EPICS in Nustar

Bastian Löher

R³B Collaboration Meeting June 2015

NDAQ TDR requirements

- Common slow-control system for all detectors
  - Cost-effective
  - Synergetic effects (security, interoperability,...)
- Open source package with large user base
  - EPICS already in use at GSI (Panda, CBM, R3B,…)
- Scalability
- No reliance on specialized hardware
- Restricted access to slow-control variables
- Builtin support for engineering units (SI units)

NDAQ TDR requirements II

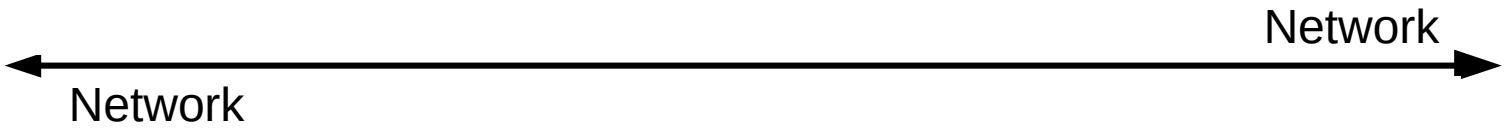
- Validity/Range checks on changed values
- Alert system for operator notification
- Fast Reconfiguration (as in case of NDAQ)
- Robust setup without single point of failure
- Access to slow-control via terminal OR GUI
- Save and restore of complete parameter set
- Mapping between physical and logical names
- Allow several clients per server
- Portability

Overview

- What is EPICS?
  - A short introduction
- How can we satisfy the requirements?
  - A few examples
- What's in store for 2017?
  - EPICS at Nustar

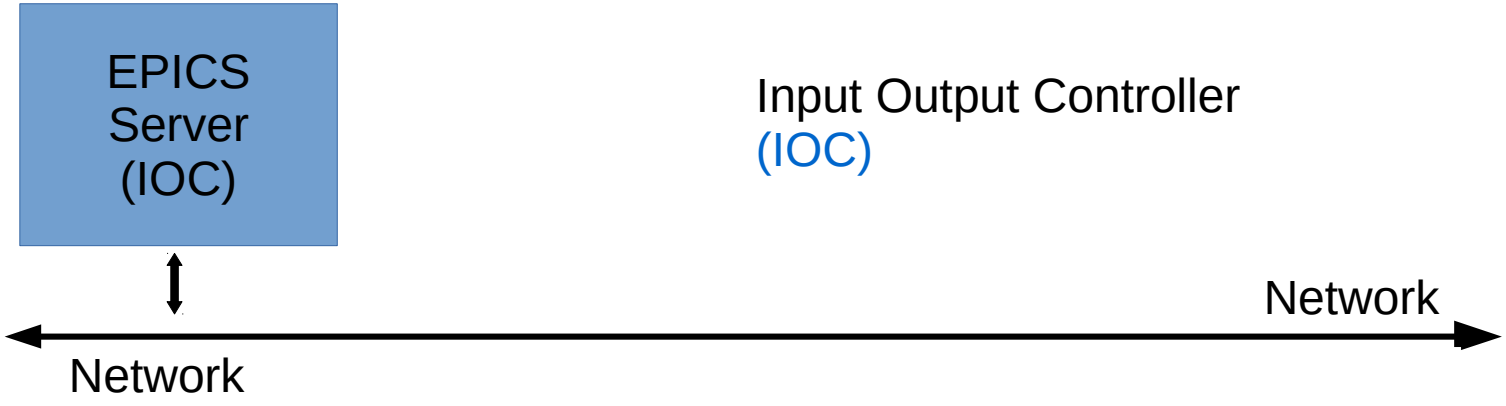
EPICS Introduction

- *EPICS is a set of Open source Software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as particle accelerators, telescopes and other large scientific experiments*



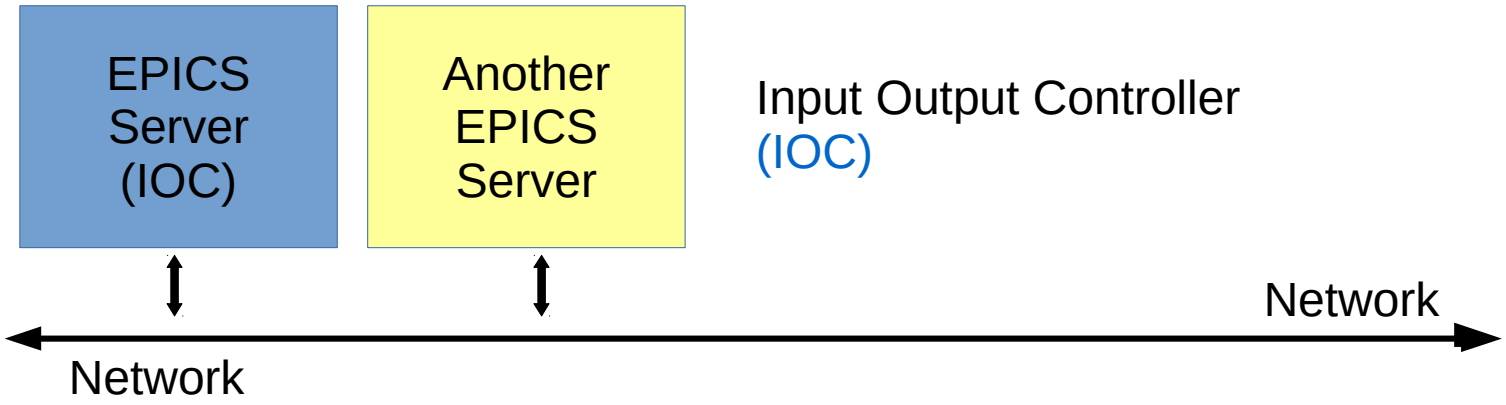
EPICS Introduction

- *EPICS is a set of Open source Software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as particle accelerators, telescopes and other large scientific experiments*



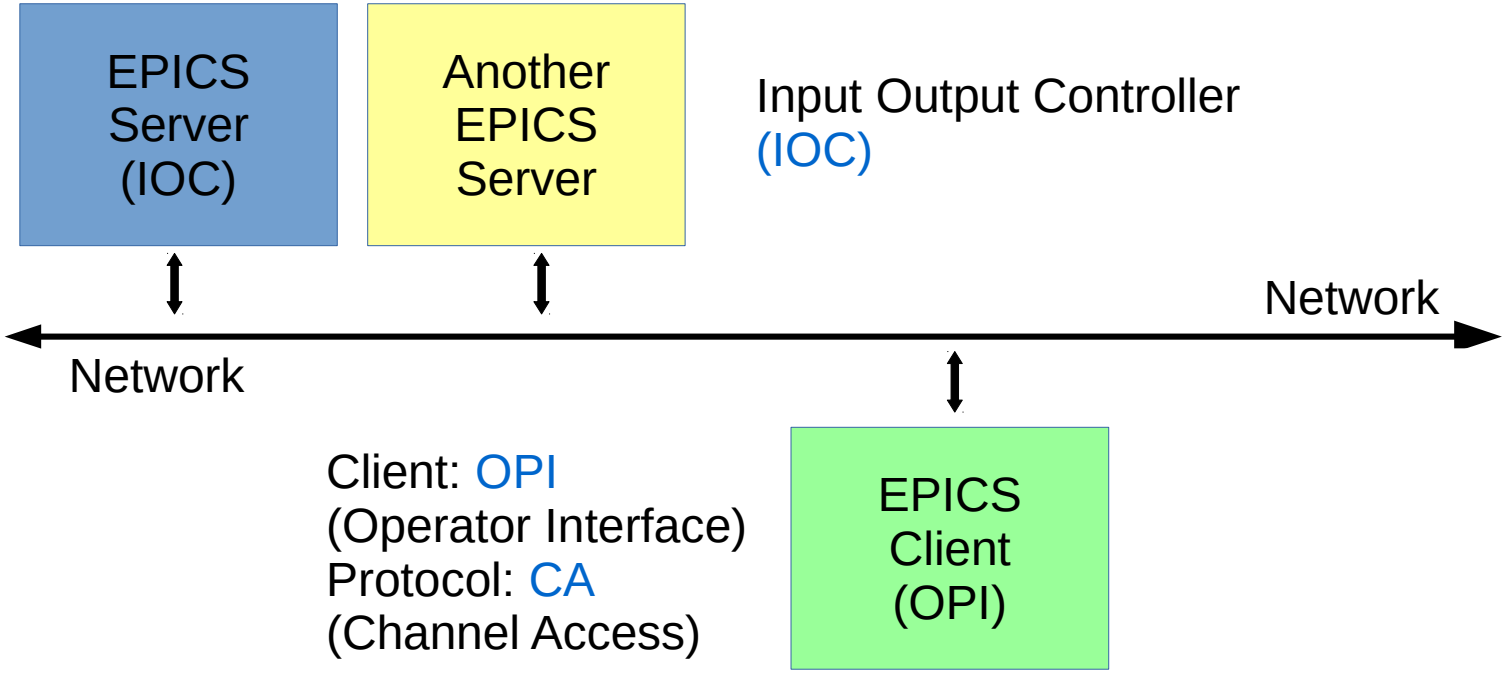
EPICS Introduction

- *EPICS is a set of Open source Software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as particle accelerators, telescopes and other large scientific experiments*



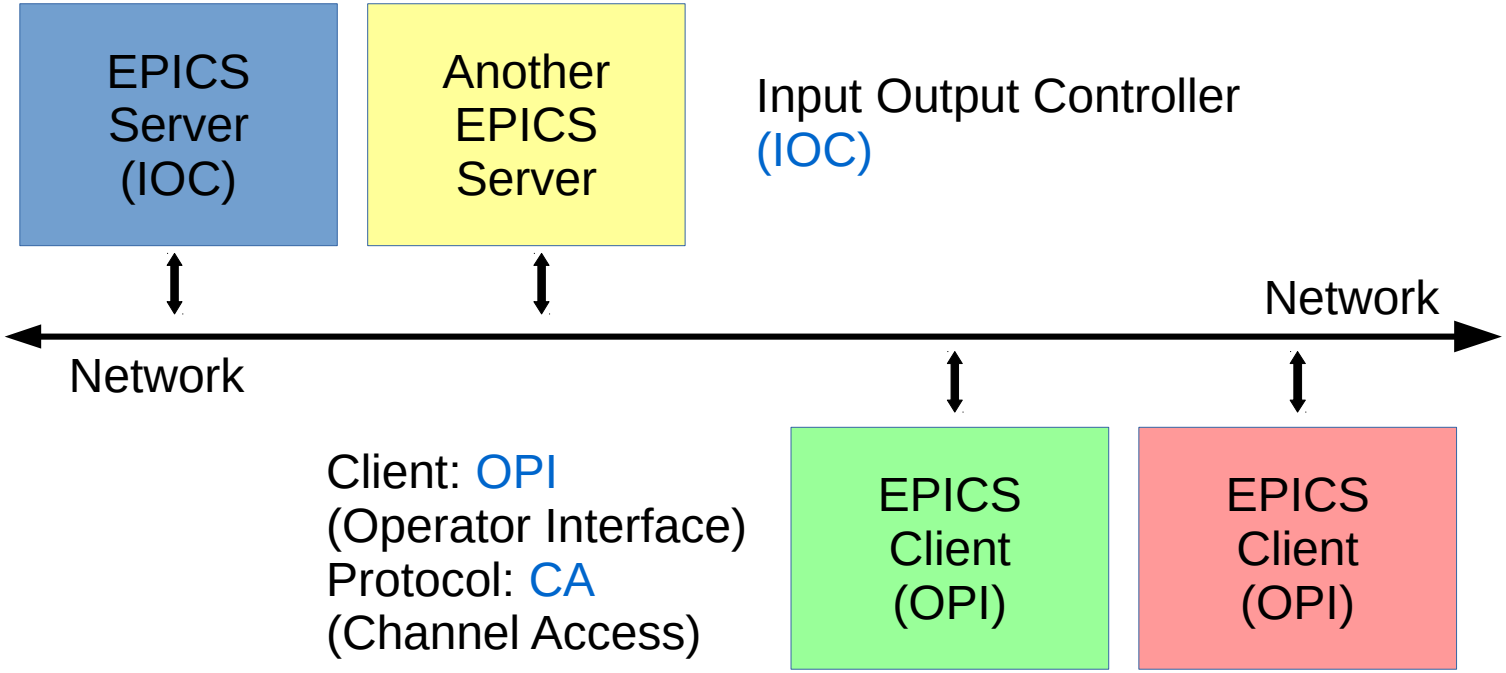
EPICS Introduction

- *EPICS is a set of Open source Software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as particle accelerators, telescopes and other large scientific experiments*



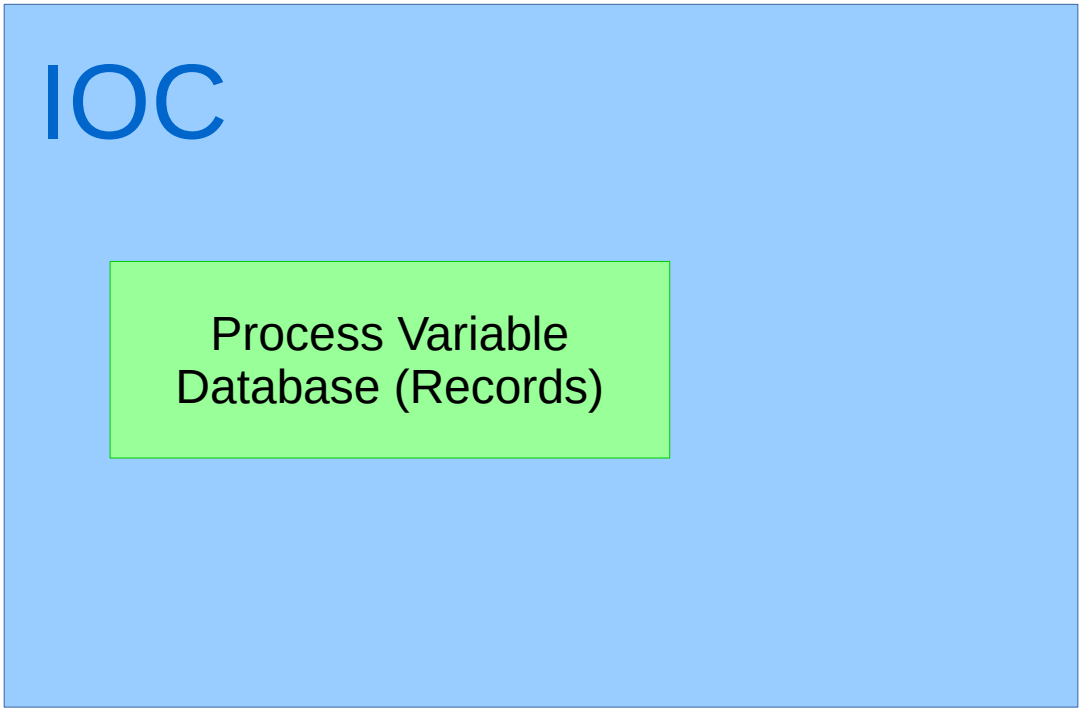
EPICS Introduction

- *EPICS is a set of Open source Software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as particle accelerators, telescopes and other large scientific experiments*



