HSF Workshop, January 20th 2015

LCLS Data Systems

Amedeo Perazzo
SLAC

U.S. DEPARTMENT OF ENERGY
Office of Science

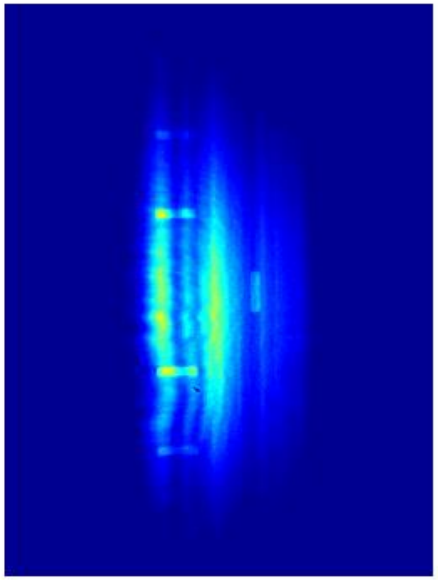
SLAC NATIONAL ACCELERATOR LABORATORY

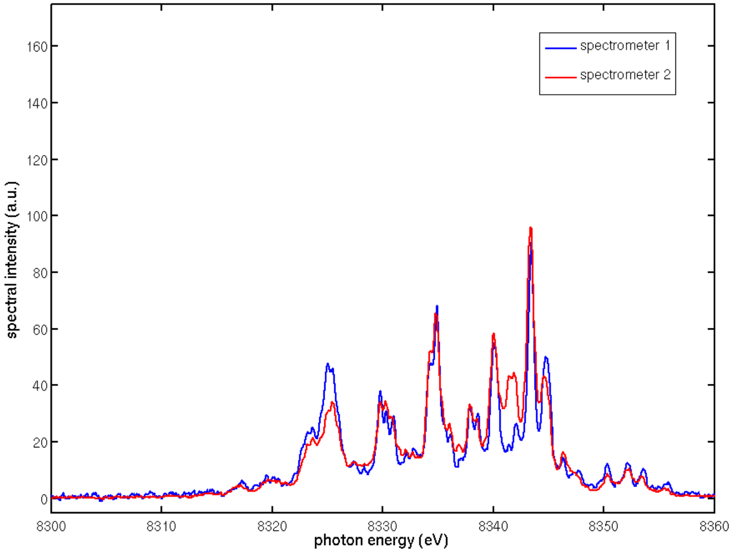
LCLS Source Fluctuations (movie)

