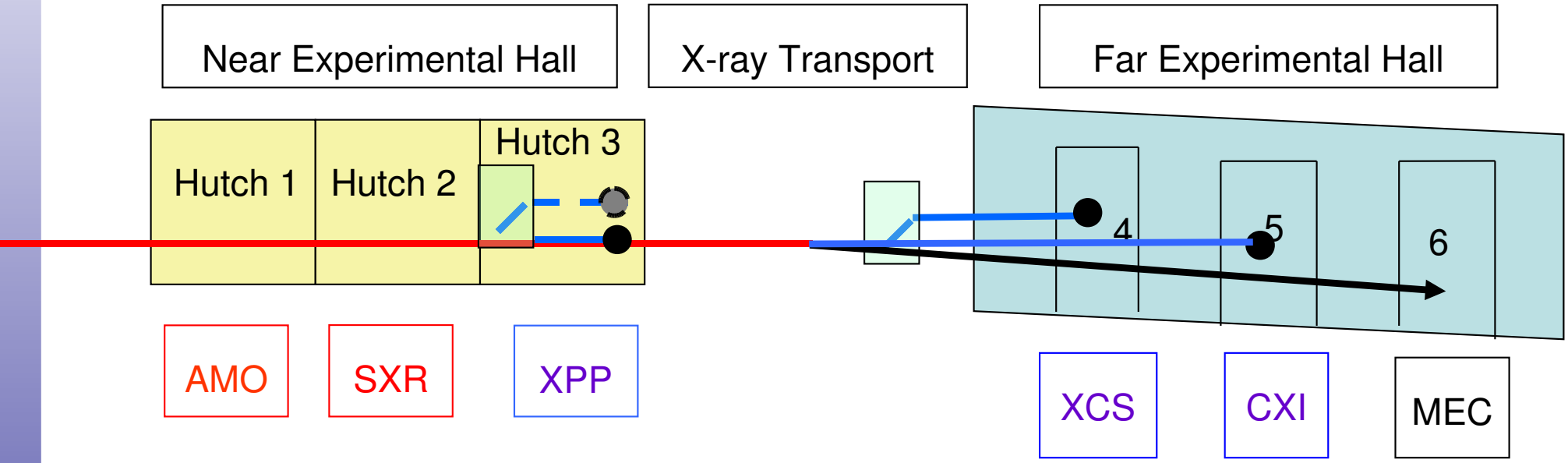
Controls and Data Systems for LCLS

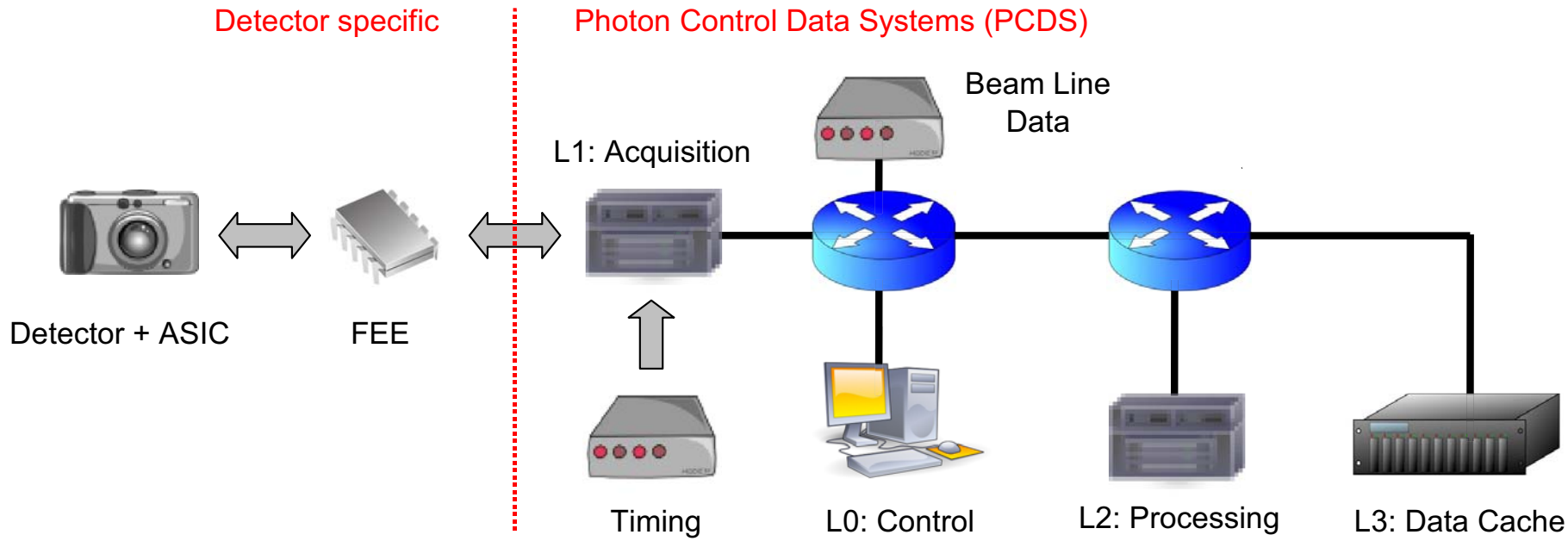
Amedeo Perazzo
LCLS Photon Controls and Data Systems

