

# Evolution of the FLASH DAQ System

## Motivation, Concepts, Evolution and Experiences

Tim Wilksen - DESY

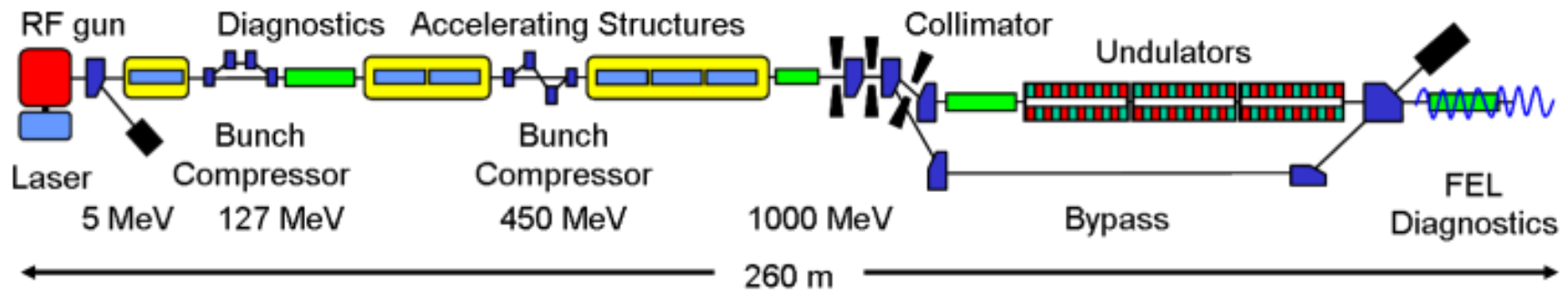
Evolution of the FLASH DAQ System

ICALEPCS 2009

Kobe, 10/12/09

# Free-Electron Laser at Hamburg (FLASH)

- Successor to TESLA Test Facility (TTF).
- User facility for photon science community providing a pulsed light source in the extreme ultraviolet and soft X-ray regime since 2005.
- Test bed for exploring and testing new superconducting accelerator technologies for the European XFEL and ILC.
- Currently being upgraded to sFLASH – restart mid next year.



Electron Energy: 0.4 ... 1 GeV  
Photon wavelength: 6 ... 32 nm  
Bunch charge: 1 ... 3 nC

Bunch repetition rate: 5 ... 10 Hz  
Bunch frequency: 1 ... 3 MHz  
Bunch train length: 800  $\mu$ s

