DAQ and Computing visions

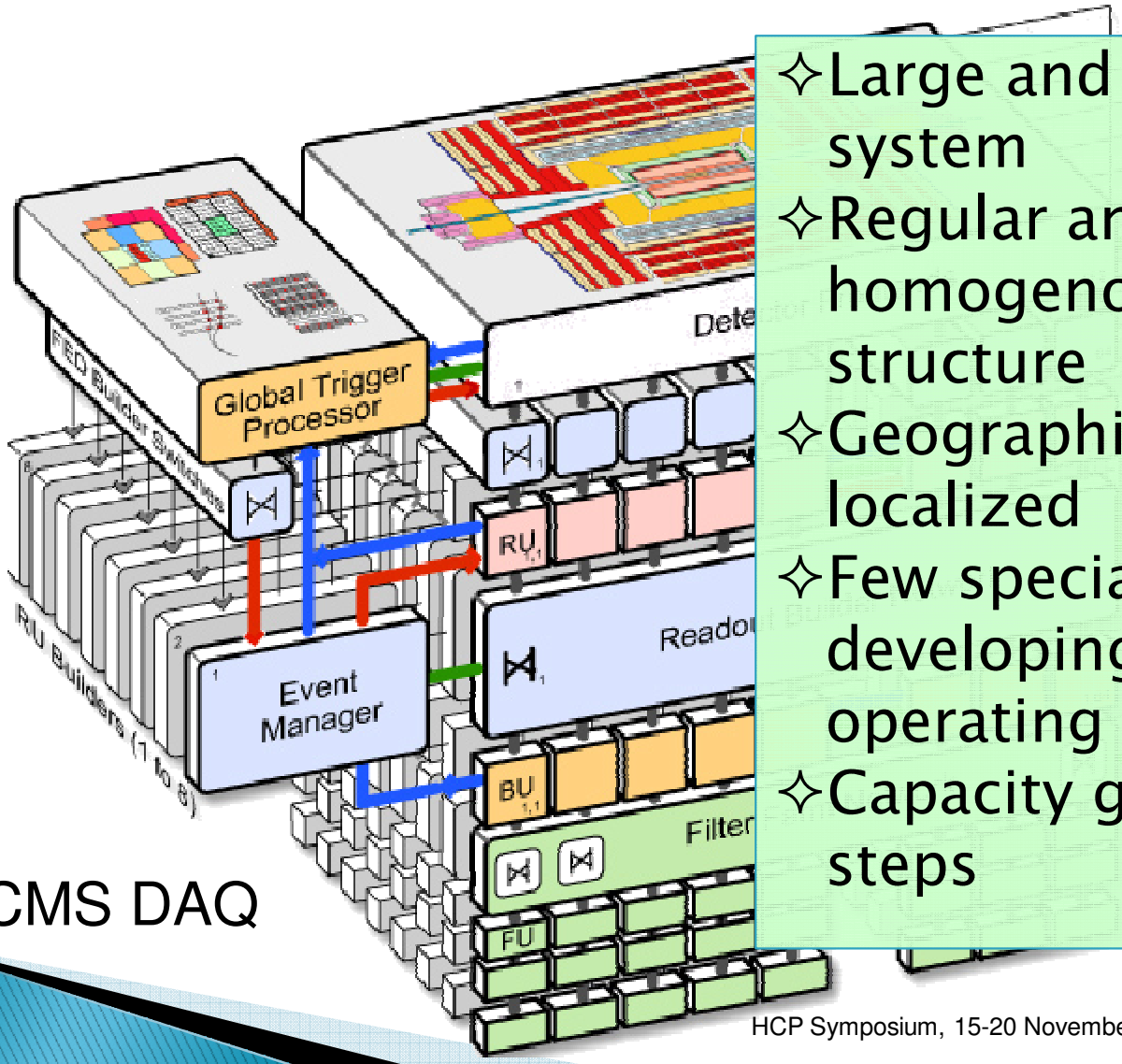
Hadron Collider Physics Symposium 15-20 November 2009, Evian, France

