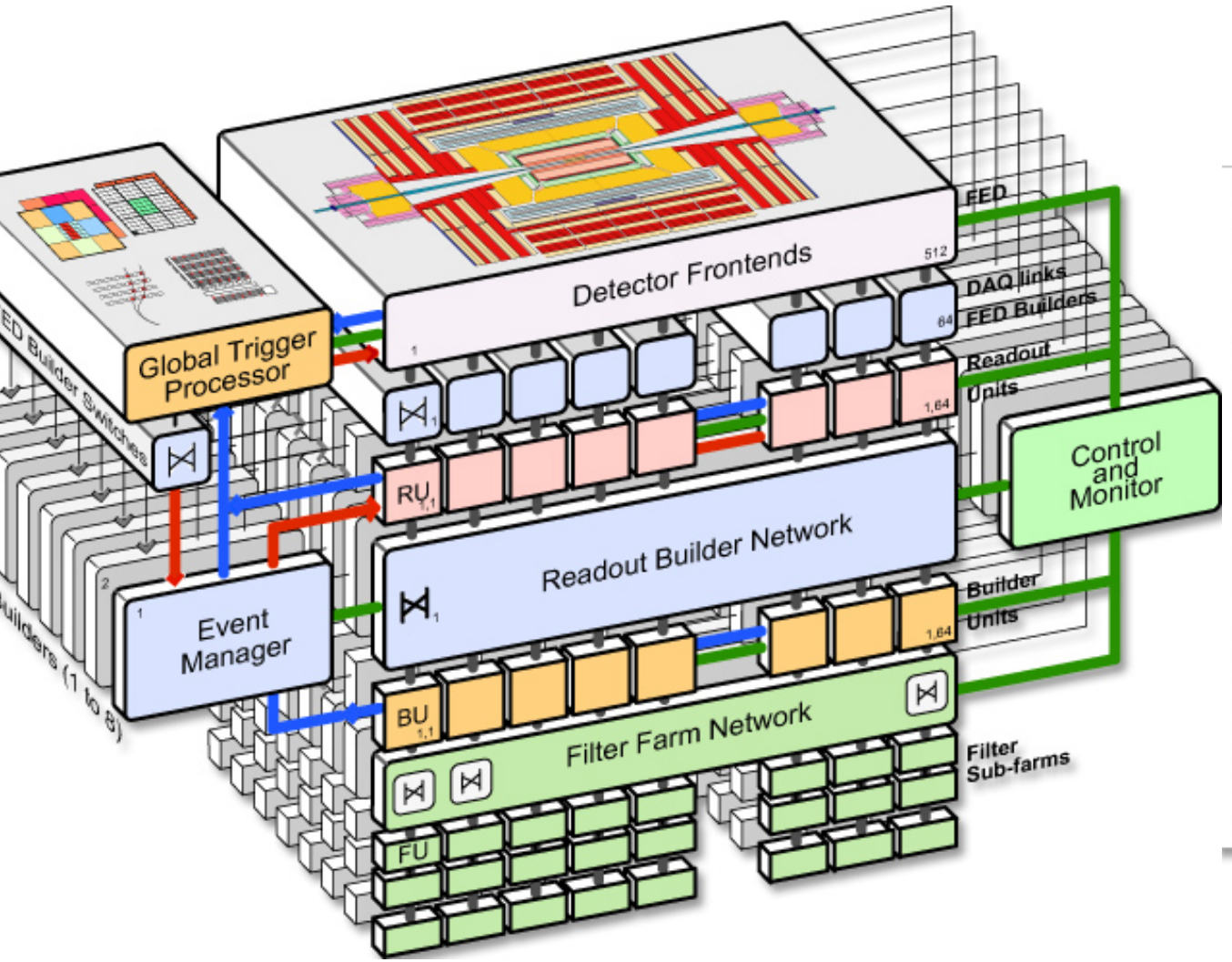
CMS DAQ Architecture



Acronyms

BCN	Builder Control Network
BDN	Builder Data Network
BM	Builder Manager
BU	Builder Unit
CSN	Computing Service Network
DCS	Detector Control System
DCN	Detector Control Network
DSN	DAQ Service Network
D2S	Data to Surface
EVM	Event Manager
FB	FED Builder
FEC	Front-End Controller
FED	Front-End Driver
FES	Front-End System
FFN	Filter Farm Network
FRL	Front-End Readout Link
FS	Filter Subfarm
GTP	Global Trigger Processor
LV1	Level-1 Trigger Processor
RTP	Regional Trigger Processor
RM	Readout Manager
RCN	Readout Control Network
RCMS	Run Control and Monitor System
RU	Readout Unit
TPG	Trigger Primitive Generator
TTC	Timing, Trigger and Control
sTTS	synchronous Trigger Throttle System
aTTS	asynchronous Trigger Throttle Syster