BNL, April 21st 2010
NSLS-II DAQ Workshop

NEH/FEH Hutch Assignments



DAQ System Architecture



- **Level 0: Control**
  - Run & configuration control
  - Run & telemetry monitoring
- **Level 1: Acquisition**
  - Image acquisition, calibration
  - Event-building with beam-line data
  - Correction using calibration constants
  - Data reduction (vetoing, compression)
- **Level 2: Processing**
  - Pattern recognition, sort, classify, alignment, reconstruction
  - Monitoring
- **Level 3: Online Archiving**
  - NEH/FEH local data-cache
  - Local cache can buffer up to 4-days worth of data
  - Offline will transport data to tape staging area in SCCS Computer Center

Languages and Frameworks

- **C/C++: DAQ core, EPICS drivers**
- **Python as high level scripting language**
  - Easy to learn, fast dev cycle, extensible, open-source, powerful, relatively fast
  - Provides common scripting language to both the DAQ system and controls
- **QT as graphical user interface**
  - Provides common interface to both the DAQ system and controls
- **EPICS: distributed control system, provides:**
  - channel access servers (iocs)
  - channel access clients
  - channel access gateway
  - archiver
  - alarm handler

Run Control

- **Main entry point screen for taking data**
- **Partition management**
  - Selects which nodes participate to a given run
- **Finite State Machine**
  - Allow operator to drive DAQ through its different states
    - allocated, configured, running, etc...
- **Configuration**
  - Operator may select from different configurations by using a run key

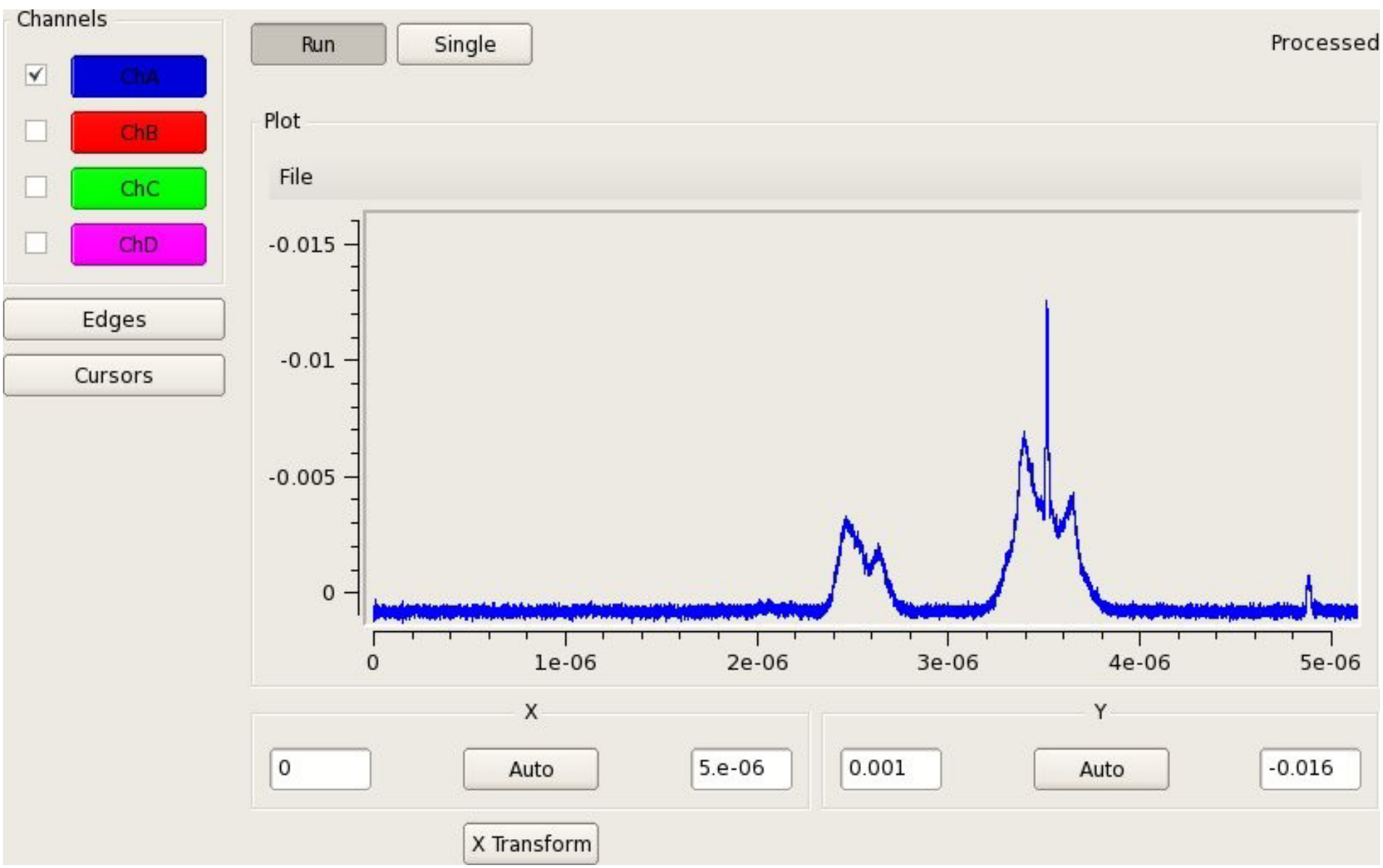
Configuration

- **The configuration software allows the operator to**
  - Create profiles of the detector
  - Upload these profiles to the various instruments before data taking starts
- **The configuration data describes**
  - Instrument settings
  - Output data types
- **Instrument configuration saved together with science data**
  - Always possible to correlate configuration with corresponding event data

