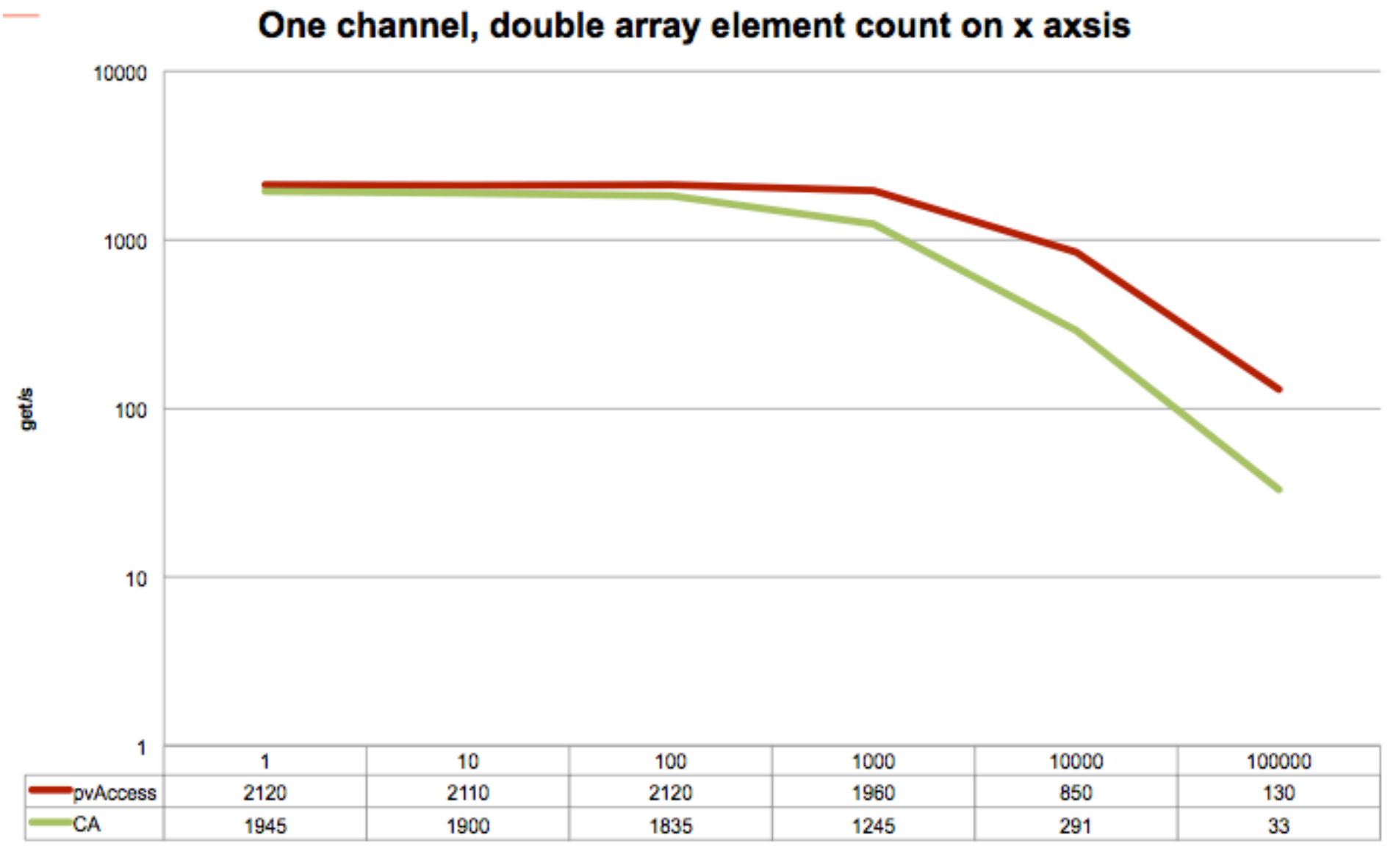


V4 client ←→ V3 server



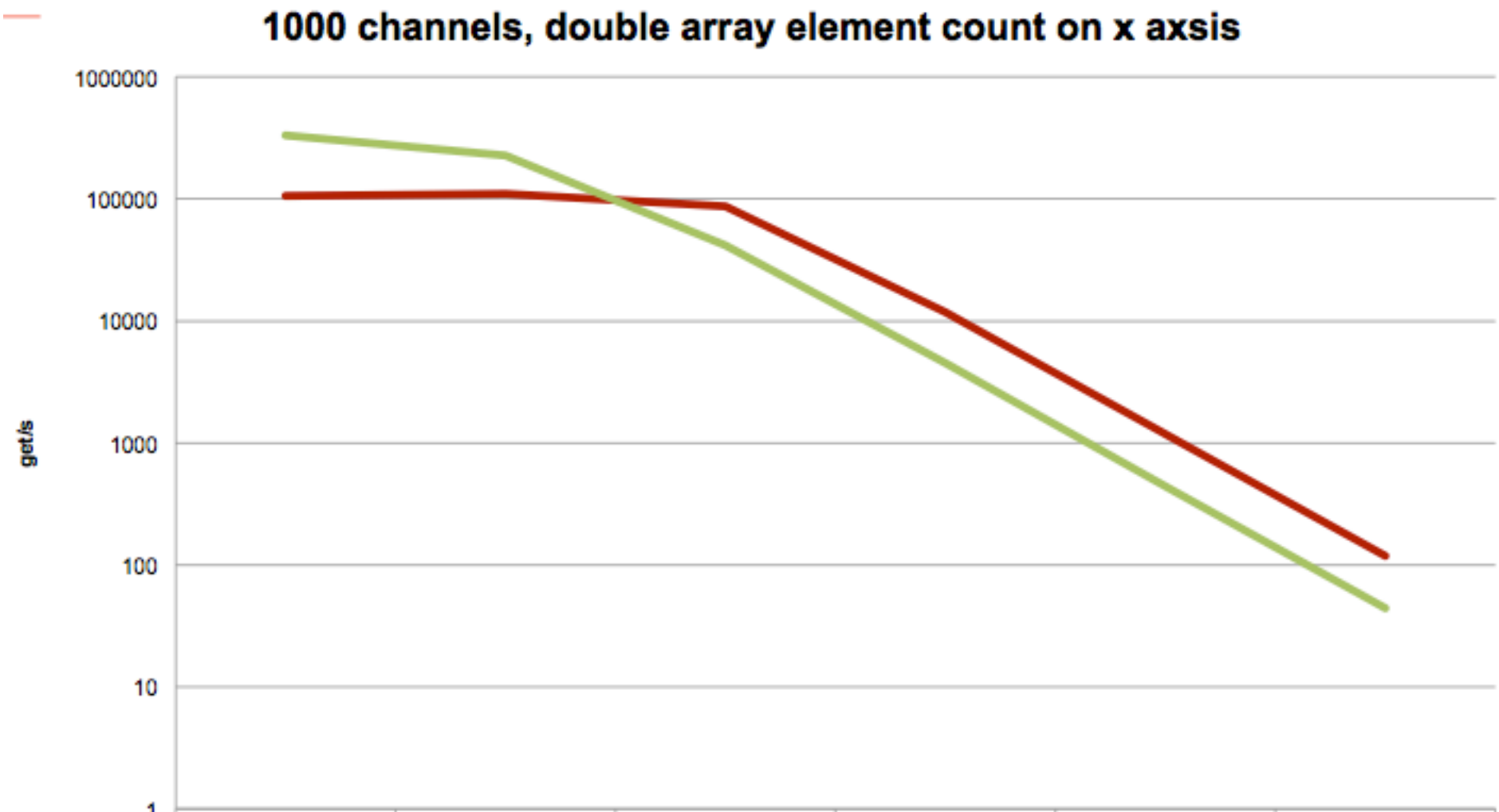
See pvIOCCPP documentation [2] Architectures Document [3], and summary in V4 FAQ [4]