Design Parameters

- The Filter Farm runs High Level Trigger (HLT) algorithms to select CMS events at the full Level 1 trigger accept rate
  - Maximum average Level 1 trigger accept rate: 100 kHz
    - Reached in two stages: a startup phase of two years at 50 kHz ("low-luminosity"), followed by a ramp-up to the nominal rate ("high luminosity")
  - Maximum acceptable average output rate ~100 Hz
  - Large number of sources (nBU ≈ 512)
  - HLT: full fledged reconstruction applications making use of a large experiment-specific code base
- Very high data rates (100 kHz of 1 MB events) coupled with Event Builder topology and implementation of HLT, dictate the choice to place the Filter Farm at the CMS experiment itself.
- DAQ architecture calls for a farm consisting of many "independent" subfarms which can be deployed in stages
- Application control and configuration must be accessible through the same tools used to control DAQ ("Run Control")

Filter Farm baseline (1 RU builder)

Event Manager

Readout Unit Builder (64x64)

Controls

Farm Manager

BU

Builder Units

Filter Farm Networks

FU

Filter Sub-farms

Filter SubFarm

Filter Data Network (FDN)

FU FU FU - - - - - - SM

"Worker nodes"

Filter Control Network (FCN)

"Head node

Farm Manager (Run Control)

CPU Requirements

Table 15-25 Summary of CPU time required for the selection of each physics objects in the HLT. The CPU figures refer to a 1 GHz Intel Pentium-III CPU.

Physics Object	CPU time per Level-1 event (ms)	Level-1 Trigger rate (kHz)	Total CPU time (s)
Electrons/photons	160	4.3	688
Muons	710	3.6	2556
Taus	130	3.0	390
Jets and E_T^{miss}	50	3.4	170
Electron + Jet	165	0.8	132
B-jets	300	0.5	150

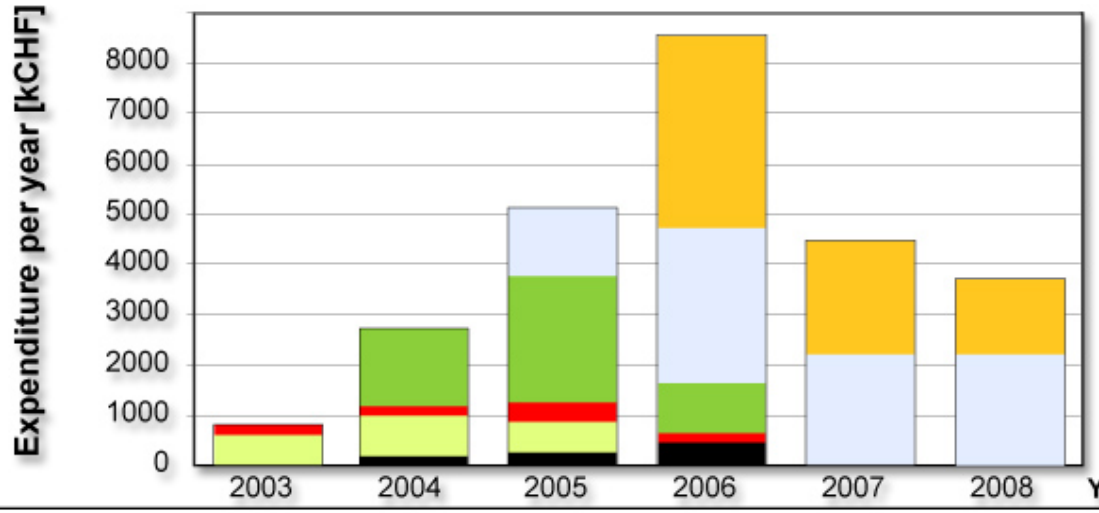
- Use a safety factor 3 in L1 budget allocation (16/50 kHz, 33/100 kHz)
- E.g. at startup: 4092 1GHz PIII to process "physics" input rate
- Use 1GHz PIII ≈ 41 SI95: @50 kHz ⇒ 6×10^5 SI95
  - Extrapolate to 2007 using Moore's law: x 8 increase in CPU power (conservative)
  - ~2000 CPU (or 1000 dual-cpu boxes) at startup, processing 1 event every 40 ms on the average
  - At high luminosity, not only the input rate, but the event complexity (and thus the processing time) increases. Assume the latter is compensated by increase in CPU power per unit, thus leading to a final figure of ~2000 dual-CPU boxes

Hardware

- Staged deployment
  - Commodity PCs
  - Stay multi vendor (multi platform)
  - Fast turnover
    - Btw, translates into need for quick integration ⇒ rapid evolution of operating system to support new hardware (see later on configuration)
  - Physical and logistic constraints (see talk by A.Racz)
- Form factor for worker nodes: at the moment not many choices:
  - Rack mount 1U boxes
  - Blades
- Head nodes with redundant filesystems (RAID etc.)
- Inexpensive commodity GE switches