## **Two-fold operation and usage requires a versatile control system.**

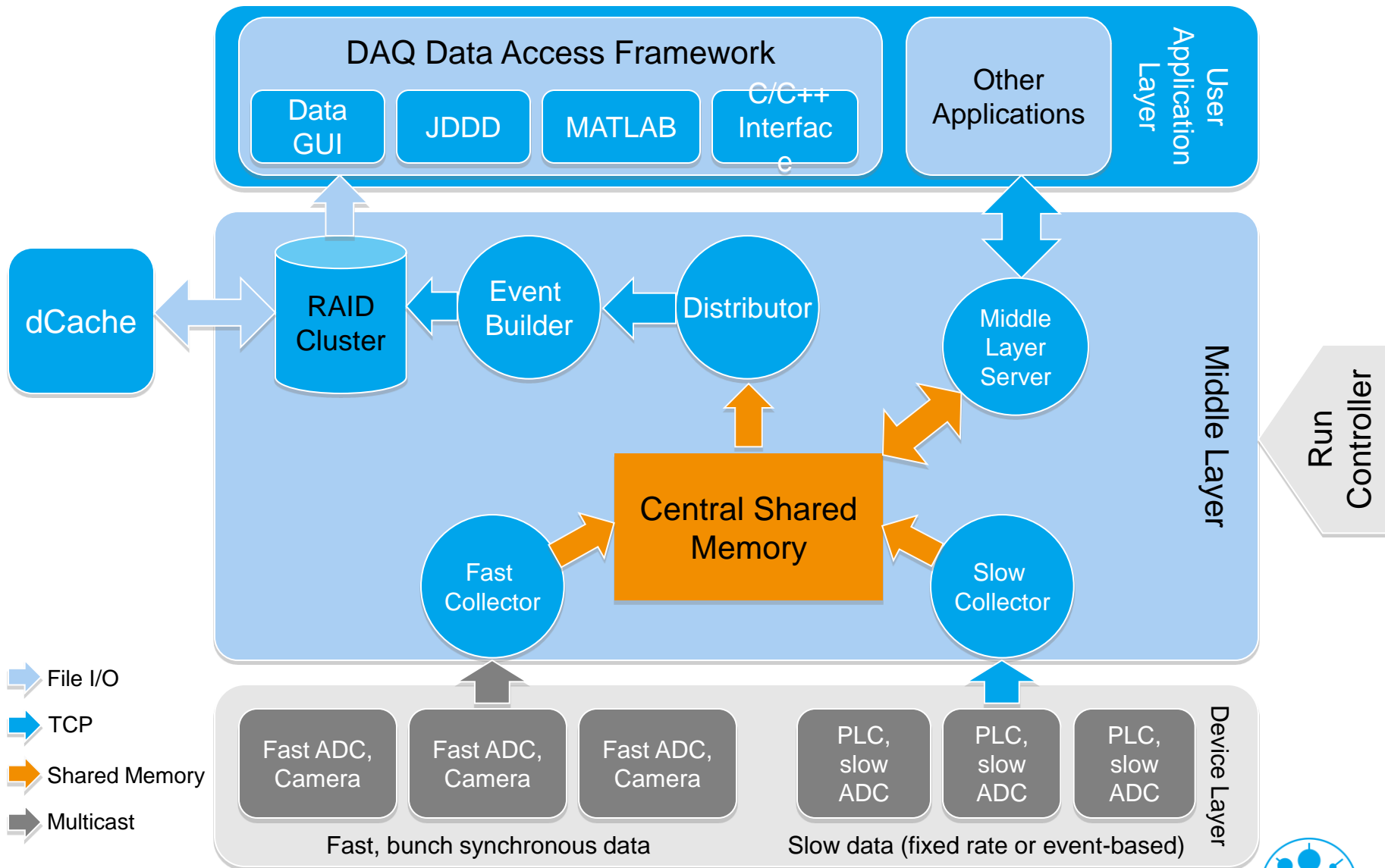
- Integrate a DAQ into DOOCS-based control system to support both operation modes.
- Store all relevant information needed for operations, R&D work and photon beam experiments.
- Combine accelerator and experiment information to enable data analysis across machine and experiment.
- Handle large number of accelerator control and diagnostic devices.
- Allow for frequent modifications of controls, diagnostics and user experiments.
- Provide tools for online and offline data access and processing.



- > **Fast, bunch-synchronous data collected from ADCs via multicast.**
  - *Slow data* is retrieved separately from slowly varying sources with fixed update rates.
- > **Bunch-synchronized event identification for all data.**
  - Timing system distributes bunch-synchronous event number matching macro pulse or shot.
- > **Centralized shared memory stores all synchronized data.**
  - A buffer manager using up to 32 Gbyte of shared memory keeps short history of synchronized data for assembling the full event record.
  - *Middle layer servers* have full read and write access to the shared memory.
- > **Scalable and extensible architecture.**
  - Distributed, concurrently running DAQ instances.
  - *Streams* for separating raw data by device group or experiment.
  - HEP-style *run control* and *configuration database* for tagging specific machine or experiment setups.



# Basic Concepts - Data Flow



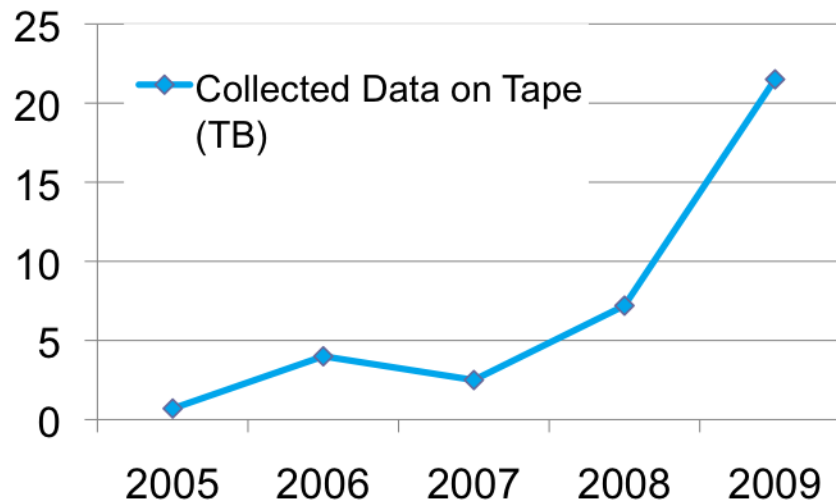
# Evolution – Hardware and Data

## > FLASH is operating since August 2005.

- Basic DAQ implementation was present and running on day 1.
- Upgraded DAQ server cluster – now two work horses (Sun E 2900 and M 4000 – each 32 Gbyte shared memory) and two Sun storage nodes (RAID 20 TB + 40 TB).