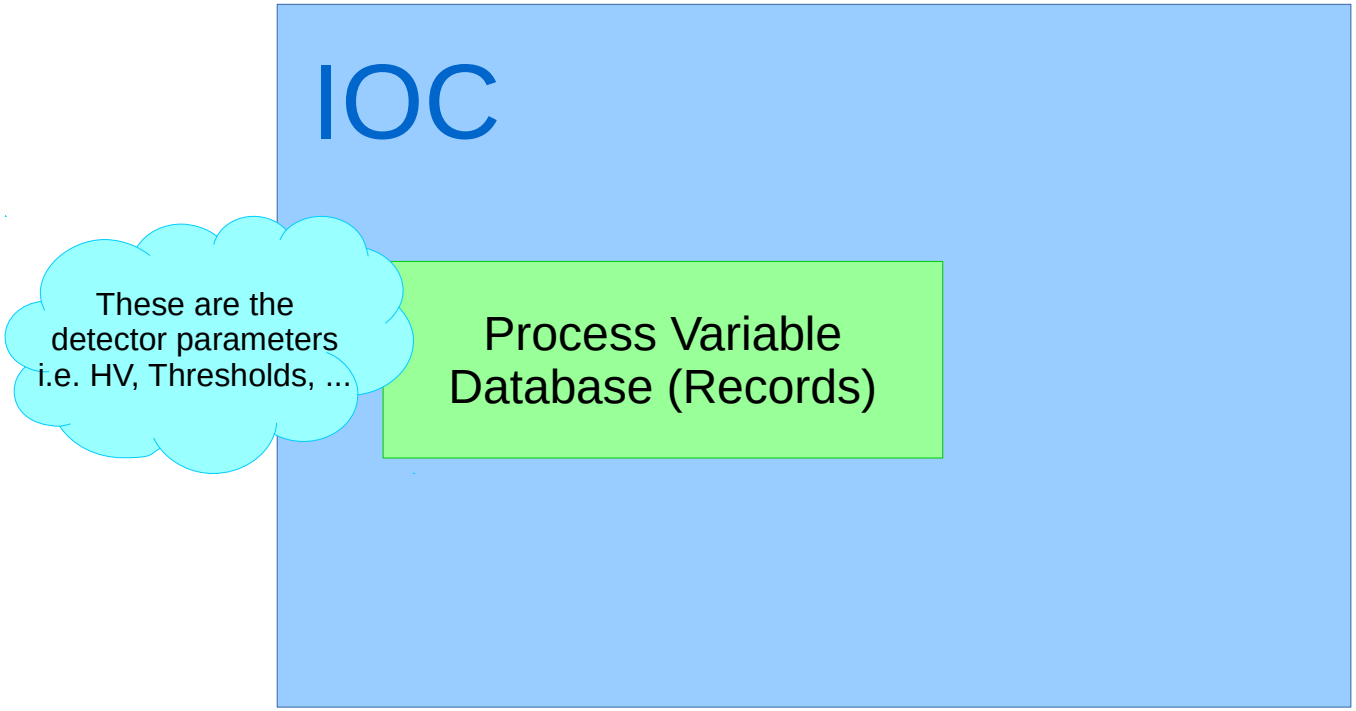
EPICS Introduction

- The EPICS Server (IOC) simplified



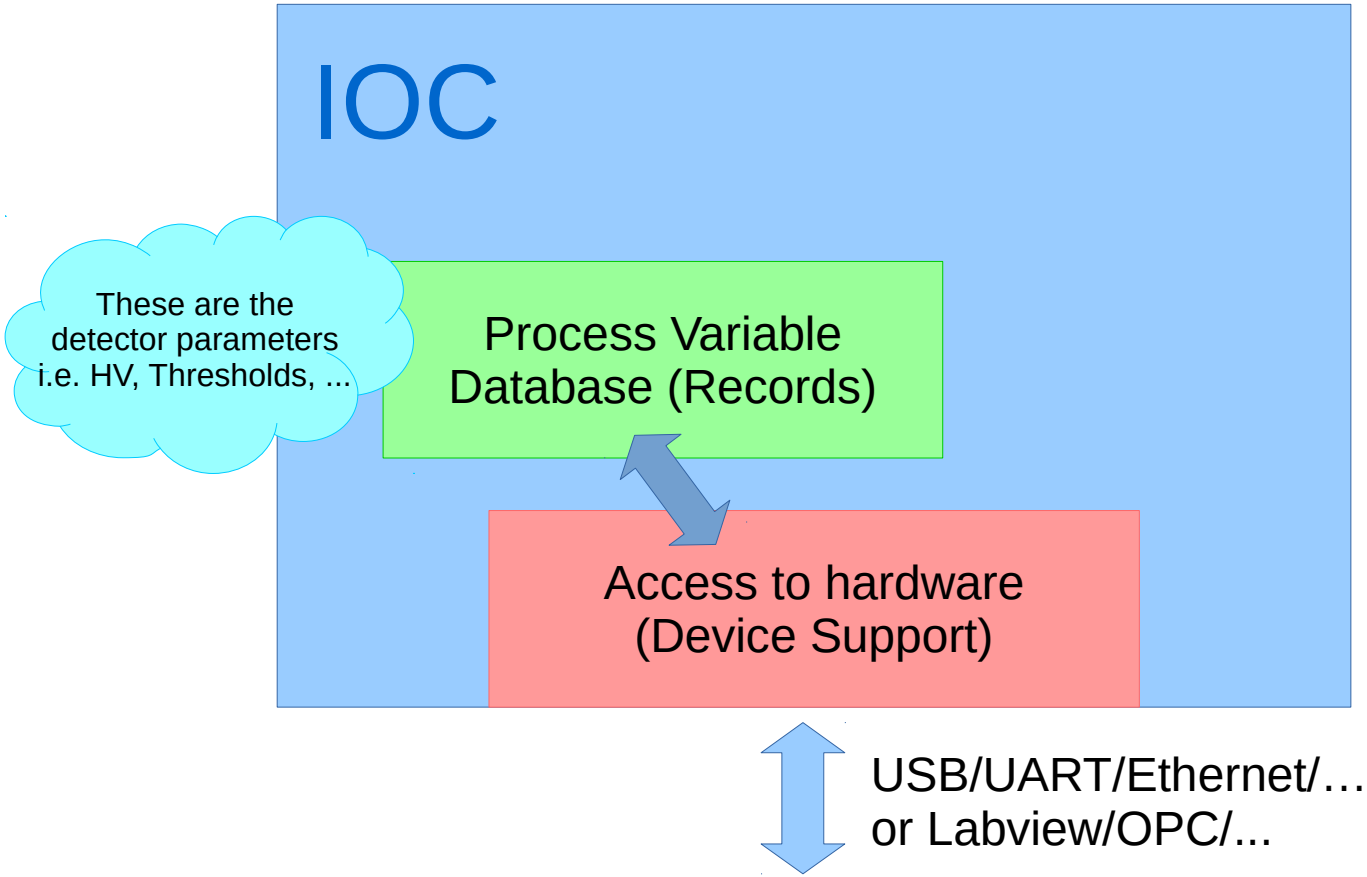
EPICS Introduction

- The EPICS Server (IOC) simplified



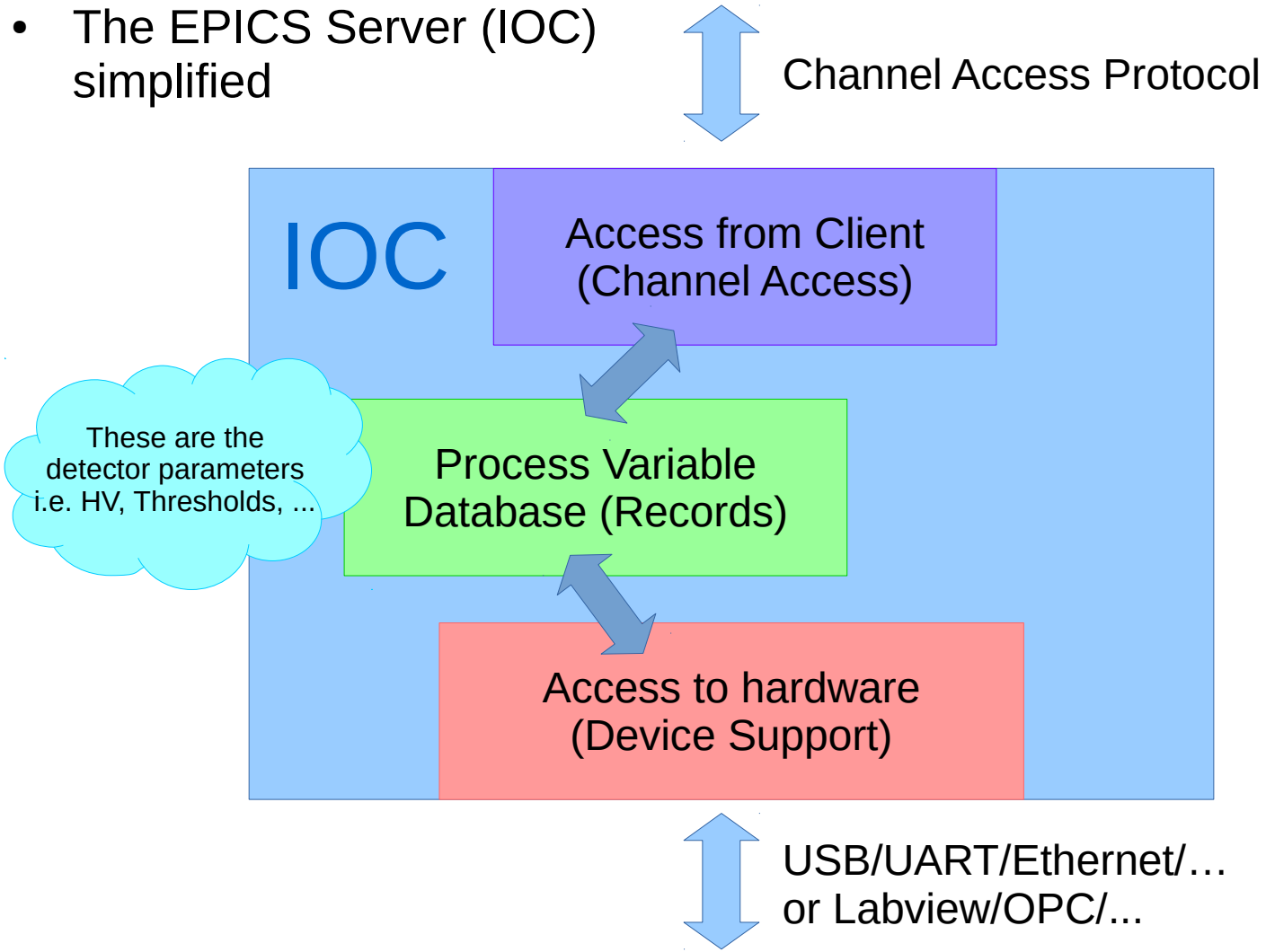
EPICS Introduction

- The EPICS Server (IOC) simplified



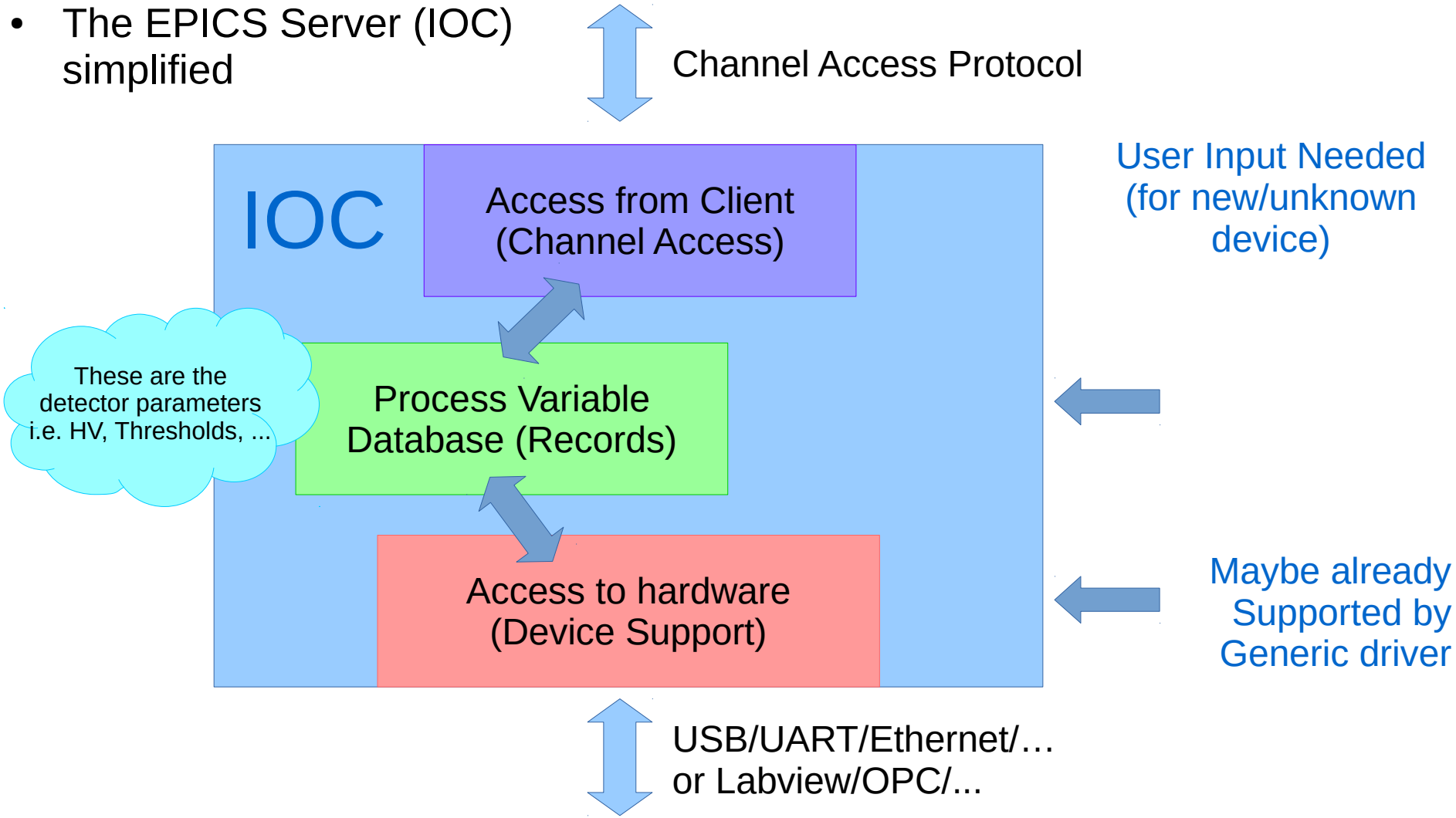
EPICS Introduction

- The EPICS Server (IOC) simplified



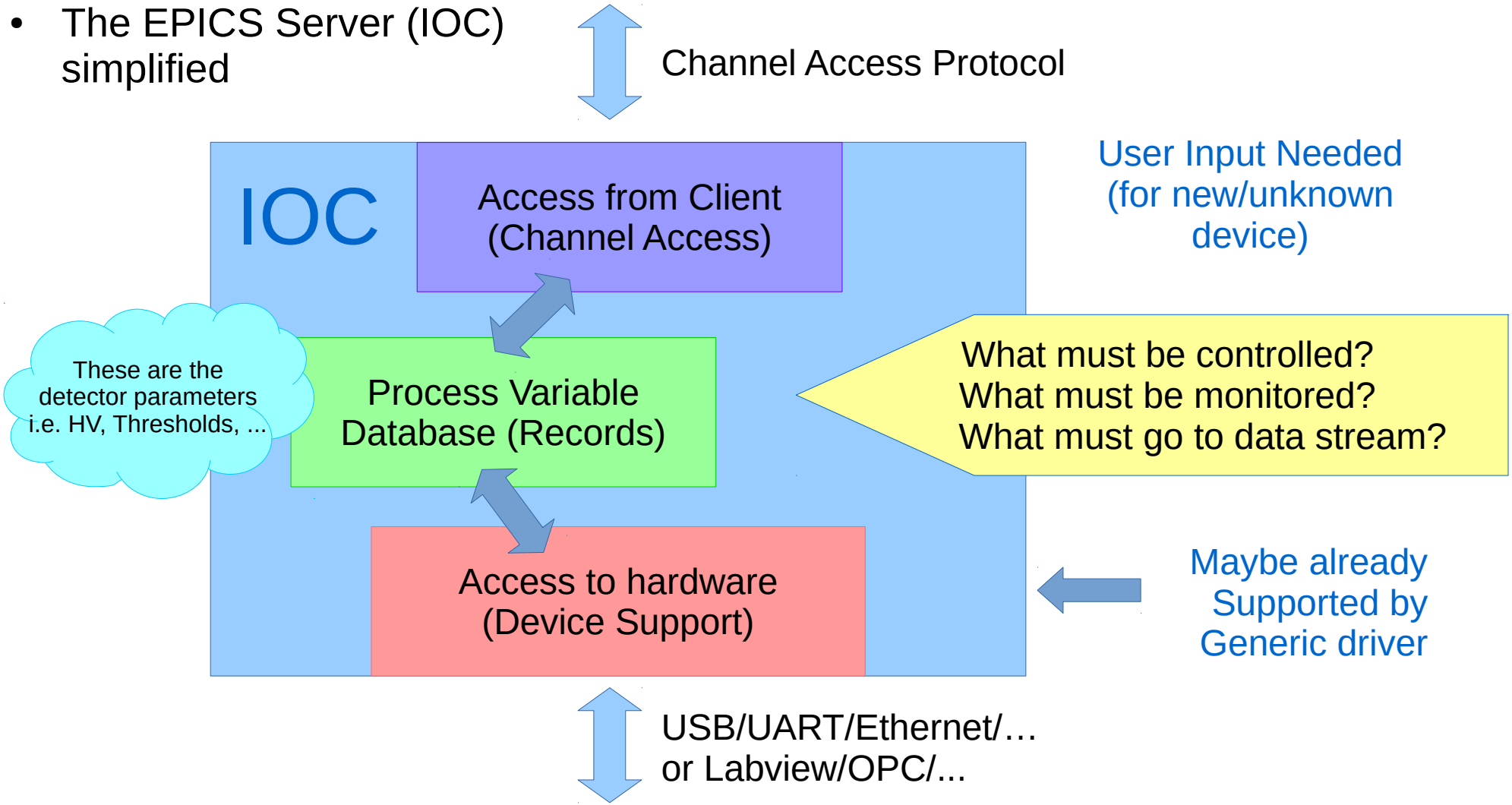
EPICS Introduction

- The EPICS Server (IOC) simplified



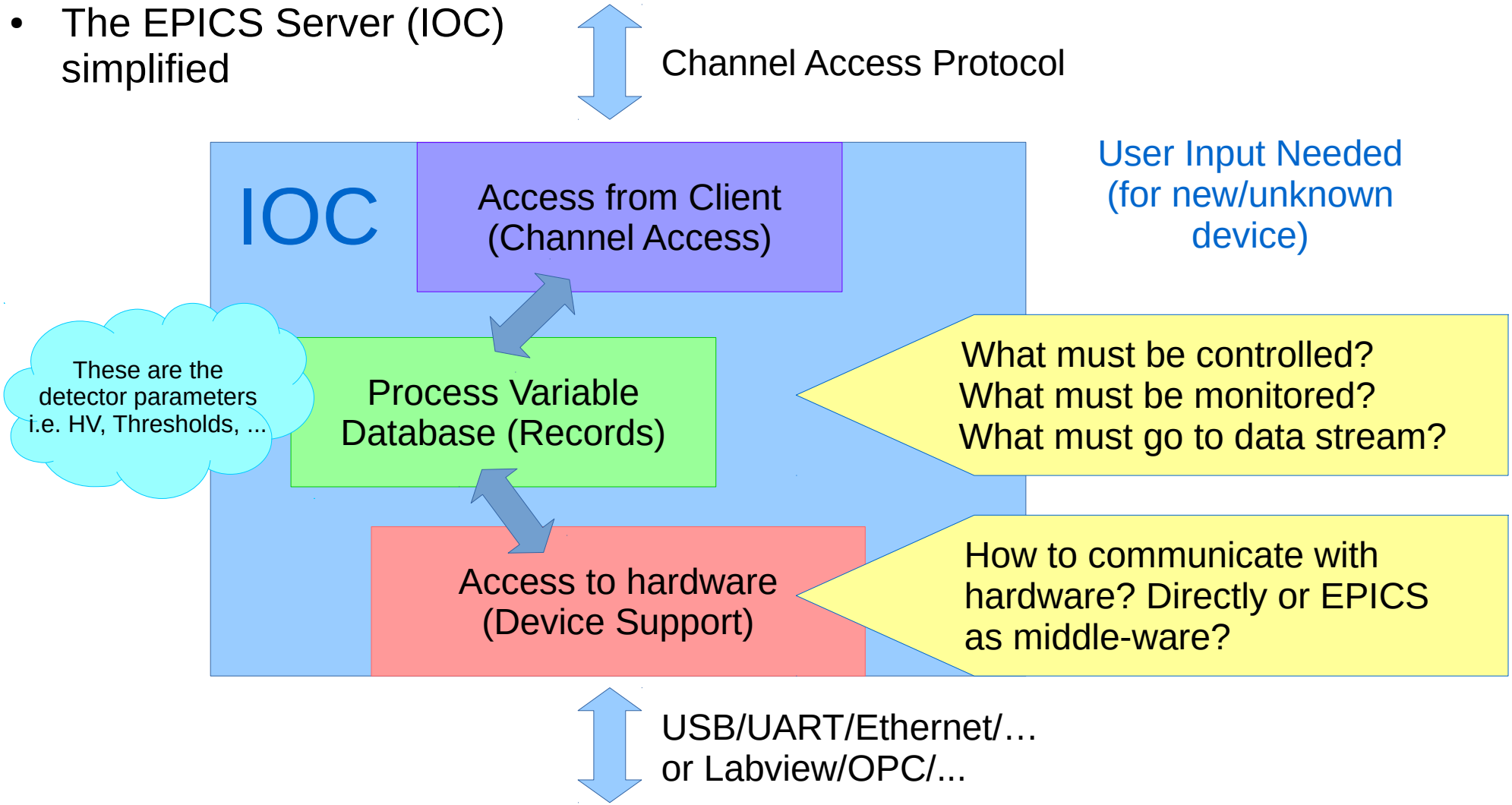
EPICS Introduction

- The EPICS Server (IOC) simplified



EPICS Introduction

- The EPICS Server (IOC) simplified



EPICS Introduction

- The EPICS Server (IOC) simplified

Channel Access Protocol

Don't start from scratch!
Many types of hardware are already supported by EPICS (> 50 protocols):