EPICS V4 Performance (1)



	1	10	100	1000	10000	100000
pvAccess	2120	2110	2120	1960	850	130
CA	1945	1900	1835	1245	291	33

EPICS V4 Performance (2)



	1	10	100	1000	10000	100000
pvAccess	106000	111000	86600	11800	1183	118
CA	333000	228000	41570	4510	443	44

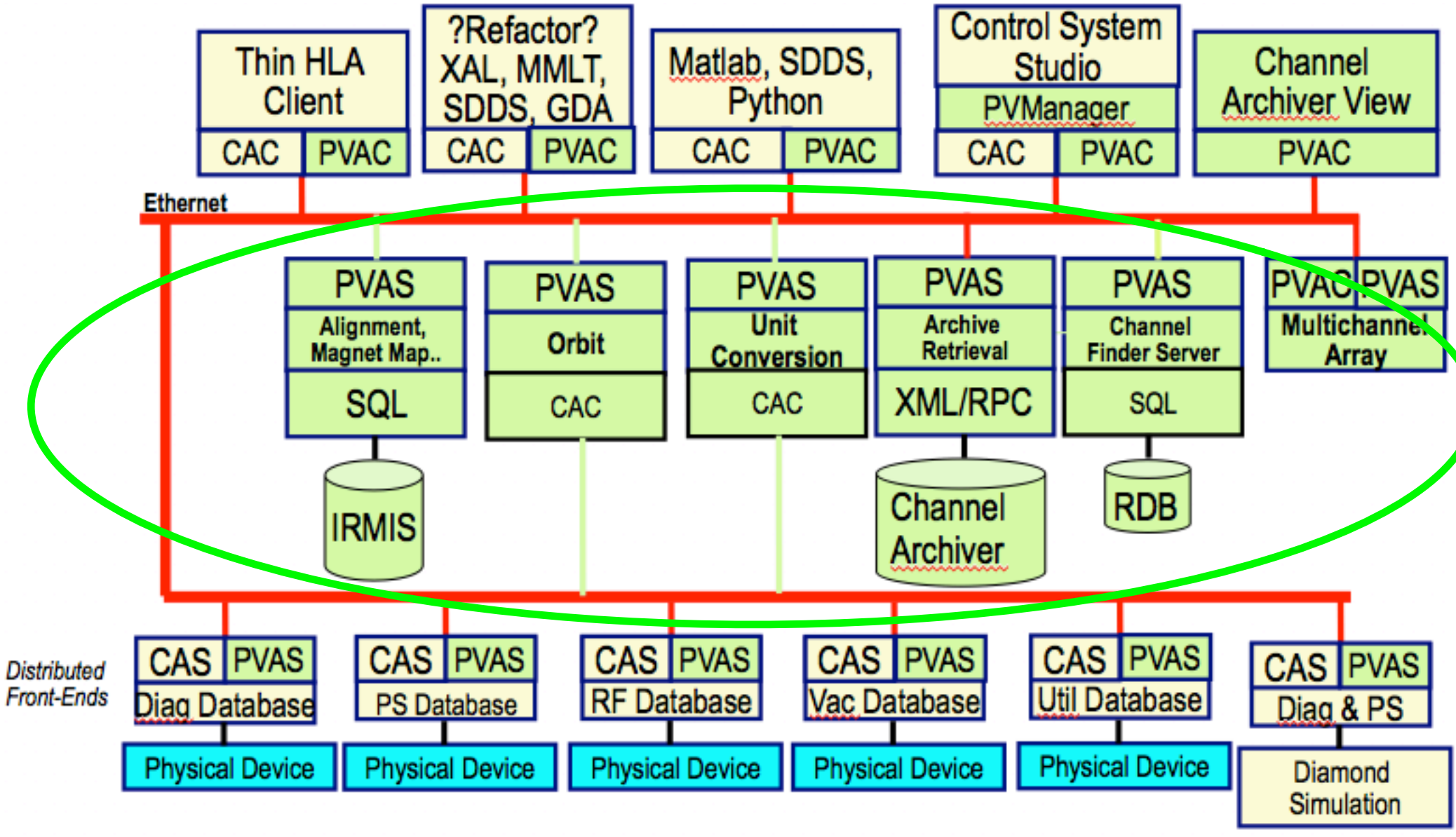
EPICS V4 Principal Science Support Additions

New Functionality	Provided in EPICS V4 by
Scientific Data Services	channelRPC
Standardized High Level Data Types	Normative Types
Data Acquisition Management Tools	pvManager, Gather platform
Directory Service	ChannelFinder EPICS V4 service
Direct Matlab and Python support	C++, Java and Python bindings

