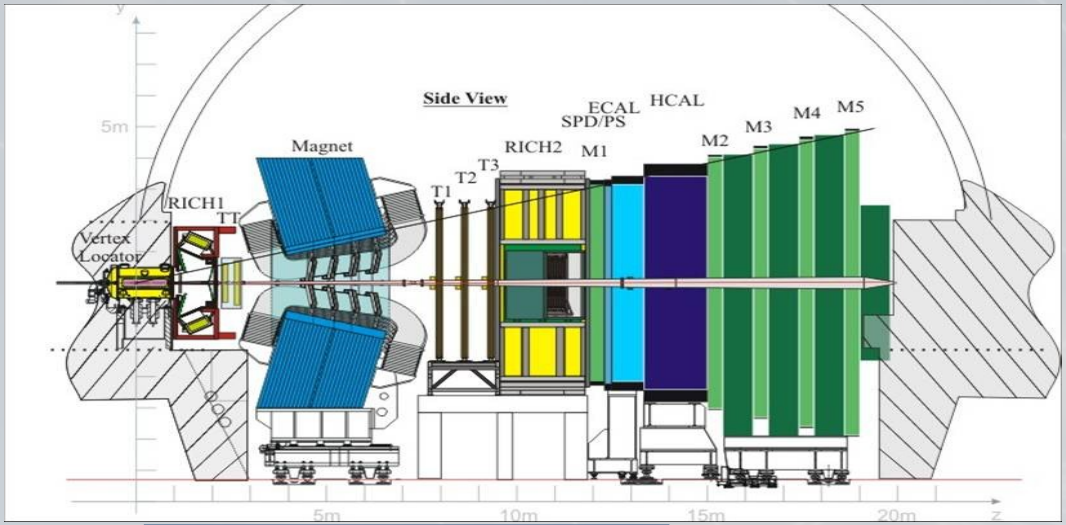
The LHCb Data Aquisition and High Level Trigger Architecture

Overview of the most important changes after LS1

- New HLT architecture – technical aspects
- Online calibrations
- Miscellanea

M.Frank, C.Gaspar, B.Jost, N.Neufeld
CERN / LHCb

LHCb Online Computing in Numbers



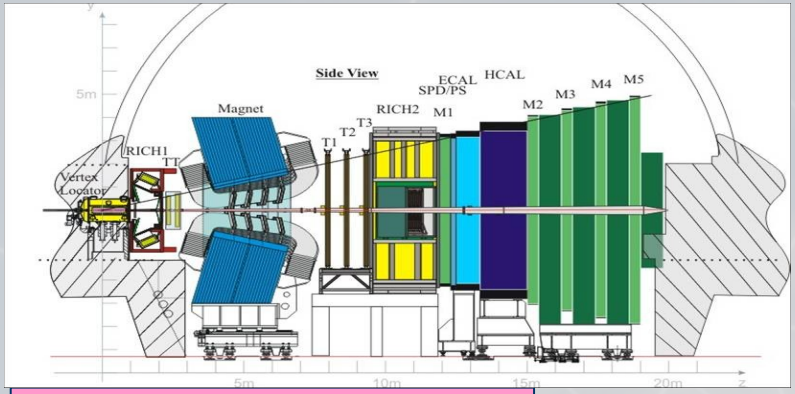
Readout Network

LHCb Online Computing Infrastructure

Substantial resources

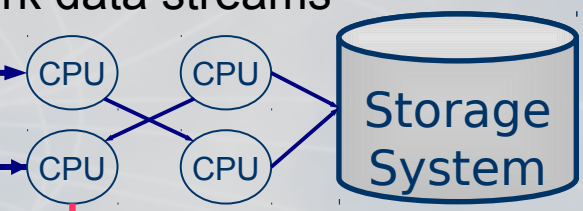
- Spectrometer for beauty and charm quark analysis at LHC
- 40 MHz collision rate
- L0 trigger (hardware)
  Accept rate: ~ 1 MHz
  Network capacity: ~ 100 GB/s
- Data sources: ~ 323
- Event packing: ~ 13
- High Level Trigger (HLT):
  HLT1 accept rate ~ 100-200 kHz
  HLT2 Accept rate ~ 10 kHz
  Event size: ~ 70 KB
- 62 Racks
  ~ 1800 Data handling nodes
  ~ 200 Controls nodes
- HLT hardware
  ~ 1750 Nodes
  ~ 25000 physical CPU cores
  ~ 50000 Trigger processes
  ~ 5000 Infrastructure tasks