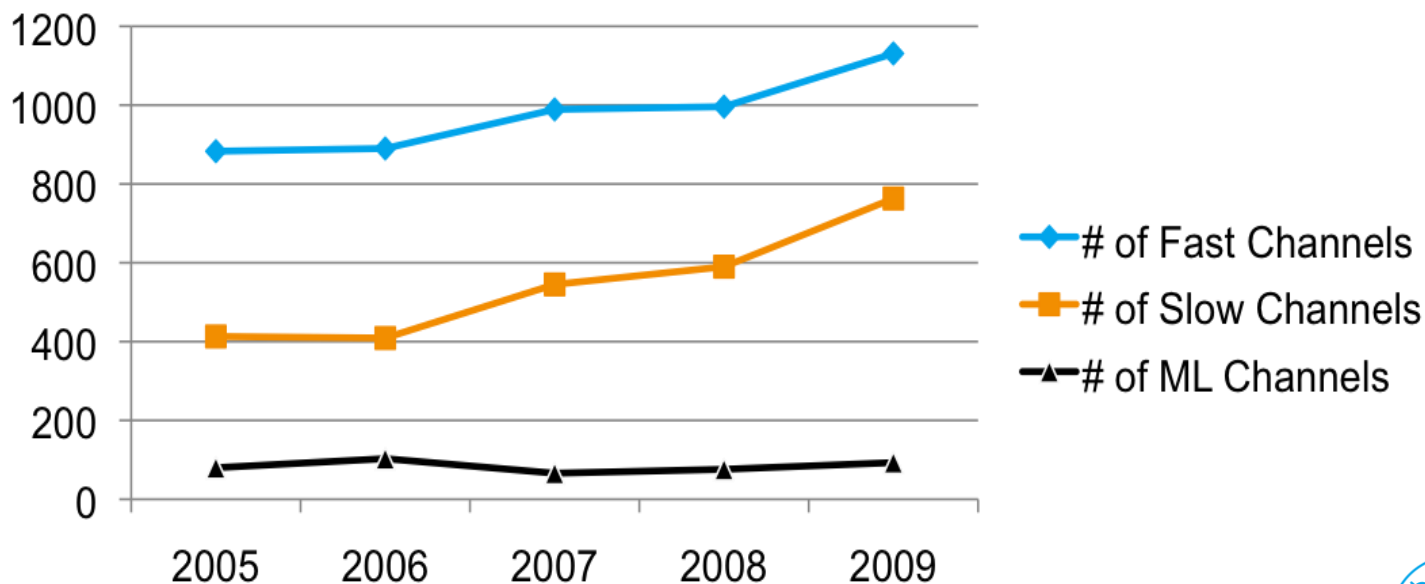
## > DAQ produces on average 1 Tbyte/day

- Most accelerator data not taped but kept for 2 – 4 weeks on disk (migration).
- Photon diagnostics, experiment and R&D data kept on disk and taped (dCache).
- Rate OK right now – might require filtering and dedicated run configuration in future.