P. Mato /CERN

Outline

- A glimpse into the LHC DAQ systems
  - Next challenges dictated by the upgrades and future accelerators
- A glimpse into the LHC Offline Computing system
  - Next challenges and the technologies at rescue
- Some key technologies identified
- Summary

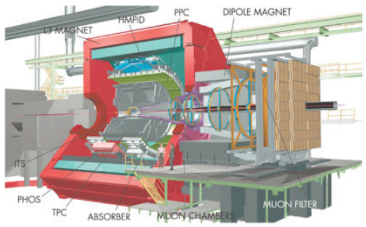
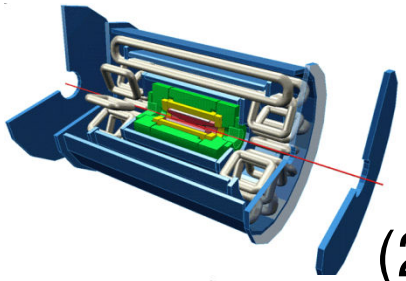
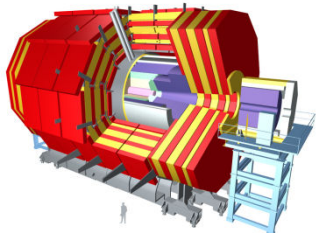
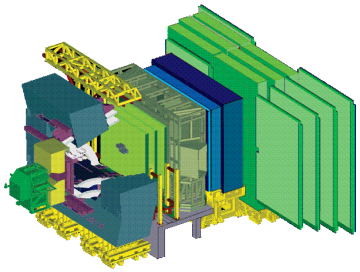
DAQ Architectures



CMS DAQ

- Large and complex system
- Regular and homogenous structure
- Geographically localized
- Few specialists developing and operating it
- Capacity grows in steps

Trigger/DAQ Parameters Today

		No.Levels Trigger		Level-0,1,2 Rate (Hz)	Event Size (Byte)	Readout Bandw.(GB/s)	HLT Out MB/s (Event/s)
ALICE		4	Pb-Pb	500	$5x10^7$	25	1250 (10^2)
			p-p	10^3	$2x10^6$		200 (10^2)
ATLAS		3	LV-1	10^5	$1.5x10^6$	4.5	300
		($2x10^2$)	LV-2	$3x10^3$			
CMS		2	LV-1	10^5	10^6	100	~1000
		(10^2)					
LHCb		2		10^6	$3.5x10^4$	35	70

Readout Architectures

- Partial vs. Full readout
  - LHCb & CMS readout everything on a first-level trigger
  - ALICE has an (optional) sparse readout
  - ATLAS has a partial, on-demand, readout (Level-2) seeded by the Region of Interest (ROI) followed by a full readout
- Pull vs. Push
  - Push is used by everybody from the front-end (with backpressure except LHCb)
    - ATLAS & CMS pull in the global event-building
    - ALICE pushes over TCP/IP (implicit rate-control)
    - LHCb uses push throughout (with a global backpressure signal and central control of FE buffers)

Point-to-Point Networks

- Readout based on buses are history
  - E.g. LEP with Fastbus and VME
- All readout is on point-to-point links in Local Area Networks
  - except the sub-event building in “readout-unit” PC-servers (the last stand of the buses)
- Many have been called forward, few have been chosen: SCI, Myrinet, ATM, Ethernet (100 Mbit), Ethernet (1000 Mbit), InfiniBand

Convergence

- Readout links of quite similar characteristics
  - The new GBT (GigaBit Transceiver) has the potential to become the standard combining DAQ, Trigger and DCS
- COTS hardware as much as possible
- LAN technology (actually all core Ethernet in the LHC DAQs comes from the same vendor)
- Standard protocols: Ethernet, UDP, (TCP)/IP
- Message coalescing techniques to keep message rates under control
  - for LHCb this is an issue even for the data packets