List of hardware support:
www.aps.anl.gov/epics/modules/index.php

Examples:
- CAEN Powersupply (Ethernet)
- Siemens S7 PLC (Ethernet)
- National Instruments Cards (PCI)
- Generic RS232, CANbus

User Input Needed (for new/unknown device)

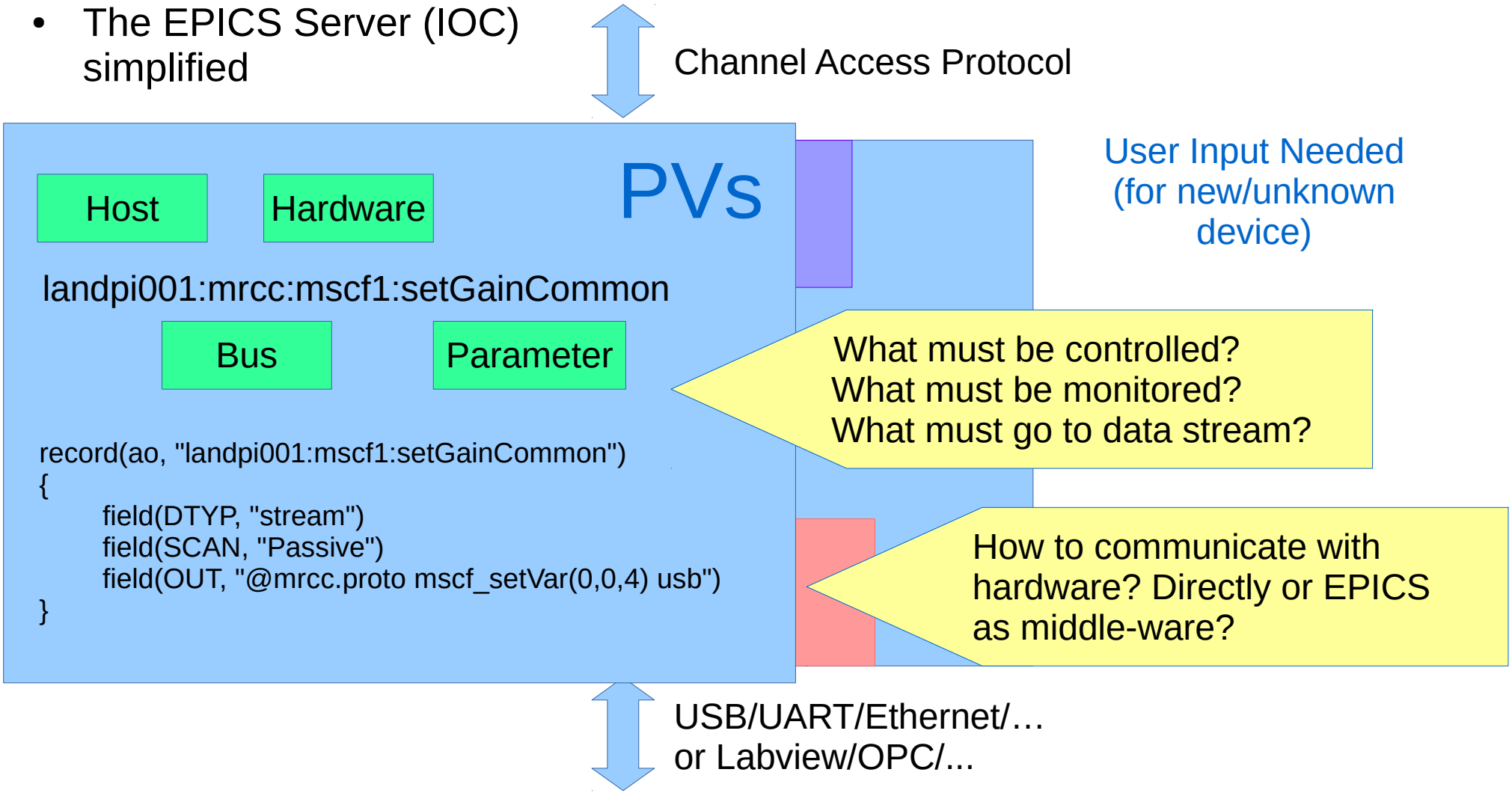
What must be controlled?
What must be monitored?
What must go to data stream?

How to communicate with hardware? Directly or EPICS as middle-ware?

USB/UART/Ethernet/…
or Labview/OPC/...

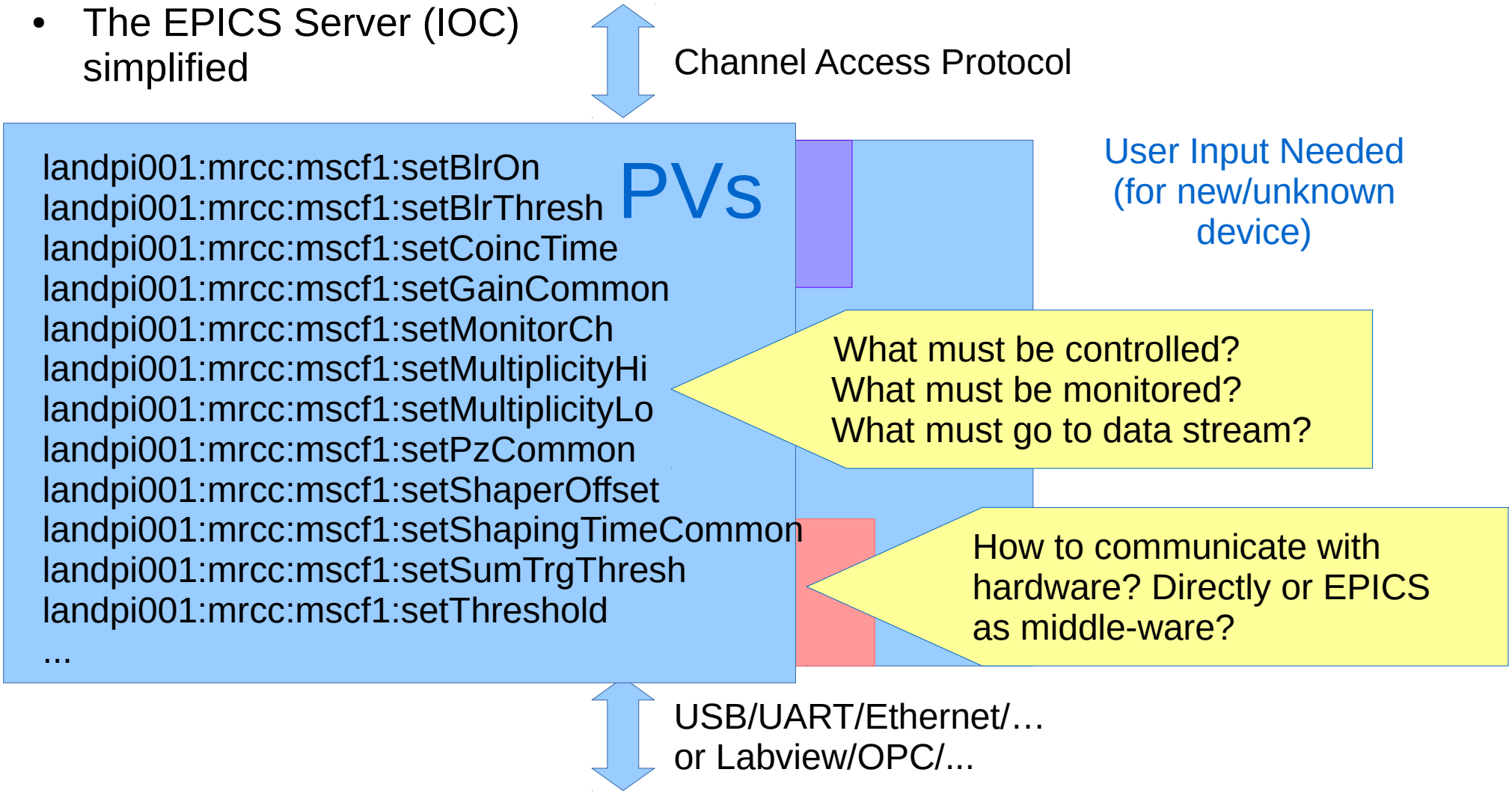
EPICS Introduction

- The EPICS Server (IOC) simplified



EPICS Introduction

- The EPICS Server (IOC) simplified



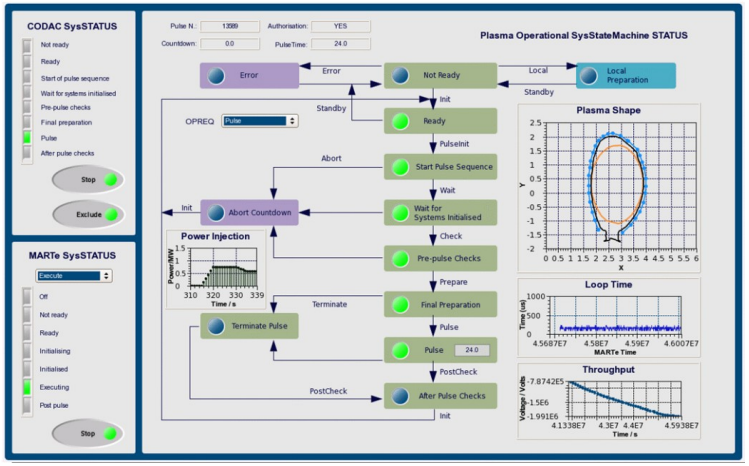
EPICS Clients (CA Clients)

- Simple clients (command line), for debugging
- Advanced clients: Backup/Restore, Gateway,…
- Graphical Clients:
  - Some old ones
  - CSS/BOY (Best OPI yet)

BOY – Best OPI Yet

- Drag and Drop creation of GUI using widgets
  - Time effective
- Scriptable, where necessary
  - Avoid repetetive tasks
- Composable
  - Easy integration of submodules within other GUI parts
  - Re-Use finished modules (complete or in parts)
- Simple export to the web (web-opi)

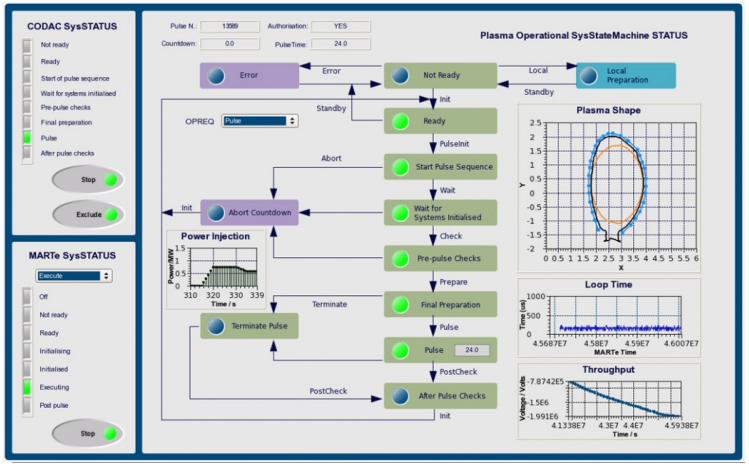
https://github.com/ControlSystemStudio/cs-studio/wiki/BOY



BOY

Strongly recommended!

- Drag and Drop creation of GUI using widgets
  - Time effective
- Scriptable, where necessary
  - Avoid repetetive tasks
- Composable
  - Easy integration of submodules within other GUI parts
  - Re-Use finished modules (complete or in parts)
- Simple export to the web (web-opi)



https://github.com/ControlSystemStudio/cs-studio/wiki/BOY

NDAQ TDR requirements

- Common slow-control system for all detectors
- Open source package with large user base
- Scalability
- No reliance on specialized hardware
- Restricted access to slow-control variables
- Builtin support for engineering units (SI units)
- Validity/Range checks on changed values
- Alert system for operator notification
- Fast Reconfiguration (as in case of NDAQ)
- Robust setup without single point of failure
- Access to slow-control via terminal OR GUI
- Save and restore of complete parameter set
- Mapping between physical and logical names
- Allow several clients per server
- Portability

NDAQ TDR requirements

- Common slow-control system for all detectors
- Open source package with large user base
- Scalability
- No reliance on specialized hardware
- Restricted access to slow-control variables
- Builtin support for engineering units (SI units)
- Validity/Range checks on changed values
- Alert system for operator notification
- Fast Reconfiguration (as in case of NDAQ)
- Robust setup without single point of failure
- Access to slow-control via terminal OR GUI
- Save and restore of complete parameter set
- Mapping between physical and logical names
- Allow several clients per server
- Portability

Already solved by EPICS base

NDAQ TDR requirements

- Common slow-control system for all detectors
- Open source package with large user base
- Scalability
- No reliance on specialized hardware
- Restricted access to slow-control variables
- Builtin support for engineering units (SI units)
- Validity/Range checks on changed values
- Alert system for operator notification
- Fast Reconfiguration (as in case of NDAQ)
- Robust setup without single point of failure
- Access to slow-control via terminal OR GUI
- Save and restore of complete parameter set
- Mapping between physical and logical names
- Allow several clients per server
- Portability

Already solved by EPICS base

Scalability example @
Advanced Photon Source:
~30000 PVs on ~300 IOCs

Spallation Neutron Source:
~300k PVs

EPICS is used at almost all
Major accelerator facilities

NDAQ TDR requirements

- Common slow-control system for all detectors
- Open source package with large user base
- Scalability
- No reliance on specialized hardware
- Restricted access to slow-control variables
- Builtin support for engineering units (SI units)
- Validity/Range checks on changed values
- Alert system for operator notification
- Fast Reconfiguration (as in case of NDAQ)
- Robust setup without single point of failure
- Access to slow-control via terminal OR GUI
- Save and restore of complete parameter set
- Mapping between physical and logical names
- Allow several clients per server
- Portability

Already solved by EPICS base

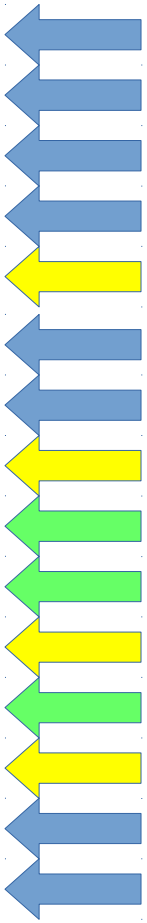
Portability:
GNU/Linux on x86 or x86_64 or arm
MacOS X on PowerPC or x86
Windows on x86
Solaris on sparc or x86
vxWorks 5.4.x, 5.5.x
vxWorks 6.x on powerPC
RTEMS

Small systems: i.e. Raspberry Pi
Powerful systems: Rack Server

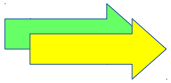
You name it → EPICS can run!

