CERN

File-based Data Flow in the CMS Filter Farm

E. Meschi – CERN PH/CMD
For the CMS Data Acquisition Group

CMS Data Acquisition for Run 2 (DAQ2)

Rationale behind the DAQ2 design already discussed in previous talk by R.Mommsen

- **Replace** obsolete technologies and benefit from **technology evolutions**
- High speed protocols (IB, 40GE), increased **memory bandwidth, many core architectures** (also…GPGPUs, MCI, ARM)**, RAM** speed up and **$/GB** down, reliable and performing **cluster file systems**

A **chance to rethink DAQ** in view of improved reliability and robustness, expanded capacity and/or increased demand (e.g. more CPU for HLT, more output bandwidth)

HLT Old and New Requirements:

- Input **after level-1** at maximum event rate of 100 kHz, Output rate (~10^3 Hz)
- Run **off-line code** on farm of **commercial CPUs** and use full event including tracking
- Up to **200 ms/event** (estimated max for Run 2), capable of handling **50 events pile-up @100kHz**
- **Allow expandability**: add HLT nodes as physics performance and/or machine conditions warrant (support diverse architectures and processing power per unit)

Hardware

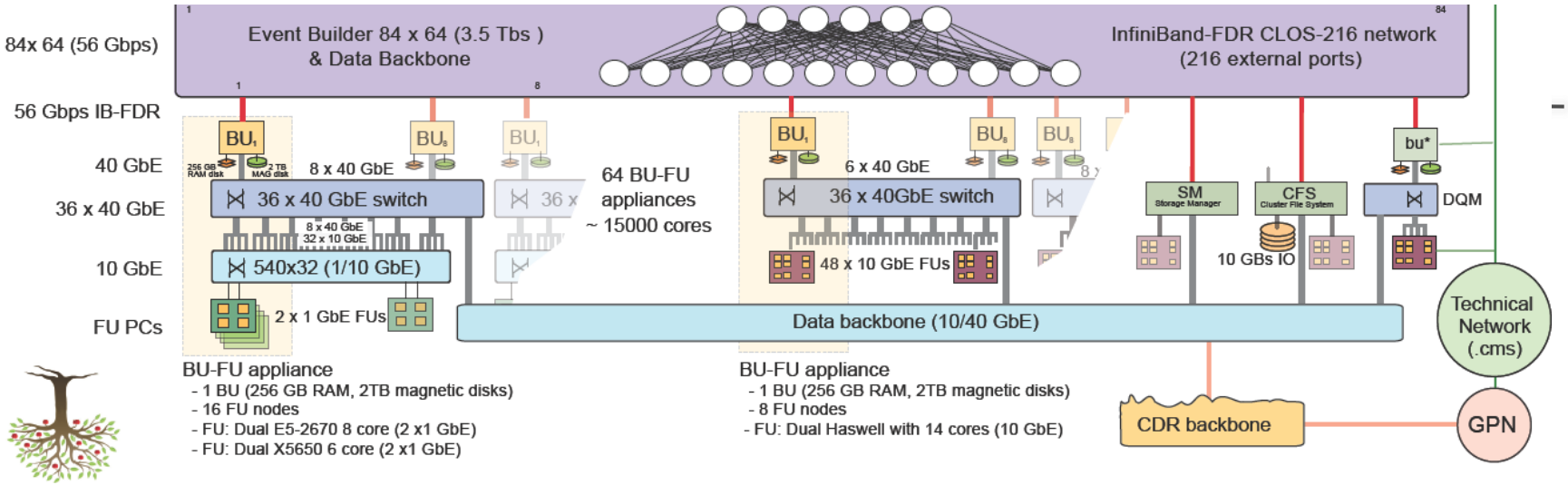
PC type	Operation	CPU per node	# cores / node	RAM (GB) / node	# nodes	#cores Run-2	HLT rel. perf. / node	
PE1950	2008-12	2*E5430	8	16	720		0.25	
C6100	2011-15	2*X5650	12	24	288	3456	0.6	
C6220	2012-16	2*E5-2670	16	32	256	4096	1.0	1GbE
s2600KP	2015-19	2*E5-2680v3	24	64	360	8640	1.66	10GbE
sum						16192		

- The CMS Online farm evolves gradually and must accommodate different generations of hardware
- **Integrate legacy processing nodes** with GbE interface not yet at end of life
- Recent dual-CPU motherboards have enough cores to require more than 1 GbE input
- 10 GbE NICs have become sufficiently inexpensive

