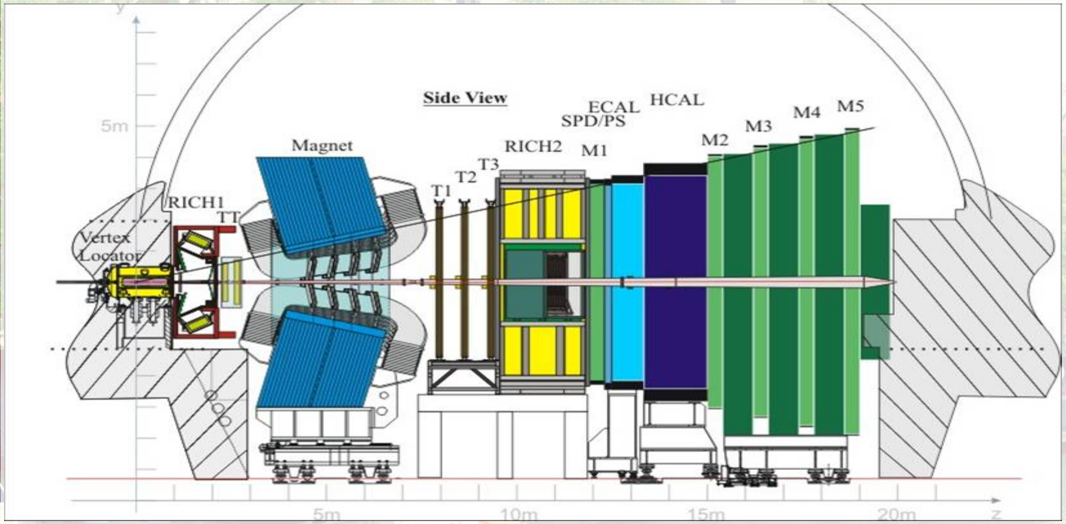
# Deferred High Level Trigger in LHCb: A Boost to CPU Resource Utilization

**The use of periods without beam for online high level triggers**

- Introduction, problem statement
- Realization of the chosen solution
- Conclusions