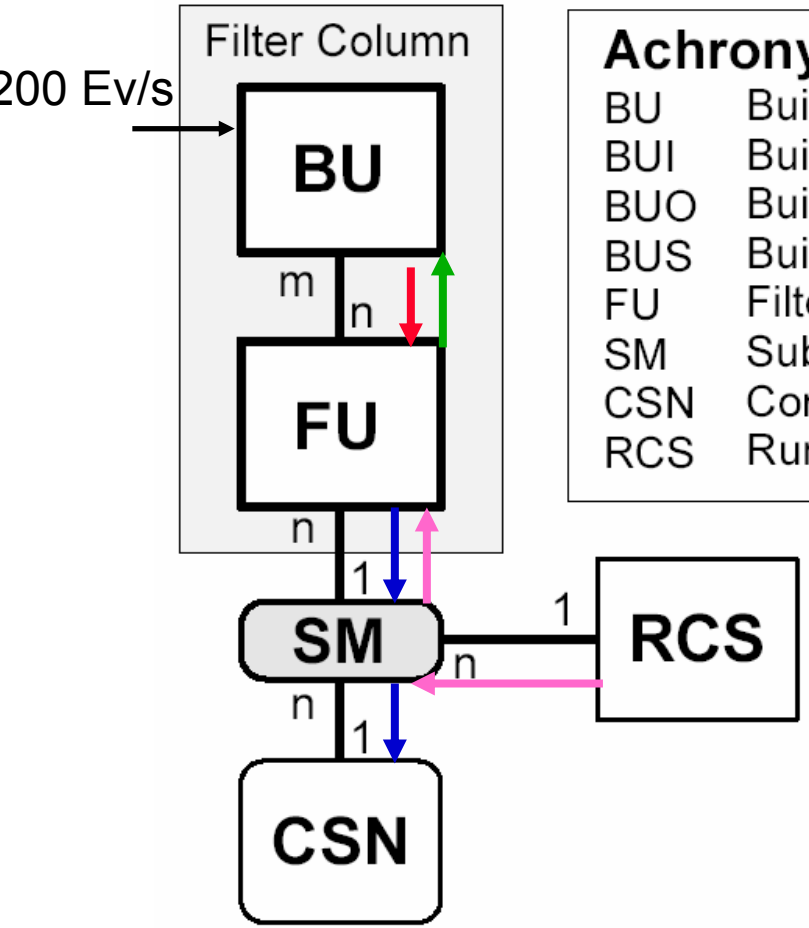
EvF Deployment

- Design figures from above drive the expenditure profile
- Main goal: deadtimeless operation @ nominal L1 max rate



Expenditures	468	2906	4904	8776	4596	3722	TOTALs
Event Filter	0	0	0	4372	2623	1749	**8744**
Readout Builder	0	0	1184	2763	1973	1973	**7893**
Data to Surface	0	1511	2519	1008	0	0	**5038**
Detector Controls	92	276	368	184	0	0	**920**
Preseries	376	939	564	0	0	0	**1879**
DAQ Integration	0	180	269	449	0	0	**898**
			Level-1 maximum rates		50kHz	100kHz	**25372**