HLT: Areas of improvement

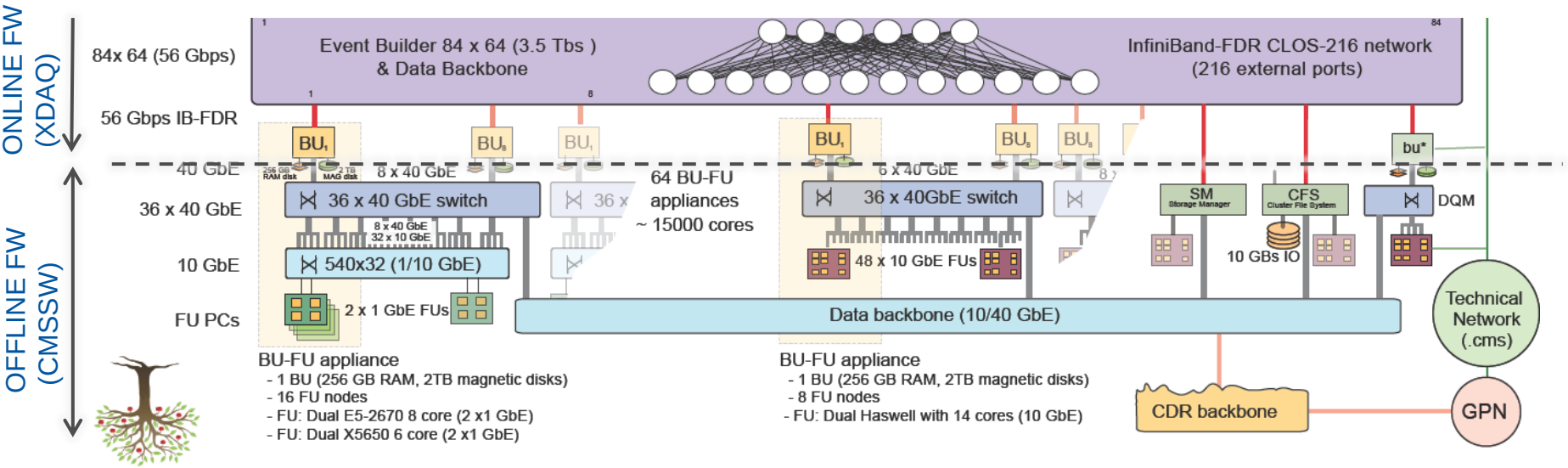
- Increase **resiliency** against failures, e.g. loss of one processing node
- **Reduce coupling** between DAQ and HLT (both time- and software-wise)
- Accommodate initialization and condition loading without generating deadtime at start of run
- A (network) file system provides:
  - Resource accounting, **arbitration, bookkeeping**
  - A **shared buffer** for time decoupling (initialization, condition loading)
  - A **network-agnostic** high level protocol for data, control and monitoring
- A file-based HLT makes “standard” use of offline software
  - Input, output of event and non-event data, monitoring and logging through **files**
  - **No need to synchronize state models** and life cycles of offline and online frameworks
  - Maintain online-offline **software release and deployment cycles** separated
  - Simplify debugging
- **A file system is “good” software we don’t need to write and maintain**

File-based Filter Farm (F^3)



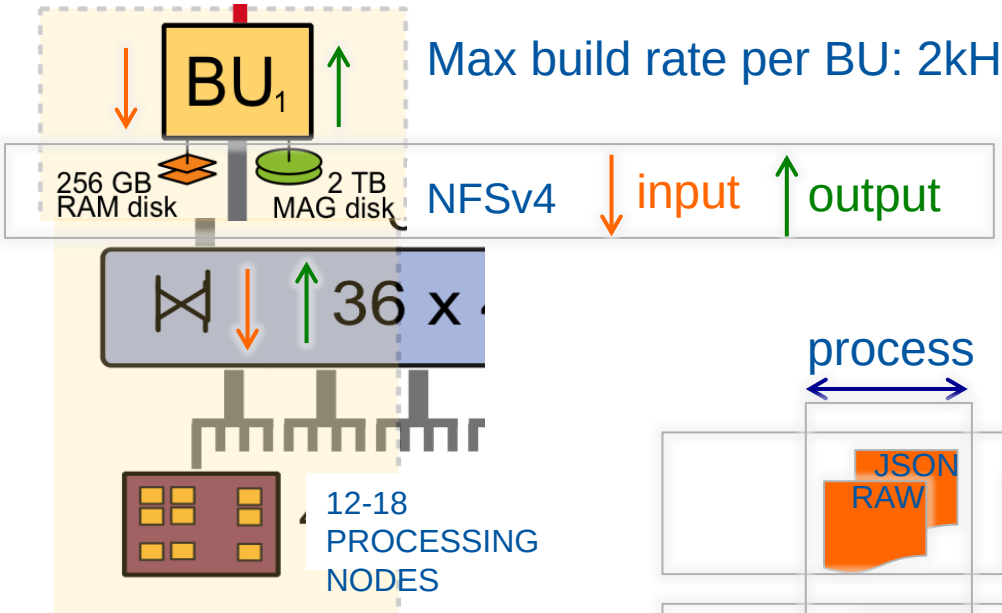
- 62 Builder Unit nodes (BU) receive event fragments **via InfiniBand**
- Complete events are stored locally **to file in a large RAMdisk** (~250 GB)
- **12-18 multi-core PCs** (Filter Units, FU) attach the RAMdisk via a 10/40 GbE switch
- Input files served to standard reconstruction processes running **offline CMS reconstruction framework (CMSSW)**
- Several **HLT processes** in a FU work on different input files and independently output to local disk

File-based Filter Farm (F^3)



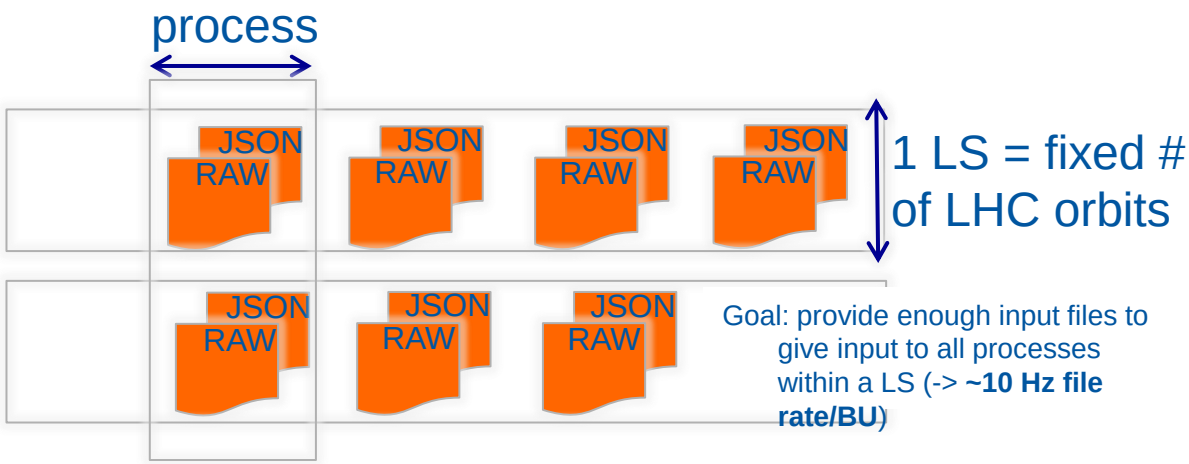
- The complex of a BU and its attached FUs form an "**HLT appliance**", i.e. a unit of computation in the filter farm
- The **HLT runs as a service** in the appliance
  - Including infrastructure to control/monitor processes and handle input/output
  - Providing fault tolerance / error recovery
- Different appliances can have different hardware: legacy 1/10 GbE switch(es) are used to attach legacy machines with 1GbE interface