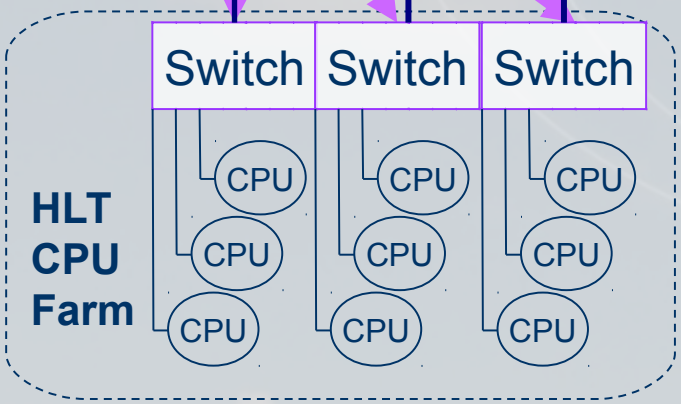
LHCb Online Computing Hardware



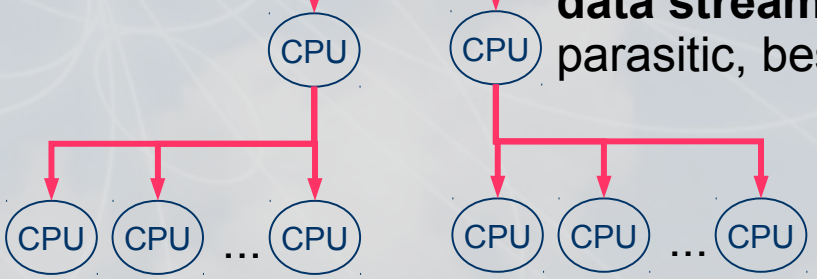
Readout Network

Storage Cluster
- File and Stream handling
- Fork data streams





Partially replicated data streams:
parasitic, best effort



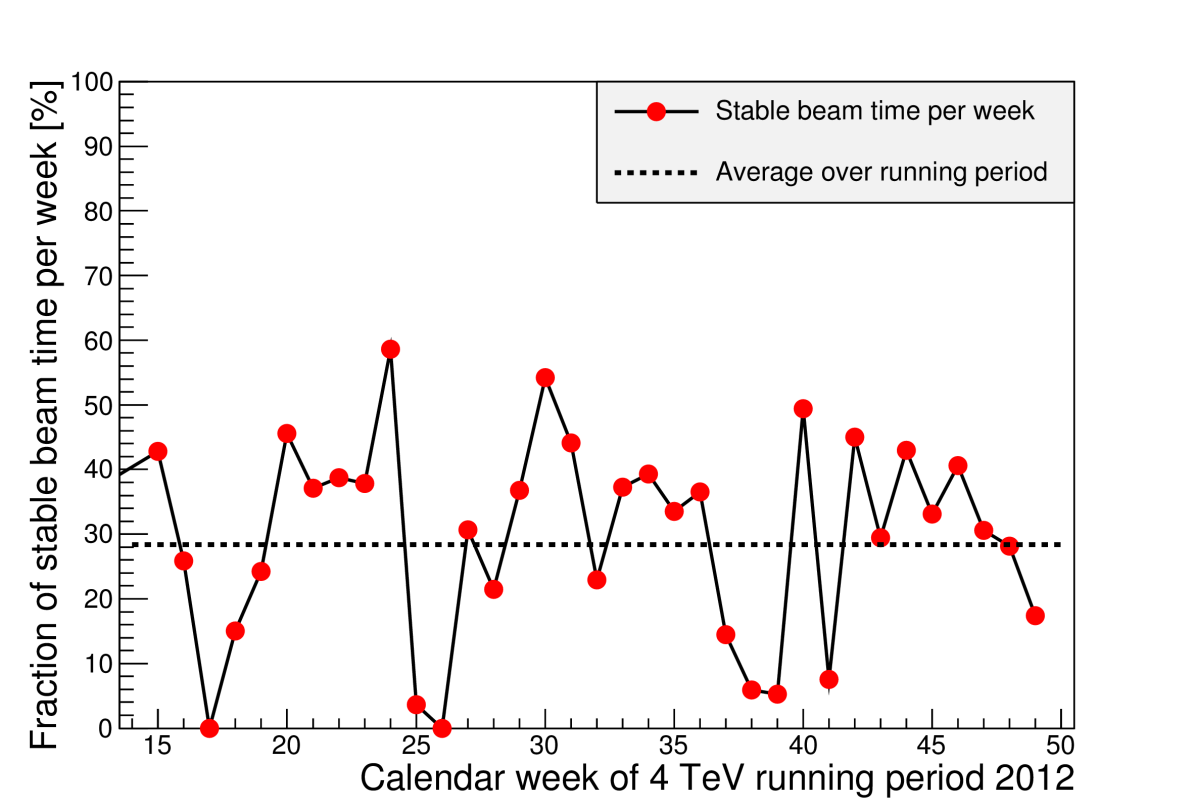
High Level Trigger
Identify the
Good the Bad and the Ugly