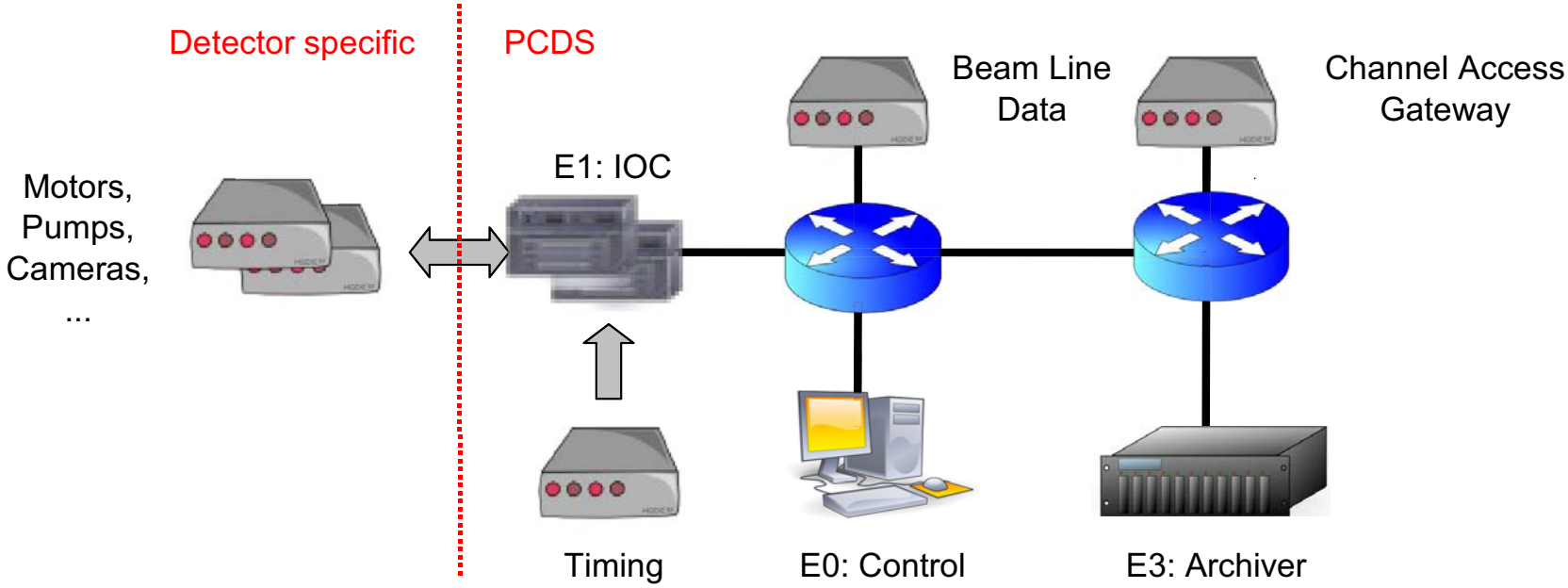
Online Monitoring

- **Science data multicast to both DAQ nodes and monitoring nodes**
  - Monitoring nodes see same data as processing nodes
- **Individual or summed spectra displayed in real-time**
  - User selectable refresh rate of up to 10 Hz
  - Time-frame of the display user selectable
    - i.e. last n shots, cumulative data set, etc.
  - Possible to zoom on the display
    - options for auto-scaling or user defined ranges
  - Storage and printing capabilities included for the displayed data
  - All data converted from hardware format to physical units before display
- **Users may introduce new monitoring objects**
  - eg if new instrument is added or different processing is required for existing instrument
  - New objects are automatically available and displayed on demand