Subfarm data paths

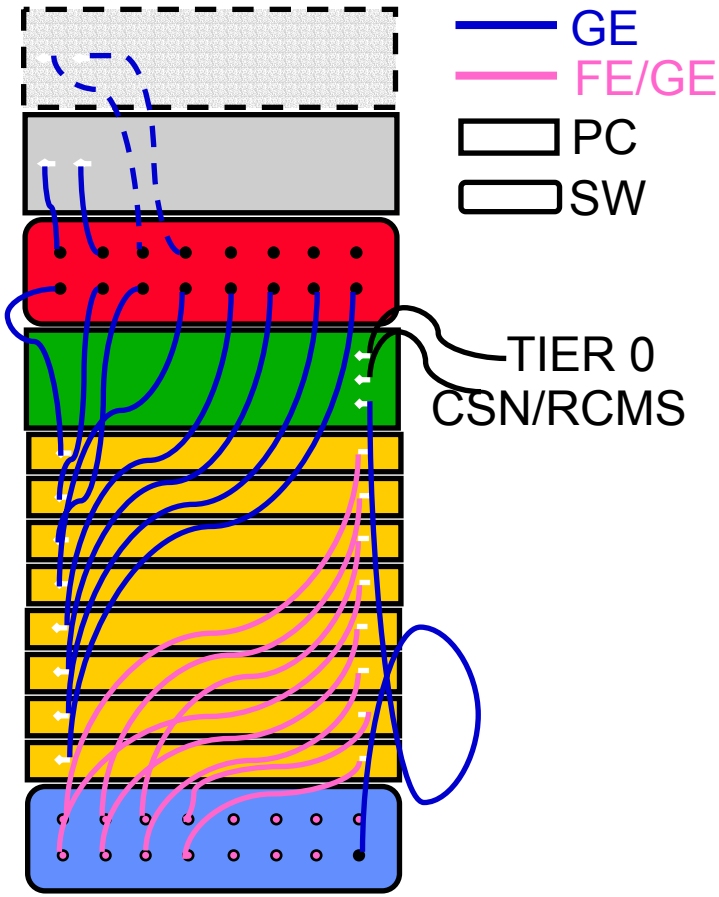


Achronyms

BU	Builder Unit
BUI	Builder Unit Input
BUO	Builder Unit Output
BUS	Builder Unit Supervisor
FU	Filter Unit
SM	SubFarm Manager
CSN	Computing Service Network
RCS	Run Control System

- Requests sent to BU
  - Allocate event
  - Collect data
  - Discard event
- Event data sent to FU
- Events sent to storage
- Control messages

Subfarm Network Topology



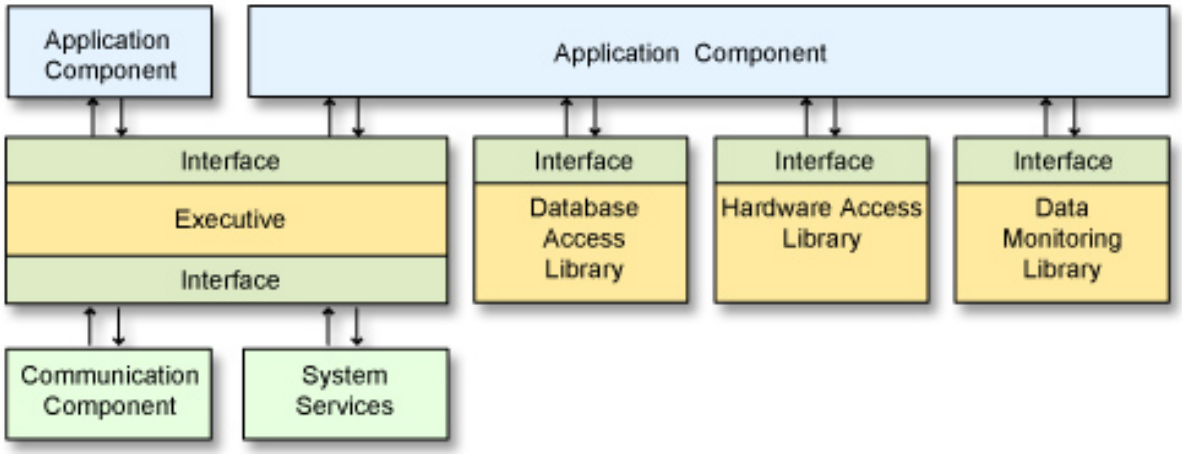
Cluster Management

- Management systems for ~2000 nodes Linux farms are currently in use
- However, availability constraints very tight for the farm
  - Can't lose two days of beam to install redhat 23.5.1.3beta
- Also need frequent updates of large application code base (reconstruction and supporting products)
  - Need a mixed installation/update policy
  - Need tools to propagate and track software updates without reinstallation
  - Need tools for file distribution and filesystem synchronization
    - With proper scaling behavior
- Significant progress in the last two-three years
  - Decided to look into some product
- Continue looking into developments and surveys of Grid tools for large fabric management

Cluster Management Tools

- Currently investigating two opensource products:
  - OSCAR http://oscar.sourceforge.net
  - NPACI ROCKS http://rocks.npaci.edu
    - Direct interaction with individual nodes only at startup (but needs human intervention) (PXE, SystemImager, kickstart)
    - Rapid reconfiguration by wiping/reinstalling the full system partition
    - Both offer a rich set of tools for system and software tracking and common system management tasks on large clusters operated on a private network (MPI, PBS etc.)
    - Both use Ganglia for cluster monitoring (http://ganglia.sourceforge.net)
    - Will need some customization to meet requirements
- Put them to test on current setups
  - Small scale test stands (10÷100) nodes
  - Scaling tests (both said to support master/slave head node configuration)