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

# LHCb Online Computing in Numbers



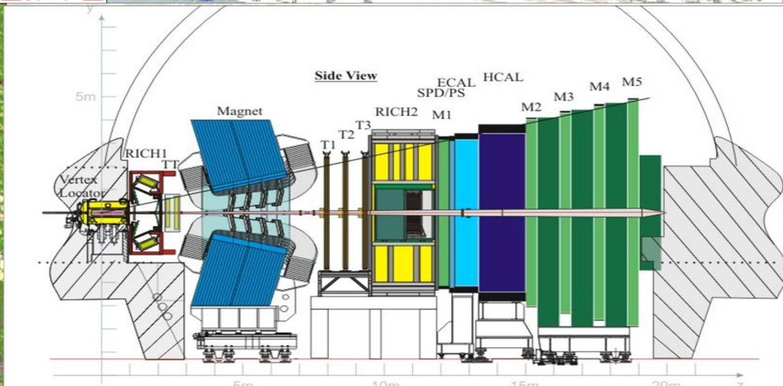

Readout Network

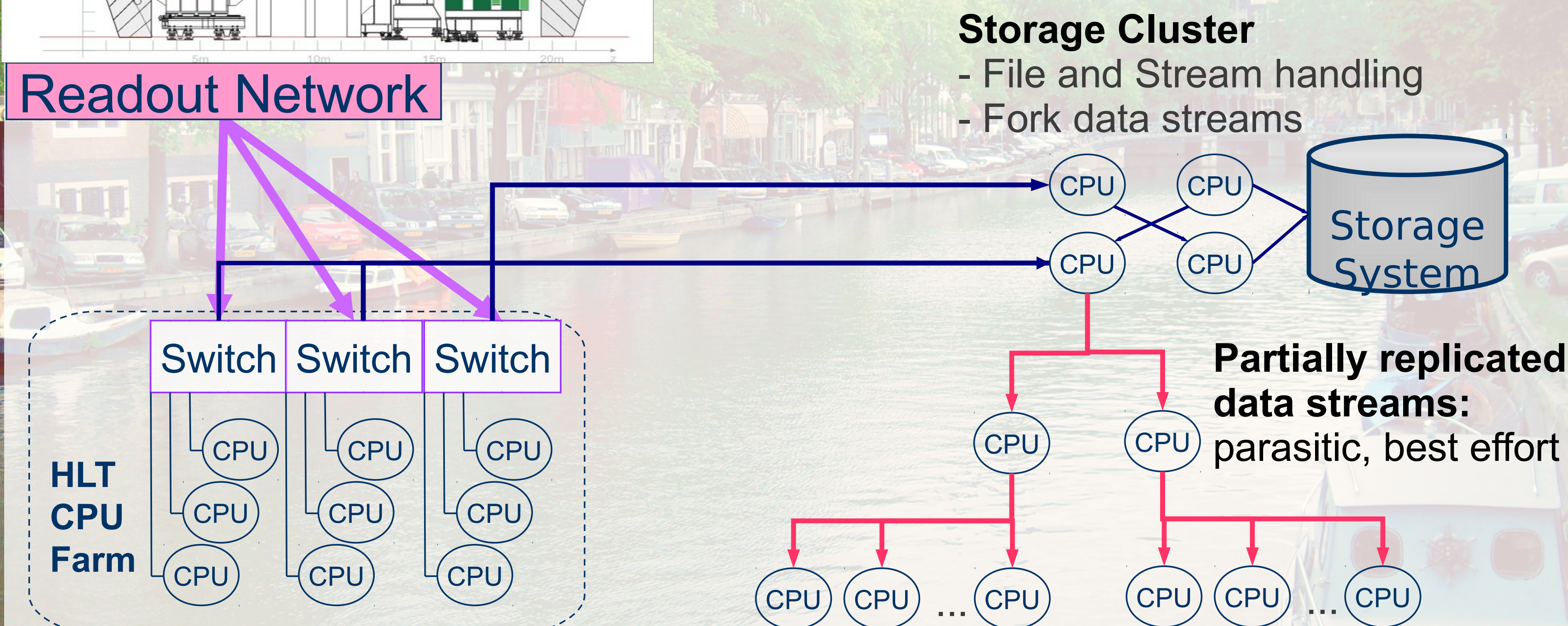
**LHCb Online**
**Computing Infrastructure**

**Substantial resources**

- **Spectrometer for b quark analysis at LHC**
- **40 MHz collision rate**
- **L0 trigger (hardware)**
  **Accept rate: ~ 1 MHz**
  **Readout NW: ~ 60 GB/s**
- **HLT (software)**
- **Accept rate: ~ 2-8 kHz**
- **Event size: ~ 50 KB**
- **Data sources: ~ 350**
- **Event packing: ~ 13**
- **56 Racks**
  **~ 1700 Data handling nodes**
  **~ 200 Controls nodes**
- **HLT (expected for 2015):**
  **~ 1600 Nodes**
  **~ 25000 CPU cores**
  **~ 45000 Trigger processes**
  **~ 5000 Infrastructure tasks**