Online Monitoring Example

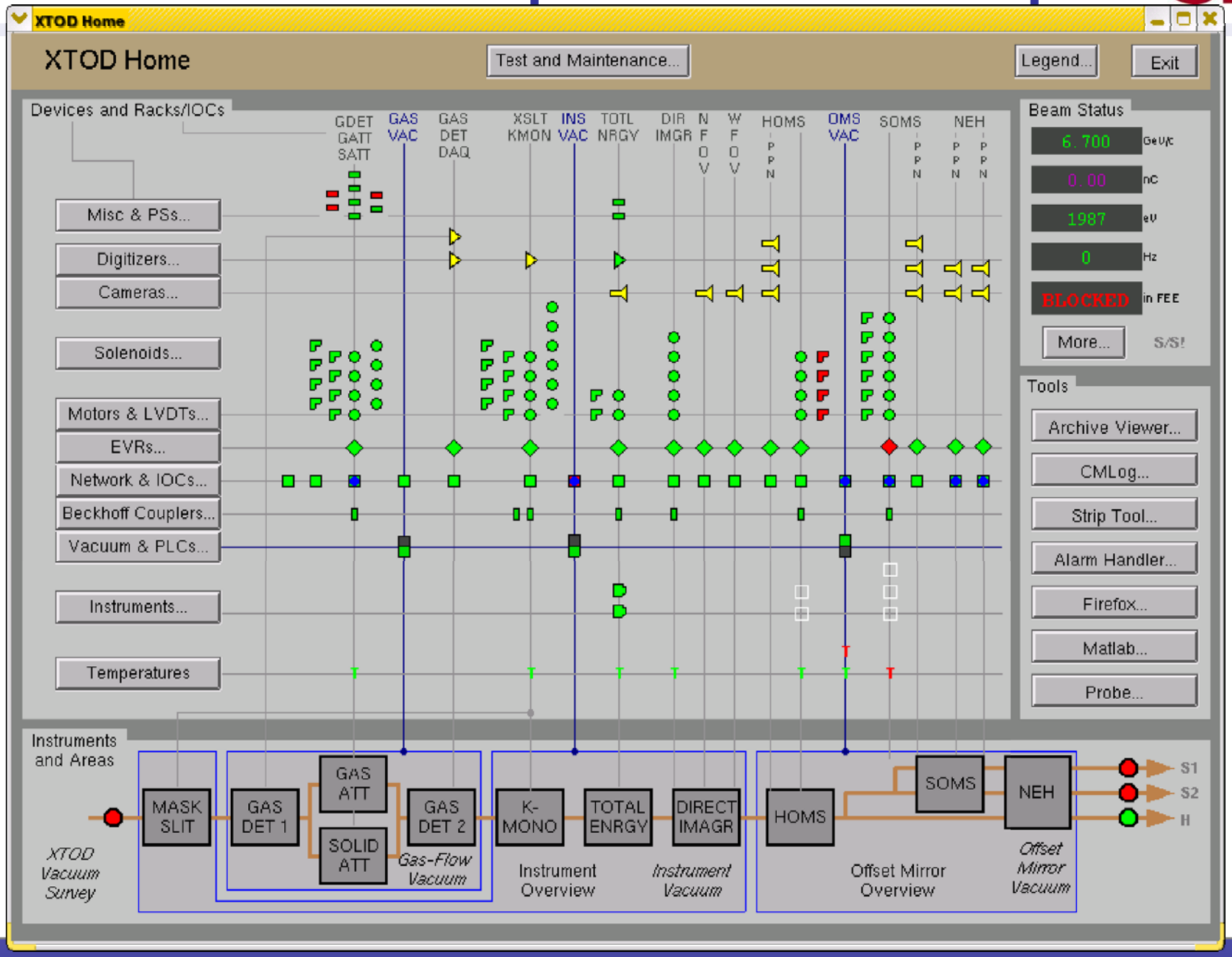


Controls Architecture



- **Control system is EPICS based**
  - Each experiment control system connects to accelerator controls and to photon-beam controls through a channel access gateway (CAG)
    - CAG as proxy for external clients and to filter traffic between different systems

