EPICS VERSION 4
OVERVIEW AND STATUS

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

Gregory White, for EPICS V4 team, 25-Apr-2013 SLAC/PSI
## Membership

The EPICS v4 Working group presently has the following members:

<table>
<thead>
<tr>
<th>Name</th>
<th>Organisation</th>
<th>Status</th>
<th>Interests</th>
<th>Charter Deliverables</th>
<th>Scribe date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gabriele Carcassi</td>
<td>BNL</td>
<td>Observer</td>
<td>General purpose services, client tools and their interoperability, such as PvManager, BOY, ChannelFinder, and data types</td>
<td>Directory Service specification, Directory Service implementation and pvlist tool, Interoperable Data Types specification, pvManager</td>
<td>7/Sep/2011</td>
</tr>
<tr>
<td>David Hickin</td>
<td>Diamond</td>
<td>Participant</td>
<td>Beamline applications, V3/V4 interoperability and pvData</td>
<td>Archive service</td>
<td>15/ Feb/2012, 4/Apr/2012, 23/Aug/2012, 19/Sep/2012</td>
</tr>
<tr>
<td>Andrew Johnson</td>
<td>APS</td>
<td>Participant</td>
<td>EPICS administration and Integration</td>
<td>Money</td>
<td>13/ Feb/2013</td>
</tr>
<tr>
<td>Anton Mezger</td>
<td>PSI</td>
<td>Observer</td>
<td>Operational tools and controls displays, physics</td>
<td>pvIOC Implementations</td>
<td>9/May/2012</td>
</tr>
<tr>
<td>Matej Sekornaja</td>
<td>Cosylab</td>
<td>Participant</td>
<td>Core architecture, protocol standards and C/C++ implementations of standards.</td>
<td>pvAccess Specification, pvAccess implementations, pvData implementations, pvIOC implementations</td>
<td>27/Feb/13, 17/Apr/13</td>
</tr>
<tr>
<td>Kunal Shroff</td>
<td>BNL</td>
<td>Observer</td>
<td>General purpose services, client tools and their interoperability, such as PvManager, ChannelFinder, data types.</td>
<td>Performance Report</td>
<td>14/Dec/2011, 11/Jan/2012, 16/May/2012, 13/Jun/2012</td>
</tr>
<tr>
<td>Sinisa Vezoli</td>
<td>APS</td>
<td>Observer</td>
<td></td>
<td>Performance Report</td>
<td>10/Apr/13</td>
</tr>
<tr>
<td>Greg White</td>
<td>PSI, SLAC</td>
<td>Participant, co-chair</td>
<td>Core architecture, model service</td>
<td>Interoperable Data Types specification, Services API Specification, Getting Started documentation</td>
<td>2/Nov/11, 21/Mar/12, 28/Mar/12, 11/Apr/12, 12/Sep/12, 3/Oct/12, 31/Oct/12, 6/Mar/13</td>
</tr>
<tr>
<td>Dirk Zimoch</td>
<td>PSI</td>
<td>Observer</td>
<td>vxWorks port</td>
<td>Money</td>
<td>10/Apr/13</td>
</tr>
</tbody>
</table>
EPICS Version 4

1. EPICS Version 4 and Relation to Version 3

2. Scientific Data Support

3. Examples and Present Activities

4. Working Group Organisation and Status
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, backlash?

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
EPICS V4 Principal Additions

New Functionality

CA => pvAccess : A Standardized protocol specification

Structured Data Exchange and PV Records

New interface to IOC (pvaccess server)

High Performance RPC Data Service Software Platform

Arguments

Send only deltas

Full Asynchronous Error and Message passing

Unsigned Int directly supported

Provided by in EPICS V4

pvAccess

pvData
EPICS V4 Principal Science Support Additions

New Functionality

Scientific Data Services
Standardized High Level Data Types
Data Acquisition Management Tools
Directory Service
Direct Matlab and Python support

Provided in EPICS V4 by

channelRPC
Normative Types
pvManager, Gather platform
ChannelFinder EPICS V4 service
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
```

<table>
<thead>
<tr>
<th>#timePastEpoch(s)</th>
<th>#value</th>
<th>#Date</th>
<th>#Alarm</th>
</tr>
</thead>
<tbody>
<tr>
<td>496169397.856321000</td>
<td>7.355487346649e-02</td>
<td>Wed Jun 21 17:49:57 2012</td>
<td>NO ALARM</td>
</tr>
<tr>
<td>496169401.996447000</td>
<td>1.682446300983e-01</td>
<td>Wed Jun 21 17:50:01 2012</td>
<td>NO ALARM</td>
</tr>
<tr>
<td>496169410.052636000</td>
<td>2.558367252350e-01</td>
<td>Wed Jun 21 17:50:10 2012</td>
<td>NO ALARM</td>
</tr>
<tr>
<td>496169420.109690000</td>
<td>3.173123300076e-01</td>
<td>Wed Jun 21 17:50:20 2012</td>
<td>NO ALARM</td>
</tr>
<tr>
<td>496169430.100015000</td>
<td>2.159405648708e-01</td>
<td>Wed Jun 21 17:50:30 2012</td>
<td>NO ALARM</td>
</tr>
<tr>
<td>496169440.081932000</td>
<td>4.953919649124e-01</td>
<td>Wed Jun 21 17:50:40 2012</td>
<td>NO ALARM</td>
</tr>
<tr>
<td>496169450.089935000</td>
<td>3.187555372715e-01</td>
<td>Wed Jun 21 17:50:50 2012</td>
<td>NO ALARM</td>
</tr>
<tr>
<td>496169450.699760000</td>
<td>0.00000000000000e+00</td>
<td>Wed Jun 21 17:50:50 2012</td>
<td>Disconnected</td>
</tr>
<tr>
<td>496169450.699760000</td>
<td>0.00000000000000e+00</td>
<td>Wed Jun 21 17:50:50 2012</td>
<td>Archive_Off</td>
</tr>
<tr>
<td>496169537.905713000</td>
<td>0.00000000000000e+00</td>
<td>Wed Jun 21 17:52:17 2012</td>
<td>Disconnected</td>
</tr>
</tbody>
</table>
**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.