sLHC Challenges

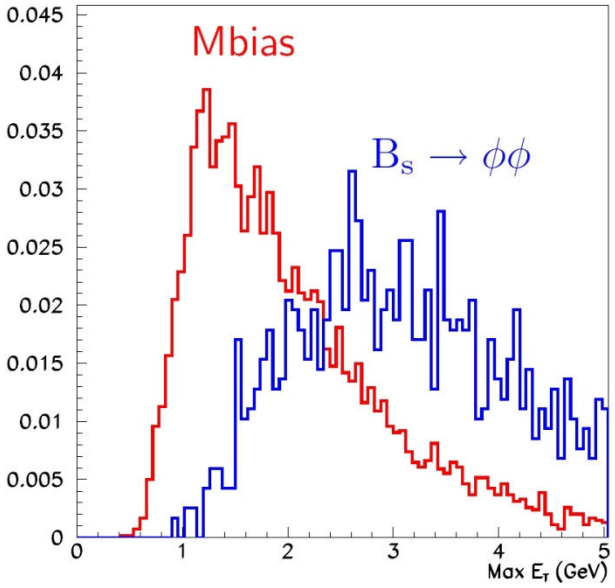
- Replacement of Vertex/Tracking detectors
  - More channels
- Participation of Si-Trackers in hardware triggers
  - More connectivity, more complex programs
- Increase Luminosity
  - More pile-up, higher trigger rates, higher data bandwidth
- No major re-designs foreseen in DAQ
  - The strategy is scaling current Readout Architectures
  - The most challenging one is perhaps the LHCb upgrade

➔ Scaling current readout architectures
  - Technology in our side (Links, FPGAs, Networks, Switches, etc.)

➔ Scaling HLT computing farms
  - More cores, Moore's law, etc.
  - One HLT process per core approach

Example: LHCb Planned Upgrade

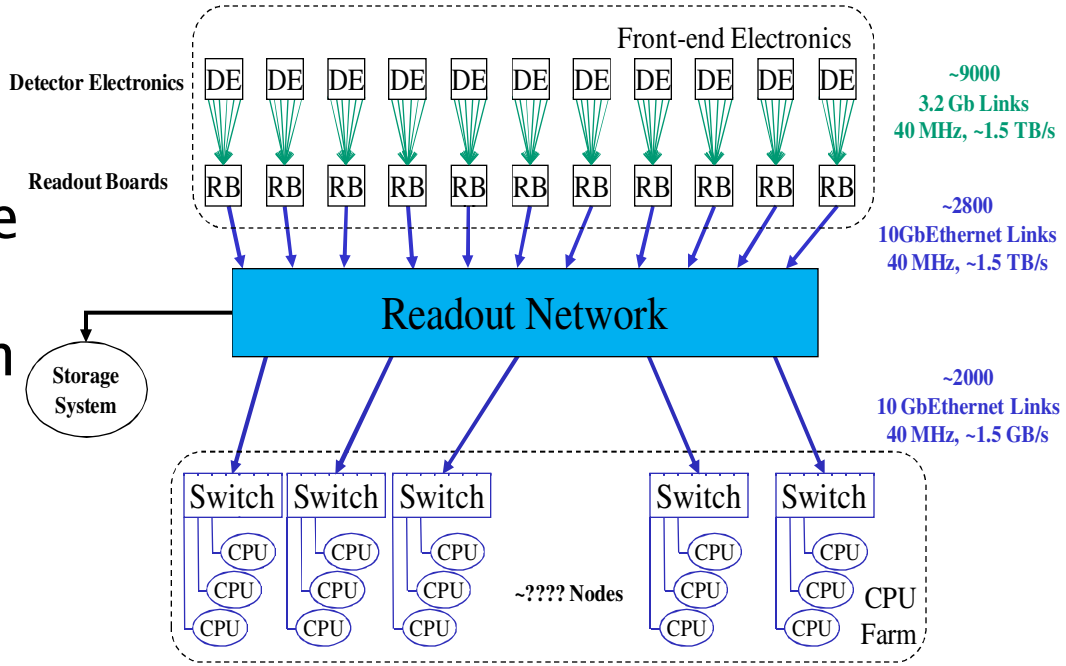
- Motivated by the low Level-0 Hadron efficiency [50-60%] compared to Muon or Electron channels [80-90%]
  - In order to keep the minimum bias contamination low (<<1 MHz) have to cut severely into the signal (E_T cut ~3 GeV)
- 40 MHz readout eliminating hardware Level-0 trigger
  - No more hardware trigger
  - Full software flexibility for event selection
- Increase Luminosity from $2x10^{32}cm^{-2}s^{-1}$→$10-20x10^{32}cm^{-2}s^{-1}$
  - More pile-up → more complexity → bigger events