Appliance Input, Lumi Section



Max build rate per BU: 2kHz, ~2-3GB/s

Every raw data file accompanied by a **metadata file in JSON format** used as a hook to indicate completion and ownership of data

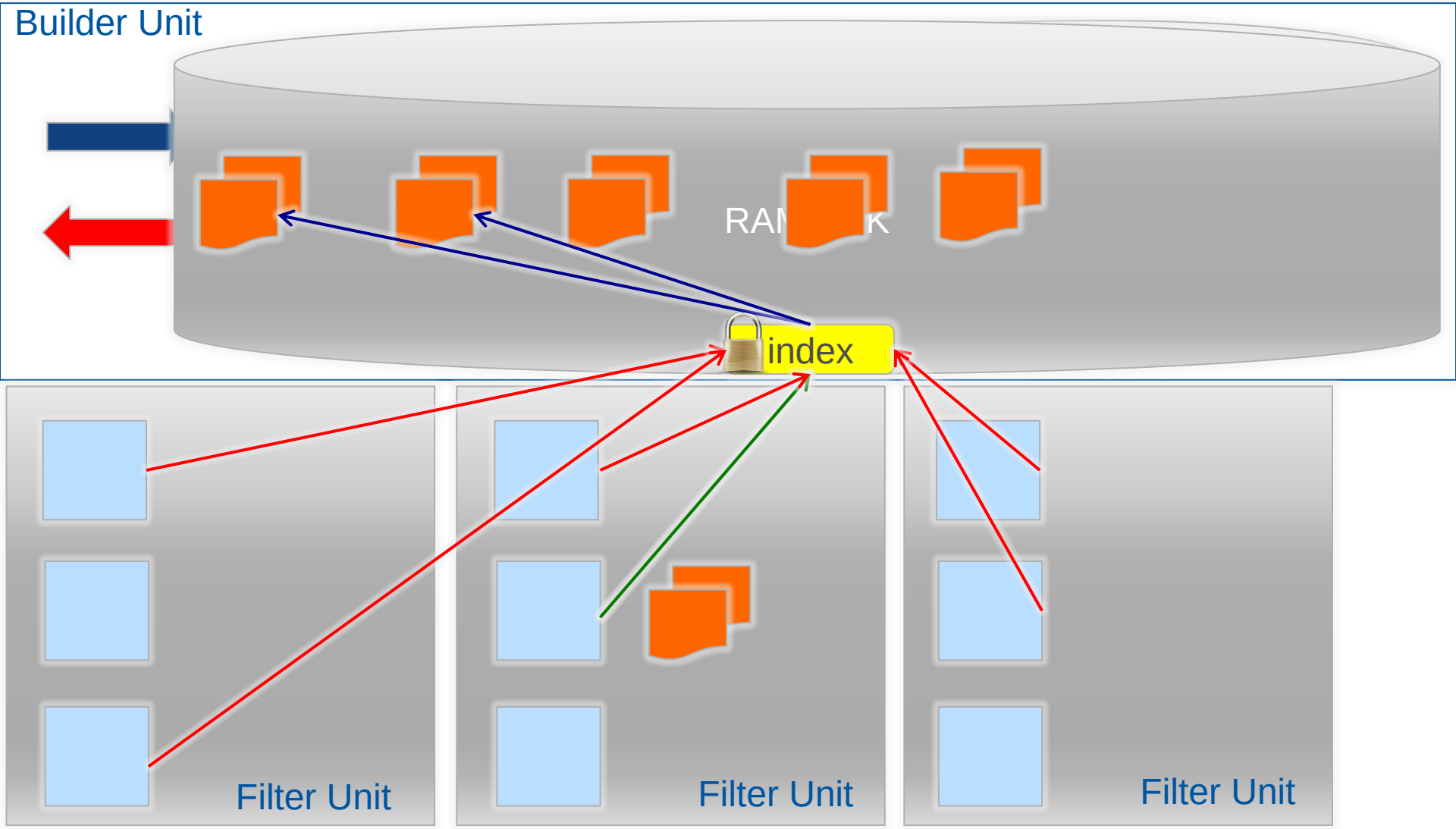


Goal: provide enough input files to give input to all processes within a LS (-> **~10 Hz file rate/BU**)