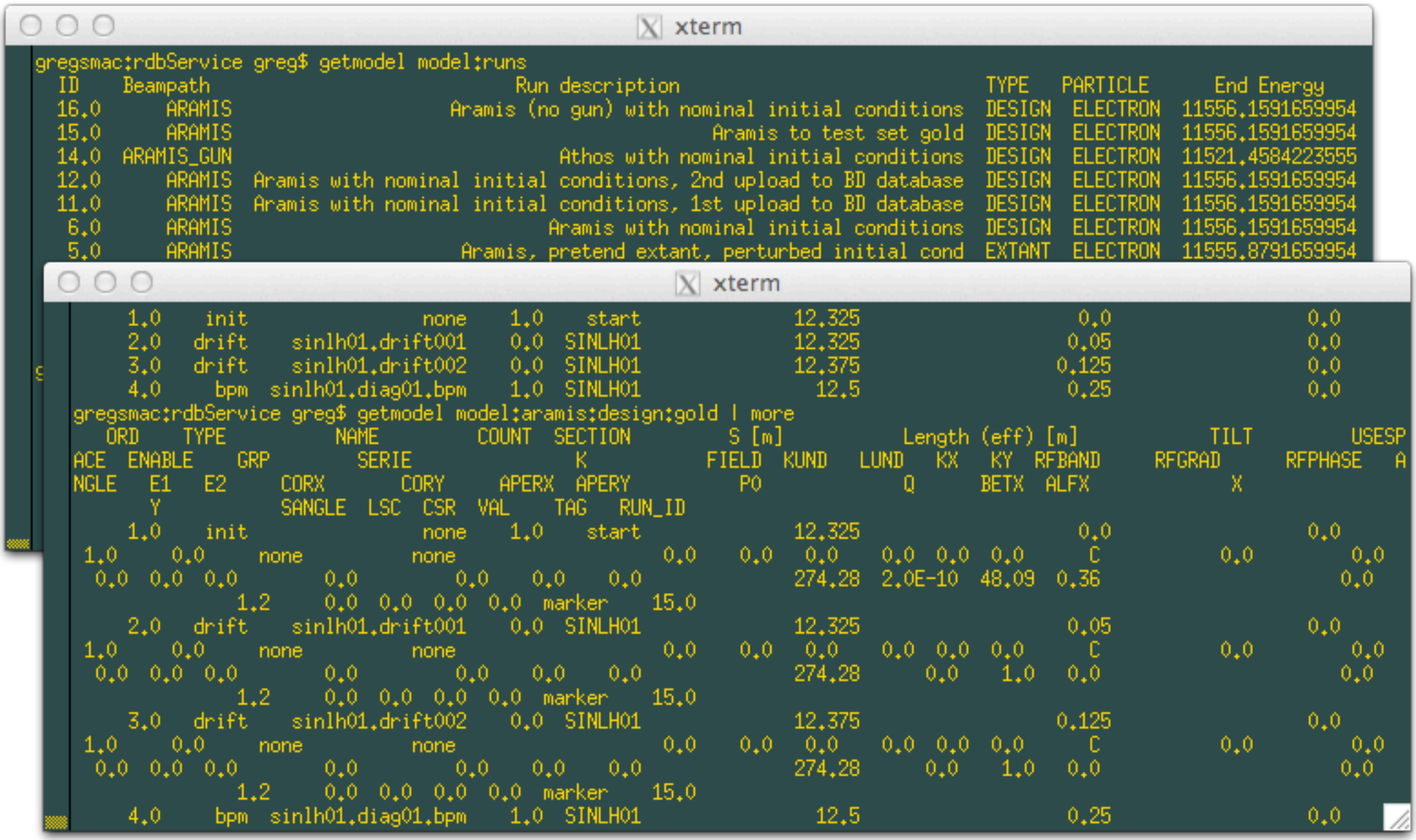
EPICS V4 Principal Science Support Additions Revisited

New Functionality	Examples	Provided in EPICS V4 by
Scientific Data Services	Lattice service, BPM Orbit service	channelRPC
Standardized High Level Data Types	Matrix, Table, "Any" data types, many others	Normative Types
Data Acquisition Management Tools	Synchronous BPM Orbit, Continuous LEM update	pvManager, Gather platform
Directory Service	lattice elem->device->channel	ChannelFinder EPICS V4 service
Direct Matlab and Python support	`>> orbit=epva.get(… 'SwissFEL:gunToARAMIS');`	C++, Java and Python bindings

Scientific Data Services Layer



Lattice Data Service

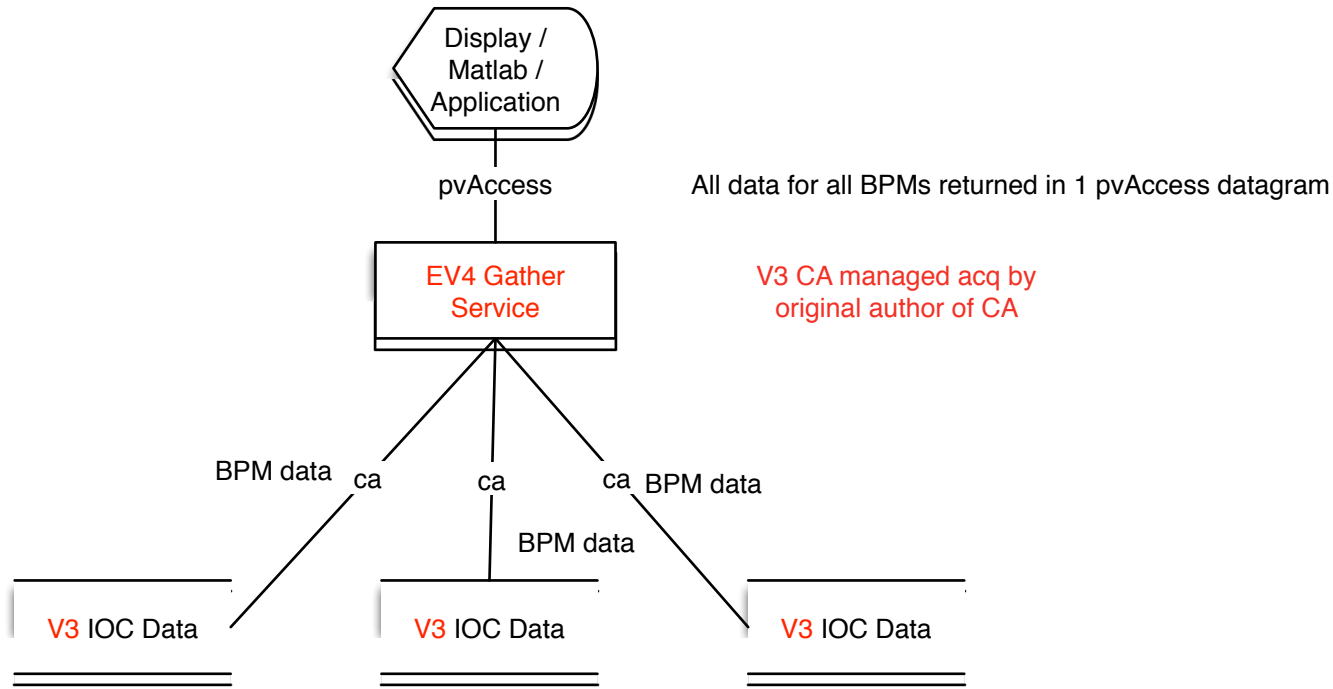


Gregory White, SLAC/PSI, for EPICS V4 team, 23-Oct-2012

BPM Orbit Data Service

Gather Service Platform: A Very Efficient PV Data Acquisition Framework for V3 PVs