NDAQ TDR requirements

- Common slow-control system for all detectors
- Open source package with large user base
- Scalability
- No reliance on specialized hardware
- Restricted access to slow-control variables
- Builtin support for engineering units (SI units)
- Validity/Range checks on changed values
- Alert system for operator notification
- Fast Reconfiguration (as in case of NDAQ)
- Robust setup without single point of failure
- Access to slow-control via terminal OR GUI
- Save and restore of complete parameter set
- Mapping between physical and logical names
- Allow several clients per server
- Portability



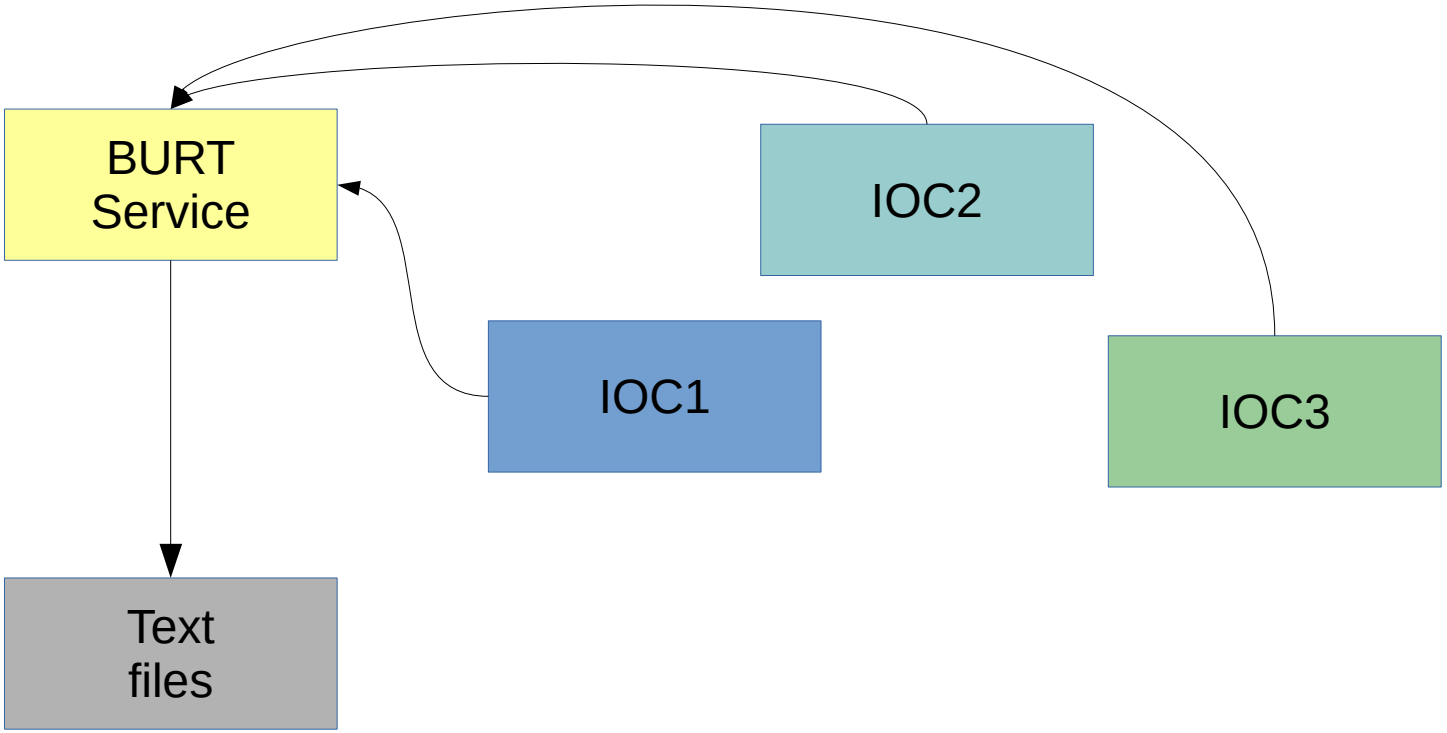
Already solved by EPICS base



Implemented by NDAQ using Available EPICS extensions

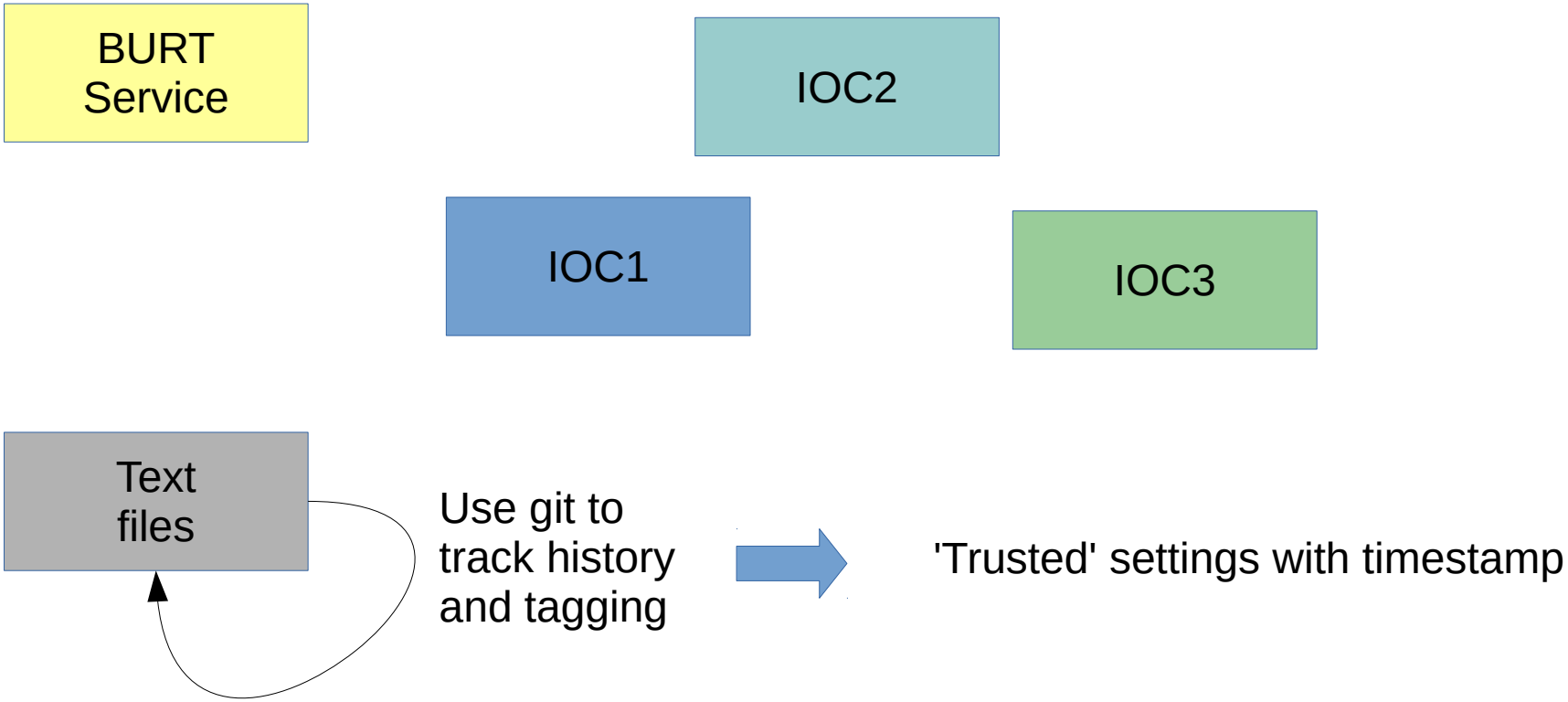
Backup & Restore

- One option: BURT – Save current state



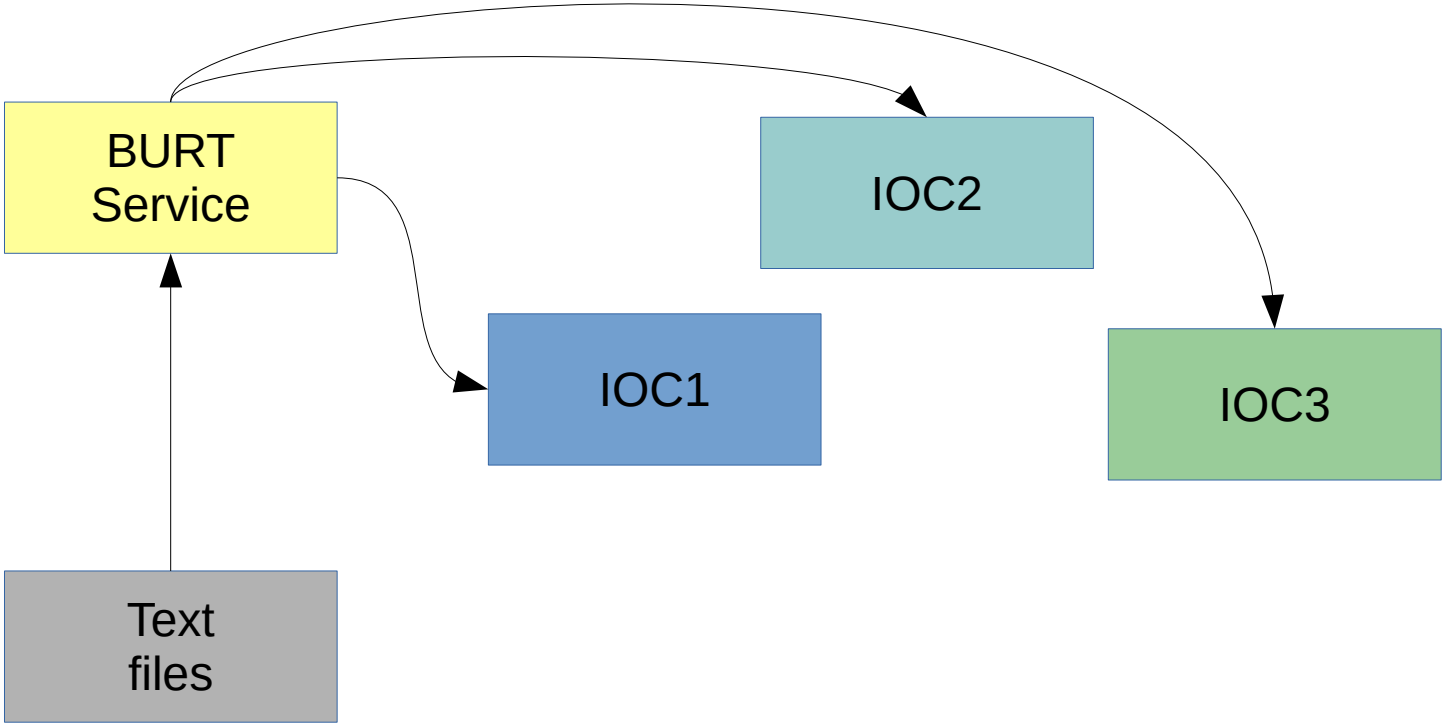
Backup & Restore

- One option: BURT – Save current state



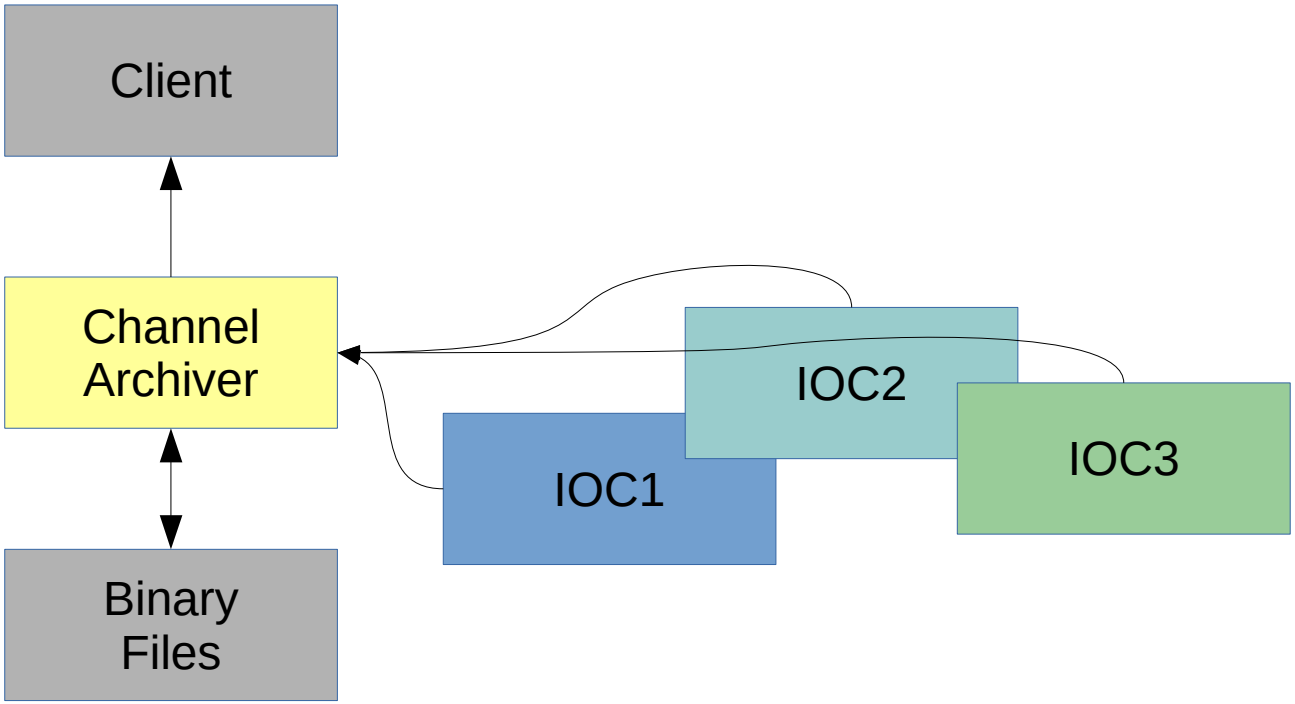
Backup & Restore

- One option: BURT – Restore previous state



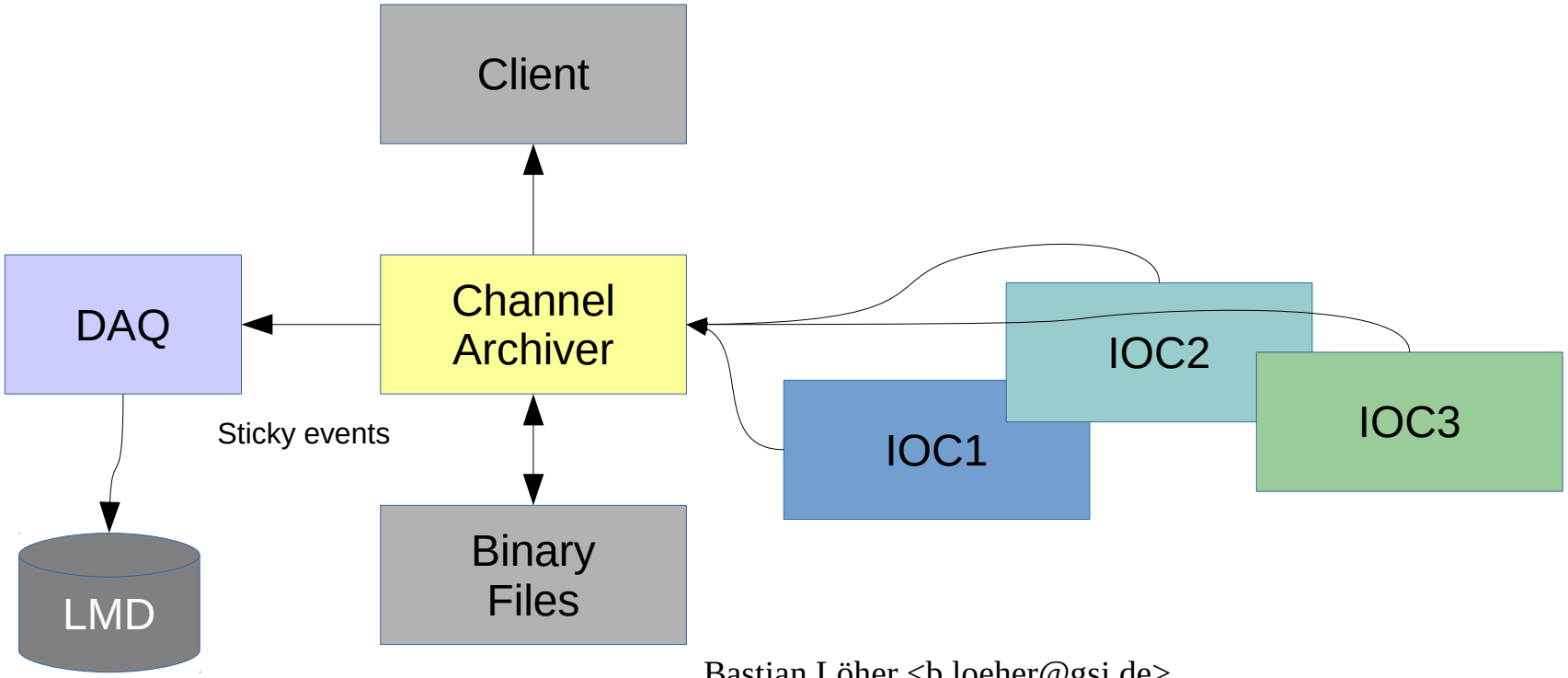
Archiving

- One option: Channel Archiver
  - Performance optimized binary storage (> 7x RDB)
    - SNS: 60k PV/s vs. 8K PV/s



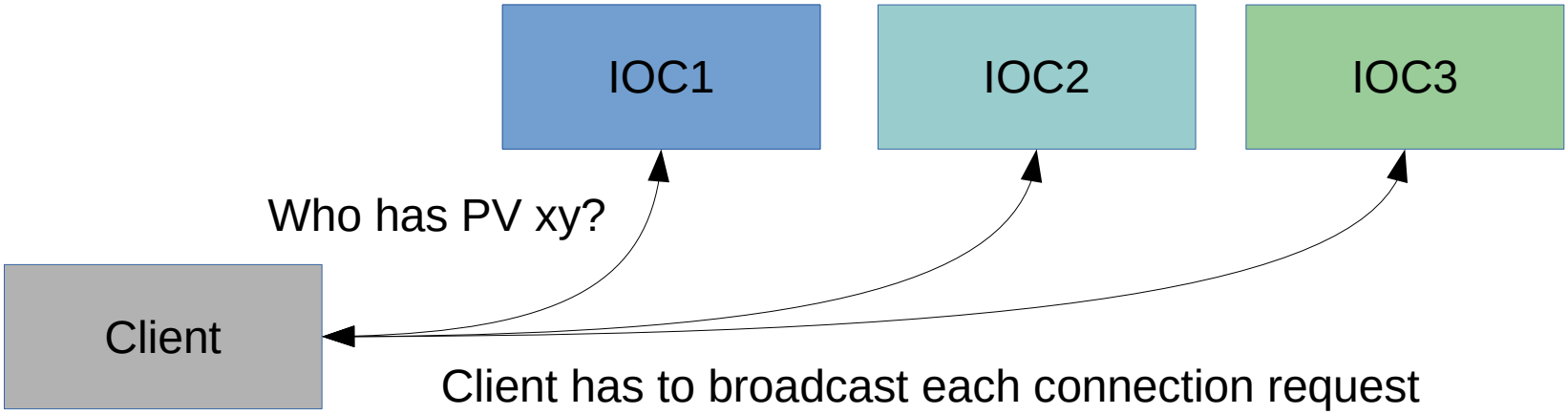
Archiving (DAQ)

- One option: Channel Archiver
  - Performance optimized binary storage (> 7x RDB)
    - SNS: 60k PV/s vs. 8K PV/s