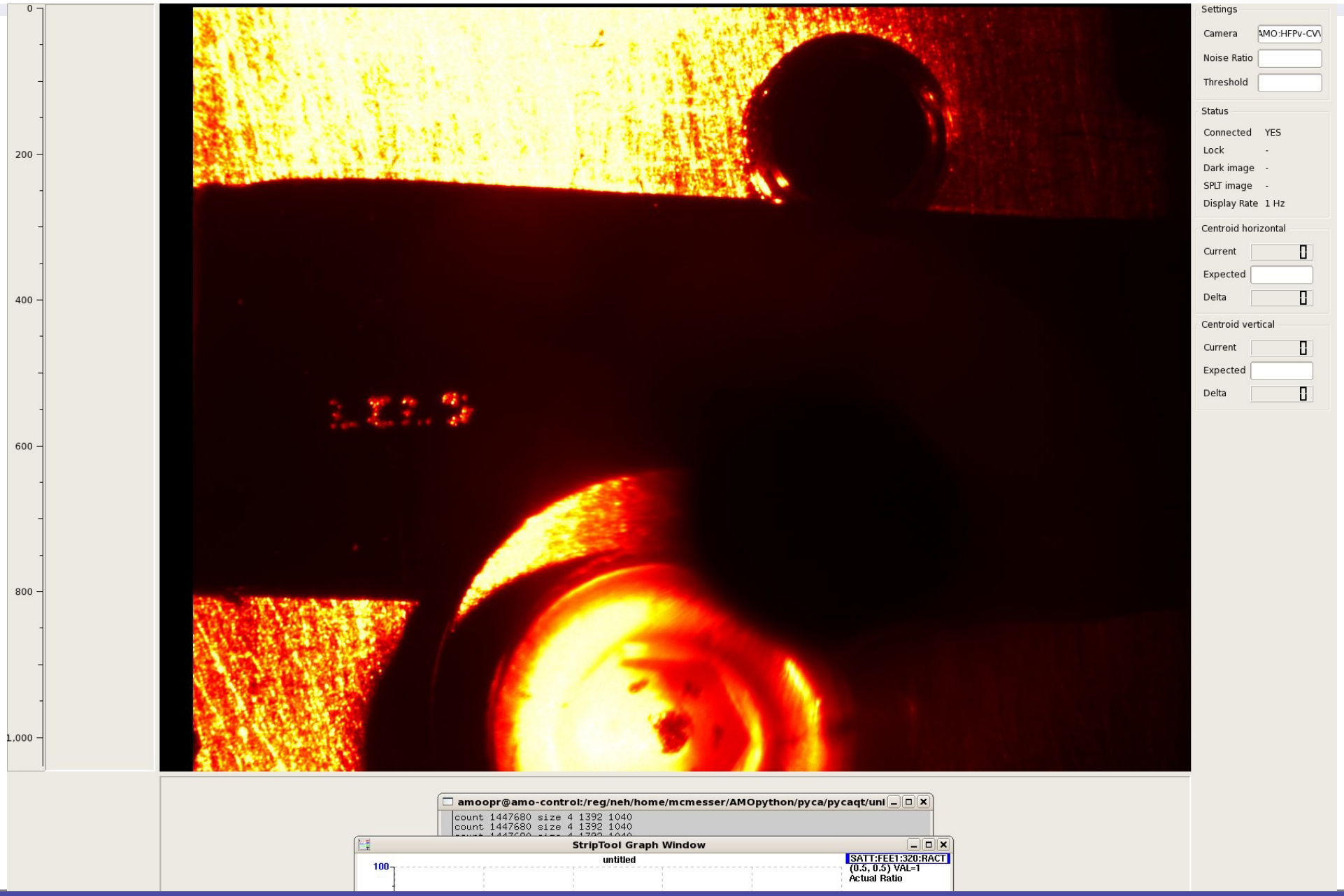
Controls Top Level GUI Example



Python Control Interface

- **Python EPICS interface (`pyca`) created for developing high level applications**
  - Based on two older existing packages (`EpicsCA` and `CaPython`), but adds two features:
    - Ability to avoid all unnecessary data copies if user decides to do so
      - *Critical for large EPICS waveforms like mega-pixel cameras*
    - Use EPICS in multi-thread mode (`ca_enable_preemptive_callback`)
      - *Allow significant simplification of the interface (500 lines of code as opposed to a few thousands of most EPICS interfaces)*
  - CPU usage reduction ~50% has been measured when using the pyca/QT combination as opposed to similar MATLAB applications or EDM displays