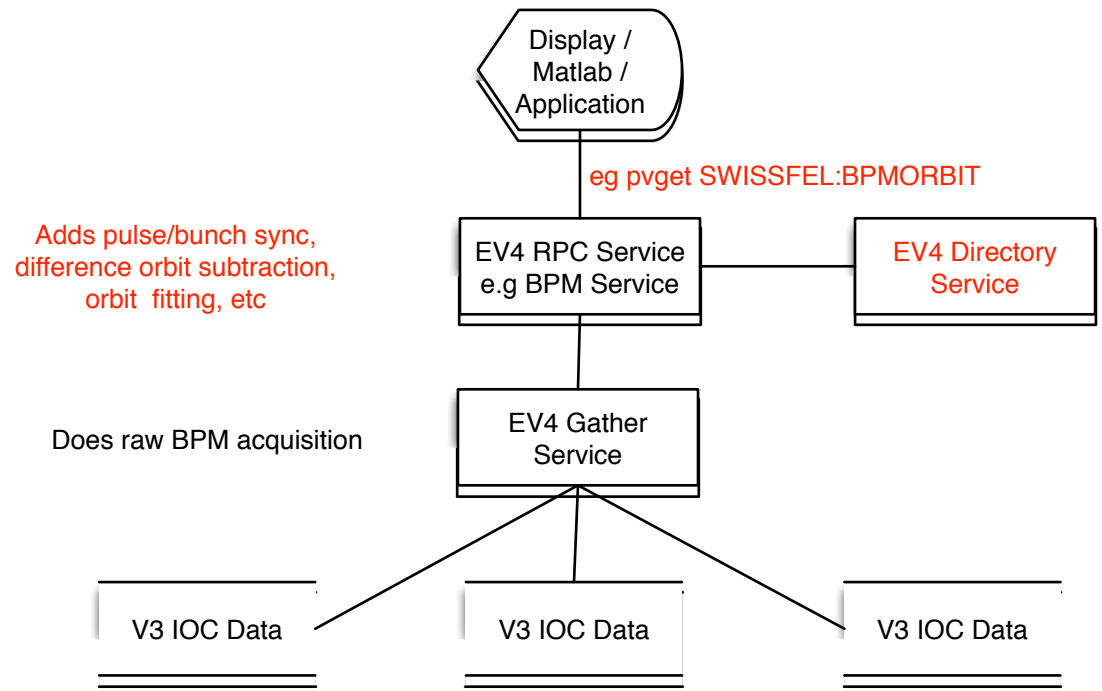
Example: Getting BPM data from many BPMs with an EPICS V4 Gather Service
NOTE: Reduces network load from M clients x N servers to M + N



EPICS V4 BASIC SCIENTIFIC SERVICE ARCHITECTURE

Beam Dynamics Services = EPICS V4 "RPC" service
+ Gather Service + Directory Service

Example: User accesses a BPM Orbit Service to "physics" oriented orbit data

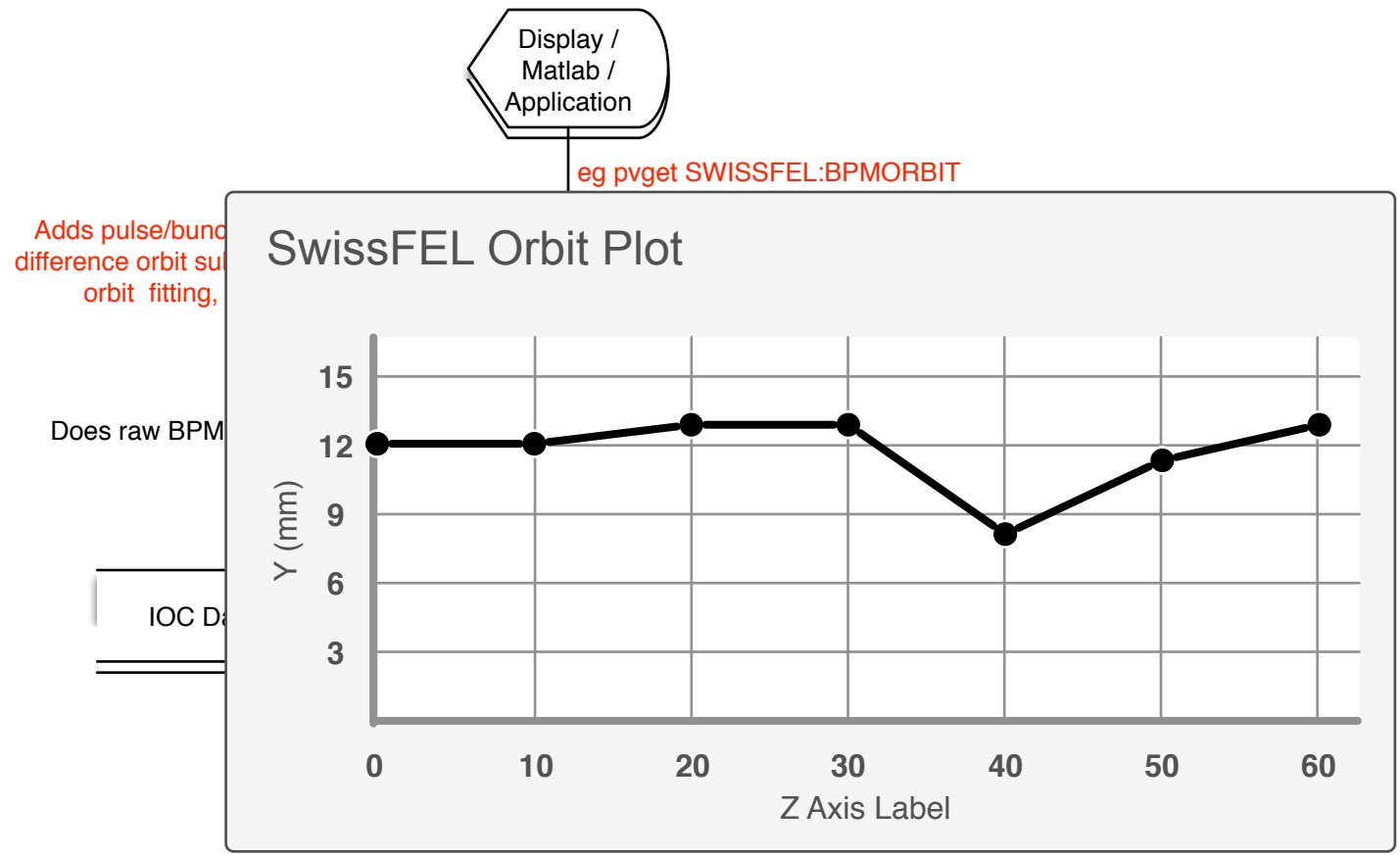


Gregory White, SLAC/PSI, for EPICS V4 team, 23-Oct-2012

EPICS V4 BASIC SCIENTIFIC SERVICE ARCHITECTURE

Beam Dynamics Services = EPICS V4 "RPC" service
+ Gather Service + Directory Service