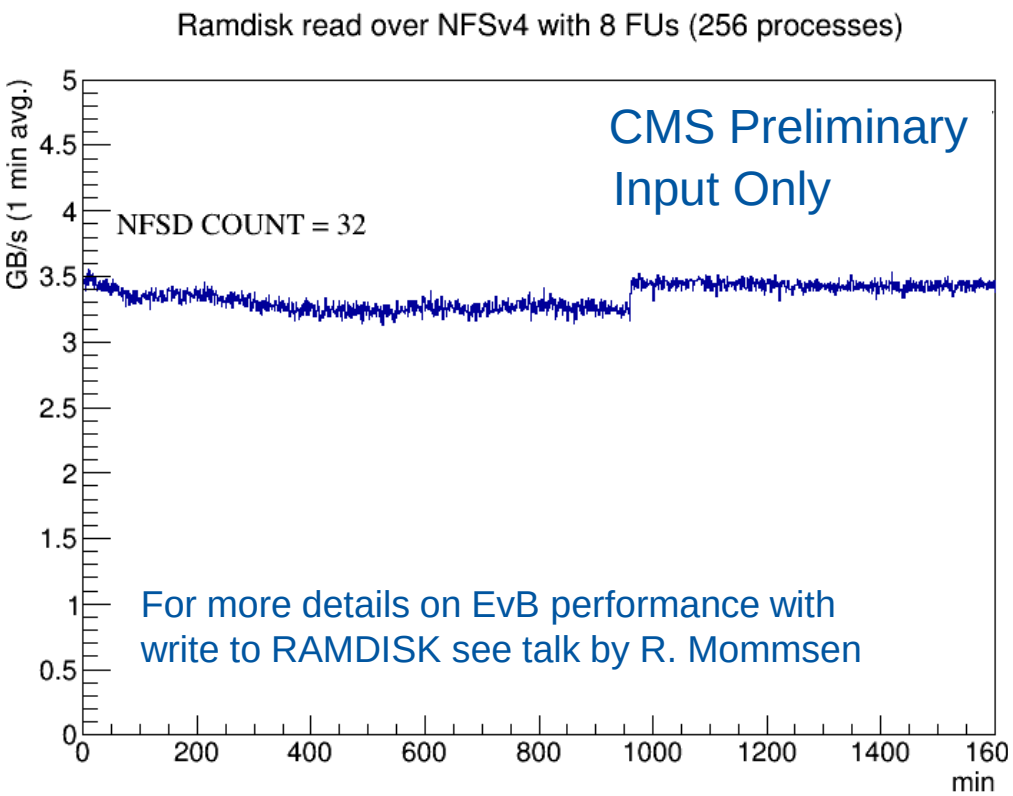
- CMS quantum of data taking(*): **Lumi Section (a fixed time span - now 23.4s**) used to close output files – defines **minimum** latency
- Input/output files **must not cross an LS boundary**

(*) controlled by Trigger Control and Distribution System (TCDS) and used later for the calculation of effective integrated luminosity