# LHCb Online Computing Hardware



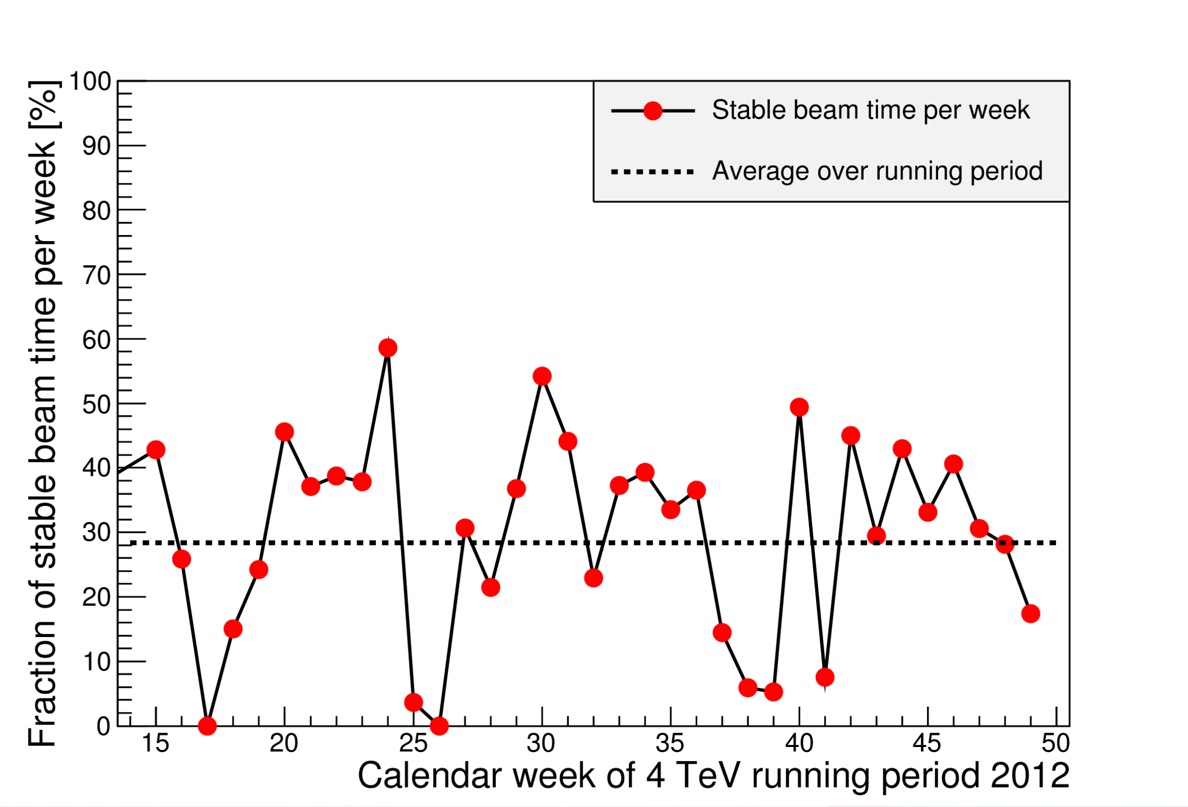

**High Level Trigger**
Identify the
Good the Bad and the Ugly

**Monitoring Cluster**
Low level monitoring using raw data

**Reconstruction Cluster**
High level monitoring with fully reconstructed events

# The Boost: Possible Gain of CPU Time

- LHC delivers roughly during 30% of the running period stable beams to LHCb
- 70% of the time the CPU resources are idle
- Take advantage of the idle-time
  - Sophisticated event filtering
  - Better selection of 'interesting' events
  - Improved physics results

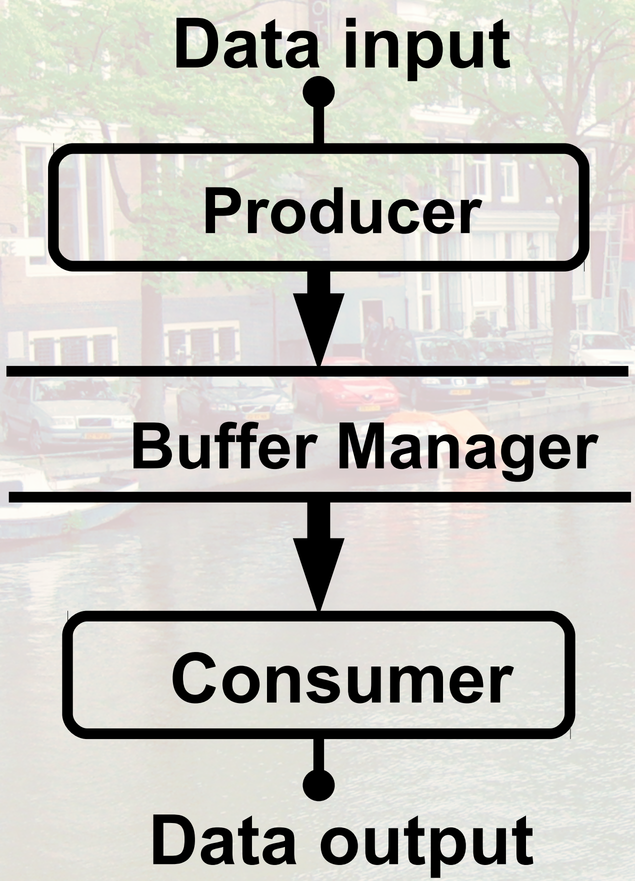


# The Roadmap: Benefit from Idle Time

- Try to defer computing needs to time without beam
  - Save events on the local disk of the worker nodes
    - ~ 8-9 hours beam time (~1 day) buffering for 1 TB disks
- Need to split high level trigger program 'Moore'
  - Only save preselected events
    - Rejection factor 6:  ~1-2 week of buffering
    - Enough to be busy during MD periods
  - First stage component responsible for pre-selection
  - Second stage component for the final event filtering
- Here I present the supporting infrastructure
  - Not the physics details of this split