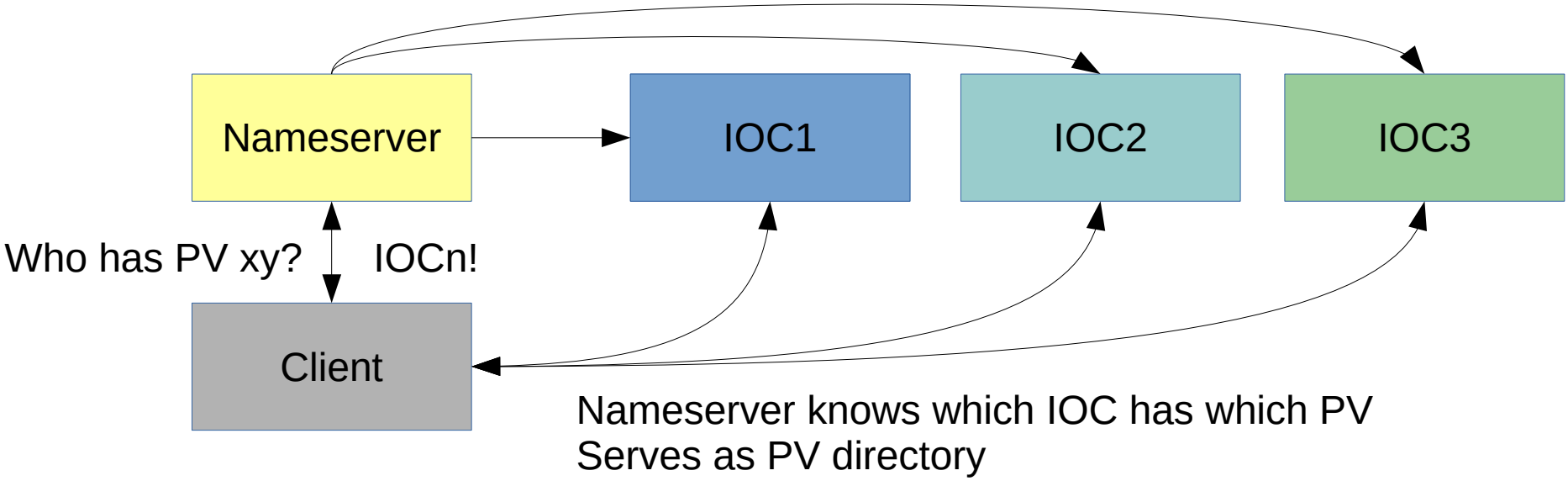
Nameserver

- Nameserver used to reduce broadcast traffic



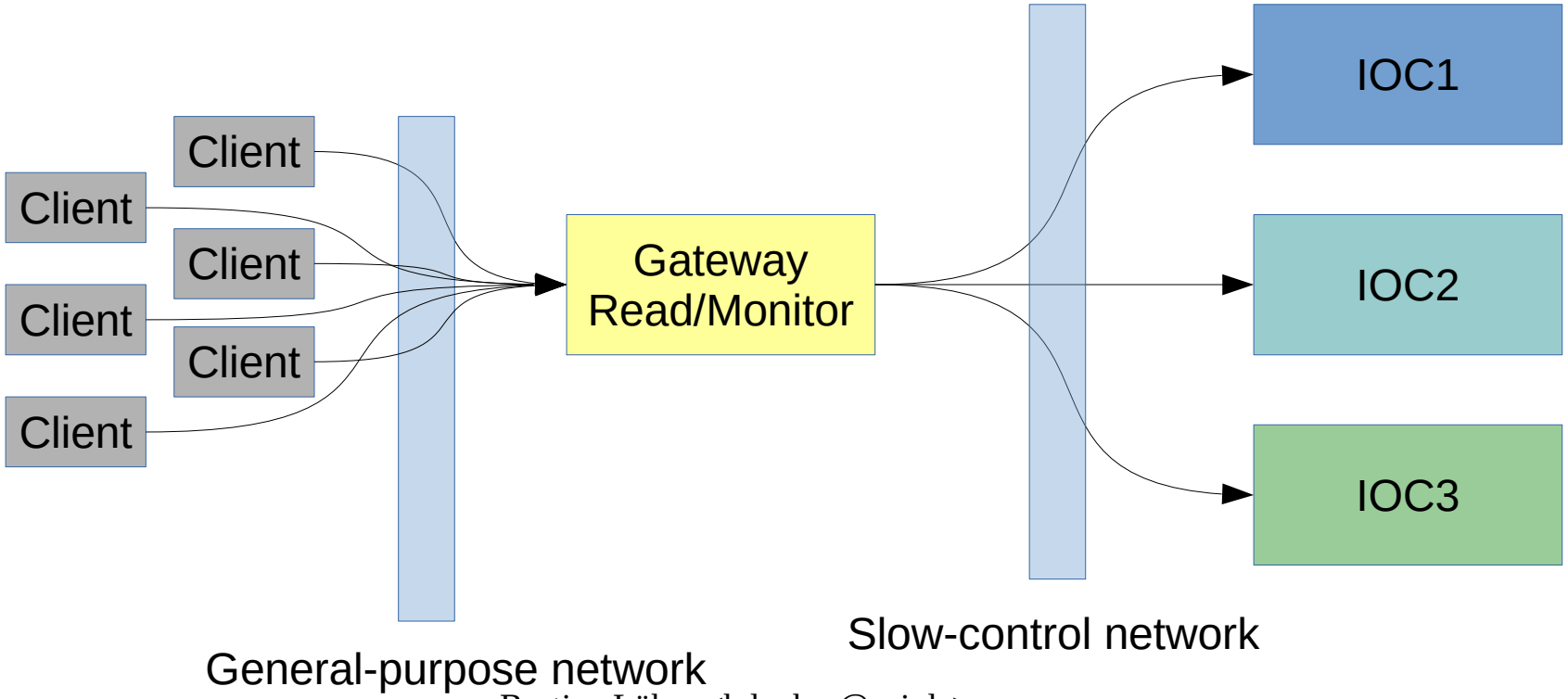
Nameserver

- Nameserver used to reduce broadcast traffic



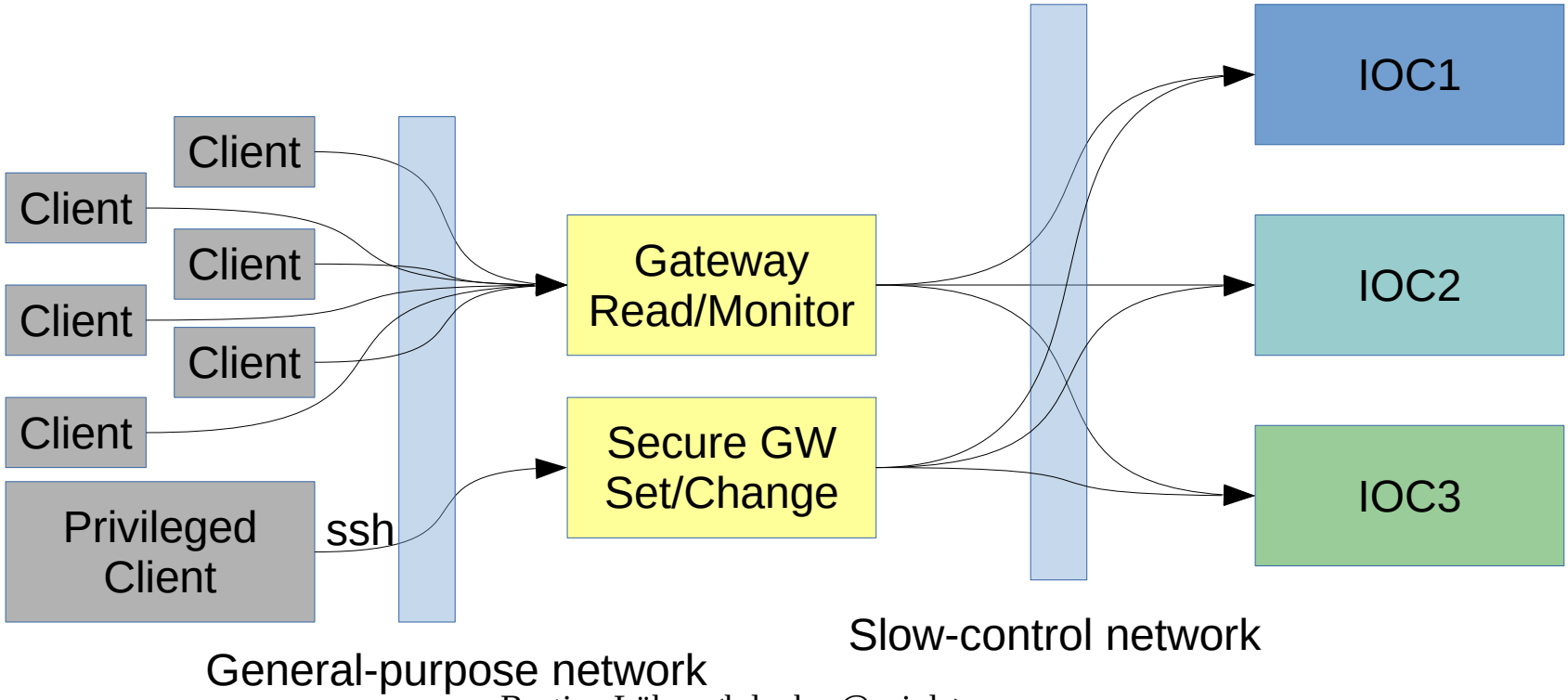
Gateway

- Provides layer between IOCs and Clients
  - Reduce load on IOCs (one connection per GW)



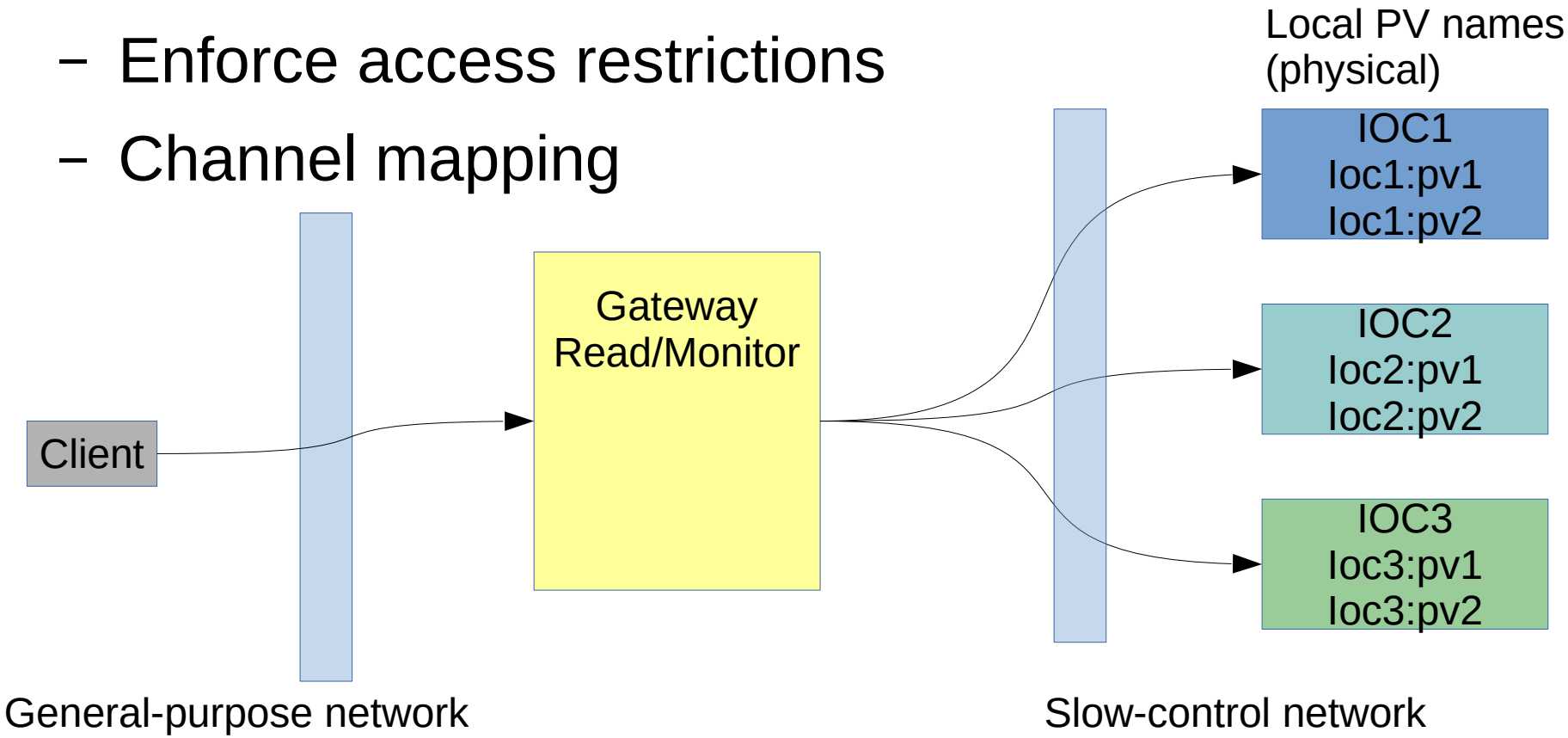
Gateway - Access

- Provides layer between IOCs and Clients
  - Reduce load on IOCs (one connection per GW)
  - Enforce access restrictions



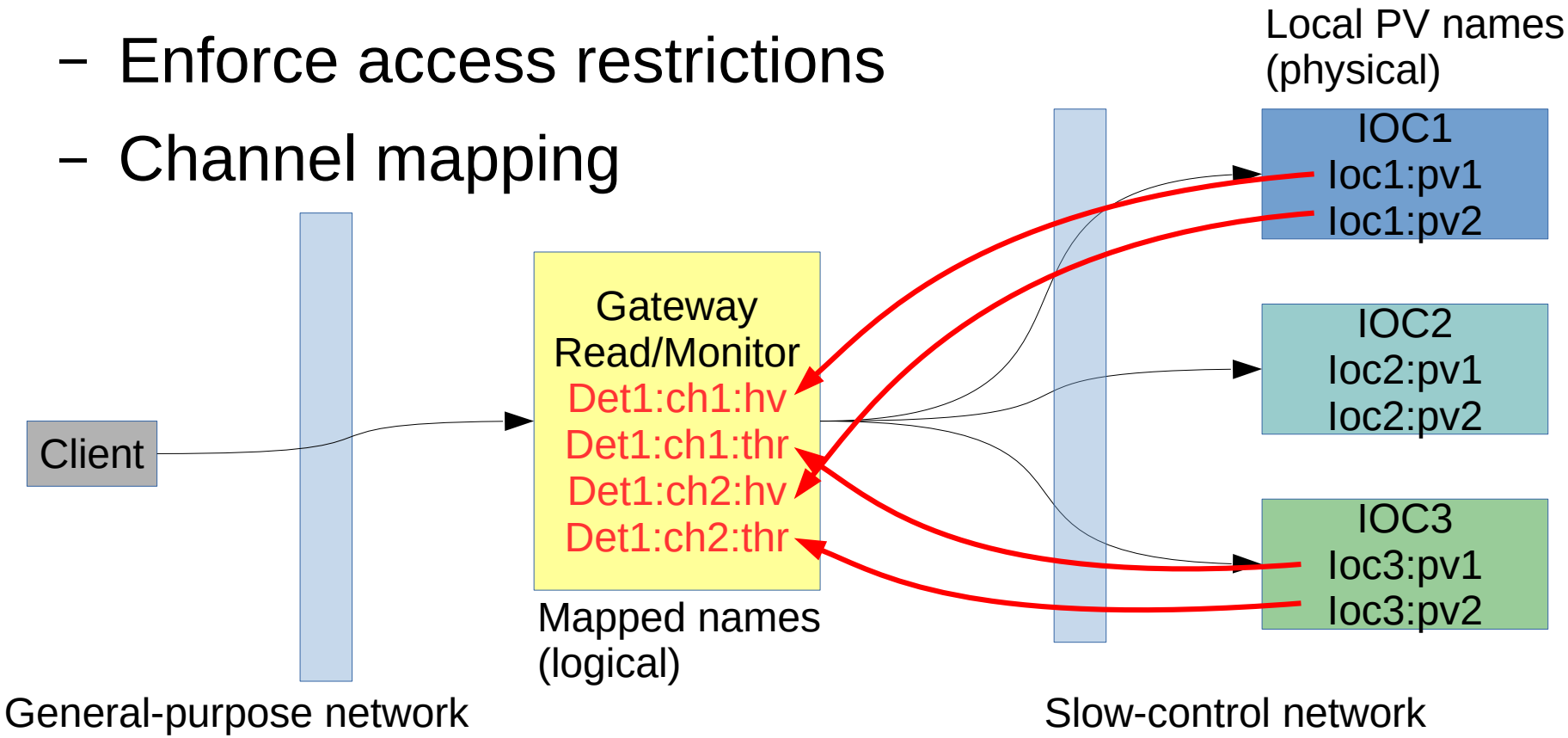
Gateway - Mapping

- Provides layer between IOCs and Clients
  - Reduce load on IOCs (one connection per GW)
  - Enforce access restrictions
  - Channel mapping



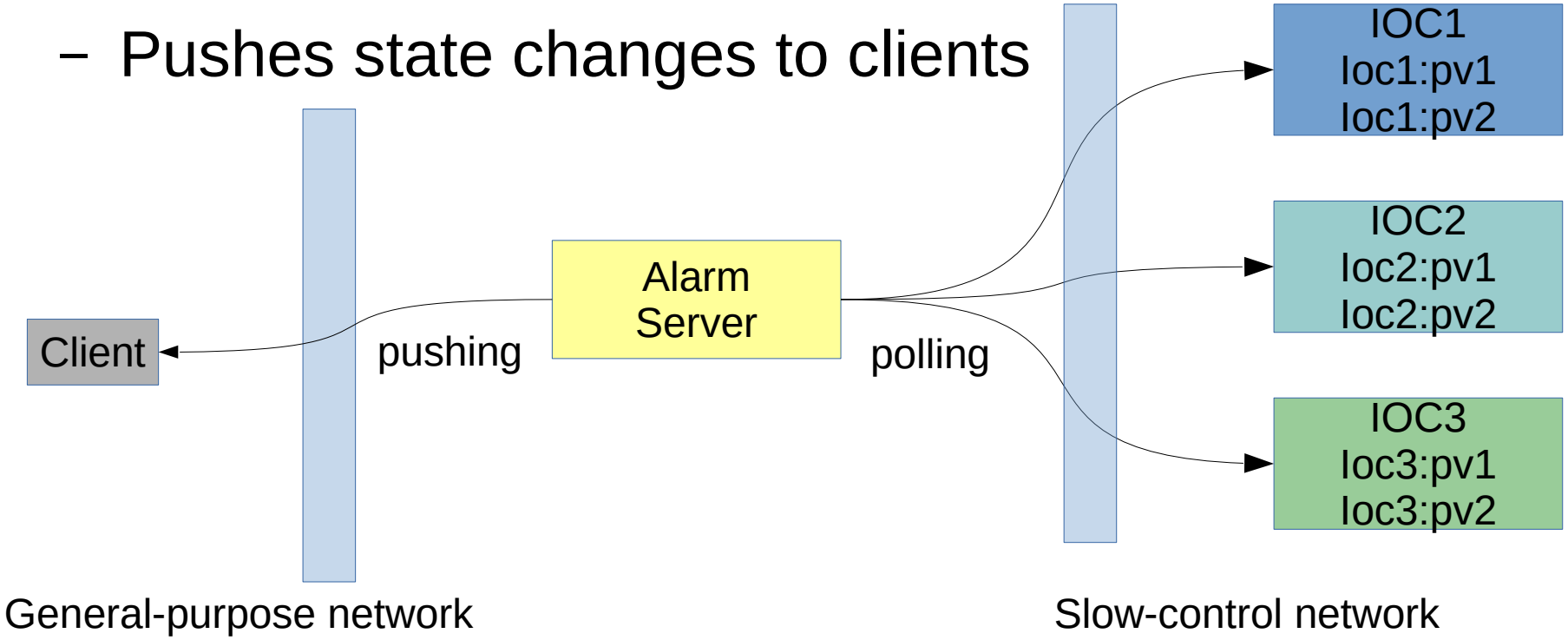
Gateway - Mapping

- Provides layer between IOCs and Clients
  - Reduce load on IOCs (one connection per GW)
  - Enforce access restrictions
  - Channel mapping