Monitoring Cluster
Low level monitoring using raw data

Reconstruction Cluster
High level monitoring with fully reconstructed events

The Boost: Possible Gain of CPU Time

- Stable beams during ~ 30% of the running period
  - 70% of the time the CPU resources are idle
- Take advantage
  - Sophisticated event filtering
  - Better select 'interesting' events
  - Improved physics



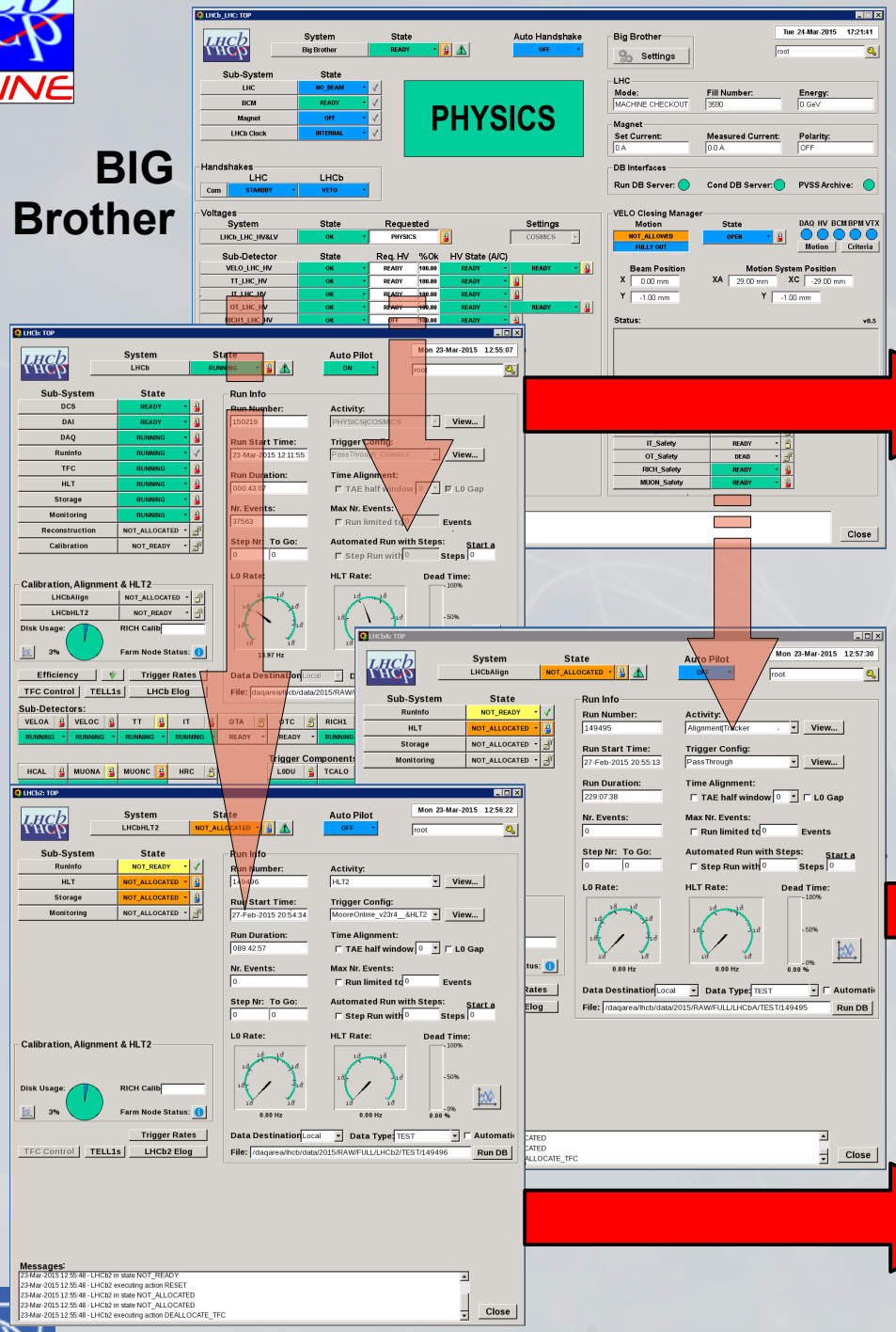