Next: Introduction of basic concepts used in the realization

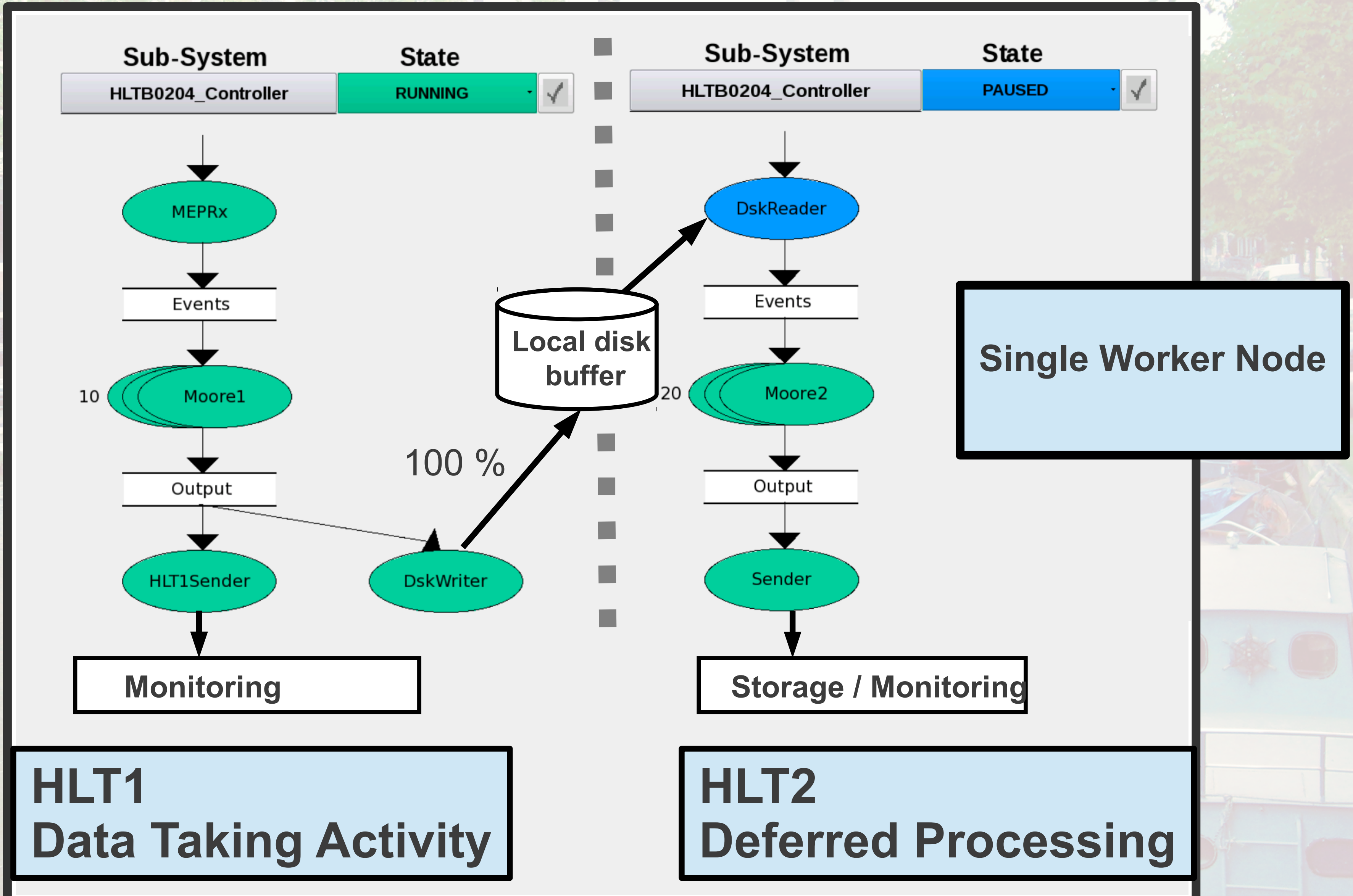
# The Basic Pattern: Buffer Manager




- **Managed shared memory**
- **Producers declare events**
- **Consumers subscribe to events**
  - **Receive interrupts when data is present**
- **Pattern used at all stages**
  - **Whenever event data have to be moved**
  - **HLT farm, storage-, monitoring- and reconstruction cluster**

See M.Frank et al., “Data Stream handling in the LHCb experiment”, CHEP 2007, Proceedings, Victoria, BC