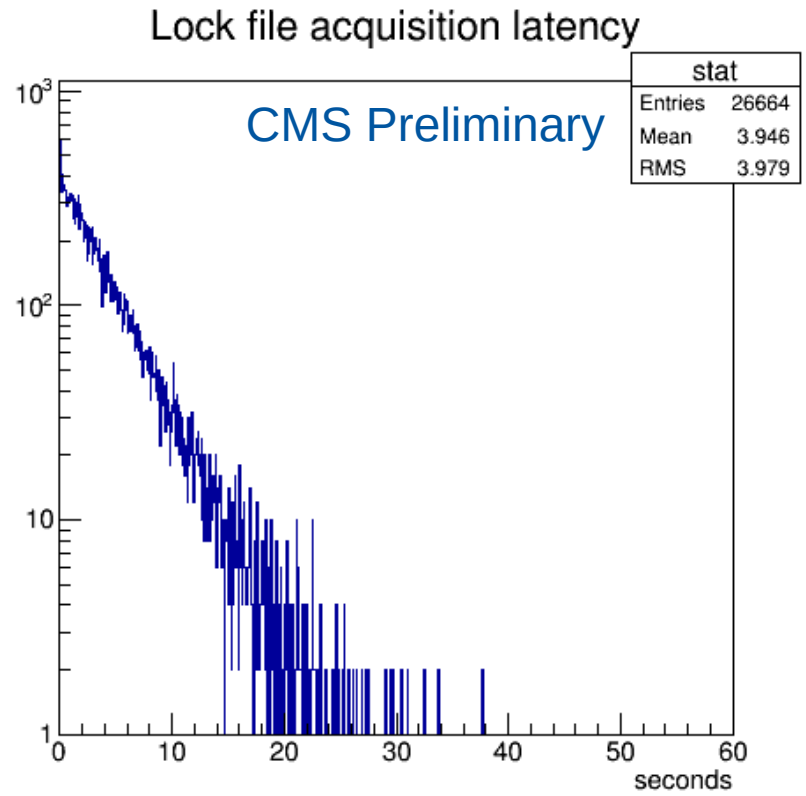
Input, backpressure, arbitration



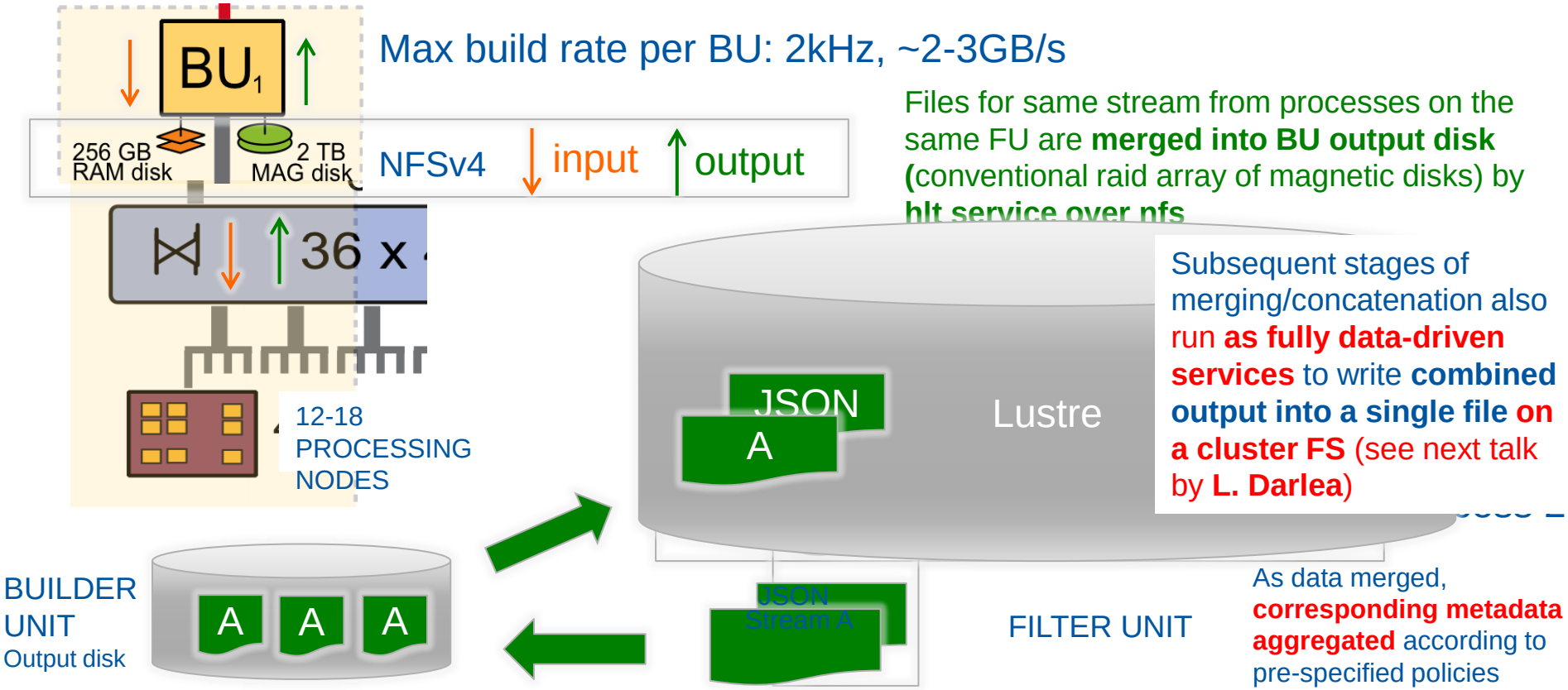
Input performance and latency



For more details on EvB performance with write to RAMDISK see talk by R. Mommsen

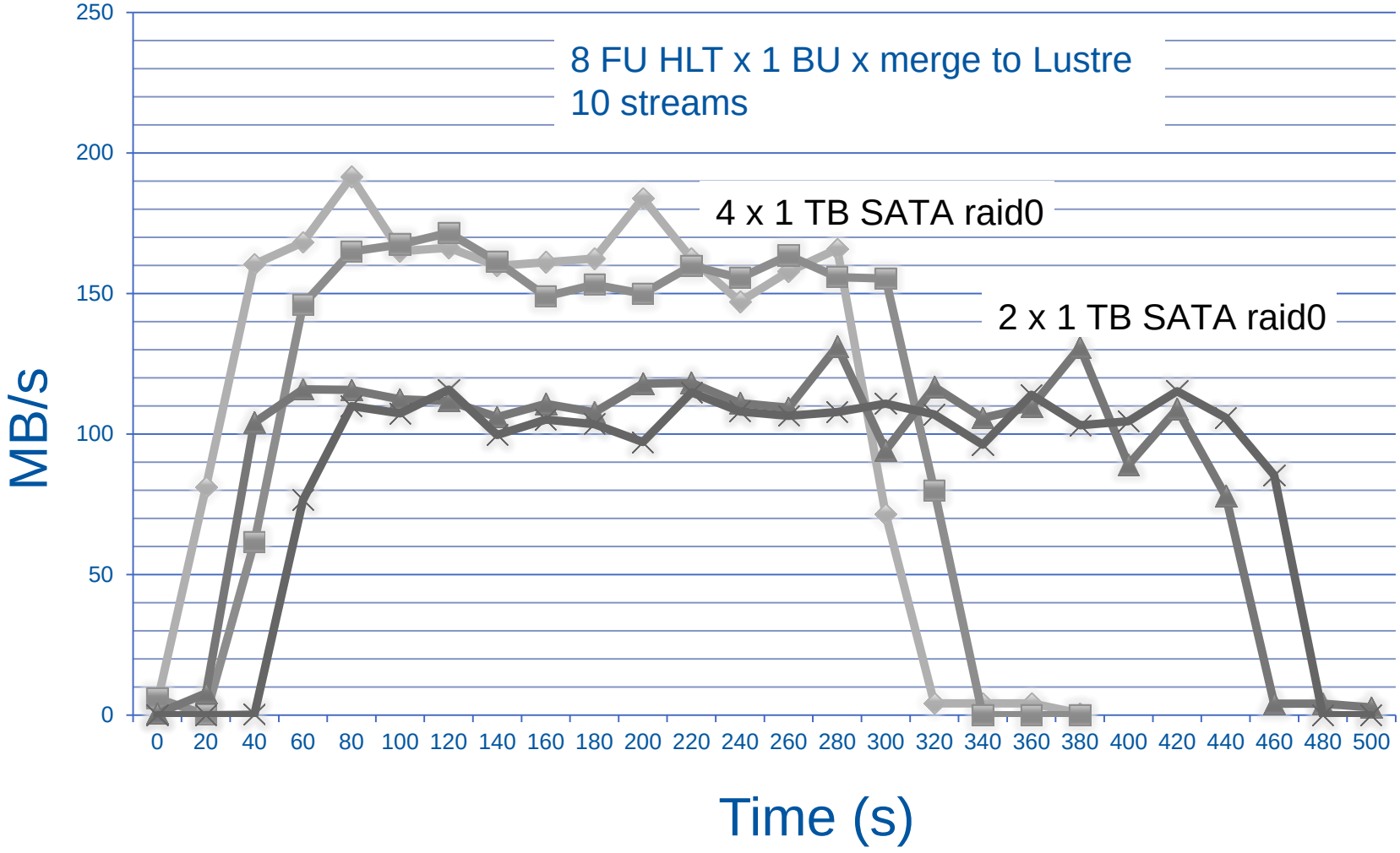


Appliance Output, Streams and Storage

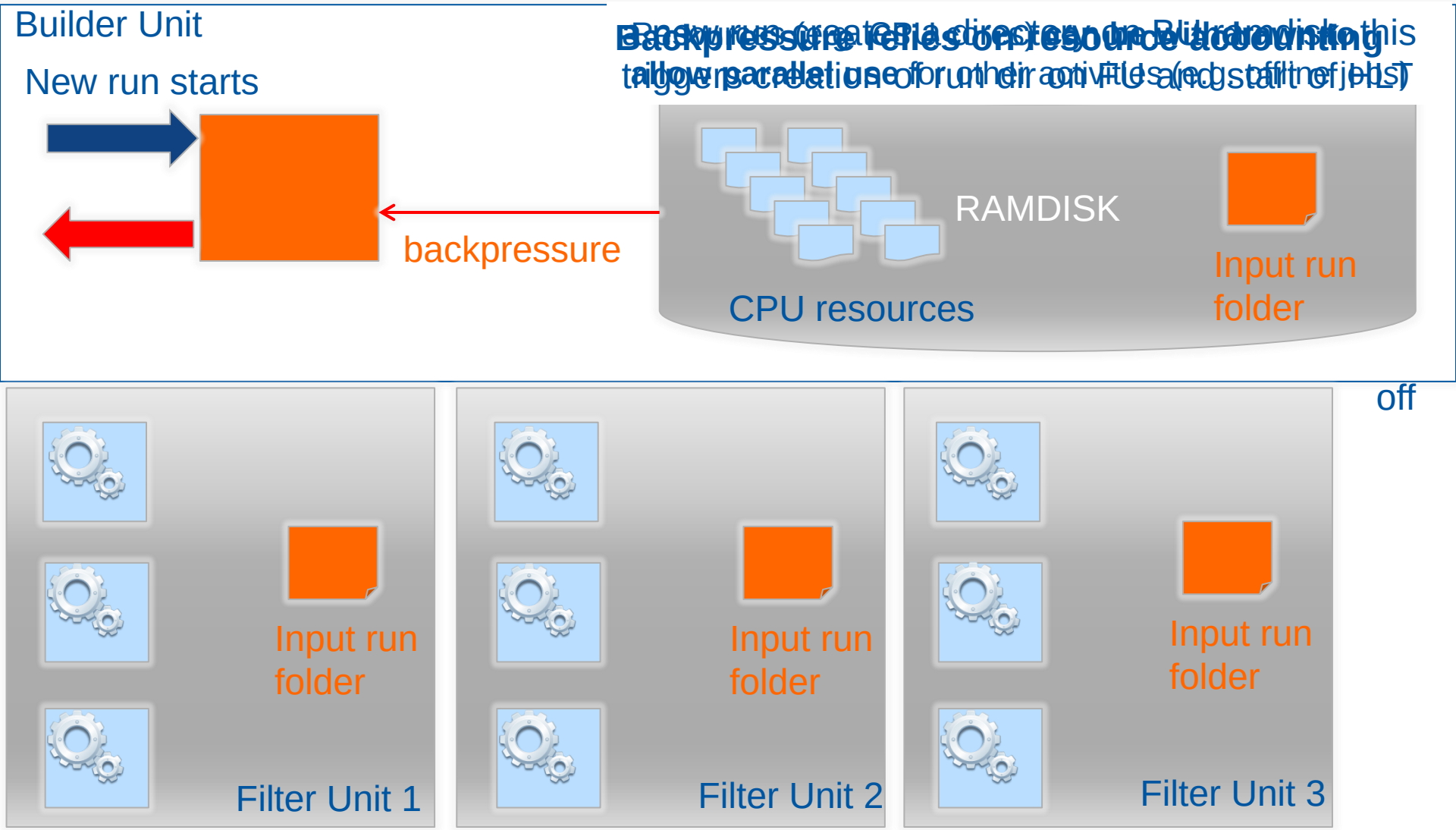


- Output categorized in **"streams"** defined by HLT configuration (i.e. physics, calibration etc.): **one stream = one file per LS** and O(10) streams in total
- **HLT processes** output to local disk using **special formats** for ease of file concatenation
- Also accommodates non-event data "streams" e.g. **Data quality histograms** and **counters for rate monitoring** of the HLT

Output performance (end-to-end)



Flow Control, Monitoring



Monitoring

Additional metadata are collected and moved around to help control the operation of the appliance