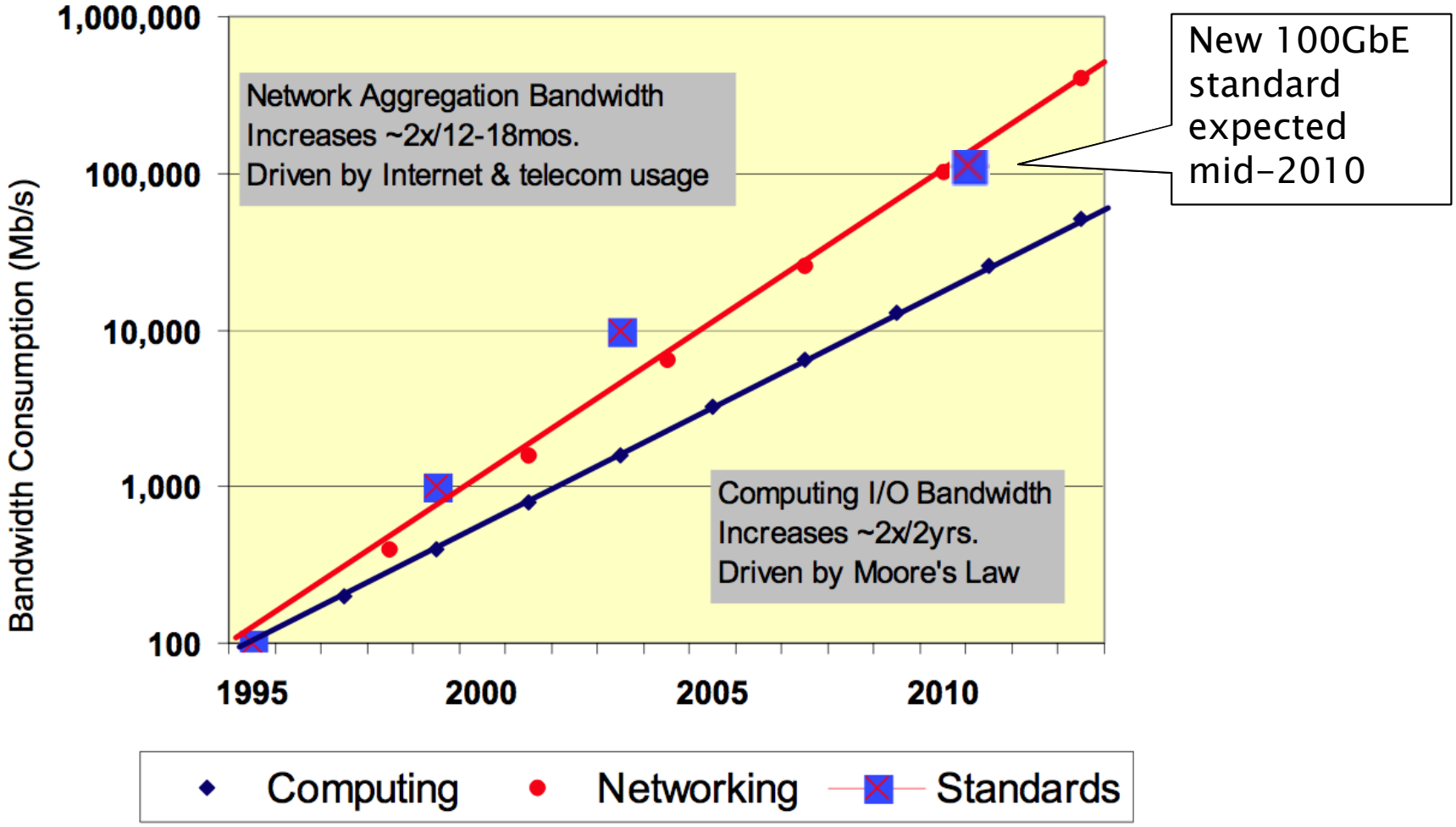
LHCb 40-MHz Readout

- On-detector electronics has to implement zero-suppression
  - Keep cost for links acceptable
- Need ~10-fold increase in switching capacity
- Need x-fold increase in CPU power (x >> 40).
- New Front-End Electronics
- More sophisticated software algorithms to cope with more pile-up