# The Process Architecture: Worker Node



# Worker Nodes: Remarks (1)

- HLT1 and HLT2 activities are entirely asynchronous
  - Loose coupling through local disk cache
  - HLT1 must execute real-time
  - HLT2 executes with lower priority
- HLT2 requires 'offline-quality' calibration
  - Calibration in real-time using fraction of HLT1 accepted events
  - Data monitoring facilities in dedicated farms crucial

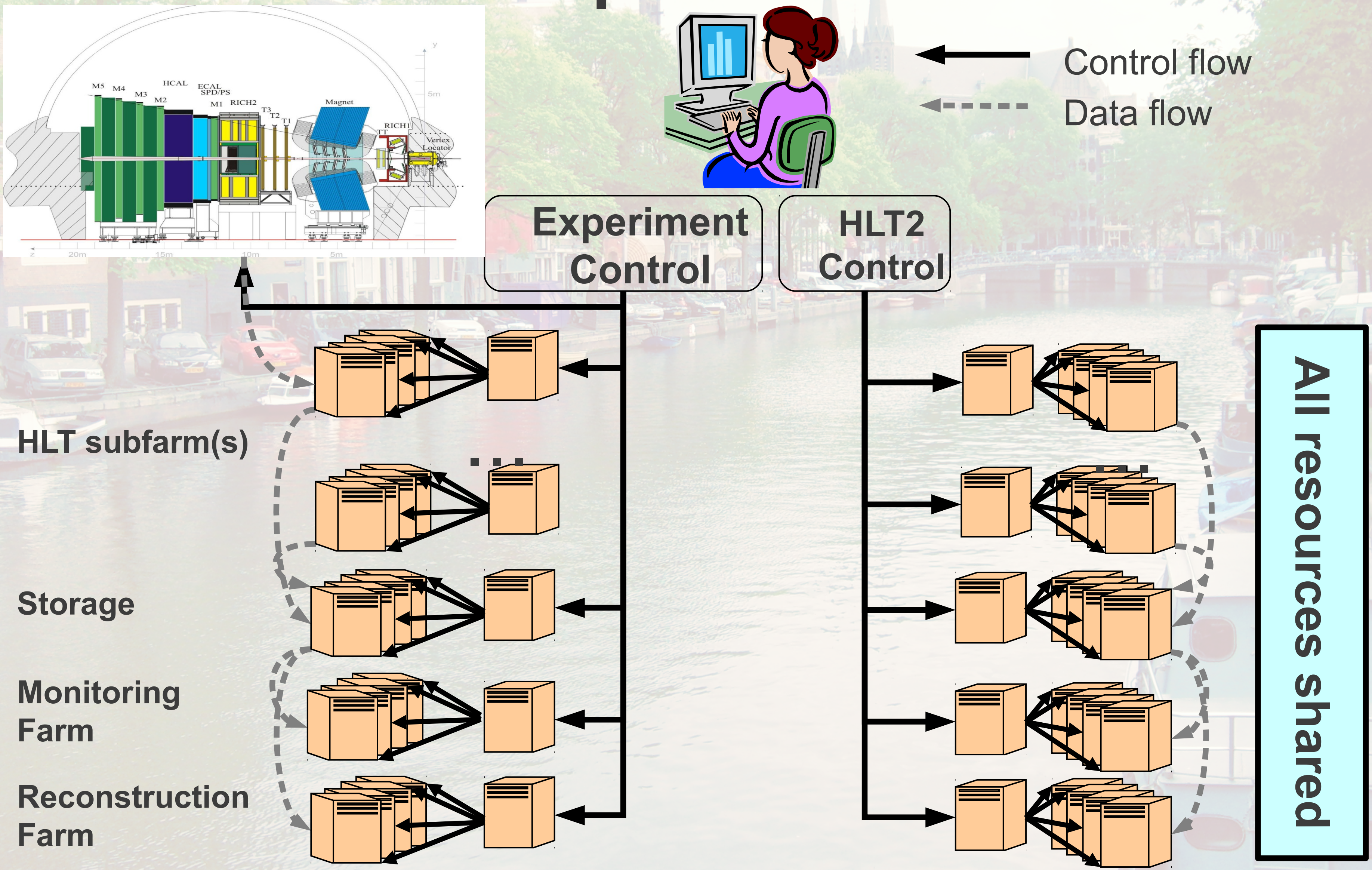
# Worker Nodes: Remarks (2)