Example: User accesses a BPM Orbit Service to "physics" oriented orbit data



Gregory White, SLAC/PSI, for EPICS V4 team, 23-Oct-2012

EPICS V4 "NORMATIVE" DATA TYPES

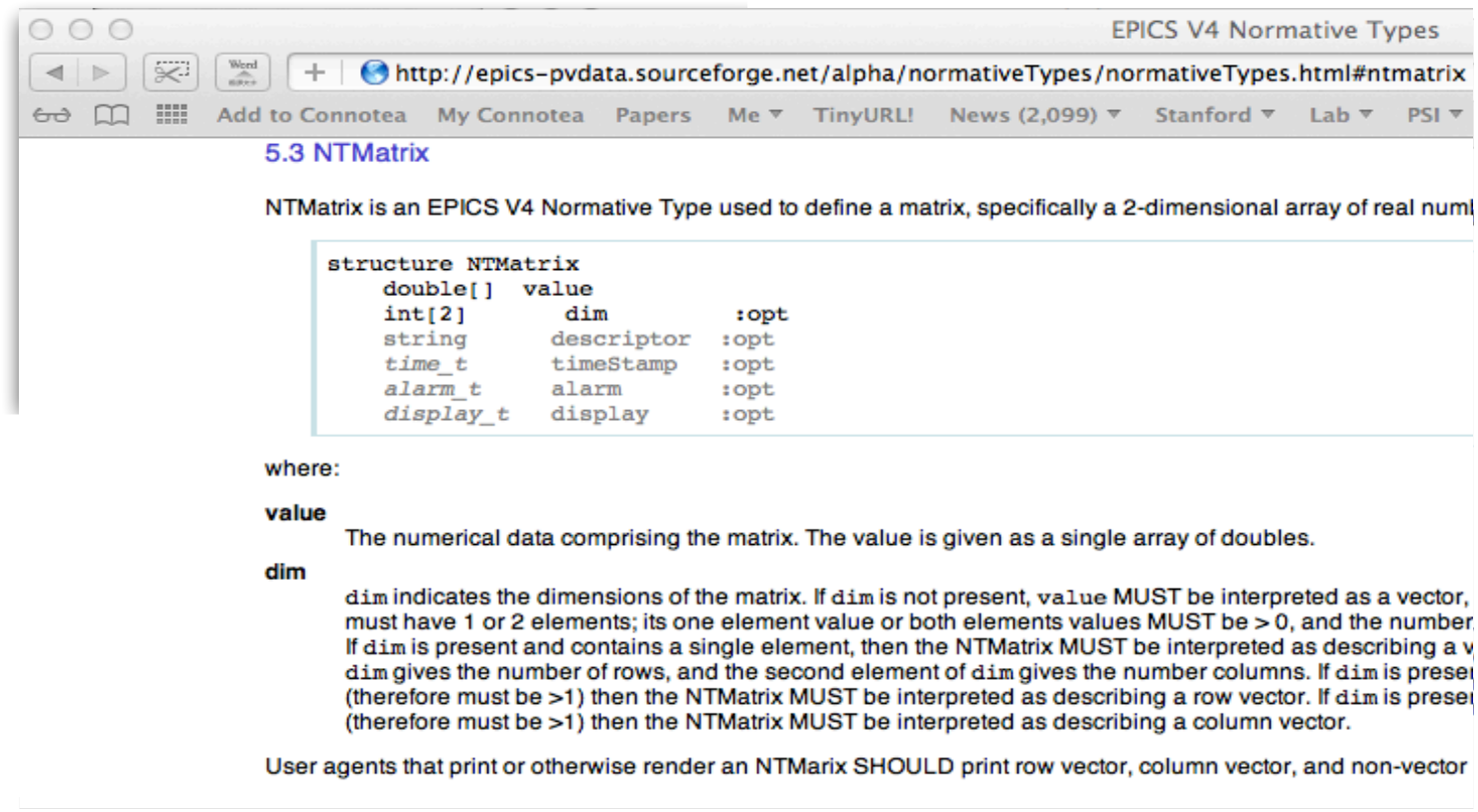
Solves the problem of high level data interoperability

E.g. New Qt based displays - how will it know it got a table, or a matrix, or an image?

All general purpose clients MUST understand the EPICS V4 Normative Types,
to be considered EPICS V4 conforming

Services SHOULD provide only EPICS V4 Normative Types.

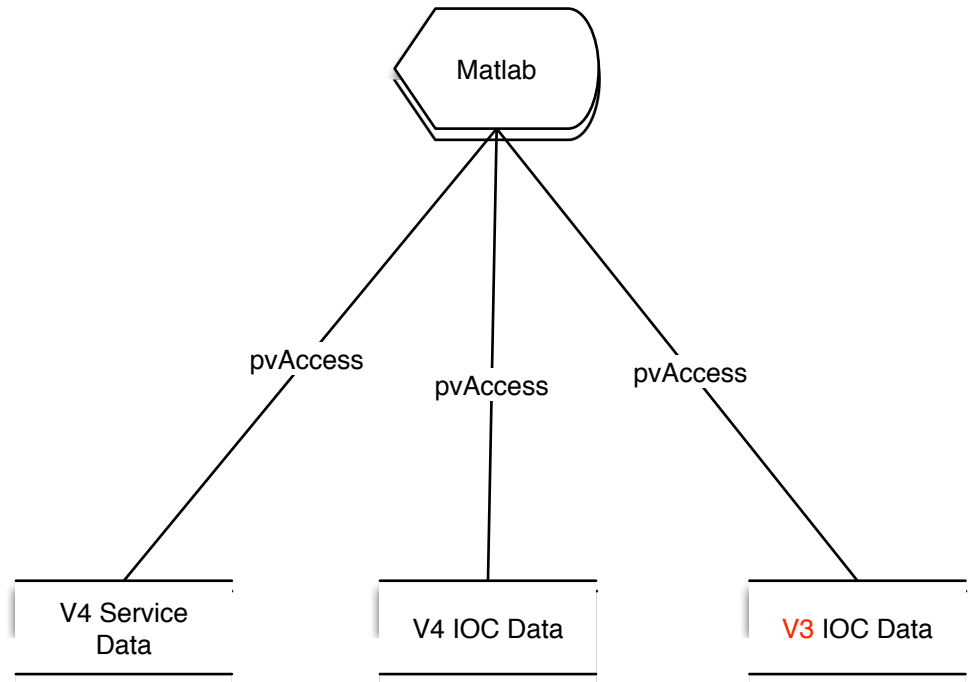
Example: NTmatrix:



See http://epics-pvdata.sourceforge.net/alpha/normativeTypes/normativeTypes.html

EPICS V4 MATLAB INTERFACE

In Matlab use EPICS V4 directly, no wrapper like lca or mex



EPICS V4 Matlab interface "EasyPVA"