40-MHz Architecture

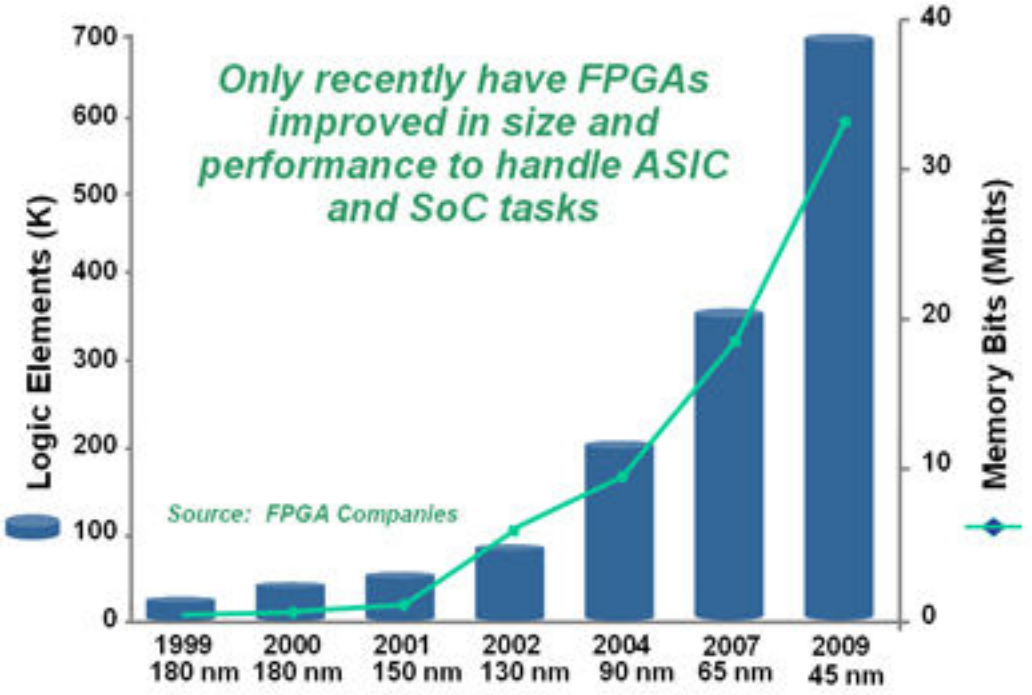


Ethernet Trends



FPGAs Trends

- FPGA trends follows Moore's law
  - FPGA's are growing in capability and capacity
  - Main manufacturers have announced devices with more than 700k logic elements
- Programming such devices will be challenging