- We heavily rely on minimizing resource usage
  - Moore processes execution simultaneously on each worker node
- Worker node resources are 'over-committed'
  More processes than CPU cores / hyperthreads
  - Memory scarce (2 GB/core) if not addressed
  - CPU and network accesses during configuration
- Resource sharing is mandatory
  - Large benefit from copy-on-write (~70% of memory)
    Trigger processes forked after configuration phase
  - Quick application startup using process checkpointing

# Worker Nodes: Control

- All processes of one activity on a worker node
  - Need to be started and configured in a well defined order following the states of a finite state machine
  - Are controlled by a dedicated process, which reports to the experiment controls system
- Consequences for the control of the activities
  - Two independent control trees (next slides)
  - HLT1 + Experiment
  - HLT2 activity

# Controls: Two Separated Control Trees

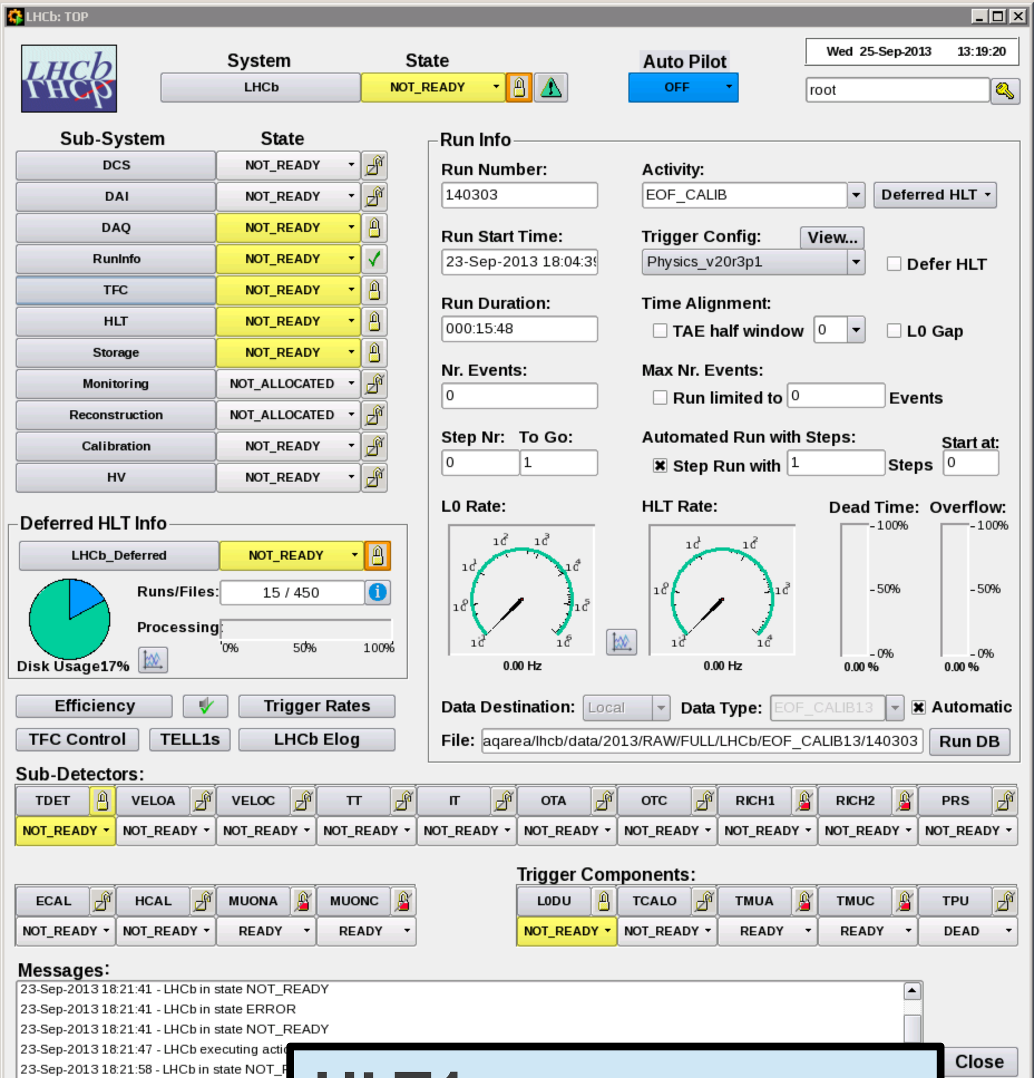


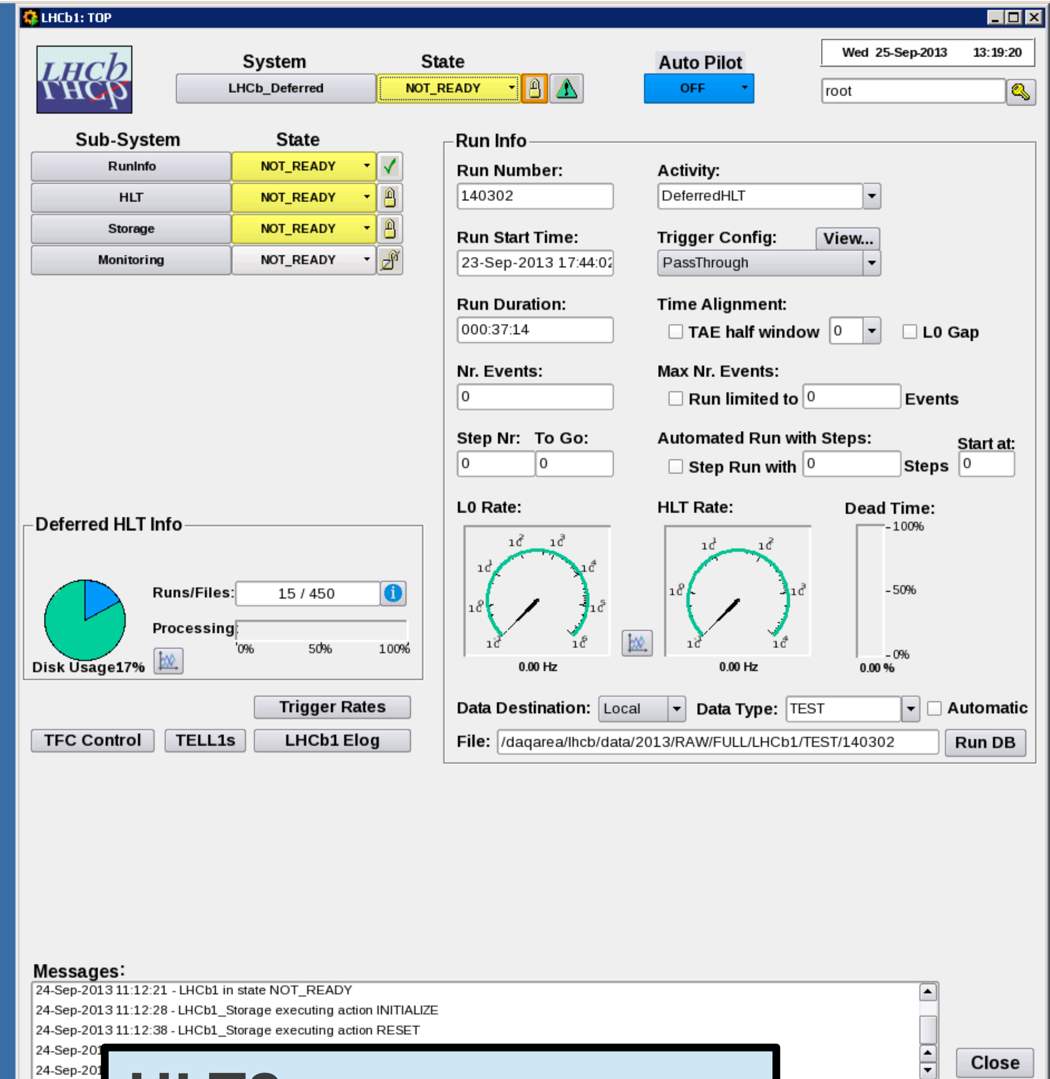
# Controls Issues

- Experiment controls system implemented in WinCC
  - Commercial SCADA (originally called PVSS)
  - Used throughout the experiment
    - Hardware configuration (slow control)
    - DAQ, Run-Control, Farm operations
- Partitioning concept realized throughout
  - Traditionally: Parallel DAQ of independent sub-detectors while no beam
  - De facto: Deferred trigger processing = Independent DAQ with data from disk

  => Presence of partitioning concept eased the implementation of deferred HLT processing