Alarms

- Make operators aware of system faults
  - Central alarm server watches PV states
  - Minor or Major alarm, depending on PV limits
  - Pushes state changes to clients



NDAQ TDR requirements

- Common slow-control system for all detectors
- Open source package with large user base
- Scalability
- No reliance on specialized hardware
- Restricted access to slow-control variables
- Builtin support for engineering units (SI units)
- Validity/Range checks on changed values
- Alert system for operator notification
- Fast Reconfiguration (as in case of NDAQ)
- Robust setup without single point of failure
- Access to slow-control via terminal OR GUI
- Save and restore of complete parameter set
- Mapping between physical and logical names
- Allow several clients per server
- Portability

Already solved by EPICS base

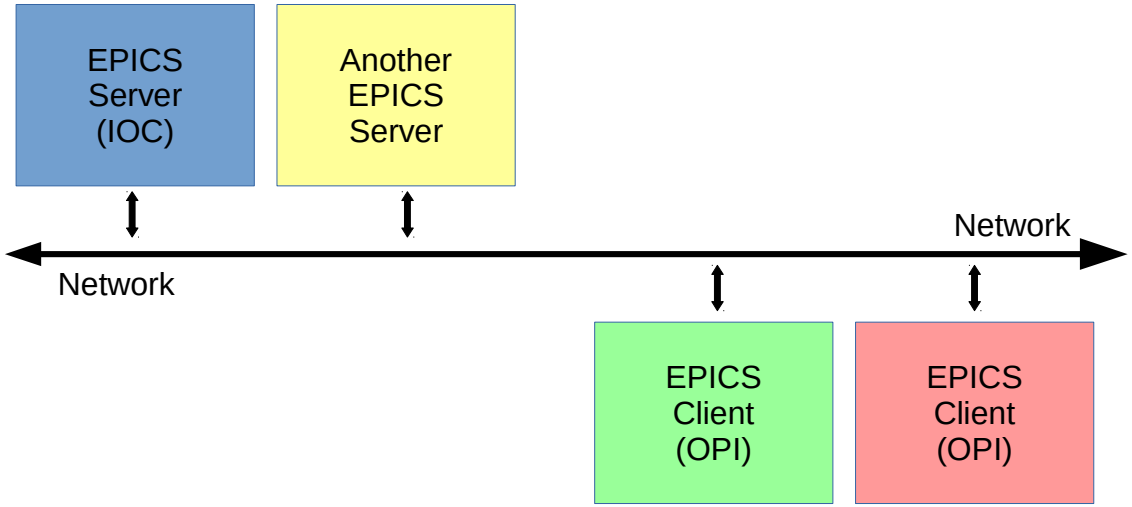
Implemented by NDAQ using
Available EPICS extensions

Need input from detector groups:

- Who can access which PVs?
- Which alarms are needed?
- Detector specific GUI needed?
- PV Mapping ↔ Cabling docu

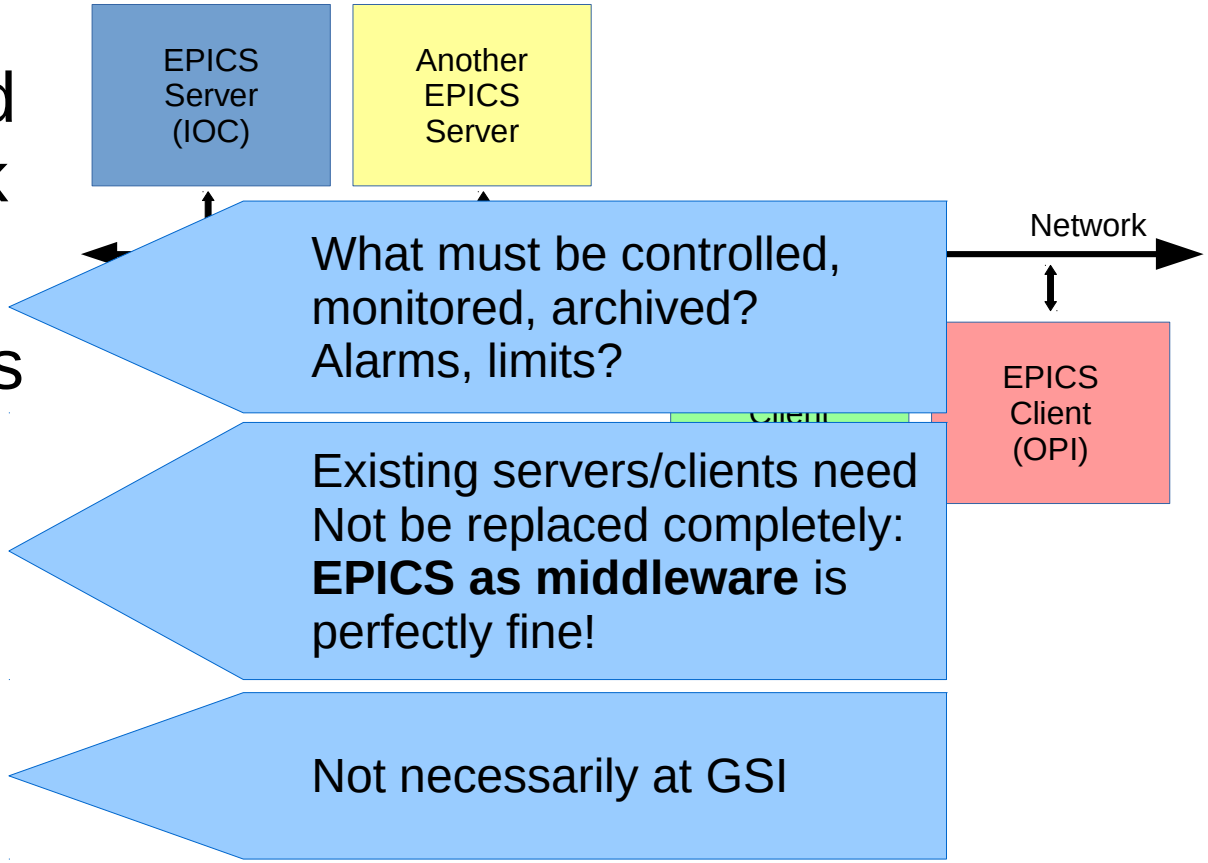
EPICS at Nustar

- Stage 1:
  - Hardware is tested on normal network
  - PVs are defined according to needs
  - IOCs and clients are built with integration in mind
  - Can be done anywhere



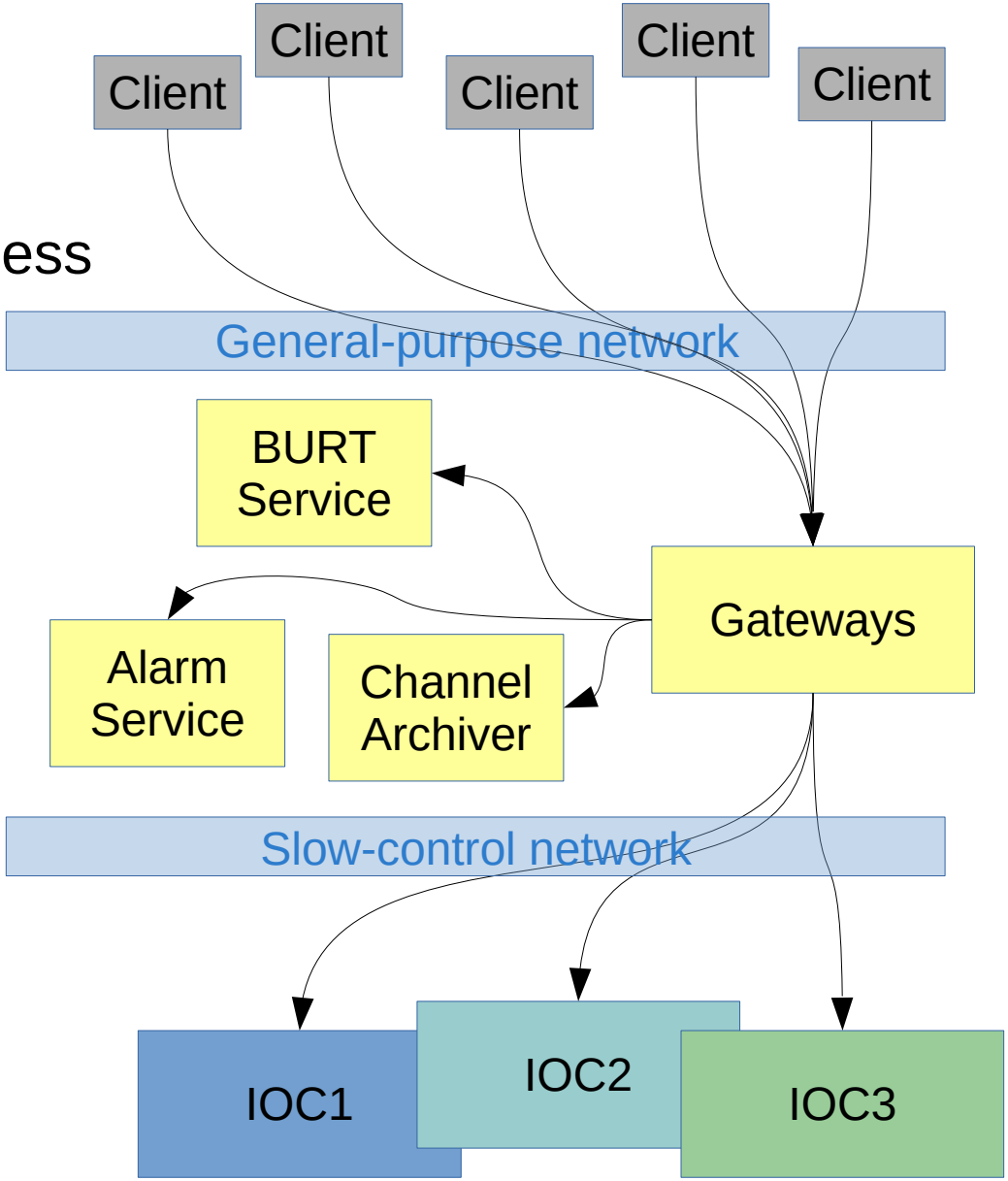
EPICS at Nustar

- Stage 1:
  - Hardware is tested on normal network
  - PVs are defined according to needs
  - IOCs and clients are built with integration in mind
  - Can be done anywhere



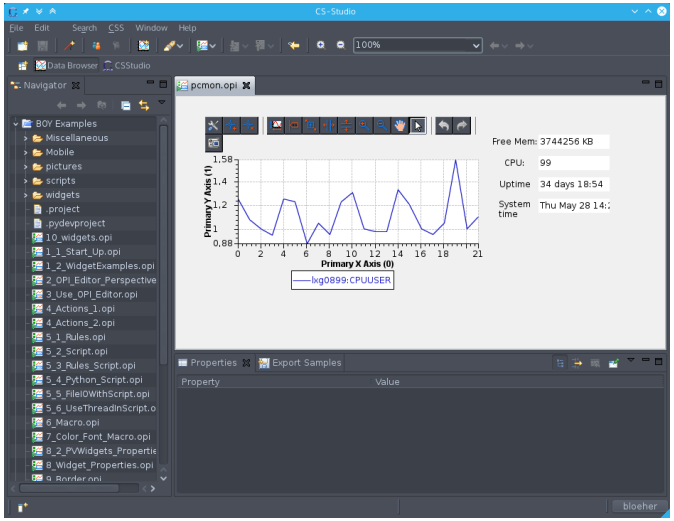
EPICS at Nustar

- Stage 2:
  - Dedicated slow-control network (VPN), no direct access to IOCs from GSI network
  - Gateways for mapping, security and name resolution
  - Central services for alarms, archiving, sticky events, ...
  - Integration of clients into common framework (which framework?)



EPICS at Nustar

- Control System Studio (CSS): A solution? http://cs-studio.sourceforge.net/
  - Development and runtime, built on Java, Eclipse as a base
  - Provides in a central location access to:
    - BEAST: Alarm system kit (alarm server)
    - BOY: Graphical Operator Interface
    - Data browser: View live and recorded data
    - Logbook support: Integration with on-site Elog
  - Integrates (using standard EPICS services):
    - Control / Monitoring
    - Authentification / security
    - Alarms / Events
    - Archiving / Logging
  - All parts are optional (FAIR-wide policy needed?)
    - Discussion with GSI EPICS team ongoing



EPICS at Nustar

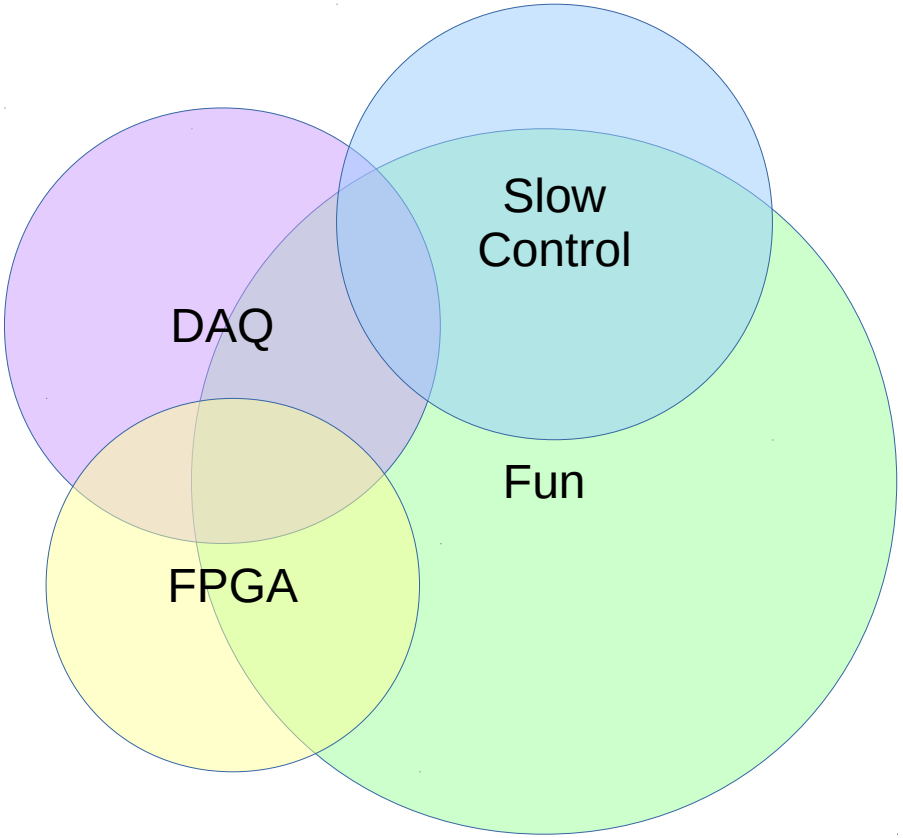
- Summary
  - NDAQ requires a common system for slow-control
  - EPICS fulfills all requirements, most automatically
  - Setup of central services for Nustar is WIP
    Many options exist, no final decision yet
  - Detector groups should become familiar with:
    - EPICS, CSS, BOY

    setup clients, and provide necessary information
  - When to start: Now!

    Getting started: http://www.aps.anl.gov/epics/docs/GSWE.php

Slow Control is fun!