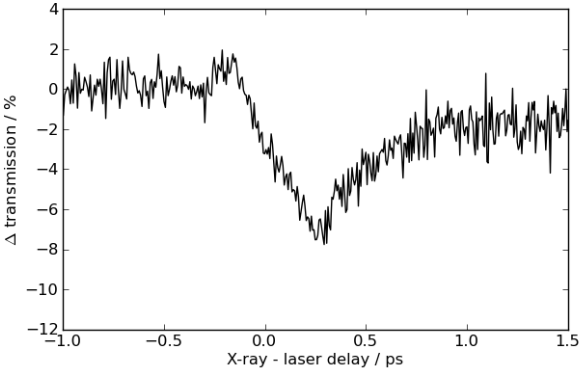
Spatial

Spectral

Temporal



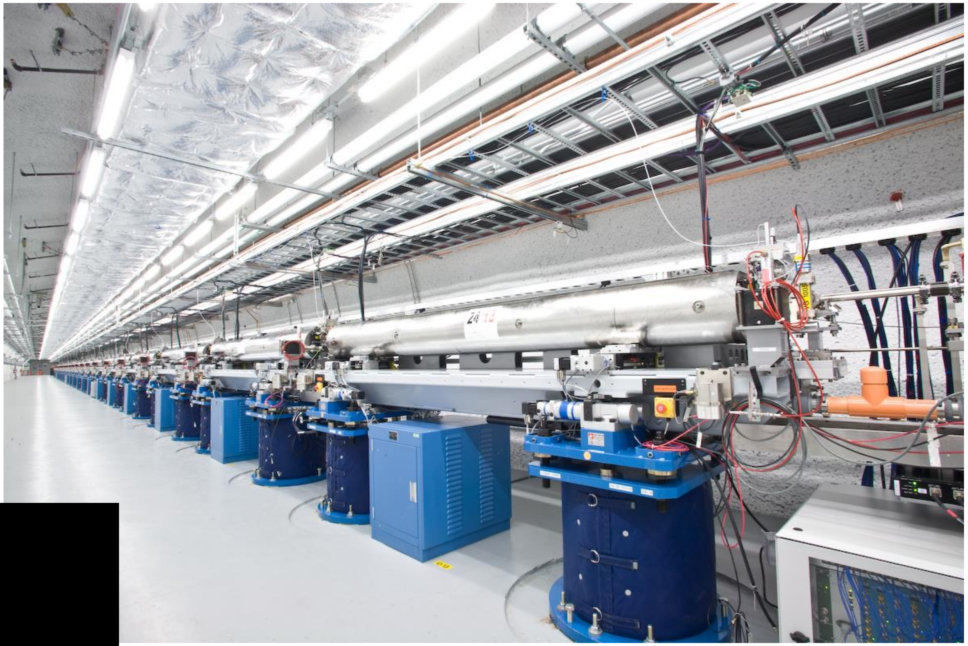


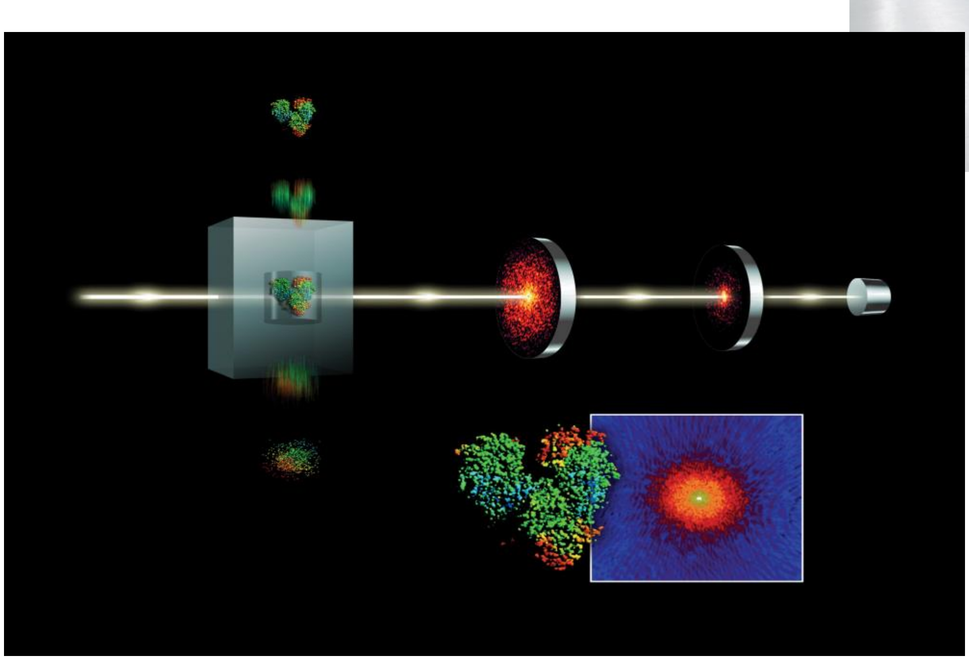


Per pulse readout of detectors and diagnostics is crucial

LCLS Parameters

X-Ray range	**250 to 11,300 eV**
Pulse length	**< 5 - 500 fs**
Pulse energy	**~ 4 mJ**
Repetition Rate	**120 Hz**