The Road map to Benefit from Idle Time

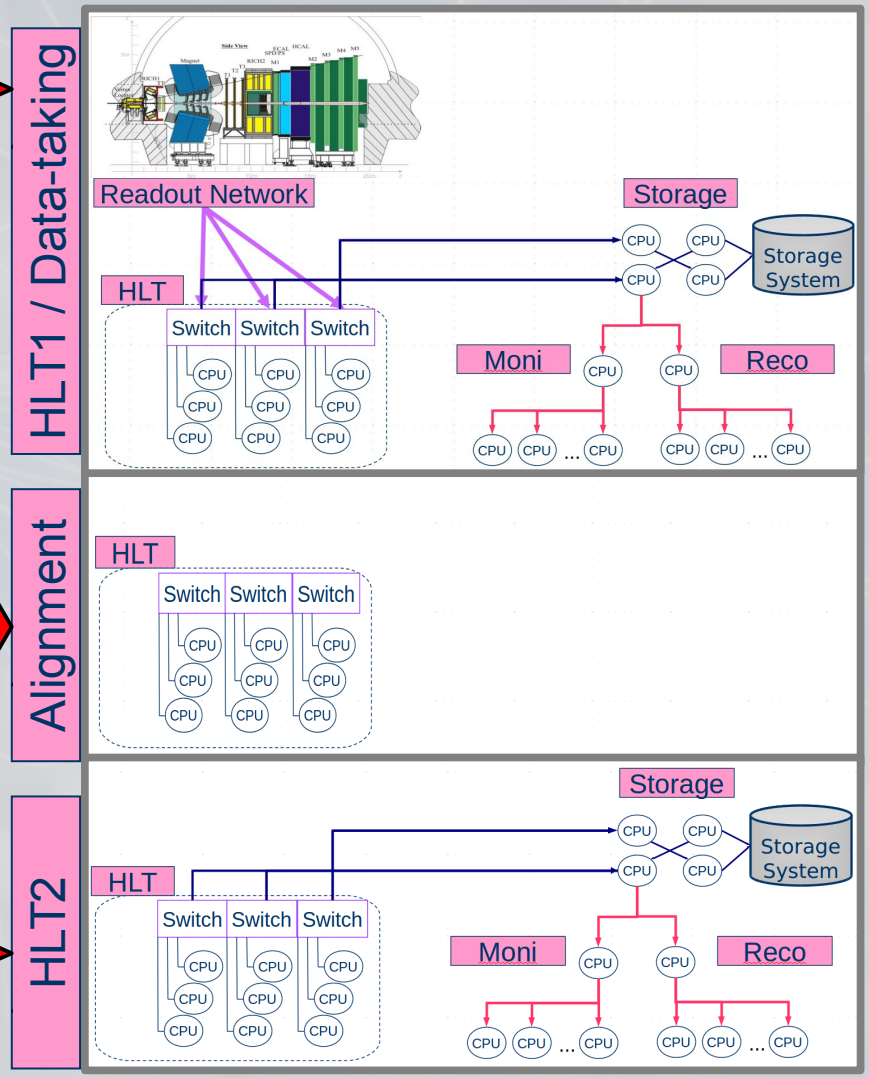
- Defer computing needs to time without beam
  - Save events on the local disk of the worker nodes
- Need to split high level trigger program 'Moore'
  - First stage saves preselected events
  - Second stage performs final event filtering
- Need calibration constants with 'offline quality'
  - Focus on online calibration and alignment activities