Application Architecture



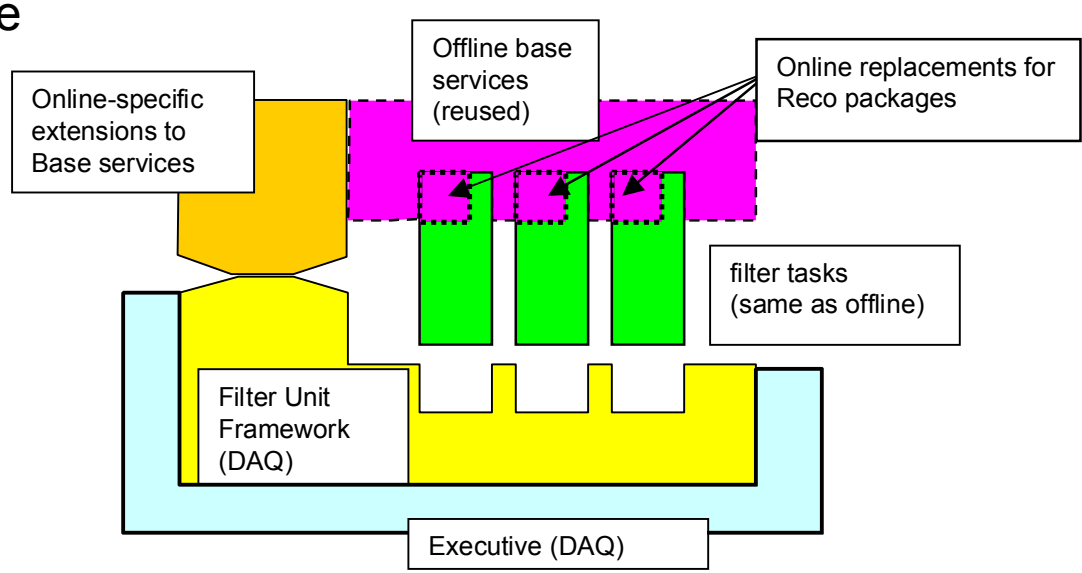
DAQ software architecture:
Layered middleware
Peer-to-peer communication
Asynchronous message passing
Event-driven procedure activation

An early decision was to use same building blocks for online and offline

- Avoid intrusion in reconstruction code: "plugin" extensions to offline framework transfer control of relevant parts to online

Online-specific services

- Raw Data access and event loop
- Parameters and configuration
- Monitoring



Controlling the Application

- Configuration of DAQ components
  - Directly managed by Run Control services
- Configuration of reconstruction and selection algorithms
  - What was once the “Trigger Table”
  - Consists of:
    - Code (which version of which library etc.)
    - Cuts and trigger logic (an electron AND a muon etc.)
    - Parameters
- Need to adapt to changing beam and detector conditions without introducing deadtime
- Therefore, complete traceability of conditions a must
  - See later on Calibrations and Run Conditions

Monitors and Alarms

- Monitor: anything which needs action within human reaction time, no acknowledge
  - DAQ components monitoring
    - E.g. data flow
  - “Physics” Monitoring, i.e. data quality control
    - Monitor information collected in the worker nodes and updated periodically to the head nodes
    - Monitor data that needs processing shipped to the appropriate client (e.g. event display)
- Alarm: something which may require automatic action and requires acknowledge
  - A “bell” sounds, operator intervention or operator attention requested
  - Alarms can be masked (e.g. known problem in one subdetector)

Physics Monitor

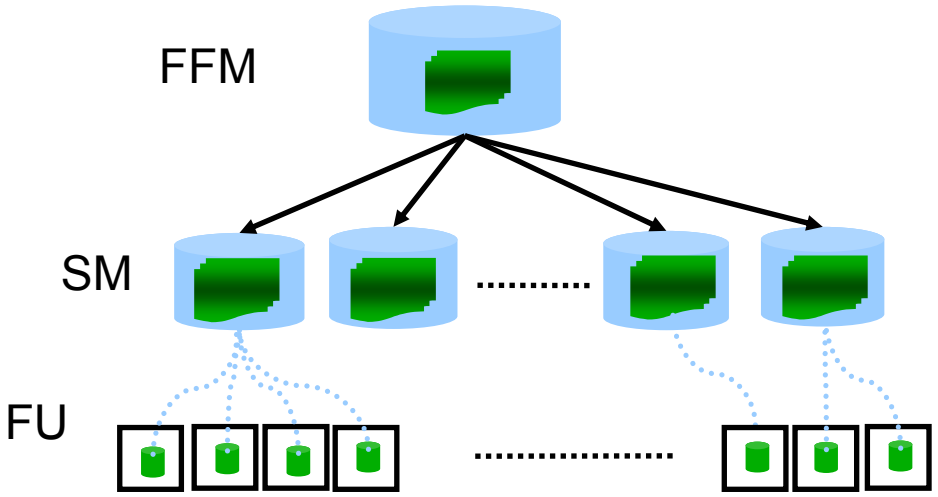
- Publish/Update application parameters to make them accessible to external clients
  - Monitor complex data (histograms etc. up to “full event”) collected in the worker nodes
    - Application backend, cpu usage etc.
    - Streaming
  - Transfer and collation of information
    - Bandwidth issues
    - Scaling of load on head nodes
- Two prototype implementations:
  - AIDA/based
    - AIDA3 “standard”
    - XML streaming
    - ASCII representation (data flow issues/compression)
  - root
    - Compact data stream (with built-in compression)
    - Filesystem-like data handling (folders, directories, etc.)
    - Proprietary streaming (and sockets, threads etc.)