# Controls: Parallel Trees in Reality

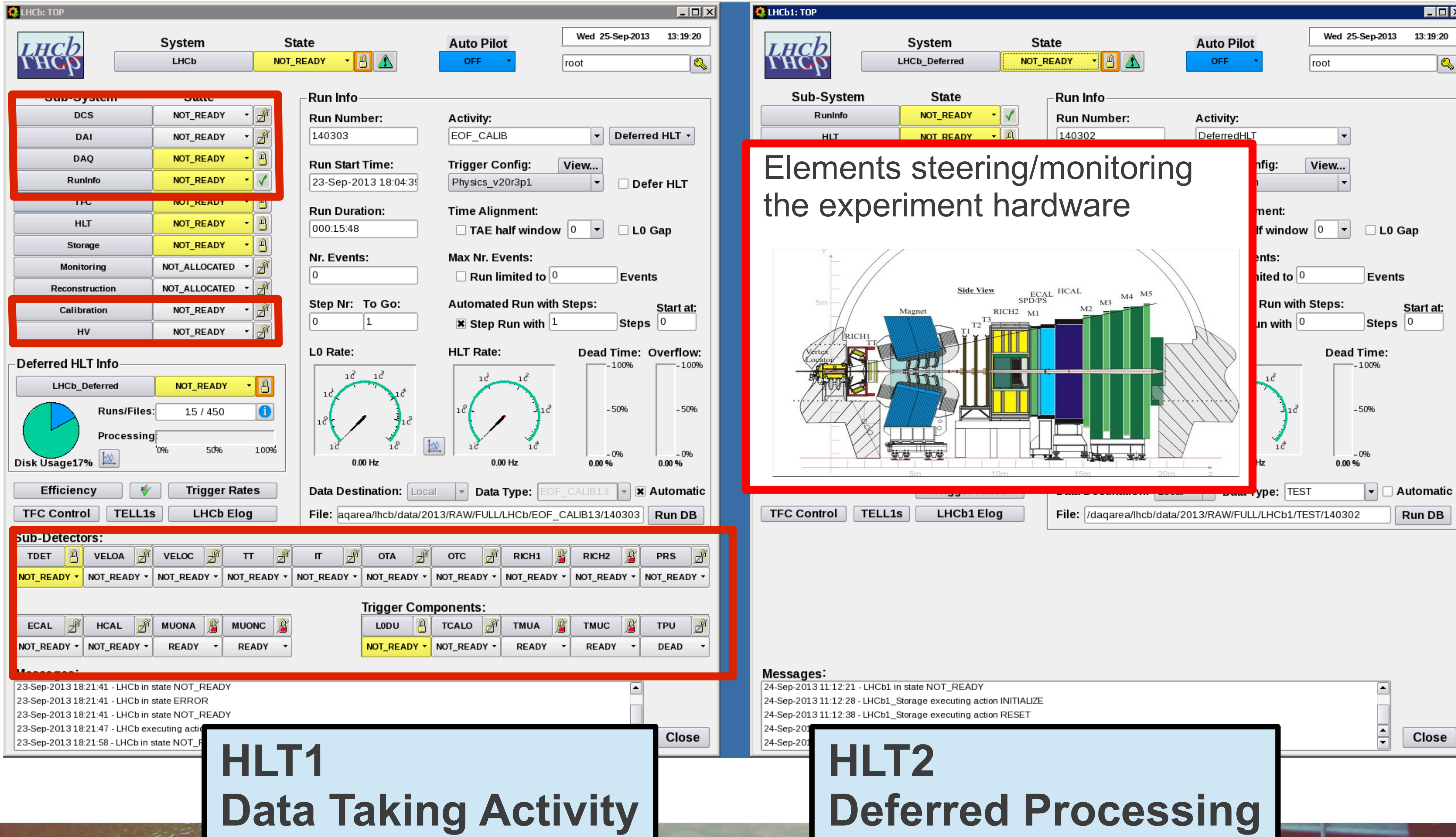



**HLT1**
**Data Taking Activity**

**HLT2**
**Deferred Processing**

# Controls: Parallel Trees in Reality



# Conclusions

- We managed a redesign of the high level trigger infrastructure to
  - Benefit from time periods without beam
  - Results in a possible increase of 200% CPU time
  - Gained CPU time to be used to improve event selection in the high level trigger
- The realization was based on two basic concepts
  - Consistent deployment of the Buffer Manager pattern throughout the dataflow
  - The partitioning concept supporting shared computing resources