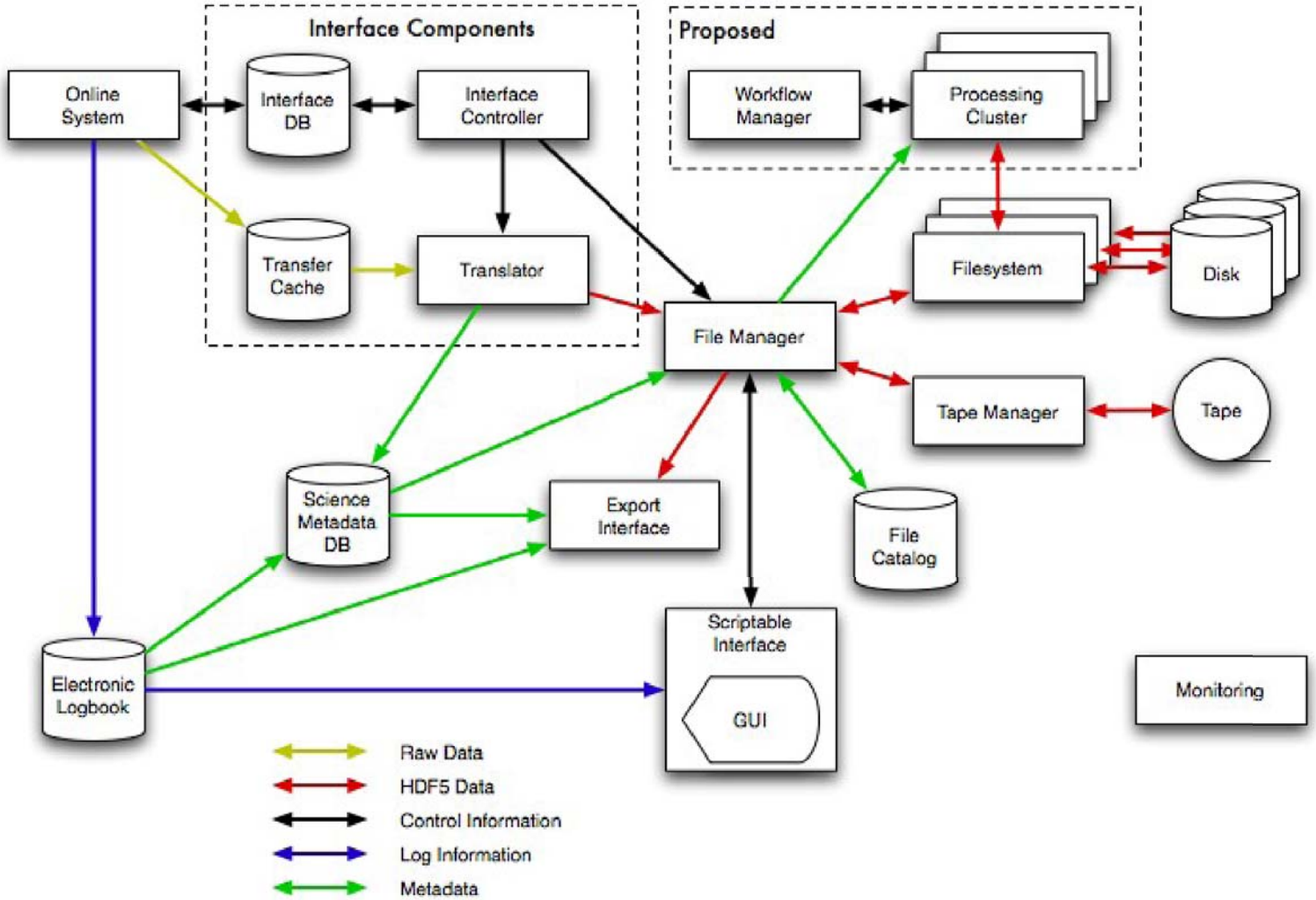
Python/QT Example



Data Management

- **Aggregates files from different processing nodes**
- **Translate data from online object-oriented format to HDF5**
- **Archive translated data to short and log term storage**
- **Provide access for scientists to archived data**
  - Data access for:
    - Processing cluster nodes
    - User workstations
    - Offsite export
- **Manage data ownership and access restrictions**

Data Management Architecture



Data Export

- **HDF5 files plus electronic logbook data**
  - HDF5 files derived from raw, object oriented, DAQ format (XTC)
- **Network transport implemented using `iRODS` and `bbcp`**
- **Data files include:**
  - Experimental Data
    - Each event stamped with time and pulse ID
  - Instrument Definition: describes each part of the experimental instrument
    - eg positions, distances, settings, calibrations
  - Beam Line Data: set of electron and photon beam parameters taken every pulse
    - eg electron beam energy and spread, photon beam pulse length and energy
  - EPICS Data
    - selected subset data from slow controls
  - Science Metadata
    - eg project identification, operator identification

Storage

- **Short term storage**
  - Disk based
    - Storage cluster located in NEH server room
  - Provides ready access to science data
    - For analysis on processing farm
    - For transfer to home institution
  - 2 PB in the beginning
    - More will be added when needed
  - 1 year retention policy
- **Long term storage**
  - Tape based
    - Staging system located in SLAC computing facility
  - Essentially unlimited retention period
  - Allows offline system to restore the science data to short term storage if data needed after 1 year

File Handling

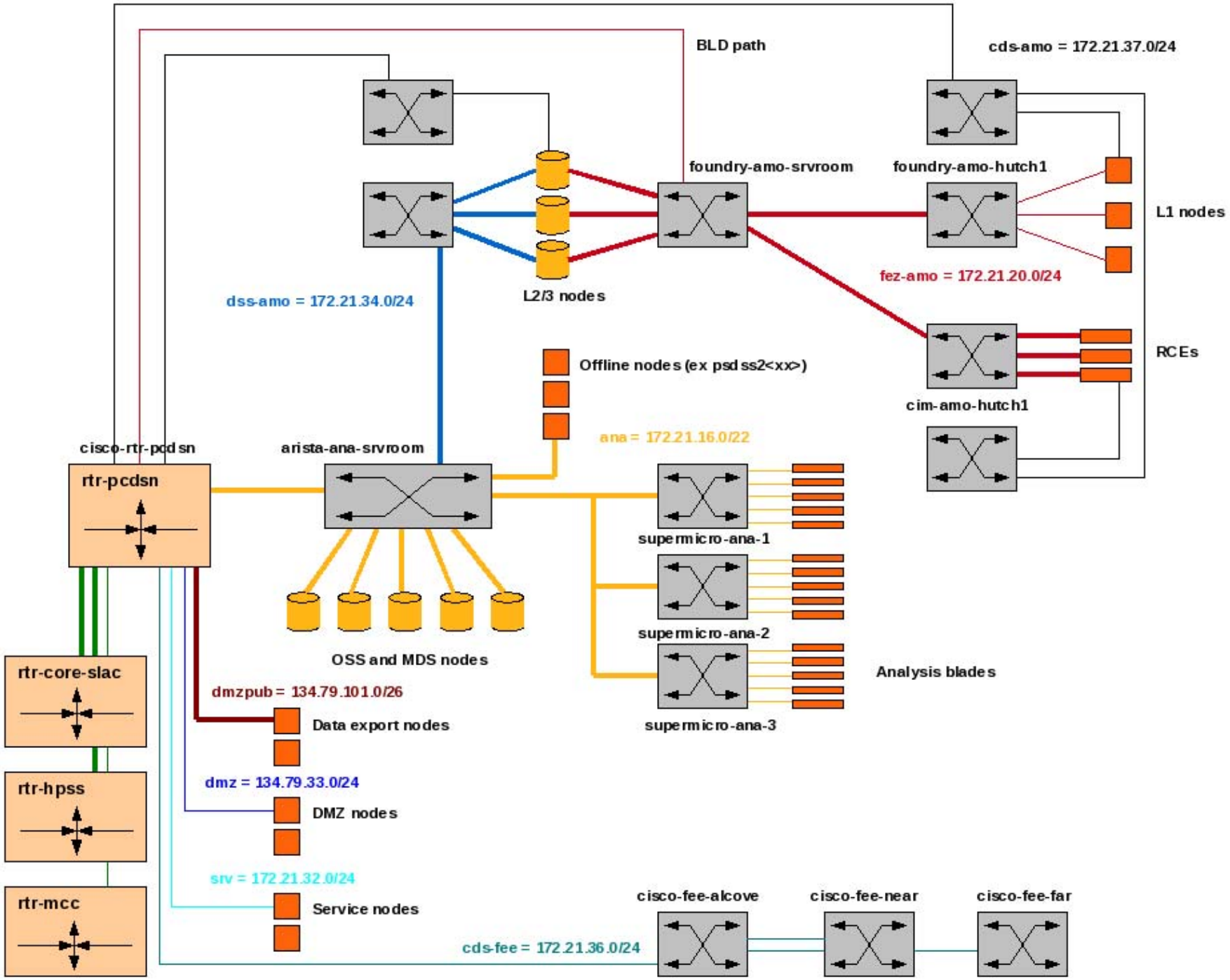
- **File Manager**
  - Routes files to appropriate destinations
  - Maintains physical location information
  - Interfaces with filesystem, tape manager, and export interface
  - Implemented with iRODS
- **Filesystem**
  - Presents standard Unix filesystem interface
  - High-bandwidth cluster filesystem
    - LUSTRE technology
    - 5GB/sec per PB read and write bandwidth
  - Scalable to petabytes, with terabytes per file
  - Average file size: 100s of MB to GBs
- **Tape Manager**
  - Writes files to tape, recording locations
  - Implemented using HPSS under iRODS