```bash
$ 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

Note: Arguments
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

Gregory White, SLAC/PSI, for EPICS V4 team, 25-Apr-2013
EPICS Version 3 Architecture

Channel Access protocol on the wire
Scientific Data Services Layer

Diagram showing various components and services related to data services, including PVAS, IRMIS, Channel Archiver, and Front-Ends like CAS, Diag Database, Diag & PS. The diagram illustrates the flow and interactions between different databases and services, with references to technologies like SQL, XML/RPC, and databases such as RDB.
EPICS V3-V4 INTEROPERATION

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

V3 client ←→ V4 server

caget or caput

cav3 module of V4 (a CA server)
pvIOC Records

V4 IOC

pvAccess "get" / "put"

V4 client ←→ V3 server

pvAccess "provider" of pvAccess API
CAV3 "provider" of pvAccess API

OR

pvAccess
ca

v3Channel module of V4 links to cav3

V3 IOCs

V3 IOC records

Regular V3 CAS (cav3)

V3 IOC records


Gregory White, SLAC/PSI, for EPICS V4 team, 25-Apr-2013
CSS talks to pvaSrv over pvAccess, pvaSrv talks to the V3 db for both puts and reads

Thanks to G. Carcassi, M. Sekornaja D. Hickin
Beginnings of an distributed AreaDetector Data Processing Pipeline

Work of David Hickin, Diamond
In Matlab use EPICS V4 directly, no wrapper like lca or mex
EPICS V4 Matlab interface "EasyPVA"

First do the setup, just once:

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

Example 1: Put a single value to a PV

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

Example 2: Get a single value from a PV

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

value =

1.9997
```

Example 3: Put an array of values to a PV

```matlab
>> 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

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

value =

1.0000
2.1000
3.3000
4.5000
5.6600
6.7000
```
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

- Display / Matlab / Application
  - All data for all BPMs returned in 1 pvAccess datagram
  - V3 CA managed acq by original author of CA

- EV4 Gather Service

- BPM data
  - V3 IOC Data
  - V3 IOC Data
  - V3 IOC Data
Beam Dynamics Services = EPICS V4 "RPC" service + Gather Service + Directory Service

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

- EV4 RPC Service (e.g., BPM Service)
- EV4 Gather Service
- EV4 Directory Service
- Display / Matlab / Application

Acts as raw BPM acquisition

Adds pulse/bunch sync, difference orbit subtraction, orbit fitting, etc.

eg pvget SWISSFEL:BPMORBIT

V3 IOC Data
BEAM DYNAMICS SERVICES = EPICS V4 "RPC" SERVICE 
+ GATHER SERVICE + DIRECTORY SERVICE

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

SwissFEL Orbit Plot

eg pvget SWISSFEL:BPMORBIT

Gregory White, SLAC/PSI, for EPICS V4 team, 25-Apr-2013
% 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 Performance (2)

1000 channels, double array element count on x axis

<table>
<thead>
<tr>
<th></th>
<th>pvAccess</th>
<th>CA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>106000</td>
<td>333000</td>
</tr>
<tr>
<td>10</td>
<td>111000</td>
<td>228000</td>
</tr>
<tr>
<td>100</td>
<td>86600</td>
<td>41570</td>
</tr>
<tr>
<td>1000</td>
<td>11800</td>
<td>4510</td>
</tr>
<tr>
<td>10000</td>
<td>1183</td>
<td>443</td>
</tr>
<tr>
<td>100000</td>
<td>118</td>
<td>44</td>
</tr>
</tbody>
</table>
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
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
3. A normative document of the EPICS V4 IOC processing pipeline
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
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
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

100% 100% 100% 30% 80% 95%
EPICS V4 Charter + Deliverables, Status 2.

- 100%
- 70%
- 100%
- 20%
- 90%

1. 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.

2. A report of interoperability of the EPICS V4 IOC with EPICS v3 record processing.

3. 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 ICIE, ASN.1, EXI Web Service.


6. 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.


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):**

\[
g^{(2)}(Q,\tau) = \frac{\langle \langle I_p(Q,t)I_p(Q,t+\tau) \rangle_\phi \rangle_t}{\langle \langle I_p(Q,t) \rangle_\phi \rangle_{0 \leq t \leq T} \langle \langle I_p(Q,t) \rangle_\phi \rangle_{t \leq t \leq T}}
\]

Thanks to DARON CHABOT, BNL

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….
2013 EPICS V4 Charter

2013 Charter concentrates 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. Windows Port

2. **Improved support for experimental data acquisition in the IOC**
   + version 4 IOC processing pipeline, particularly for areaDetector processing
   + 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.
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 (though we are now starting work on caching proxy server)
5. High Performance Web Server on the IOC (e.g. IBM XML screamer + W3C EXI)
6. Services
   - Snapshot save and Restore (Done by BNL)
   - BPM Orbit
   - Model (Transverse optics being done by PSI/SLAC/BNL, others would be cool)
   - Linac Energy estimation (for correcting Quad focusing w.r.t. Energy)
   - Archive service
7. 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

A V4 IOC is a V3 IOC that can talk pvAccess as well as CA

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