Top Level Controls:

3 Trees managed by BIG Brother

BIG Brother



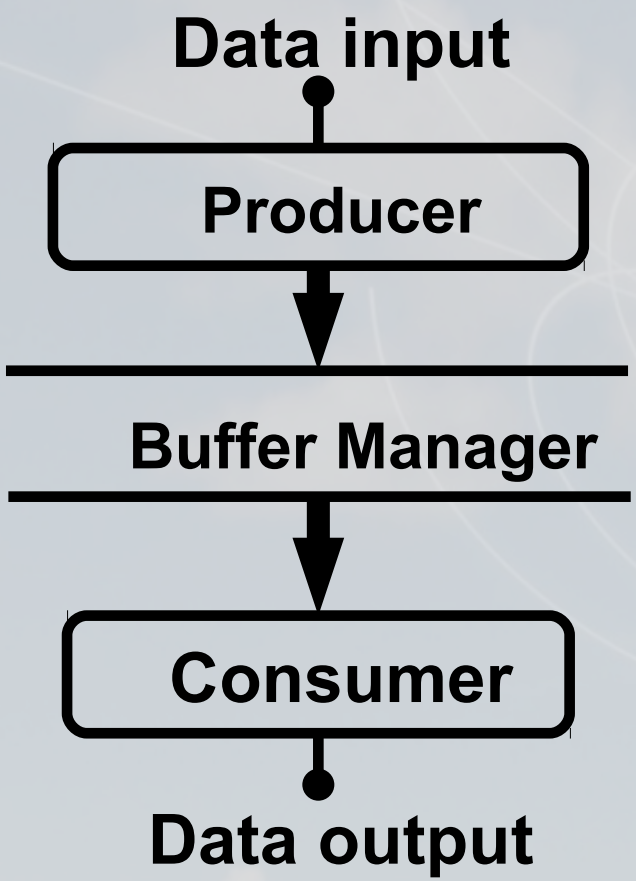


Controls Aspects

- At top level 3 simultaneous activities
  - Orchestrated by Big Brother
  - HLT farm is shared resource

- At low level
  - Node controllers orchestrate processes on one node
  - 1 controller / activity