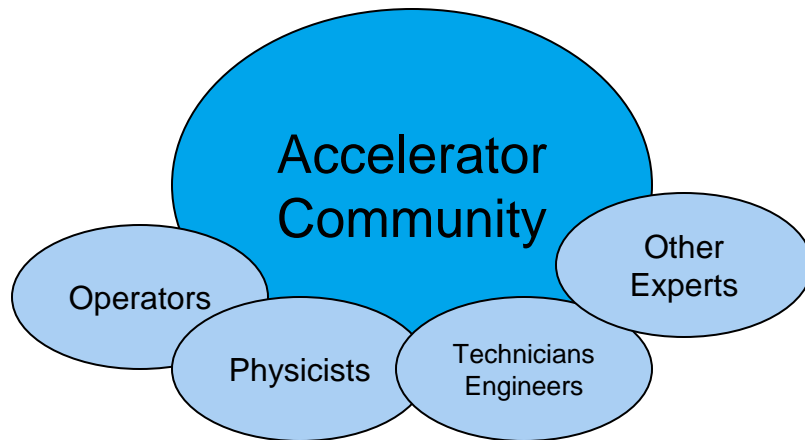


## > Scalability and Extensibility

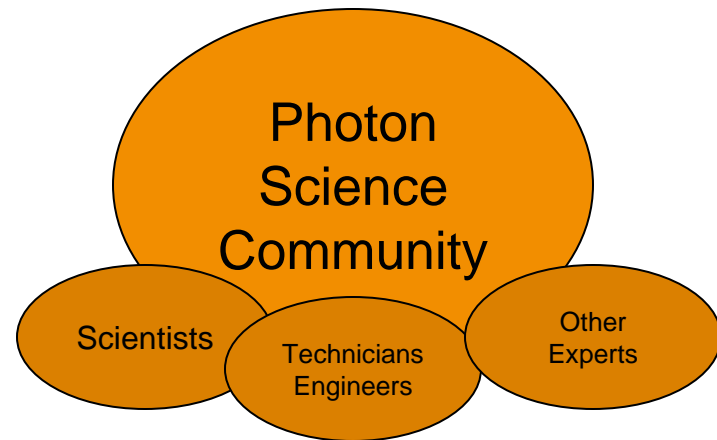
- Relatively simple to add another complete DAQ instance: script for cloning DAQ instance, then add run control configuration with Java-based GUI and adjust.
- Instances share memory, I/O and CPU resources otherwise completely separated.
- Four concurrently DAQ instances are running, seven are ready to run.
- Easy to include new (supported) devices or individual channels via run control configuration database.



# Experiences – Tools = Acceptance?



- Interested, not yet explored full DAQ potential.
- ILC R&D utilized complete DAQ in '09.
- Want ready-to-go applications or known framework for writing some (MATLAB).



- Interested from the beginning using it.
- Work habits and culture matches more HEP-style design of DAQ.
- Could make use of initial ROOT-based tools.

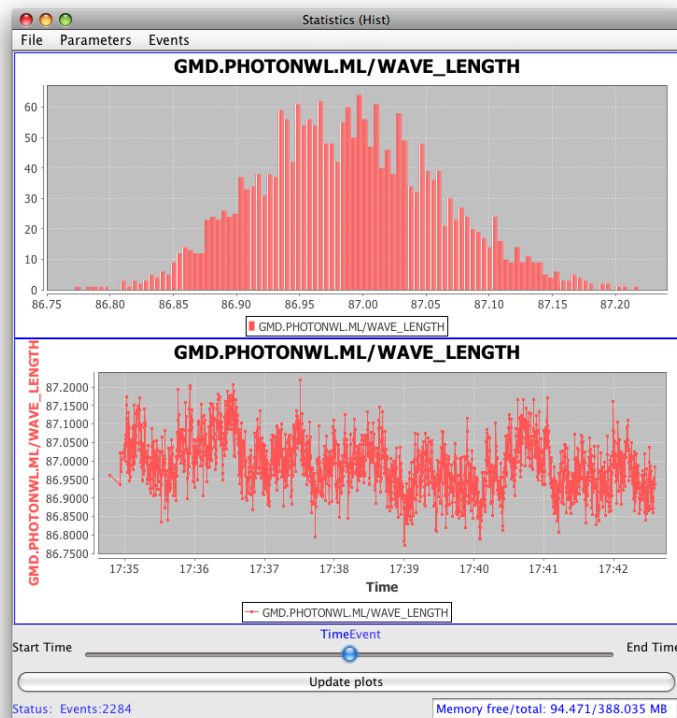
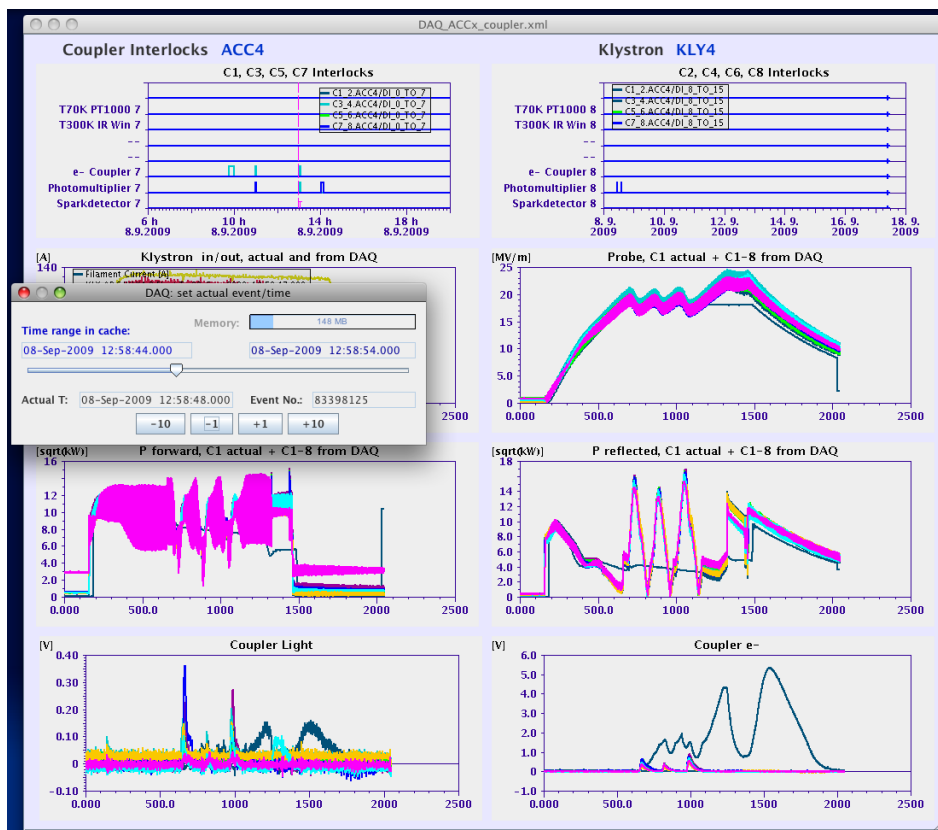
- 
- Started with ROOT-based applications. Lacked I/O performance.
  - Now more Java-based tools using own I/O-interface. Better performance.
  - Added and enhanced MATLAB support.