Run Conditions and Calibrations

- Affect the ability of the application to correctly convert data from detector electronics into physical quantities
- Online farm must update calibration constants ONLY if they reduce CMS efficiency to collect interesting events or to keep the output rate within the target maximum rate
  - Define to what level we want to update CC (e.g. when efficiency or purity drop by > x%?, when rate grows by > y%)
  - Update strategy: guarantee consistency, avoid overloading of central services, avoid introducing deadtime (more on next slide)

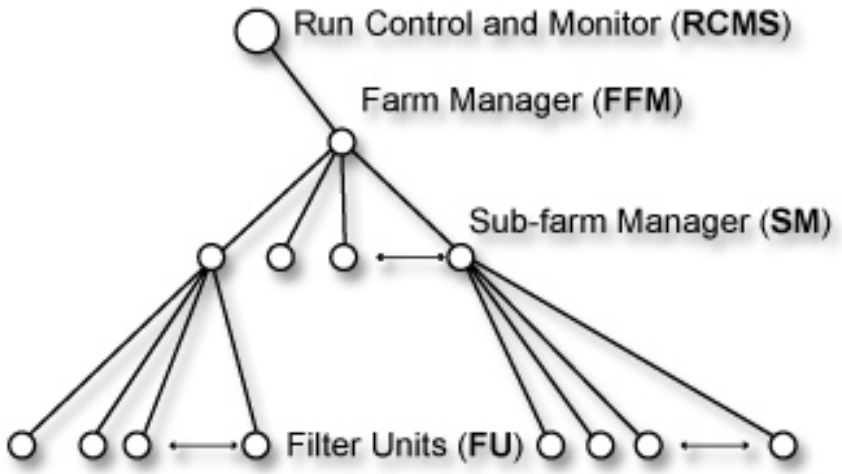
Dealing with changing conditions

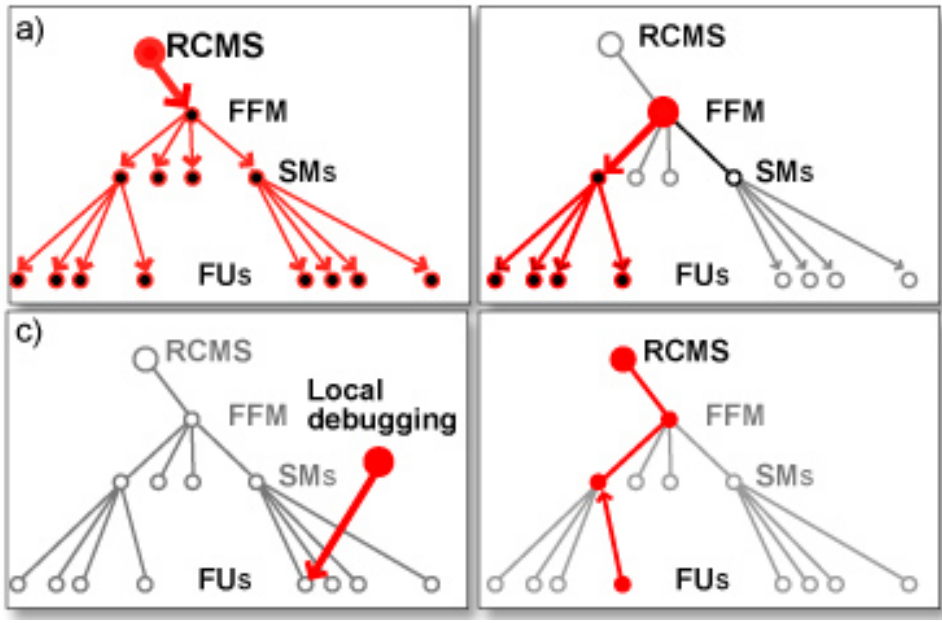
- Calibration & Run Conditions Distribution
  - Central server decides (or is instructed to) update calibrations
  - Head nodes prepare local copy



  - When instructed by Run Control, worker nodes preload new calibrations
  - New calibrations come into effect
- Traceability of Run Conditions and Calibrations
  - Mirroring/Replica of DB contents on SM, distribution, link to offline
  - Conditions key (e.g. use time stamp): synchronization issues across and among large clusters