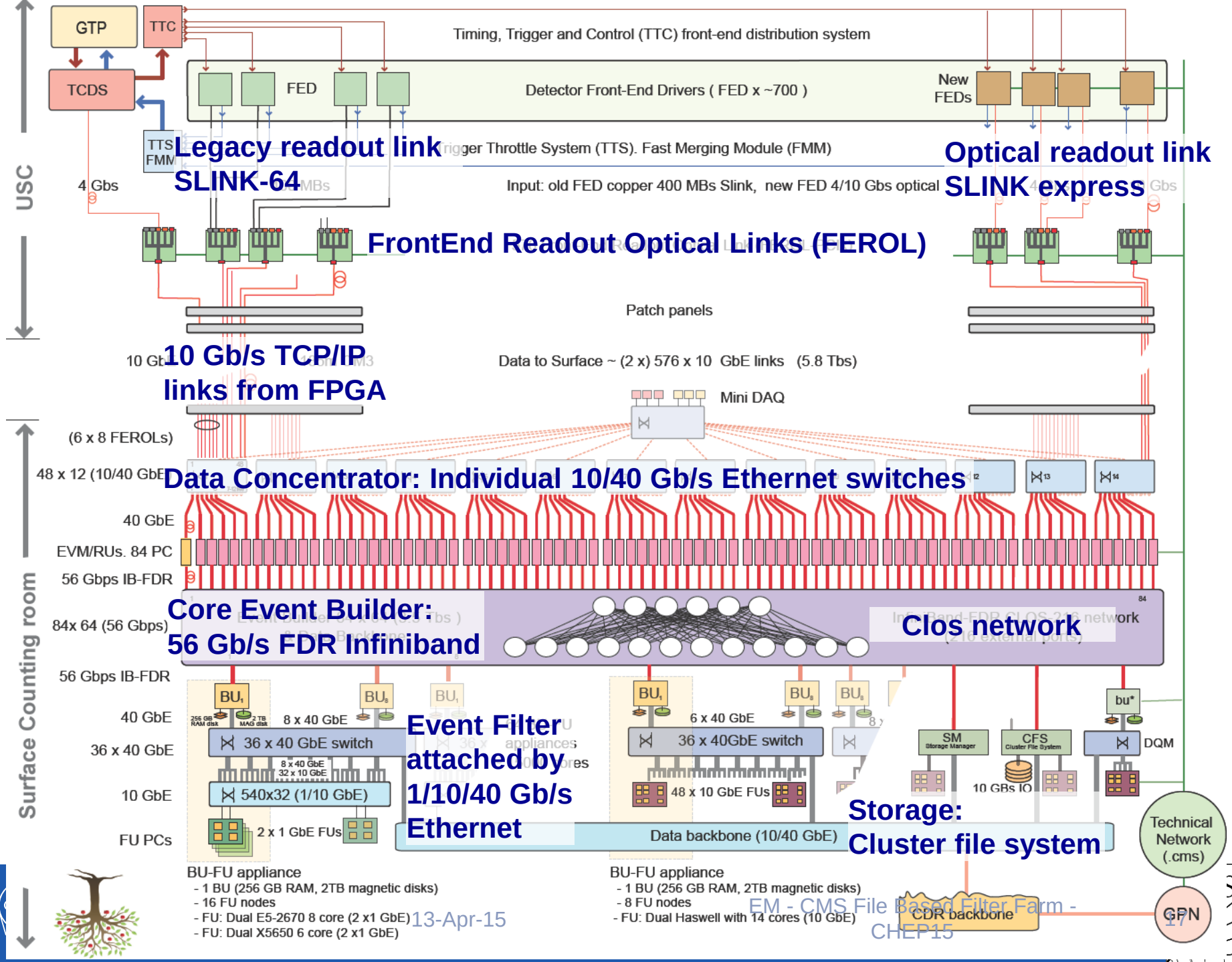
- Resource availability (disk space, number of available cores for processing)
- HLT execution flow information (state of processes, input, processing, output etc.)

As they are aggregated they are **also injected in the elasticsearch**-based monitoring (**see following talk by S. Morovic**)

Summary

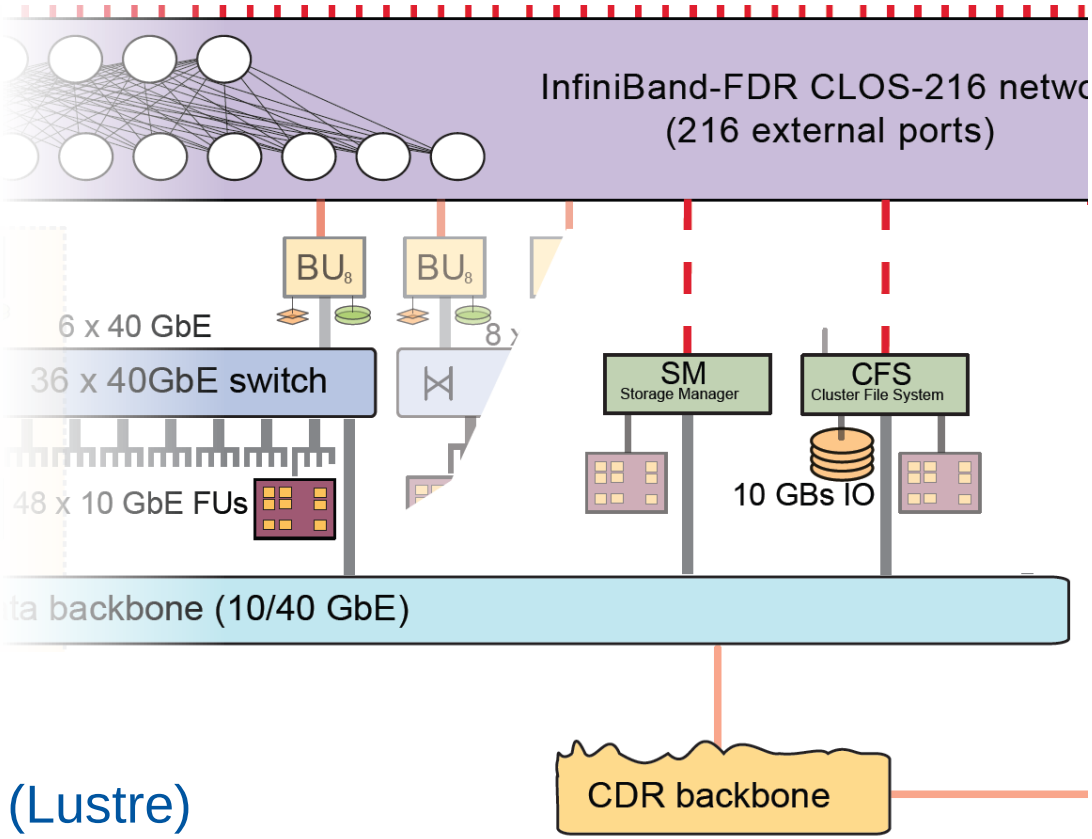
- The control, monitoring, **and data** flow of the CMS Filter Farm for Run 2 are **entirely based on files and file systems**
- The system is now in production and **performing as expected so far** (though Run 2 has not properly started yet)
- Initial measurements on the production system indicate it will meet specifications:
  - 100 kHz input, up to 200 ms HLT CPU time, ~1 kHz aggregated output
- An example of innovation by **use of conventional technology in (slightly) unconventional** ways