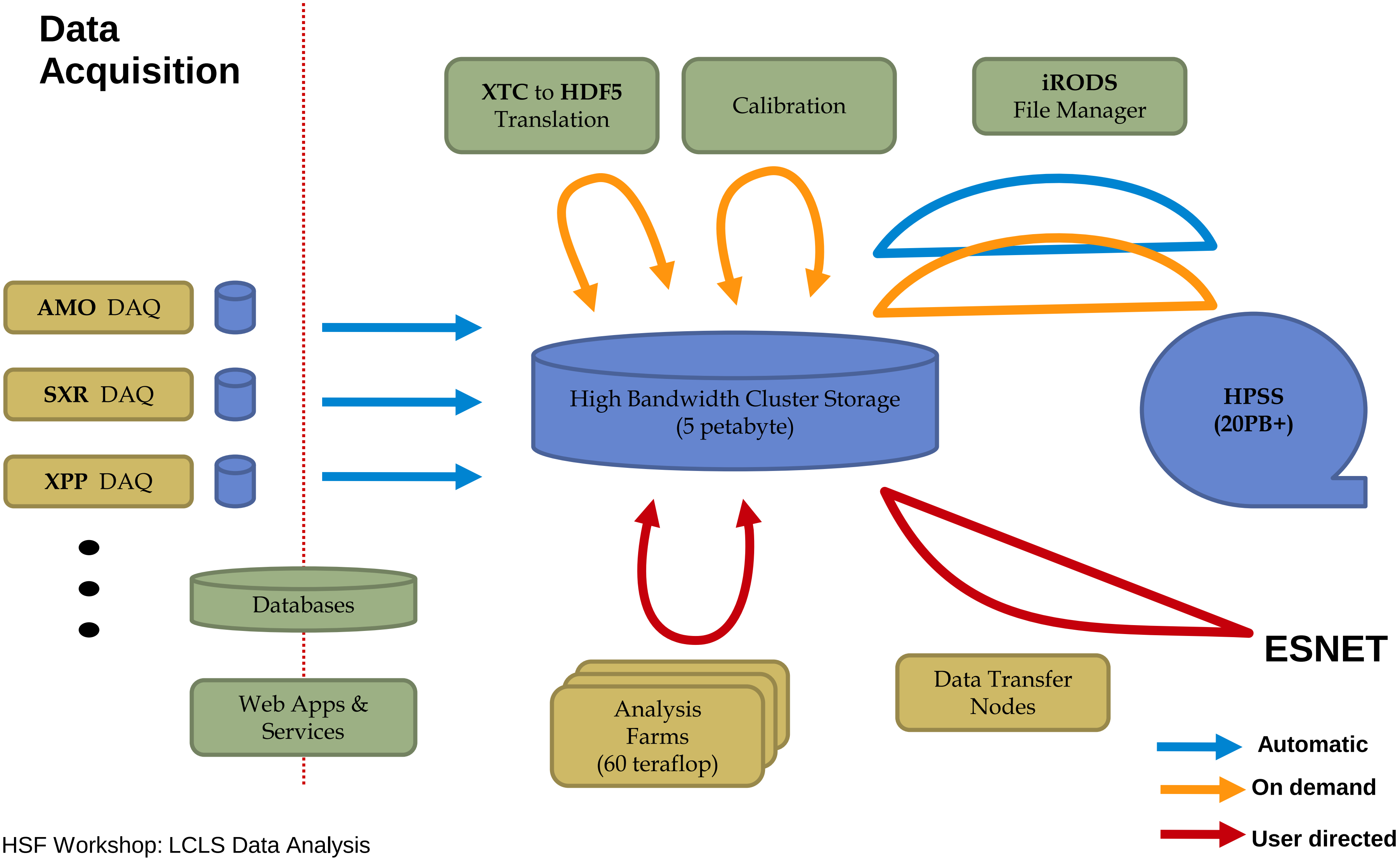




LCLS Data Infrastructure

- **DAQ systems dedicated per hutch, user analysis system shared across instruments**
- **Four storage layers**
  - Online cache (flash), fast-feedback (disk), medium term (disk), long term (tape)
  - Medium-term storage currently 5 petabytes
    - Each PB aggregated throughput of 12GB/sec
  - Long-term storage uses tape staging system in the SLAC central computing facilities
    - Can scale up to several petabytes
- **Science data files policies:**
  - Kept on disk for 2 years (quota enabled after 6 months), on tape for 10 years
  - Access to the data for each experiment granted only to members of that experiment
- **60 teraflop processing farm**

Data Systems Architecture



LCLS Data Management Framework

- **Data Management system handles all content-opaque operations**
  - Moves data across storage layers (online cache, fast-feedback, offline storage, tape)
  - Handles data policies (security, access, retention)
  - Handles DAQ generated data or data resulted from centralized processing (eg HDF5 translation, compression, filtering)
  - File catalog and tape operations are based on iRODS
  - File migration implemented as a collection of distributed services written primarily in Python
  - Using LSF for processing HDF5 translation services and other operations
- **Currently handling 11PB LCLS data, raw and user generated**
  - 5PB on disk, 6PB on tape
- **User accessible through LCLS web-portal (electronic logbook)**
  - Web front-end based on HTML5, CSS3, JavaScript, and a bunch of modern JavaScript tolkits/libraries
  - Server-side backend: RESTful Web services, mostly PHP and relevant libraries, Pylons (Python-based Web framework for some Web services), MySQL, LDAP and Apache

LCLS User Data Analysis

- **Main data analysis framework is `psana`**
  - Event-driven batch framework to parse the raw data
  - Allows mixing of python and C++ modules
  - Powerful, but, until recently, not widely adopted, threshold too high for many users
    - Many groups used `myana` (simple C++ program developed by DAQ group to parse the raw data), `Matlab`, `ami` (this is the the same framework used for on-the-fly data monitoring but run against data on disk), `cass` (originally developed for CAMP detector) and `cheetah` (CFEL)
- **Beside parsing the data, currently providing basic capabilities:**
  - Calibration modules
  - Modules for time-correlation analysis
  - Data browser
  - Peak finding algorithms
- **We are currently looking at two main projects in the data analysis arena:**
  - Develop advanced algorithms for LCLS users
  - Build an ecosystem for data analysis at FEL facilities