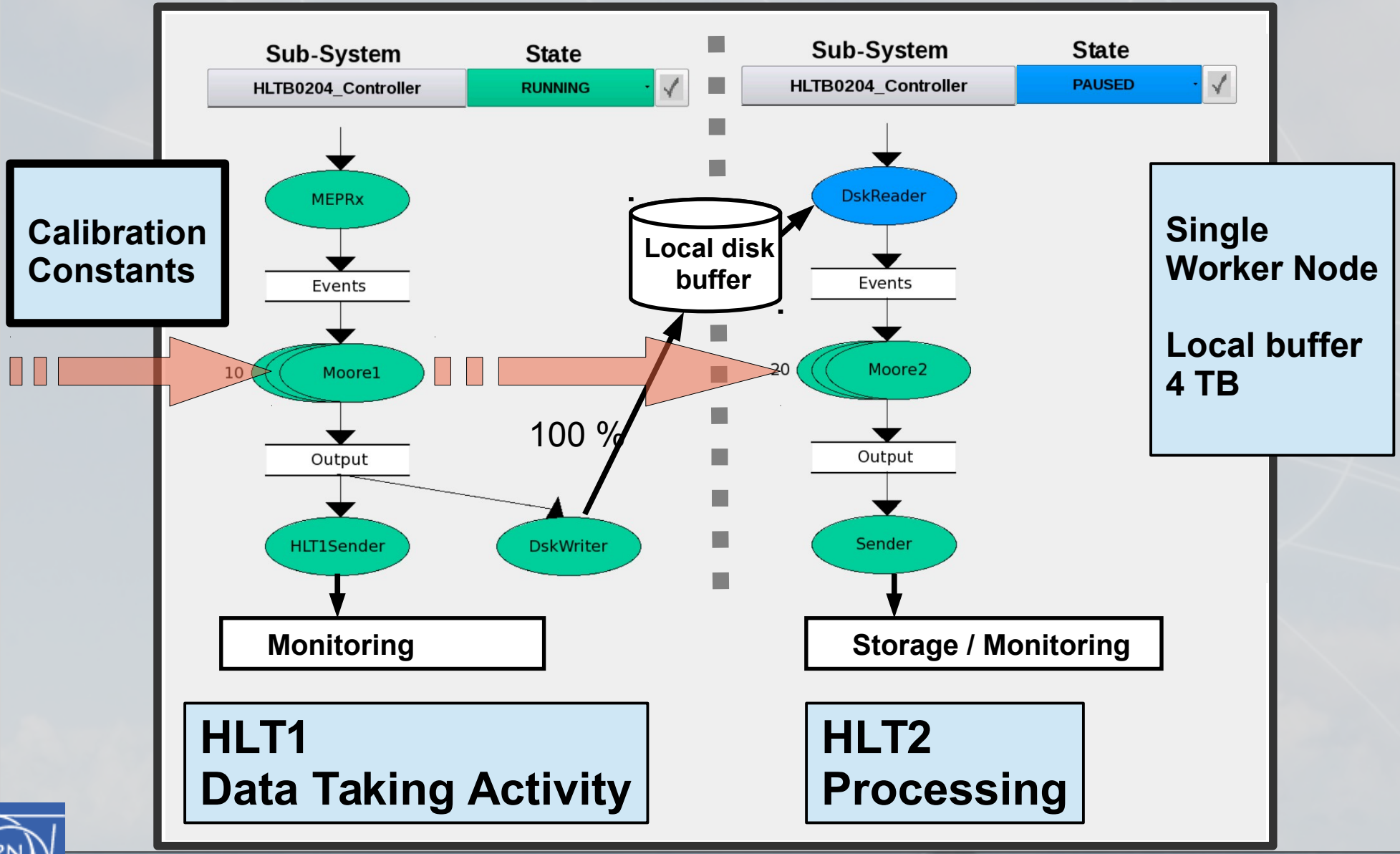
The Basic Pattern: Buffer Manager



- Managed shared memory
- Producers declare events
- Consumers subscribe to events
  - Get notified on data present
- Pattern used whenever event data are moved
  - HLT farm, storage-, monitoring- and reconstruction cluster

See M.Frank et al., "The LHCb High Level Trigger Software Framework", CHEP 2007, Proceedings, Victoria, BC, CA

The Process Architecture: Worker Node



Action Sequence During Fill

- When VertexLocator (VELO) subdetector is closed:
  - Accumulate data for tracking detector alignment O(5 min)
  - Perform tracking detector alignment on these data O(6 min)
  - Change run in HLT1 partition and load tracking alignment
  - Take data for the rest of the fill using offline quality alignment
  - Change run every ~60 minutes
- At end of each run taken with the DAQ / HLT1
  - Perform RICH calibrations
  - Start HLT2 processing
  - Start HLT2 monitoring
- At end of each run processed by HLT2
  - Start data quality monitoring

Operational Remarks (1)

- **HLT1 and HLT2 and calibration activities are asynchronous**
  - **Loose coupling through local disk cache**
  - **Pre-selection of events used for tracker alignment**
  - **HLT1 <u>must</u> execute real-time**
  - **HLT2 executes later**
  - **Optimize usage of disk cache <=> HLT1 rejection (CPU) => physics group**

Operational Remarks (2)

- HLT1 requires 'offline-quality' tracking alignment
  - For first run of each fill
    => Use alignment of previous run
    => Collect events to align tracking detectors
  - Calibrate alignment with collected event-sample
  - HLT1 picks up new constants on Run-Change
- At HLT1 end-of-run (after every ~60 min)
  - Start calibration of other subdetectors
  - Then mark run as 'HLT2 ready' (allow processing)