backup


USC
Surface Counting room
GTP
TTC
TCDS
Timing, Trigger and Control (TTC) front-end distribution system
FED
Detector Front-End Drivers (FED x ~700)
New FEDs
TTS FMM
Legacy readout link SLINK-64
Trigger Throttle System (TTS). Fast Merging Module (FMM)
Optical readout link SLINK express
4 Gbs
Input: old FED copper 400 MBs Slink, new FED 4/10 Gbs optical
FrontEnd Readout Optical Links (FEROL)
Patch panels
10 Gb/s TCP/IP links from FPGA
Data to Surface ~ (2 x) 576 x 10 GbE links (5.8 Tbs)
Mini DAQ
(6 x 8 FEROLs)
48 x 12 (10/40 GbE)
Data Concentrator: Individual 10/40 Gb/s Ethernet switches
40 GbE
EVM/RUs. 84 PC
56 Gbps IB-FDR
Core Event Builder: 56 Gb/s FDR Infiniband
84x64 (56 Gbps)
Clos network
56 Gbps IB-FDR
40 GbE
36 x 40 GbE
10 GbE
FU PCs
8 x 40 GbE
36 x 40 GbE switch
540x32 (1/10 GbE)
2 x 1 GbE FUs
Event Filter attached by 1/10/40 Gb/s Ethernet
6 x 40 GbE
36 x 40GbE switch
48 x 10 GbE FUs
Data backbone (10/40 GbE)
SM
CFS
DQM
10 GBs IO
Storage: Cluster file system
Technical Network (.cms)
GPN
CDR backbone
BU-FU appliance
- 1 BU (256 GB RAM, 2TB magnetic disks)
- 16 FU nodes
- FU: Dual E5-2670 8 core (2 x1 GbE)
- FU: Dual X5650 6 core (2 x1 GbE)
BU-FU appliance
- 1 BU (256 GB RAM, 2TB magnetic disks)
- 8 FU nodes
- FU: Dual Haswell with 14 cores (10 GbE)

Merging and storage

- File-Based Filter Farm produces output files
  - After merging on FU: 800 files x 10 streams scattered over 64 BUs every 23 seconds
  - To be merged to 1 file per stream in a central place
- Idea: Merging can be done by file system
  - Just need to find a file system that can handle it
- Solution: Global File System (Lustre)
  - Merger process on BU reads data from all FUs in appliance
  - Data are written directly from the BUs to a single output file in the global file system



Lustre Hardware (NetApp)

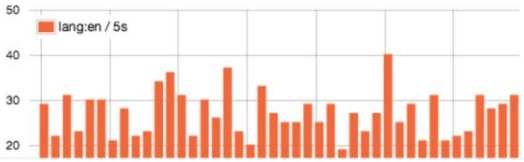
ElasticSearch

what is elasticsearch?

distributed restful search and analytics

real time data

Data flows into your system all the time. The question is … how quickly can that data become an insight? With Elasticsearch, real-time is the only time.



real time analytics

Search isn't just free text search anymore – it's about exploring your data. Understanding it. Gaining insights that will make your business better or improve your product.

distributed

Elasticsearch allows you to start small, but will grow with your business. It is built to scale horizontally out of the box. As you need more capacity, just add more nodes, and let the cluster reorganize itself to take advantage of the extra hardware.

multi-tenancy

A cluster can host multiple indices which can be queried independently or as a group. Index aliases allow you to add indexes on the fly, while being transparent to your application.

Multi-tenancy

```
$ curl -XPUT http://localhost:9200/kimchy

$ curl -XPUT http://localhost:9200/elasticsearch

$ curl -XPUT http://localhost:9200/elasticsearch/tweet/1
```

document oriented

Store complex real world entities in Elasticsearch as structured JSON documents. All fields are indexed by default, and all the indices can be used in a single query return results at breath taking speed.

Document oriented

```
$ curl -XPUT http://localhost:9200/twitter/user/kimchy -d
'{
    "name" : "Shay Banon"
}'
```

schema free

Elasticsearch allows you to get started easily. Toss it a JSON document and it will try to detect the data structure, index the data and make it searchable. Later, apply your domain specific knowledge of your data to customize how your data is indexed.

ES config for Eventfilter