First do the setup, just once:

```
>> import org.epics.ca.easyPVA.*
>> easyPVA = EasyPVAFactory.get()
```

Example 1: Put a single value to a PV

```
>> easyPVA.createChannel('double01').createPut('record[process=true]field(value)').putDouble(1.9997);
```

Example 2: Get a single value from a PV

```
>> value = easyPVA.createChannel('double01').createGet().getDouble()

value =

    1.9997
```

Example 3: Put an array of values to a PV

```
>> mydata=[1.0 2.1 3.3 4.5 5.66 6.7];
>> easyPVA.createChannel('doubleArray01').createPut().putDoubleArray(mydata,length(mydata));
```

Example 4: Get an array of values from a PV

```
>> value = easyPVA.createChannel('doubleArray01').createGet().getDoubleArray()

value =

    1.0000
    2.1000
    3.3000
    4.5000
    5.6600
    6.7000
```

```
% SwissFEL orbit correction in 1/2 page of matlab
%
import org.epics.ca.easyPVA.*;
easyPVA = EasyPVAFactory.get();

% Get the names of all the Correctors and BPMs from the Directory Service
corrNamesChan = easyPVA.createChannel('DS:SwissFEL:GUN_to_ARAMIS');
corrNamesChan.addArgument('DEVICETYPETAG','XCOR');
corrNames = corrNamesChan.createGet().getStringArray();
bpmNamesChan = easyPVA.createChannel('DS:SwissFEL:GUN_to_ARAMIS');
bpmNamesChan.addArgument('DEVICETYPETAG','BPMS');
bpmNames = bpmNamesChan.createGet().getStringArray();
Ncor = length(corrNames);
Mbpm = length(bpmNames);

% Get BPM x orbit from the BPM service.
b = easyPVA.createChannel(...
    'BPMORBIT:SwissFEL:GUN_to_ARAMIS').createGet().getDoubleArray();

% Form the Ax-b problem getting Rmats from the Model Service
modelmatrixChan = easyPVA.createChannel('model:aramis:gold:extant:R');
for bpmi = 1:Mbpm;
    modelmatrixChan.addArgument('to',bpmNames(bpmi));
    for corj = 1:Ncor;
        modelmatrixChan.addArgument('from',corrNames(corj));
        PVStructure = modelmatrixChan.createGet().getPVStructure();
        RmatCorToBpm=PVStructure.toMatrix();
        A(bpmi, corj) = RmatCorToBpm(1,2);
    end
end
x = inv(A)*b;              % Solve Ax-b
newBDESes = -KtoB(x);      % new B field values from K to B

% Deploy the new magnet settings.
magSetChan = easyPVA.createChannel('MAGNETSET');
magSetChan.addArgument('magnetlist',corrNames);
magSetChan.createPut().putDoubleArray(newBDESes,length(newBDESes));
```

EPICS V4 Charter + Deliverables, Status

Status at completion of 2011-2012 Charter

6.1 Deliverables

The group is expected to produce the following normative deliverables:

1. A normative document of the pvAccess protocol — 100%
2. A normative document of the pvData protocol. The document must include the user API - how a programmer creates data objects for the wire, and extracts them on the other side — 100%
3. A normative document of the EPICS V4 IOC processing pipeline — 30%
4. A reference implementation of pvAccess in each of C++ and Java language bindings
5. A reference implementation of pvData in each of C++ and Java language bindings — 80%
6. A reference implementation of the EPICS V4 IOC in each of C++ and Java language bindings. The Java version has high priority
7. A normative document of the EPICS V4 interoperable data types. These data types must be universally understood by every client and service which claims EPICS V4 compatibility. The requirement for this deliverable is distinct from the pvData document deliverable, since pvData can encode any type, this deliverable recommends the confined set of data objects that will be used by EPICS V4 interoperable services — 95%
8. A directory service accessible through the EPICS V4 API itself, from which can be found at least PV and entity names, and associated service names — 70%

EPICS V4 Charter + Deliverables, Status 2.

100% 9. A normative document of the EPICS V4 services API. This defines the form for encoding parameters and status descriptions between clients and services and back

10. A report of interoperability of the EPICS V4 IOC with EPICS v3 record processing 20%

70% 11. A performance report, comparing EPICS v3 to EPICS V4 for some common EPICS v3 control and read tasks, plus report of the expected performance of EPICS V4 service support. For instance, round trip time for network encoding/deserialization of results of 4 or 5 common service queries such as archive data, orbit data, whole beamline model etc. Comparisons to at least 2 other common high performance data interconnects should be made, eg ICE, ASN.1, EXI Web Service.

12. A "Getting Started" document for EPICS V4 Service developers 100%

100% 13. A User Guide for EPICS V4 IOC control application developers

14. A command line tool similar to caget (call it say pvget), which understands all the interoperable data types above, and conforms to the EPICS V4 services API above. 85%

20% 15. A normative document of the EPICS V4 Directory Service function, API, and unix command line tool.

20% 16. A reference implementation of the EPICS V4 Directory Service.

The 3 Directory Service deliverables, and the Interoperability report make up 2/3 of delivery failure.
That will be fixed shortly.
The remaining IOC Processing Pipeline design will be a major part of next year's charter.

2012-2013 EPICS V4 Charter

2011-12 Charter soon complete: basic data architecture for scientific data exchange

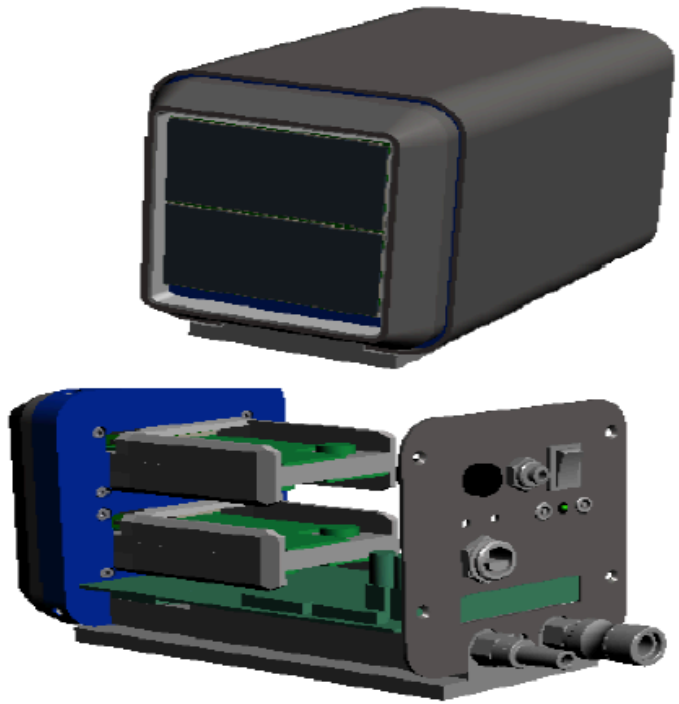
2012-13 Charter will concentrate on:

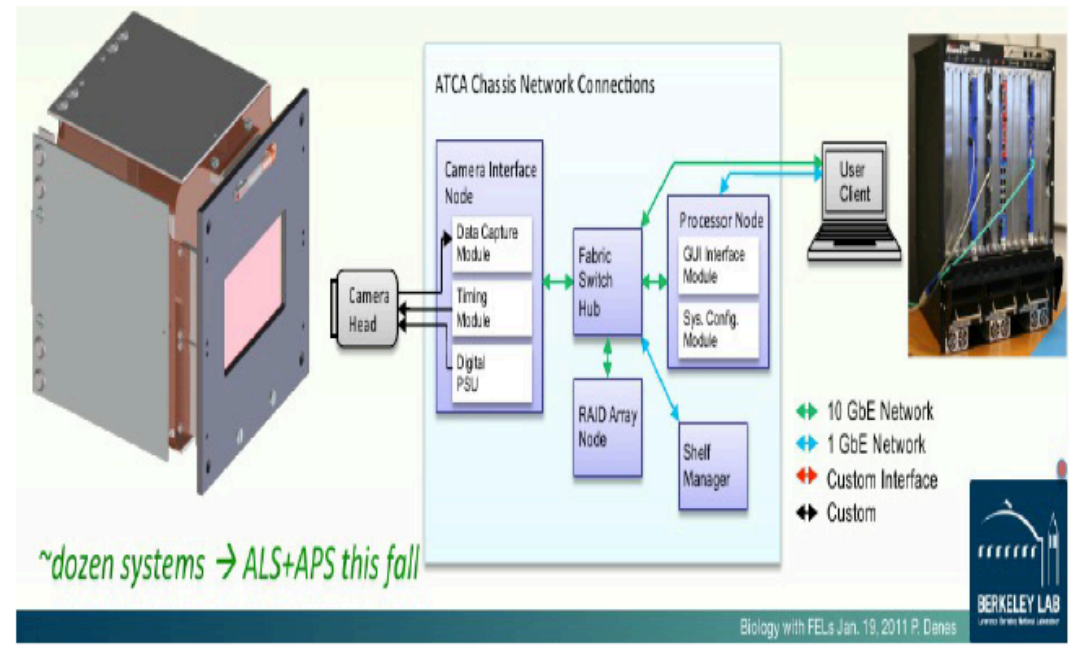
1. V4 in the classic "V3" IOC
   + Get/put lockset of V3 channels through V4 structure file
   + Documentation on V3-V4 interop, and develop standard architecture
   + Proper vxWorks port. Possibly RTEMS?
2. Improved support for experimental data acquisition in the IOC
   + NDArray support - areaDetector. Normalizing areaDetector using Normative Types
   + Image Library - tools for manipulating images and packaging as NTImage
   + Monitors suitable for data acquisition.
     Guaranteed in-order delivery and configurable queue size and replacement
   + areaDetector driver (like simDetector) connected to a V4 record layer;
     dynamically created fields according to the underlying parameter library
   + Possibly a coordinate space conversion library. Mapping coordinate space to reciprocal space
3. Develop a proposed design for the version 4 IOC processing pipeline
   pvIOC is only a straw man and alpha implementation.
   Need to make it go through public review and community process.
4. GUIs. pvManager integration. caQtDM. Matlab reference examples.

2013-14 Charter will likely implement the V4 IOC processing pipeline from 2012-13 charter.

AFTER THE NEXT STEPS, NEW V4 IOC REQUIREMENTS

- Eiger (Dectris/PSI)
  - 1-4 Mpix @ 2-24 kHz
    - 47 Gbps @ 3 kHz (1Mpix)
- LBNL FastCCD
  - 2 Mpix @ 200 Fps
    - 6.4 Gbps