Worker Nodes Resource Management

- **We must minimize resource usage**
  - **HLT1 and HLT2 processes execution simultaneously**
  - **Nodes are 'over-committed'**
    More processes than CPU cores / hyper-threads
  - **Memory scarce: 2 GB/core**
  - **Limit CPU and network accesses during configuration**
- **Resource sharing is mandatory [1]**
  - **Copy-on-write mechanism saves us ~70% of memory**
    Trigger processes forked after configuration phase
  - **Quick application startup using process checkpointing**

[1] See M.Frank et al., "Optimization of the HLT Resource Consumption in the LHCb Experiment", CHEP 2012, Proceedings, New York, NY/US

Monitoring

- **Detector and HLT1**
  - **Detector performance monitoring with small data stream with HLT1 accepted events in dedicated monitoring farm(*)**
- **HLT2 monitoring**
  - **Based on files with HLT2 accepted events**
    - **Performed on dedicated facility**
    - **Cannot be done online: Simultaneous processing of many runs**
- **Data quality monitoring**
  - **Based on data files**
  - **Performed on dedicated facility**

(*) See M.Frank et al., "Online Data Monitoring in the LHCb Experiment", CHEP 2007, Proceedings, Victoria, BC, CA

Miscellanea

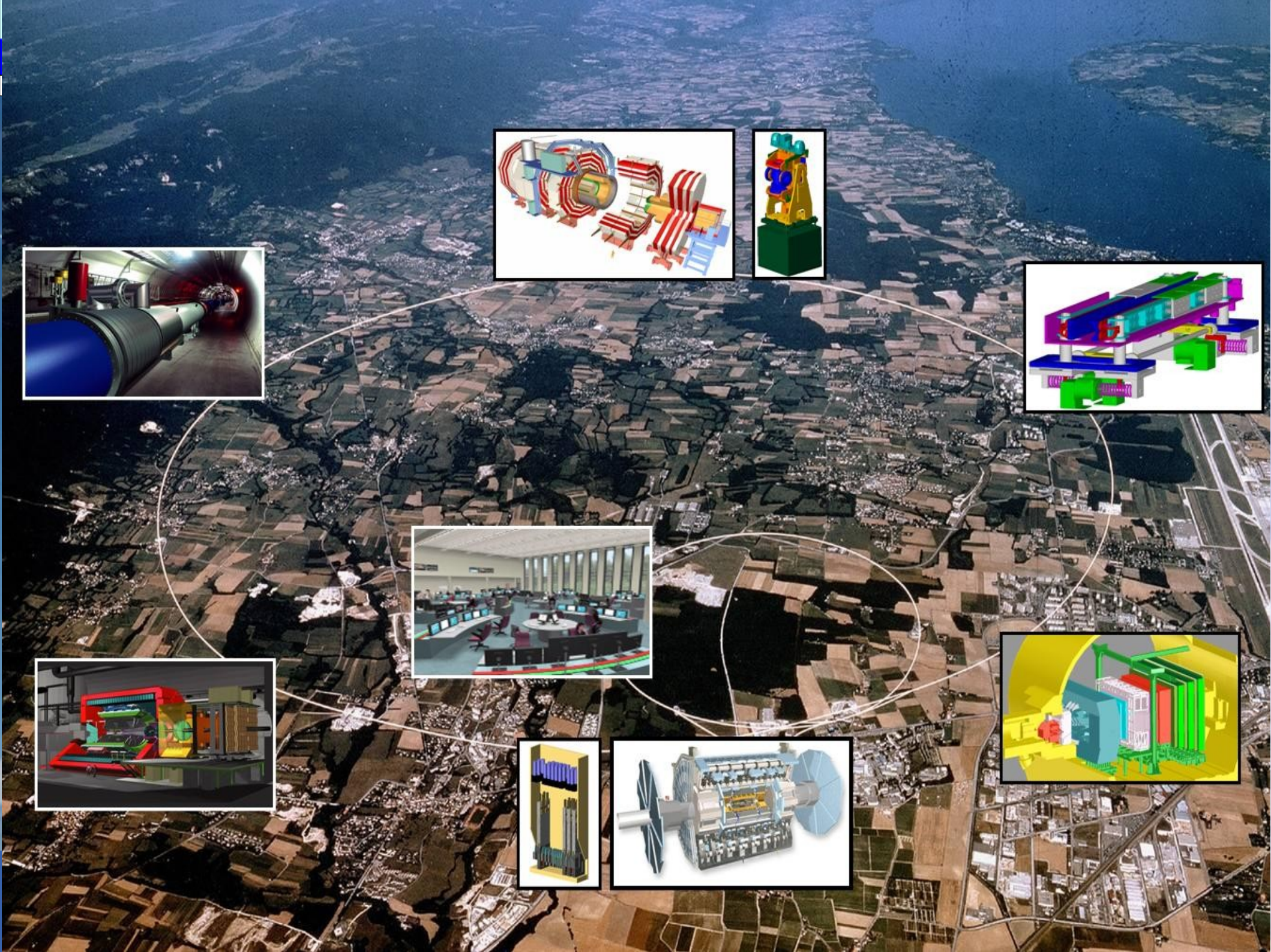
- Basic DAQ architecture unchanged
- All servers > 6 years exchanged
- All control nodes are virtualized
- All web servers are virtualized
- Local disk space on HLT worker nodes