Hierarchical Structure

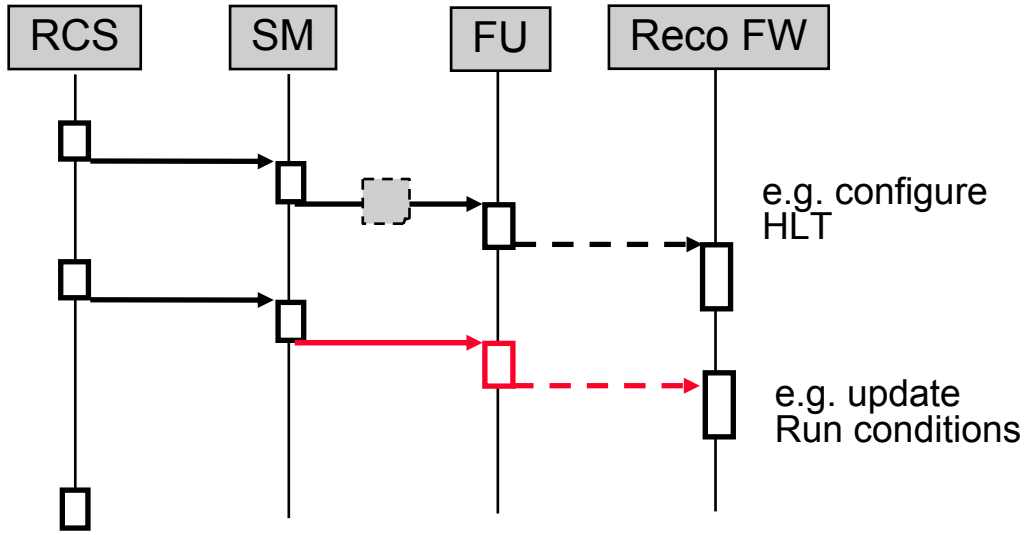




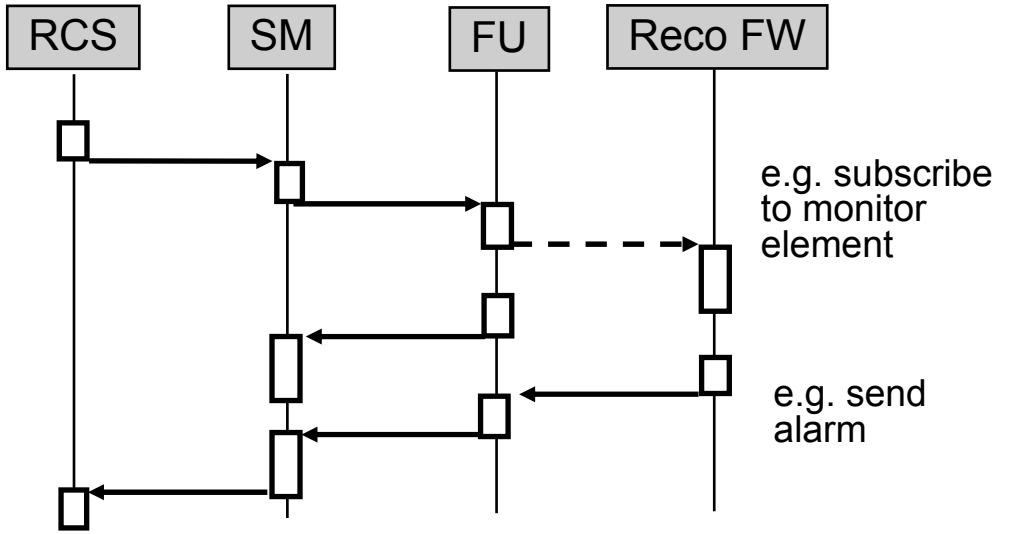
- Ensure good scaling behavior
- Worker nodes don't interact with other actors directly
- Access to external resources managed in orderly fashion