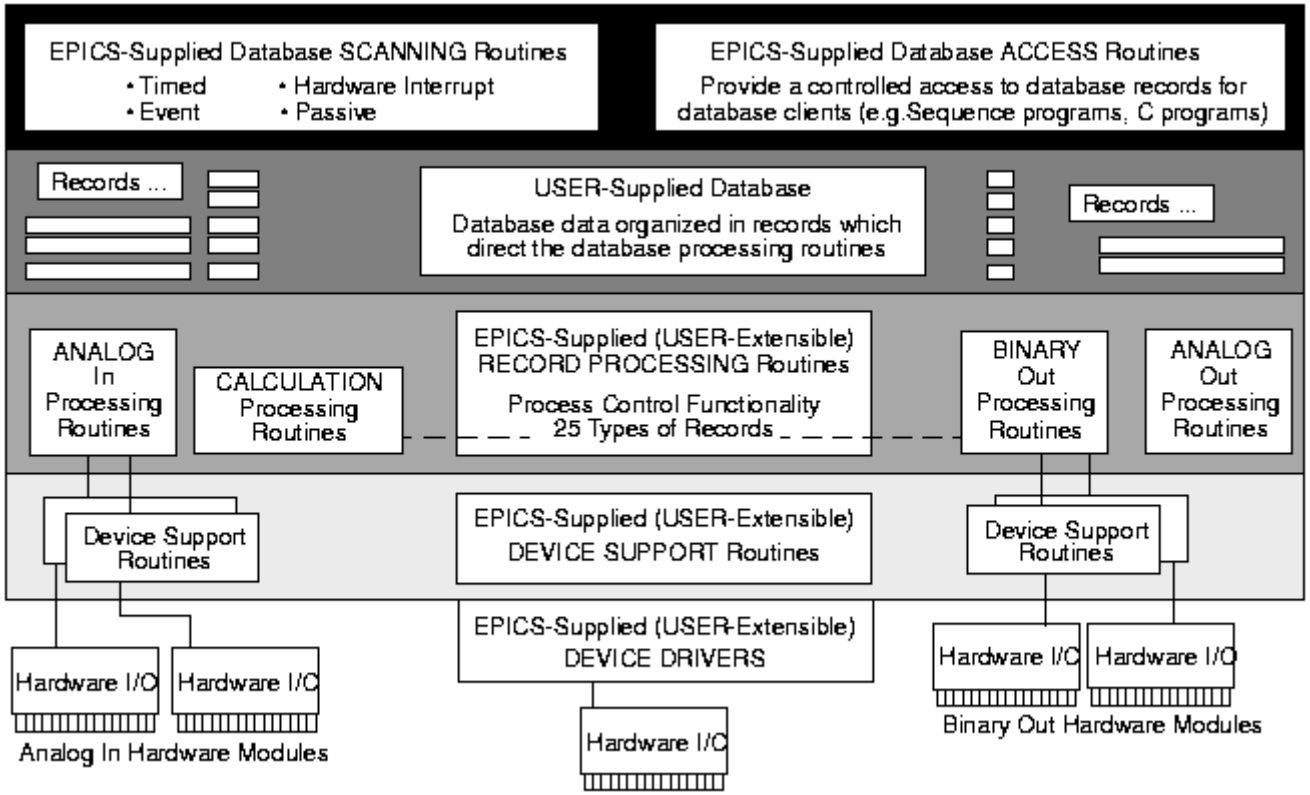
- Your EPICS at Nustar team
  - Haik, Hans, Bastian
  - **You!**



Blank Slide

EPICS Introduction

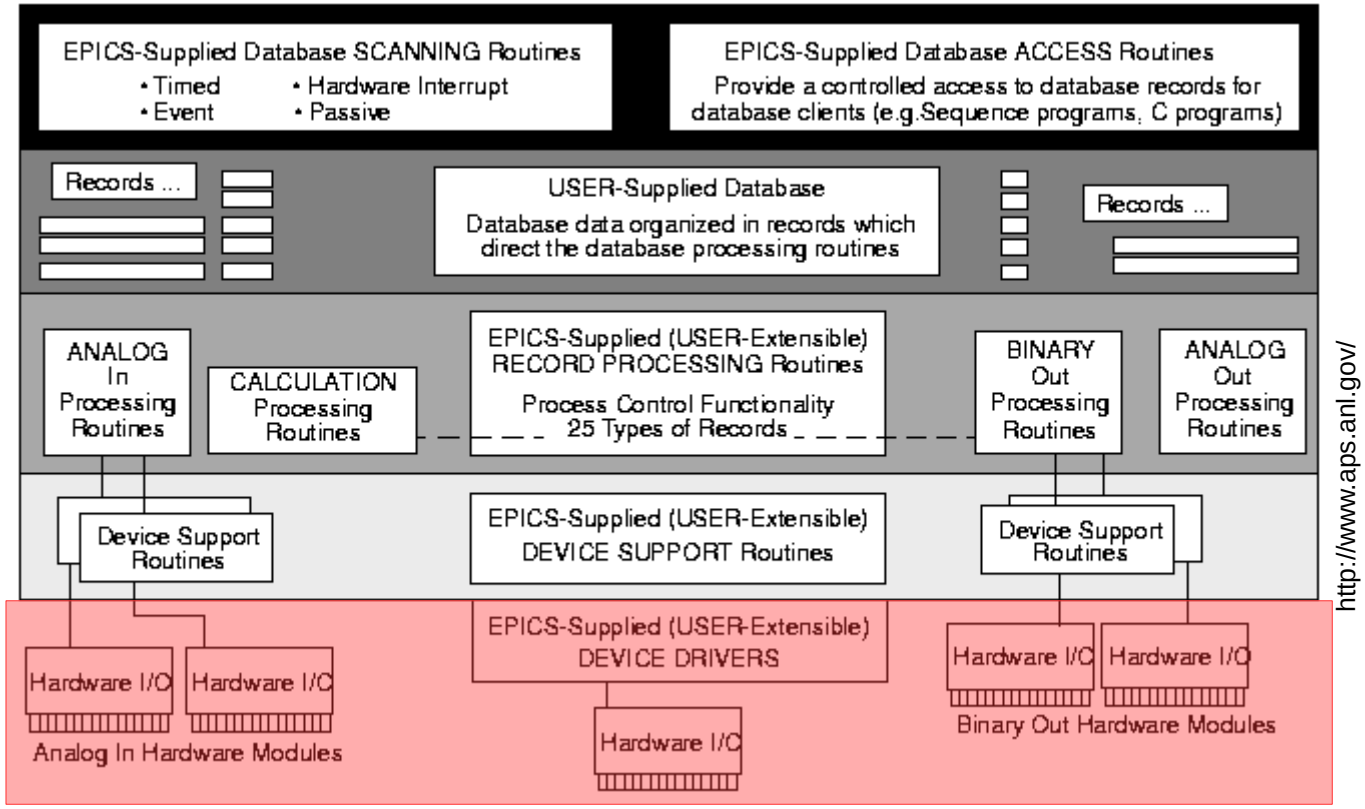
- The EPICS Server (IOC)



http://www.aps.anl.gov/

EPICS Introduction

- The EPICS Server (IOC)

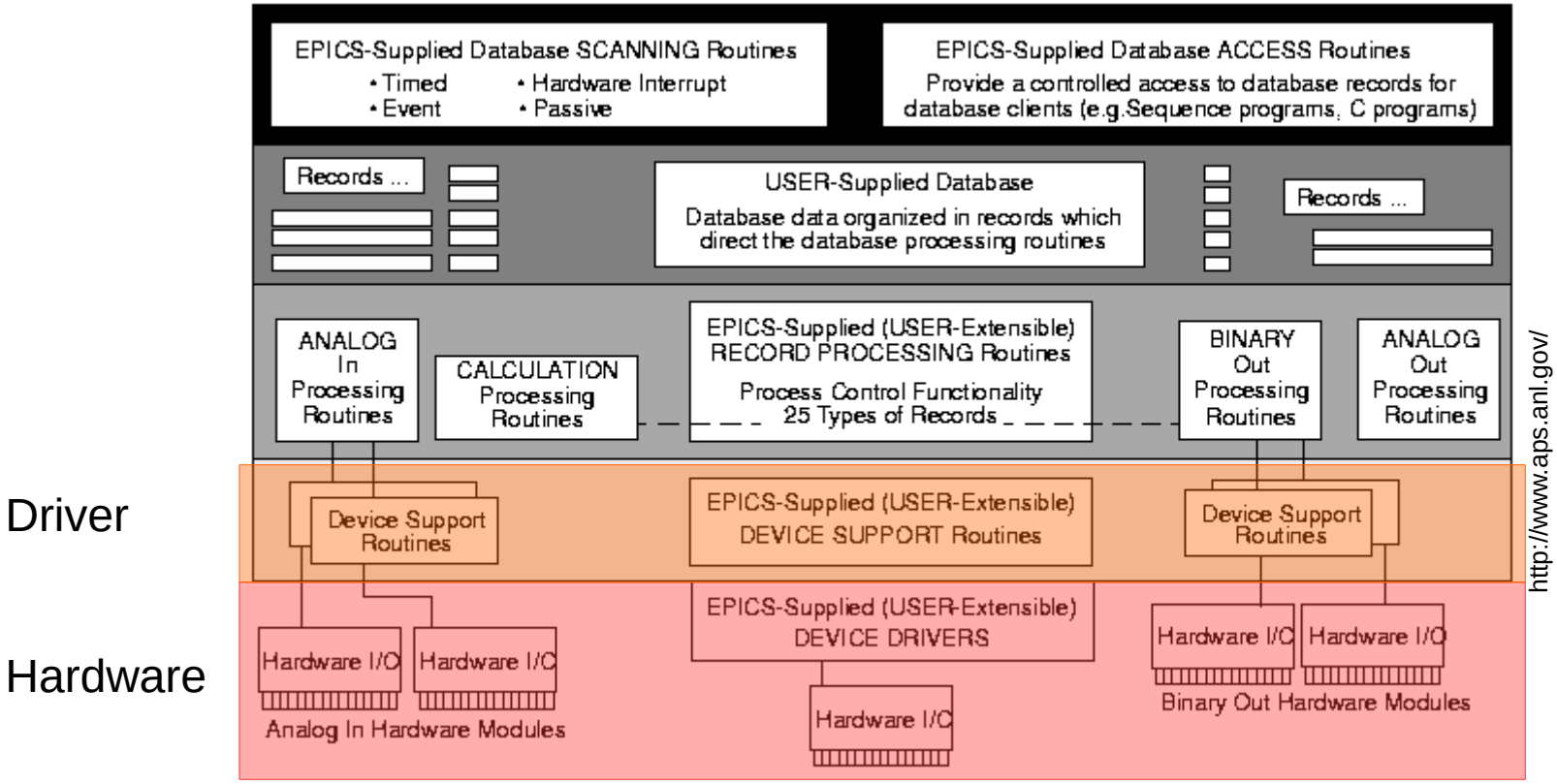


http://www.aps.anl.gov/

Hardware

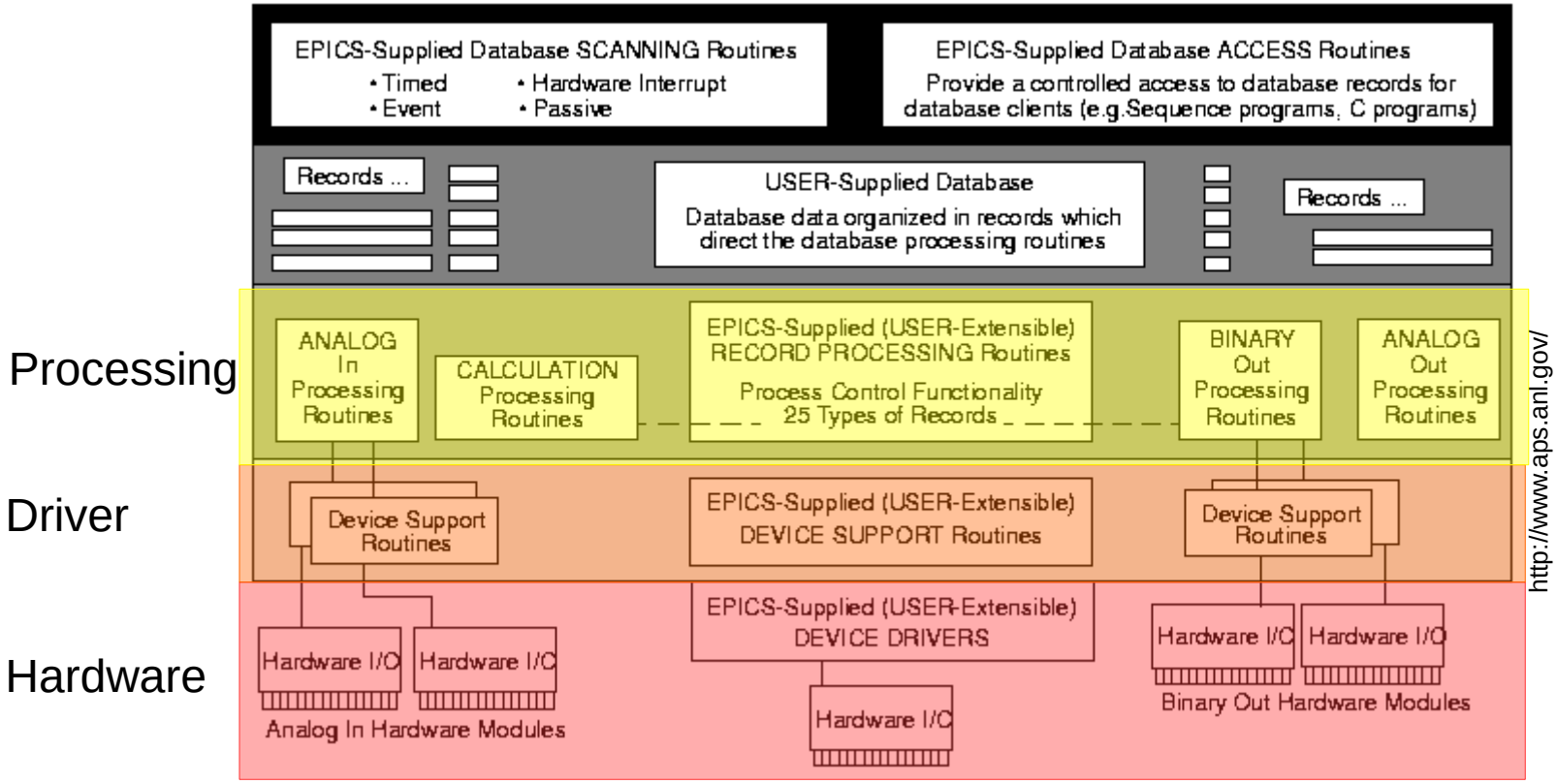
EPICS Introduction

- The EPICS Server (IOC)



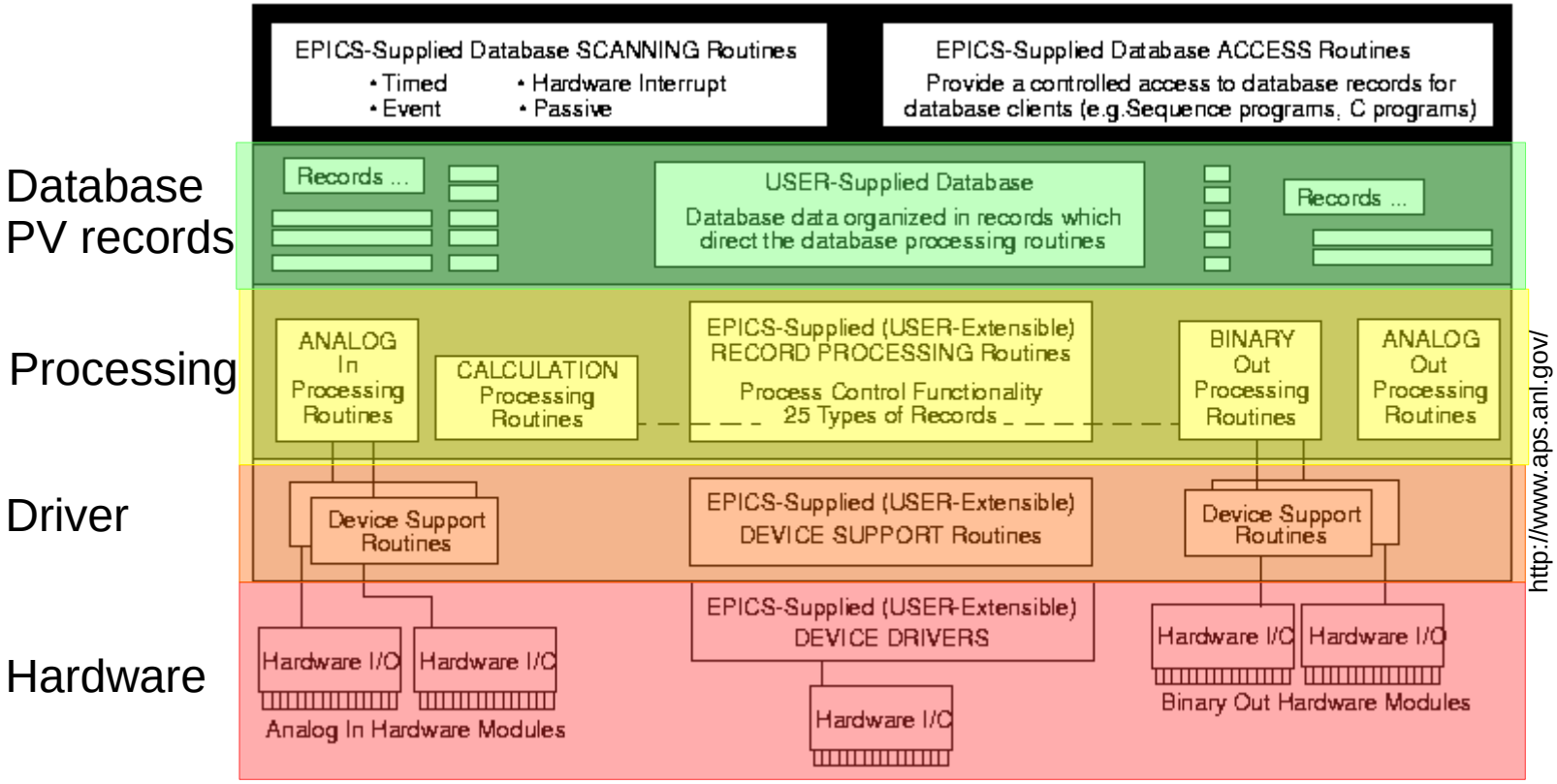
EPICS Introduction

- The EPICS Server (IOC)