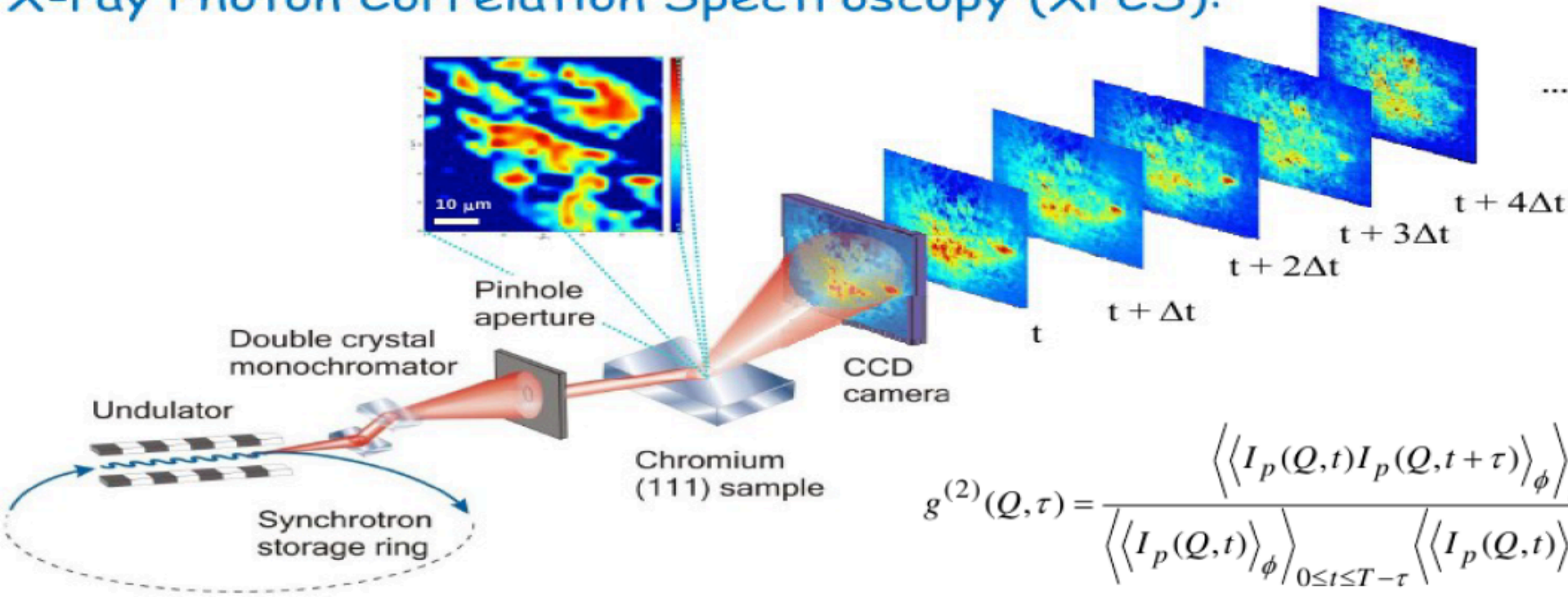




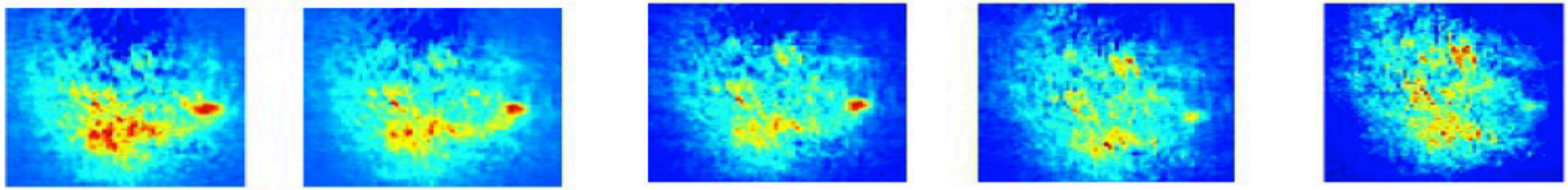
Thanks to DARON CHABOT, BNL

Data Analysis Example

X-ray Photon Correlation Spectroscopy (XPCS):



$$g^{(2)}(Q,\tau) = \frac{\left\langle \left\langle I_p(Q,t) I_p(Q,t+\tau) \right\rangle_\phi \right\rangle_t}{\left\langle \left\langle I_p(Q,t) \right\rangle_\phi \right\rangle_{0 \le t \le T-\tau} \left\langle \left\langle I_p(Q,t) \right\rangle_\phi \right\rangle_{\tau \le t \le T}}$$



O. G. Shpyrko et al., *Nature* **447**, 68 (2007)

ANALYTICAL CRYSTAL LATTICE SPACES

1) Reciprocal, or Q or K-space. The original lattice in fourier space

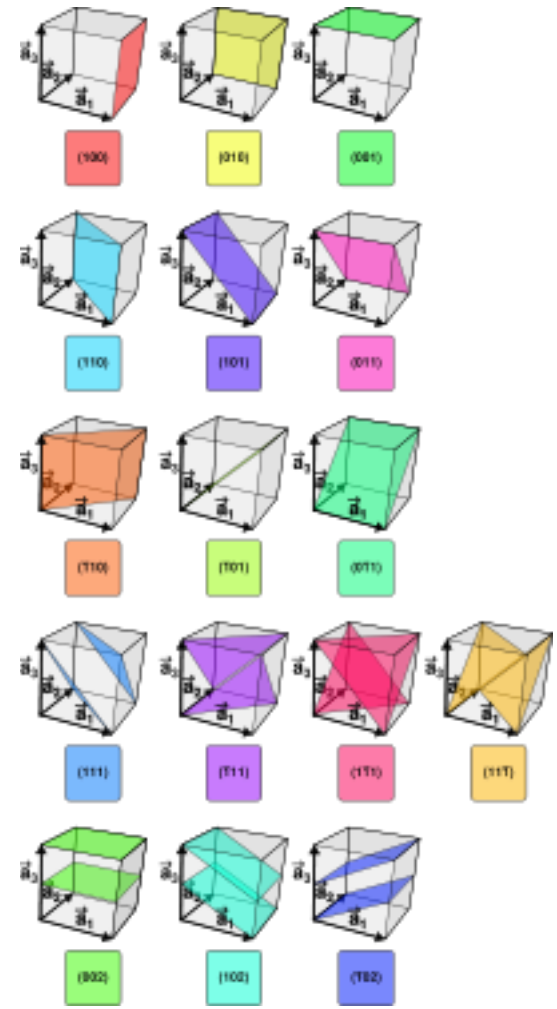
2) HKL space [h,k,l], or Bravais-Miller indeces,
each give the orientation of a plane orthogonal to the basis of the reciprocal lattice space.

Commonly used in crystallography

Orientation of [h,k,l] maps to various physical axes

Eg. Multiple defractor angles and detector position

Others too….



Other Opportunities
Not in scope of the Working Group's Charter, but useful

1. Independent Performance Measurement

2. HDF5 data save

3. pvAccess Access Security

4. Gateway

5. IOC Record and module support. May be a significant effort to move EPICS into large data and parallel processing

6. High Performance Web Server on the IOC (e.g. IBM XML screamer + W3C EXI)

7. Services
   - Snapshot save and Restore (Done by BNL)
   - BPM Orbit (Being done by PSI)
   - Model (Being done by PSI)
   - Linac Energy estimation (for correcting Quad focusing w.r.t. Energy)
   - Archive service

8. pvAccess python deserializer

MOST OF ALL - JUST USE IT TO SOLVE PROBLEMS AND PROVIDE FEEDBACK

CONCLUSIONS

V4 orients EPICS to science in addition to control

V4 includes V3. V4 is a significant version upgrade to V3, not an alternative to V3

EPICS V4 is technically ready for host based service development - beta.

EPICS V4 IOC is not ready for control, but that's ok, do control with V3 IOC

Full Interoperation: You can supply V3 data to V4 clients, and V3 clients can get V4 simple data

V4 gives complex data, efficiently network managed by shared memory system

V4 gives PV values according to arguments

Direct matlab through Java API, and possibly python, no wrappers

The EPICS V4 working group has been very successful at creating a new platform for scientific data

Standards driven. Allows Independent implementation

It seems real. It's good. Works, fast, well documented.

REFERENCES

[1] pvAccess Protocol Specification, http://epics-pvdata.sourceforge.net/pvAccess_Protocol_Specification.html

[2] V3/V4 Interoperation: See pvIOCCPP documentation, sections 3 and 4

http://epics-pvdata.hg.sourceforge.net/hgweb/epics-pvdata/pvIOCCPP/raw-file/tip/documentation/pvIOCCPP.html#overview_of_ioccore,_pvaccess_and_pvioc and #cav3/v3record_<==>_pvioc/pvaccess

[3] EPICS V4 Architectures, http://epics-pvdata.sourceforge.net/arch.htm

[4] EPICS V4 Normative Types, http://epics-pvdata.sourceforge.net/alpha/normativeTypes/normativeTypes.html (Editor's Draft)

[5] Gather Service, http://epics-pvdata.sourceforge.net/alpha/gatherStatus.html

[6] EasyPVA, http://epics-pvdata.hg.sourceforge.net/hgweb/epics-pvdata/alphaCPP/raw-file/tip/easyPVA/documentation/easyPVA.html

[7] EPICS V4 FAQ, http://epics-pvdata.sourceforge.net/faq.html

[8] PSI EPICS V4 SwissFEL Installation and Programmers Guide Example, http://epics-pvdata.sourceforge.net/exampleinstall.txt