Data Ownership

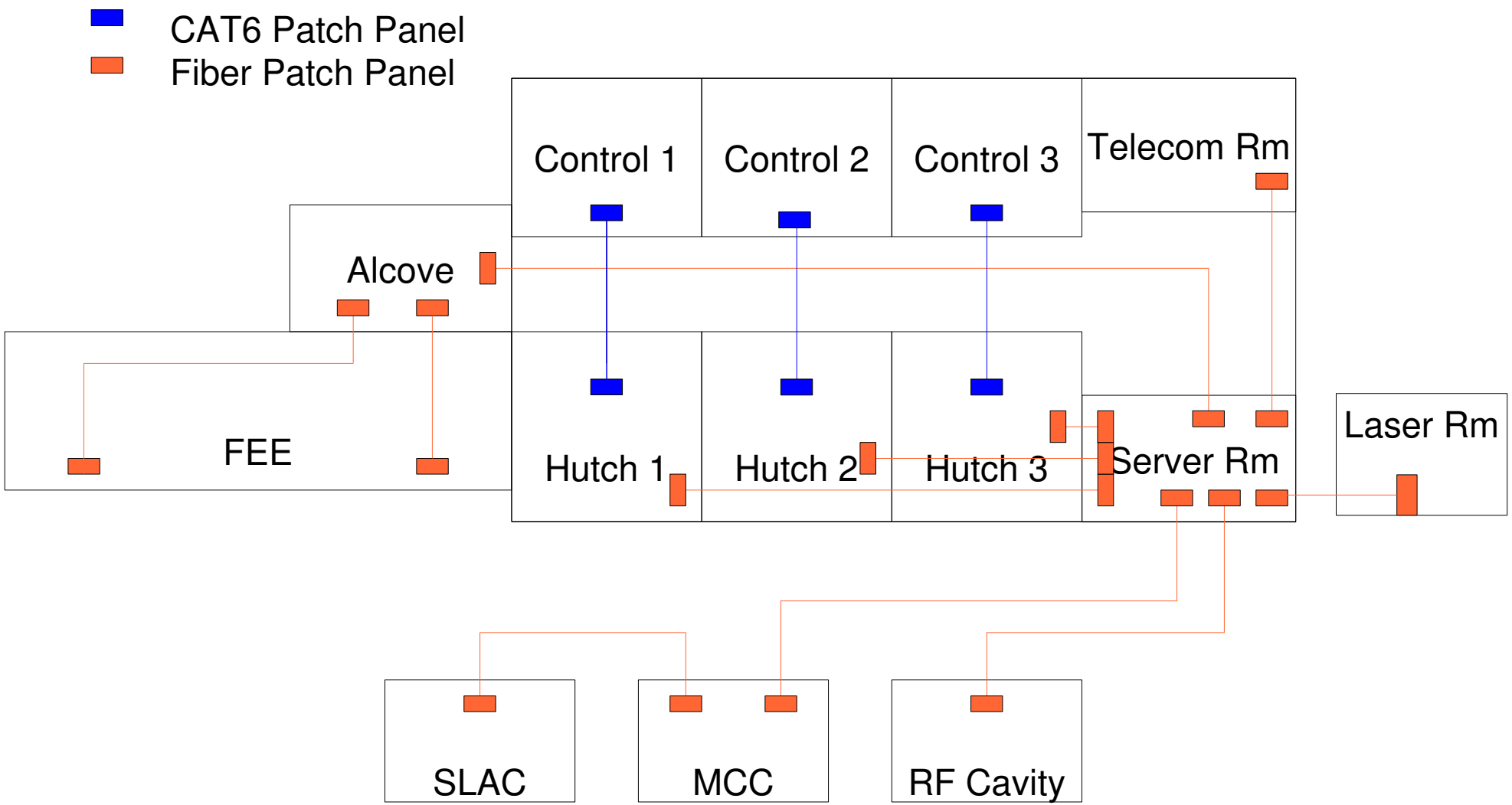
- **Ownership assigned to projects:**
  - Research activity with specific purpose
    - Scientist analyzing sample on existing instrument for days or weeks
    - Collaboration with new instrument running for months or years
  - Sub-projects possible
  - Individuals and groups assigned to projects
- **Data tagged with project identifier**
- **Grant and revoke access to project data based on individual or group**
    - Implemented with POSIX ACLs on LUSTRE file system