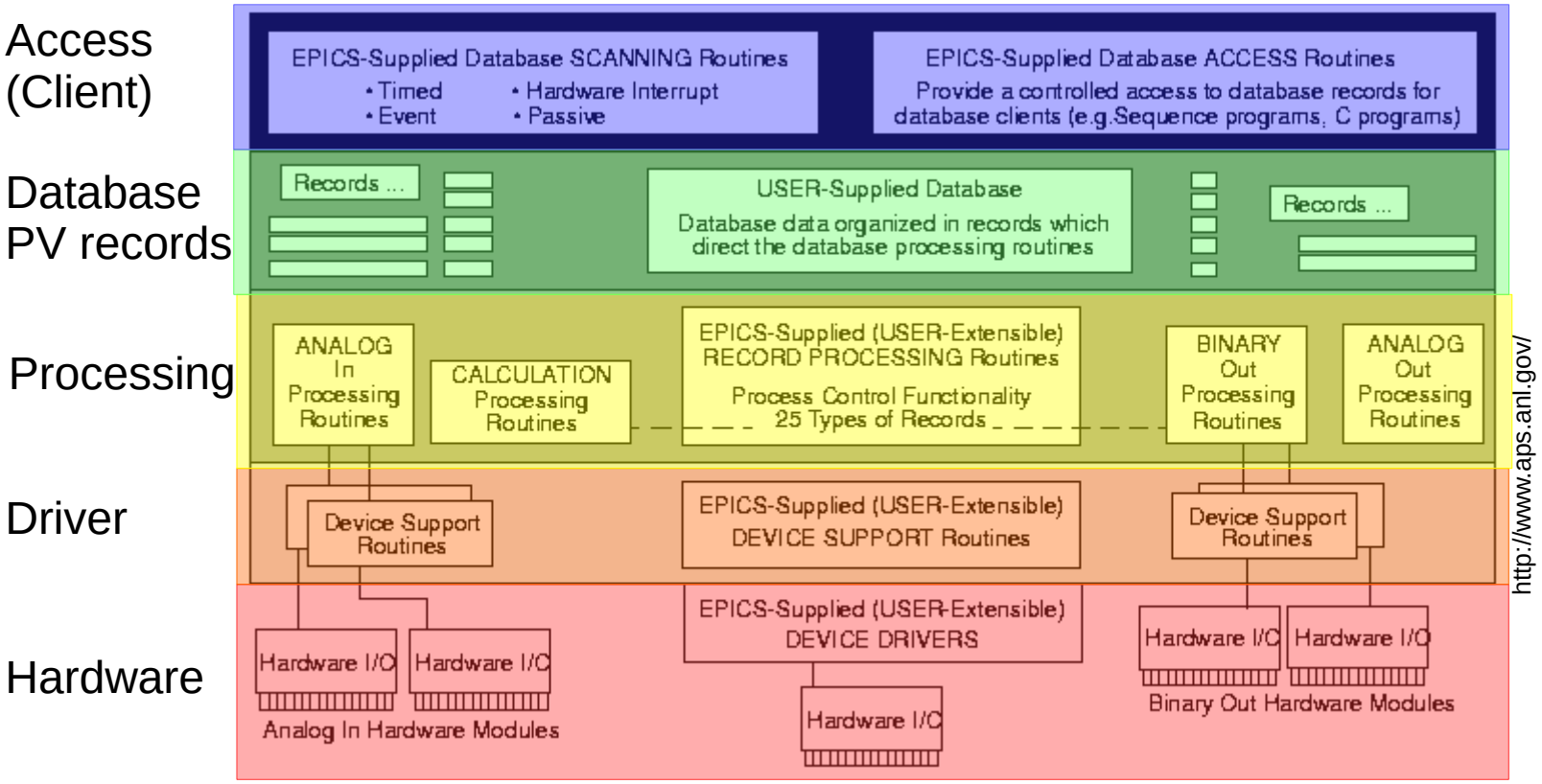
EPICS Introduction

- The EPICS Server (IOC)



EPICS Introduction

- The EPICS Server (IOC)



EPICS Introduction

- The EPICS Server (IOC)

User Input Needed (for new/unknown device)

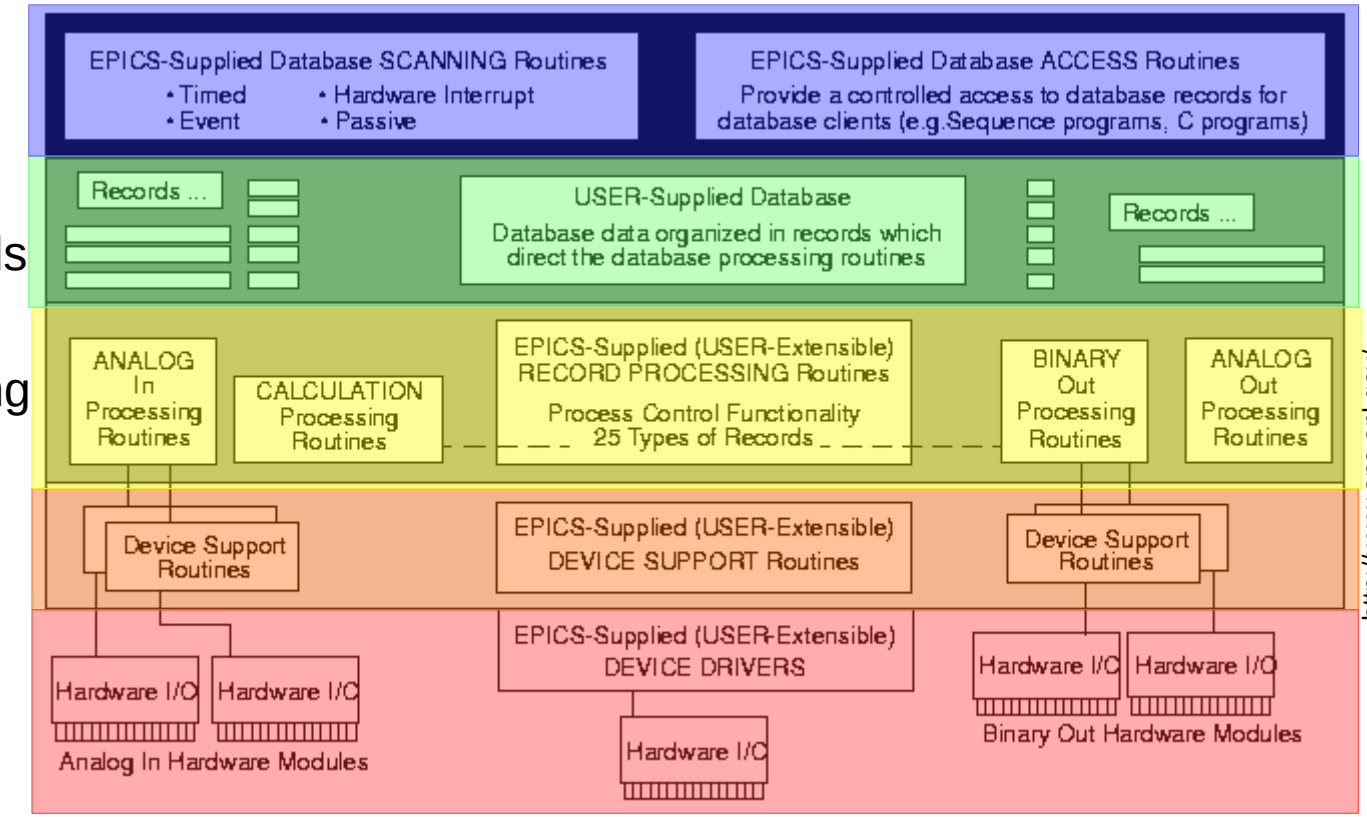
Access (Client)

Database PV records

Processing

Driver

Hardware



http://www.aps.anl.gov/

Maybe already Supported by Generic driver

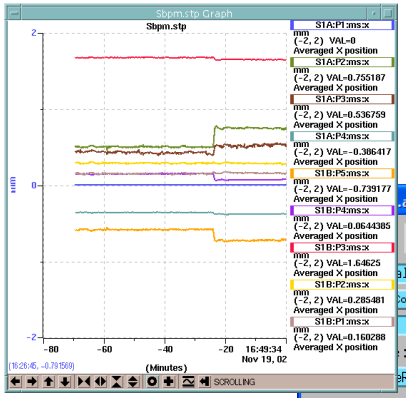
EPICS Clients (CA Clients)

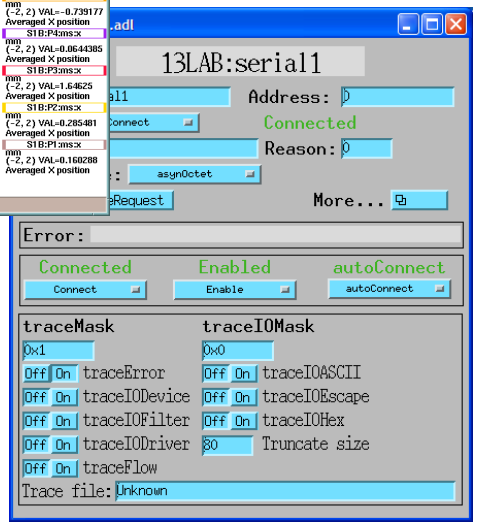
- Simple clients (command line)
  - Read a value: `caget <PV>`
  - Set a value: `caput <PV> <value>`
  - Monitor a value: `camonitor <PV>`
- Not meant for production operation!
- Great for debugging / setting up

EPICS Clients (CA Clients)

- Advanced clients provide central services:
  - BURT: Backup and Restore tool
  - Gateway: Bridge between networks
  - Nameserver: PV name resolution
  - Caxy: Tunneling over SSH
  - Many more, see: http://www.aps.anl.gov/epics/extensions/

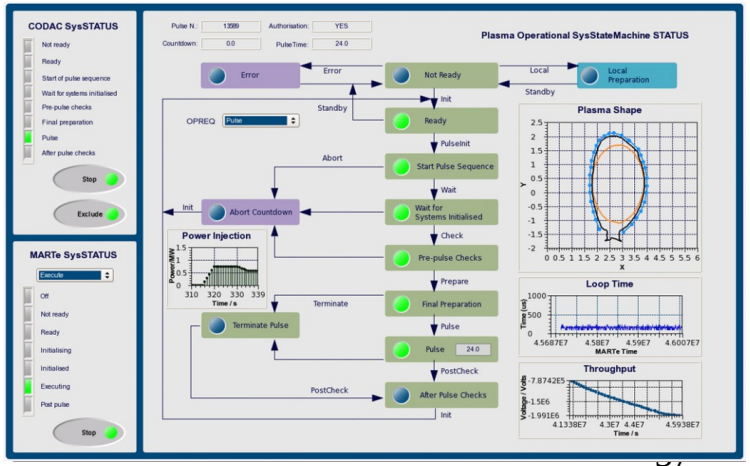
EPICS Clients (CA Clients)





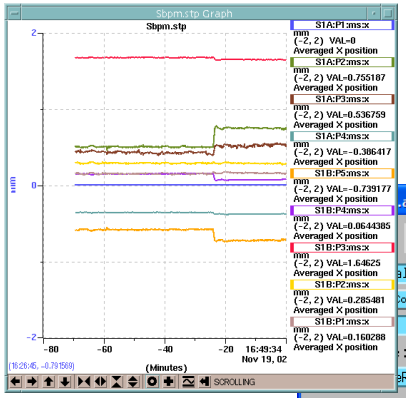
Graphical clients:

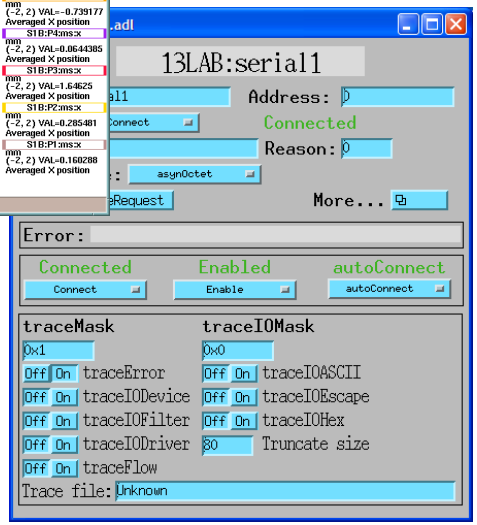
- StripTool: Display PV vs. Time
- MEDM: Motif editor and display manager
  - Old and dusty, but simple to set up
- EPICS-Qt: Modern GUI library based on Qt
  - Needs programming skills
- BOY: Best OPI Yet
  - Drag&Drop GUI, used with CSS (Control system Studio)



portal.slac.stanford.edu

EPICS Clients (CA Clients)

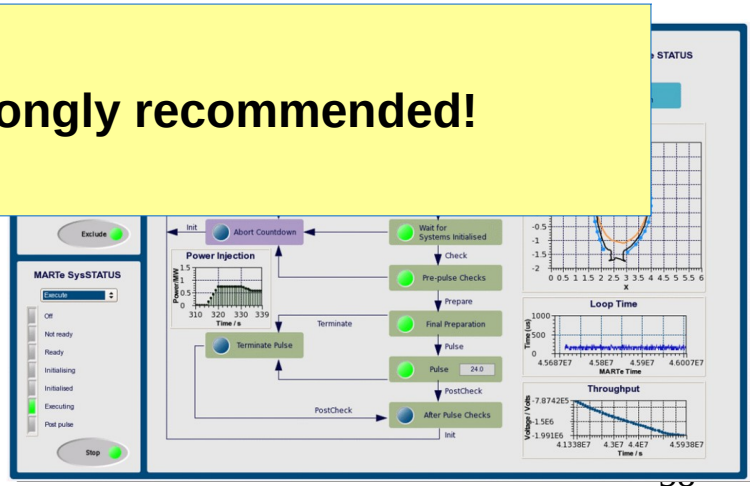




Graphical clients:

- StripTool: Display PV vs. Time
- MEDM: Motif editor and display manager
  - Old and dusty, but simple to set up
- EPICS-Qt: Modern GUI library based on Qt
  - Needs programming skills
- BOY: Best OPI Yet
  - Drag&Drop GUI, used with CSS (Control system Studio)

Strongly recommended!



portal.slac.stanford.edu

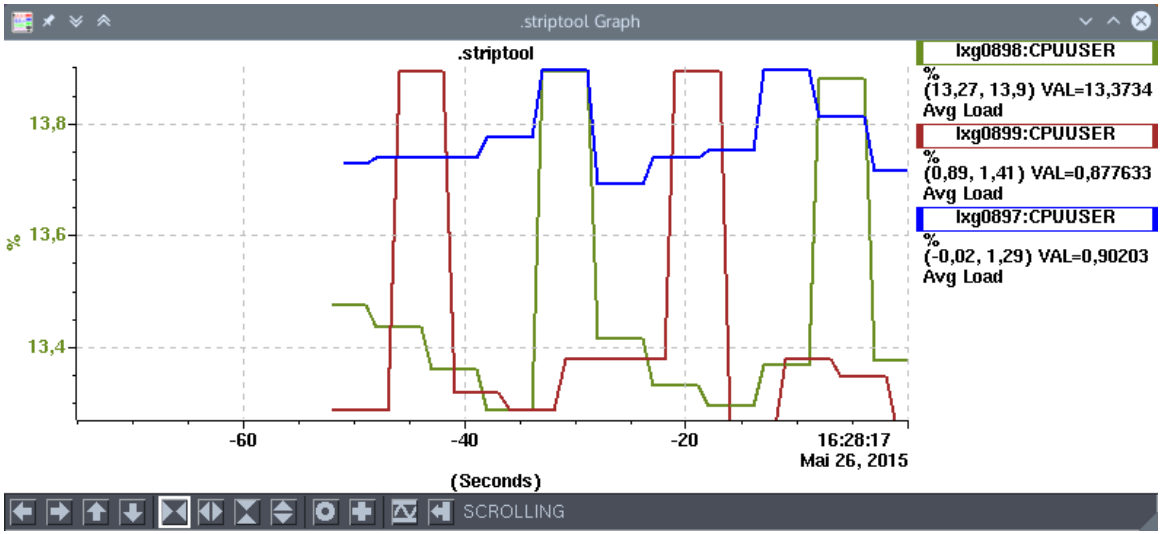
Simple example IOC

- pcMonitor: Watch system parameters (CPU,…)
  - PVs: CPU, MEM, LOAD, UPTIME, …
  - Three IOCs
  - StripTool used to display values

lxg0897: CPUUSER

lxg0898: CPUUSER

lxg0899: CPUUSER



EPICS Introduction

The record system (Database)

A record defines a set of parameters
For each process variable (PV), i.e.
for each system parameter

Examples:

- (AO) Aladin:currentSet
- (AI) Aladin:currentMeasured
- Many more: BI, BO, CALC,...

Standard contents of an EPICS record:

- Name, Description
- Device Type
- Time stamp
- Access security group (ASG)
- Current alarm status (STAT)
- Alarm severity (SEVR)
- Undefined (Not yet processed)
- Scanning algorithm (SCAN)
- Forward link (FLNK)
- and a few more...

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS Introduction

The **AI** record (Analog Input)

```
struct aiRecord  {
  /* start of ai specific fields */
  double          val;   /* Current EGU Value */
  struct link     inp;   /* Input Specification */
  short           prec;  /* Display Precision */
  unsigned short  linr;  /* Linearization */
  float           eguf;     /* Engineer Units Full */
  float           egul;     /* Engineer Units Low */
  char            egu[16];  /* Engineering Units */
  float           hopr;  /* High Operating Range */
  float           lopr;  /* Low Operating Range */
  float           aoff;  /* Adjustment Offset */
  float           aslo;  /* Adjustment Slope */
  float           smoo;  /* Smoothing */
  float           hihi;  /* Hihi Alarm Limit */
  float           lolo;  /* Lolo Alarm Limit */
  float           high;  /* High Alarm Limit */
  float           low;   /* Low Alarm Limit */
  unsigned short  hhsv;  /* Hihi Severity */
  unsigned short  llsv;  /* Lolo Severity */
  unsigned short  hsv;   /* High Severity */
  unsigned short  lsv;   /* Low Severity */
  double          hyst;  /* Alarm Deadband */
  double          adel;  /* Archive Deadband */
  double          mdel;  /* Monitor Deadband */
  double          lalm;  /* Last Value Alarmed */
  double          alst;  /* Last Value Archived */
  double          mlst;  /* Last Val Monitored */
  double          eslo;  /* Rawto EGU Slope */
  long            roff;  /* Raw Offset */
  caddr_t         pbrk;  /* Ptrto brkTable */
  short           init;  /* Initialized? */
  short           lbrk;  /* LastBreak Point */
  long            rval;  /* Current Raw Value */
  long            oraw;  /* Previous Raw Value */
};
```

Standard contents of the **AI** record:

- Raw value: 'rval'
- Value (in engineering units): 'val', 'prec'
- Recipe for input: 'inp'
- Range conversion: 'roff', 'aoff', 'aslo'
- Unit conversion: 'egu*', 'linr', 'smoo'
- Range checking: 'hopr', 'lopr'
- Alarm limits: 'hihi', 'high', 'low', 'lolo'
- Hysteresis: 'hyst', 'adel', 'mdel'

EPICS in R3B

- Systems using EPICS:
  - LAND/Neuland (HV, TACQUILA settings)
  - TOF (HV)
  - ALADIN (Current)
  - Crystal Ball (HV, Shaper modules)
  - PSP (HV, work-in-progress)
  - LOS/ROLU (HV, controls)

EPICS in R3B

- Status quo, or 'The big jumble of windows'
  - No archiving, alarm handling, security, continuity,...

EPICS in R3B

- Status quo, or 'The big jumble of windows'
  - No archiving, alarm handling, security, continuity,…
- How can we improve?