LCLS User Data Analysis (continued)

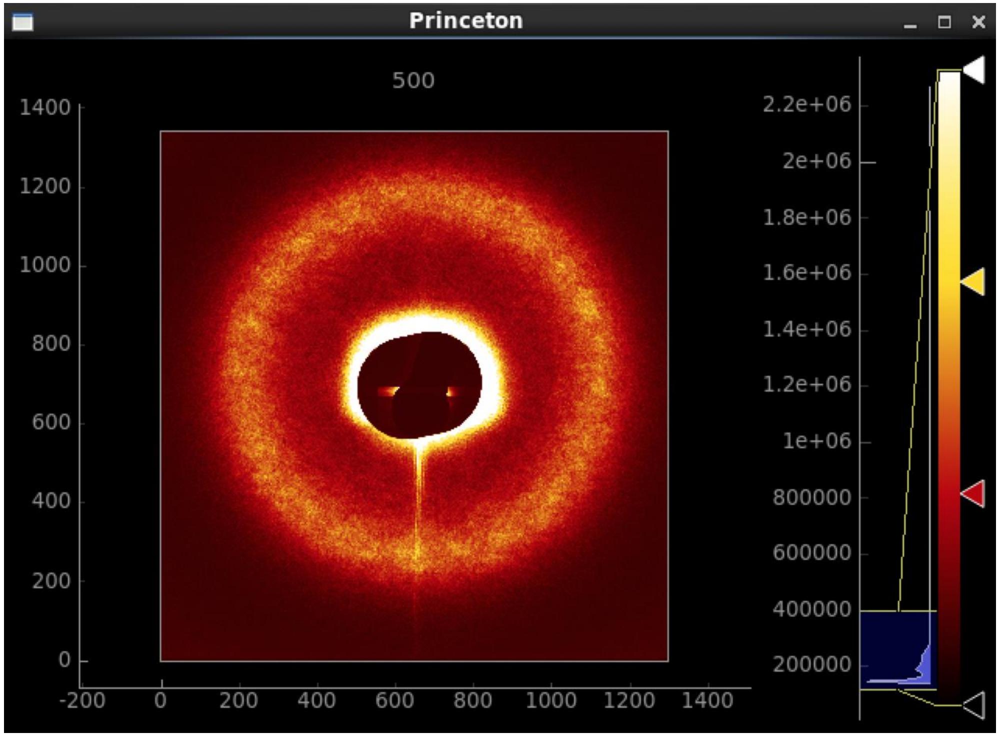
Developed python based interactive framework `ipsana` to complement the psana batch framework

Adoption of `psana` significantly increased after `ipsana` was introduced:

- Can write analysis code with simple python scripts
- All documentation on one page:

`https://confluence.slac.stanford.edu/display/PSDM/psana+-+Python+Script+Analysis+Manual`

- Can run the same simple scripts offline and online (with real-time plotting)
- Can analyze a run (online and offline) in parallel on hundreds of cores using MPI
- Many experiments have used this to analyze all 120Hz, online in real-time



Lesson Learned 1 or Why Vetoing Events for FEL Experiments Can Be Tricky

- **Very hard to implement effective trigger/veto system**
  - Not a technical/computing issue: the ability to veto events is already implemented in the system
  - Vetoing based on beam parameters not effective (most pulses are good)
  - Hard to get help from users in setting veto parameters which define event quality
    - Users themselves often don't know what these parameters or their thresholds should be
    - Users are usually very suspicious of anything which can filter data on-the-fly
- **Benefit of vetoing events based on the event data potentially very large for those experiments with low hit rate**
  - factor 10-100

Lesson Learned 2 or Why HEP Style Online-Offline is Not Enough

- **HEP style online/offline separation doesn't work**
  - The core online monitoring is not enough for many experiments
  - The skill level required to write on-the-fly analysis code is too high for most users
  - As a consequence some experiments felt they were flying blind
- **Critical to provide users the ability to run offline style code for fast feedback**
  - This was an issue for:
    - High data volume combined with low hit rate experiments: offline designed to keep up with DAQ only in average, not instantaneously; fast feedback nodes which look at subset of the data don't provide enough statistics
    - HDF5 based experiments: must wait for additional translation step