# Experience - Tools

## ➤ Data Retrieval – Browsing large amounts of data

- Streams and run configuration helped dealing with Tbytes of data.
- Data server cluster provides access to all data on disk via TCP. Interfaces to MATLAB, JDDD, Java-based browser and C/C++ library. Used a lot by users.



FLASH Java-based DAQ GUI



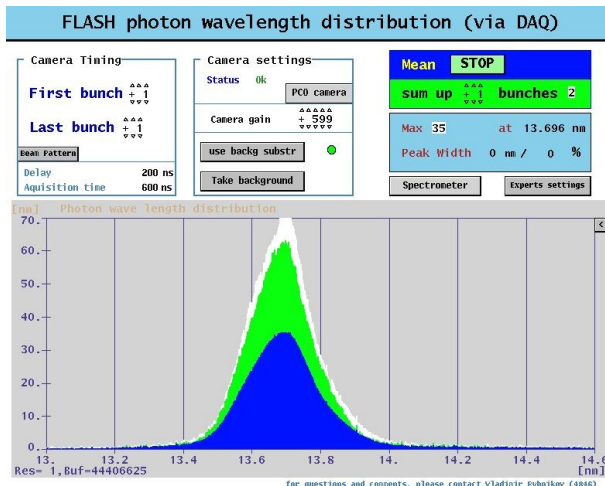
# Experiences – File Format

## > File Format

- Initially based on ROOT: good compression (GZIP), highly flexible C++ analysis framework in HEP-style. But poor performance with tree-based layout. No threading for event writer processes running at 8 Mbytes/s maximum.
- Looked at other solutions, ended up with own, ZLIB-based implementation. Performance much better, multithreaded, writing at ~58 Mbytes/s via NFS, reading an event takes 200 – 700  $\mu$ s, files are available after 1 – 2 min.

## > Middle layer servers became essential for operation over time.

- Uses MATLAB or C/C++ for computations with access to full data in shared memory.
- Sufficient response time for many operational tasks and some automation purposes.



Examples: Energy measurement, photon wavelength and energy measurement, gas monitor detector server, LLRF server, orbit server, quench detection server, ...

Added new DOOCS data classes for direct shared memory access, currently being tested for faster performance.



# Conclusions

- Powerful concept and proven architecture in general.
- Reliably running since 2005 for accelerator and photon science community and its experiments.
- Recording accelerator data continuously and providing middle layer server for operations. DAQ is crucial now for FLASH operations.
- Basic concepts seem to be right, though data storage and applications need more evaluation and experience. Essential for future projects (European XFEL).
- Used with great success for ILC R&D work in 2008 and 2009 (high beam current run at up to 3 MHz bunch frequency) – about 18 Tbyte of recorded data.