Raw space	Usable	100 kHz	Buffer	LHC eff. 30 %
12 PB	~ 6 PB	10^6 seconds	~ 11 days	~ 38 days

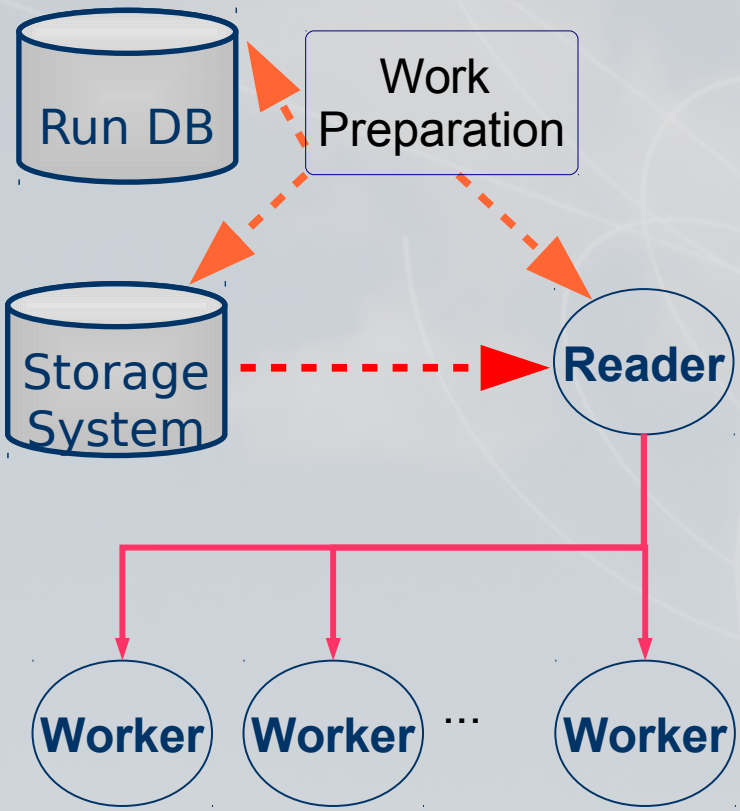
  - Sufficient to keep farm busy during major MD phases

Conclusions

- We managed a redesign of the high level trigger infrastructure
  - Using periods without beam boost CPU usage by 200 %
  - Improve event selection, better physics
  - Farm upgraded, replacement of obsolete equipment
- Benefits from consequent application of patterns
  - Multiple instances of functionally similar entities
  - Buffer manager, node control, run control



HLT2 and Data Quality Monitoring



- Scan run DB and data area for runs to be processed
- Prepare work for reader node
- Distribute events to workers
- Combine and save histograms from workers
- Processes controlled by WinCC (as on farm)