Data Paths

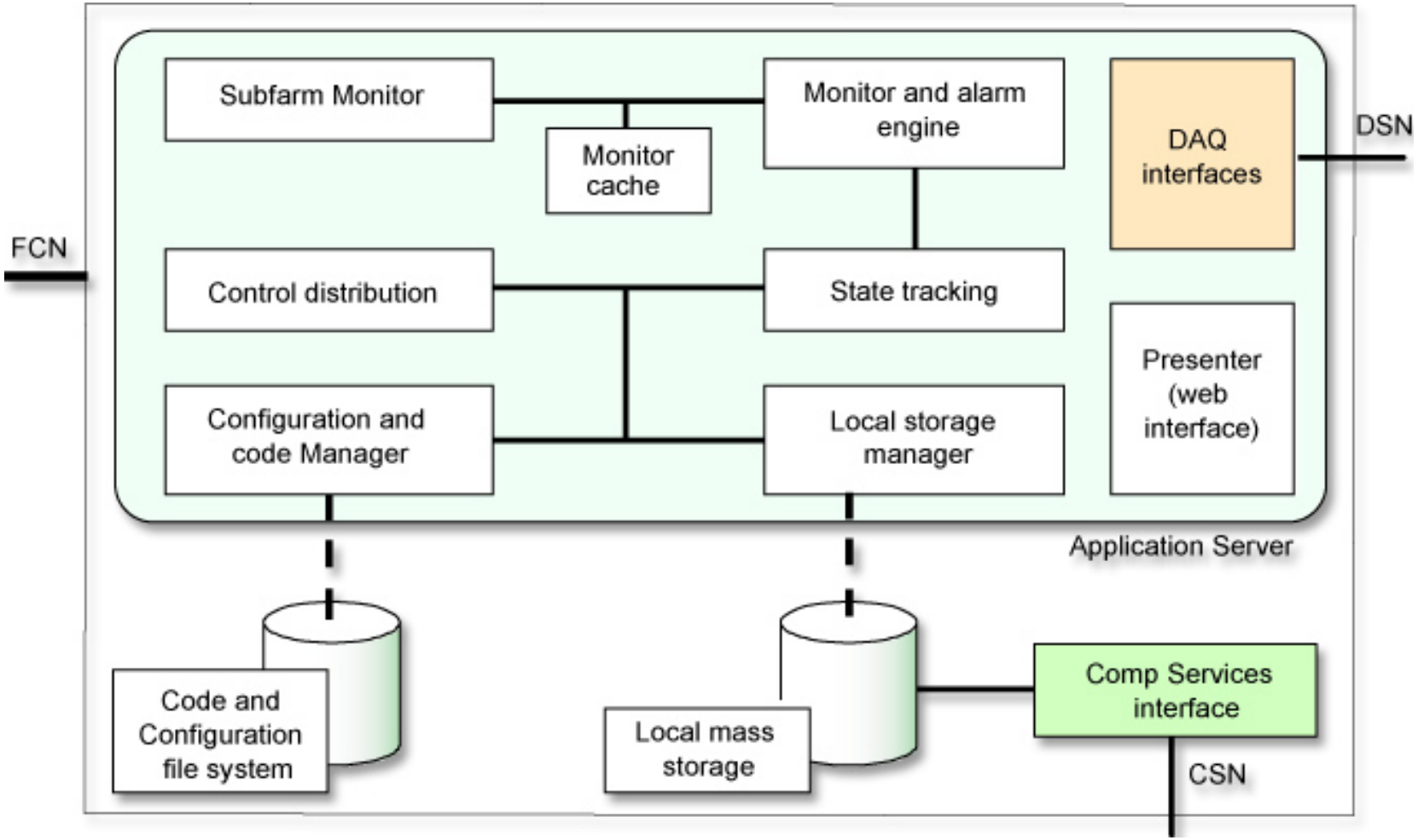
- Configuration and control
  - DAQ protocols (XDAQ)
    - SOAP message with callback
  - Control Network (FCN)
- Integration with System Management tools



- Monitoring
  - DAQ protocols (XDAQ)
    - SOAP message with callback
  - Control Network (FCN)
  - Special protocol for physics monitor data



Subfarm Manager (head node)



- **Plus typical cluster management tools**

Output Data Management: Local Buffering

- A necessary tool, when commissioning detector and DAQ
  - Guarantee local storage of detector data for tests, debugging, calibration etc.
- A possible working mode to minimize coupling with computing services
- A safety measure, in case link to CCC down
  - Guarantee 24 hrs worth of data taking with no contact to Tier0: emergency standalone system (assume 1.5MB/ev @ 100Hz)
    - Estimated in the order of 13 TB
- Three working modes:
  - No Buffering: events are shipped to Tier 0 by the head nodes as they come (in physical or logical streams): tight coupling between head nodes or switchboard and remote servers at CCC
  - Temporary mirroring: events are buffered on the head nodes while being shipped to tier0 and removed as soon as storage is acknowledged by the remote tier0 server
  - Short term storage: events are stored locally and files are shipped by the head nodes to tier0 when they reach a certain size (corresponding to order 1hr of data taking)
- Requires file catalogs and mass storage management, RAID, etc.

Application Quality Control

Application quality is critical in an environment where unavailability means deadtime of the experiment

- Define a set of coding guidelines to deal with critical quality issues
- Provide profiling and validation tools for early testing
  - Leak detection
  - CPU and memory profiling
  - Fault injection
- Define a validation and burn-in sequence to ensure stability and reliability of production application that make it to the farm
  - Offline validation on production data
  - Online test stand validation using Raw Data playback
  - Online burn-in in test partition (mirror mode)

Summary

- The Filter Farm basic architectural and management design choices are driven by well known and specific requirements
  - Guarantee deadtimeless operation at nominal L1 output rate
  - Get input from DAQ Event Builder
  - Use online control and monitoring data paths
  - Guarantee complete traceability of run conditions on an event by event basis
- Still a lot to learn as details are filled
- Some commonalities with Tier 0
  - Mainly on boundary between the two “worlds” of offline and online
  - Depending on certain strategic choices