Processing

- **Few hundred cores**
  - Located in NEH server room
  - Direct access to short term storage filesystem
- **Both batch and interactive nodes**
- **Basic tools provided by PCDS group**
  - eg: data iteration interface
  - Supported languages: C/C++ and Python
  - Users may run their own code


BLD path
cds-amo = 172.21.37.0/24
foundry-amo-srvroom
foundry-amo-hutch1
L1 nodes
fez-amo = 172.21.20.0/24
L2/3 nodes
dss-amo = 172.21.34.0/24
RCEs
Offline nodes (ex psdss2<xx>)
cim-amo-hutch1
ana = 172.21.16.0/22
cisco-rtr-pcdsn
arista-ana-srvroom
rtr-pcdsn
supermicro-ana-1
supermicro-ana-2
OSS and MDS nodes
Analysis blades
supermicro-ana-3
rtr-core-slac
dmzpub = 134.79.101.0/26
Data export nodes
rtr-hpss
dmz = 134.79.33.0/24
DMZ nodes
srv = 172.21.32.0/24
rtr-mcc
Service nodes
cisco-fee-alcove
cisco-fee-near
cisco-fee-far
cds-fee = 172.21.36.0/24

NEH Data Cable Trunks



Amedeo Perazzo
perazzo@slac.stanford.edu

Users Workshops

- **Two requests emerged from various users workshops**
  - Ability for the users to plug-in their applications to both online and offline systems
  - Ability for the users to run their analysis code on their own machines without the need to export the data
  - Ability for the users to access the data on disk as soon as the file is close and before the files are copied/translated to the offline system
- **User plug-in to online**
  - Subset of online monitoring nodes allocated for user applications
  - User code communicates with online core applications through shared memory
    - User code parse event using DAQ data format
- **User mount short term storage**
  - Implemented through sshfs to a node which is also read-only client to short term storage
    - Considering XROOTD server to remove performance limitations imposed by using ssh

Lessons Learned

- **Timing system**
  - Proprietary EVG/EVR from MRF
  - Relatively expensive per board
  - Not enough form factors
    - Limit choice of DAQ/controls nodes
- **Choice of EDM as main EPICS GUI**
  - Not supported anymore
- **Underestimated needs for server room**
- **Underestimated needs for an offline system**
- **RTEMS**
  - Good, fast RTOS, but
  - Non trivial development environment
    - Introduce strong dependency on few expert developers
  - Currently gradually converting to Linux on photon side
- **On-boarding process**
  - Computer account creation requires user signature, cannot be done online

Lessons Learned (2)

- **All 10Gb connections are optical**
  - Expensive transceivers
  - SMF very sensitive to dust
  - Existing technology allows 10Gb over CAT6
- **EPICS gateway**
  - Required to manage EPICS traffic among the different networks (accelerator, x-ray tunnel, instruments, etc) but
  - Very painful deployment due to a few bugs in the channel access gateway
    - Under certain conditions (heavy load together with motor operations) entire EDM session would freeze
    - Good response from EPICS community (in particular Jeff Hill)
    - Few problems remain to be ironed out, but currently usable
- **LUSTRE**
  - Mixed feelings
    - Good at putting together scalable large file system using cheap disks
    - Bug in 1.8.1 which caused painful memory leak (fixed in 1.8.2)
    - Performance not as good as underlying hardware
  - Next run will use LUSTRE on offline side, but not online

Lessons Learned (3)

- **Very positive experience with high-performance, relatively cheap, L2 edge switches**
  - L3 traffic done only in central router
- **Users access to instrument subnets**
  - Implemented through netgroups in LDAP database
- **E-log**
  - Developed from scratch
    - Based on PHP
    - Uses MySQL and LDAP backends
    - Allows fine granularity for user access
- **Use of UDP multicast between L1 and L2 DAQ nodes**
  - Natural way to provide data to observer (monitoring) nodes 'for free'
- **Keep the core DAQ and analysis groups small and with very good, highly motivated people**
- **Choose our priorities right**
  - Focus on online for Run1

Conclusions

- **LCLS Run2 will have a fully dedicated offline system**
  - During Run1 we simply provided resources from the online system for user analysis
- **Future online tasks:**
  - Online data filtering (trigger)
  - Online data reduction
- **Future offline tasks**
  - Expand user analysis library
  - Currently evaluating use of GPUs for offline processing
    - Both AMD and NVIDIA