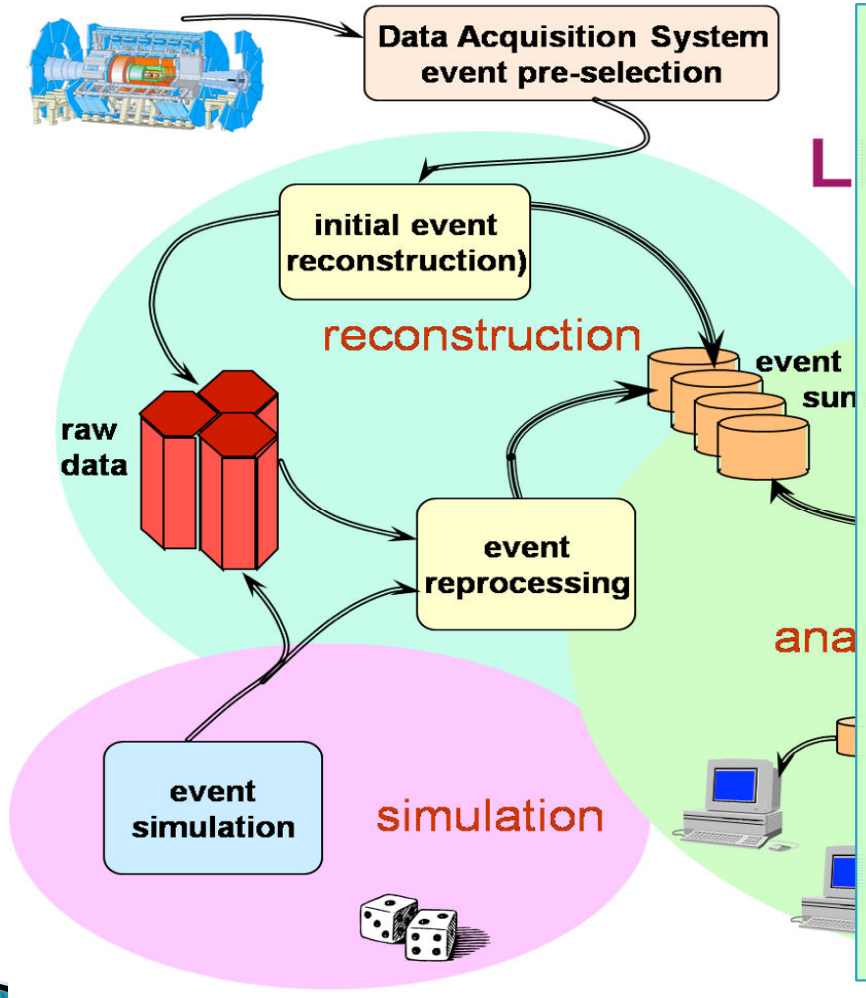
Outlook on Future Accelerators

Beat Jost, Niko Neufeld (LHCb)

- The future will be in trigger-free DAQ systems
  - Either dictated by the bunch structure of the machine
    - 0.6 ns bunch spacing at CLIC
    - 'Digital Oscilloscope' connected to each channel
  - Or by choice since technology allows it (LHCb upgrade) because of flexibility and (possibly) improved efficiency and simplicity
- Special case might be continuous high-frequency accelerators
  - E.g. SuperB: 2 ns bunch spacing continuous operation relatively low interaction rate
    - 500 MHz collision rate
    - $\Upsilon(4S)$ Cross section ~5 nb
    - Luminosity 1036cm-2s-1
      → hadronic event-rate ~5 kHz
    - Activity trigger might be indicated

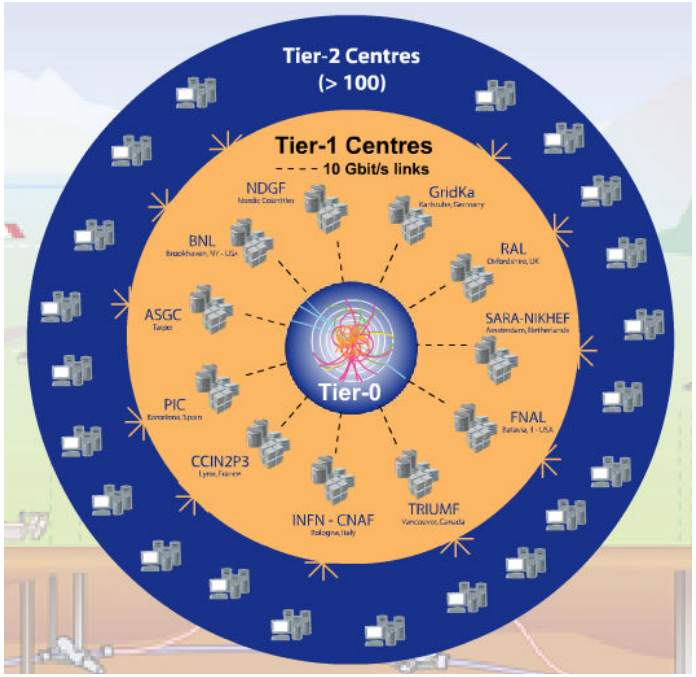
Offline Computing

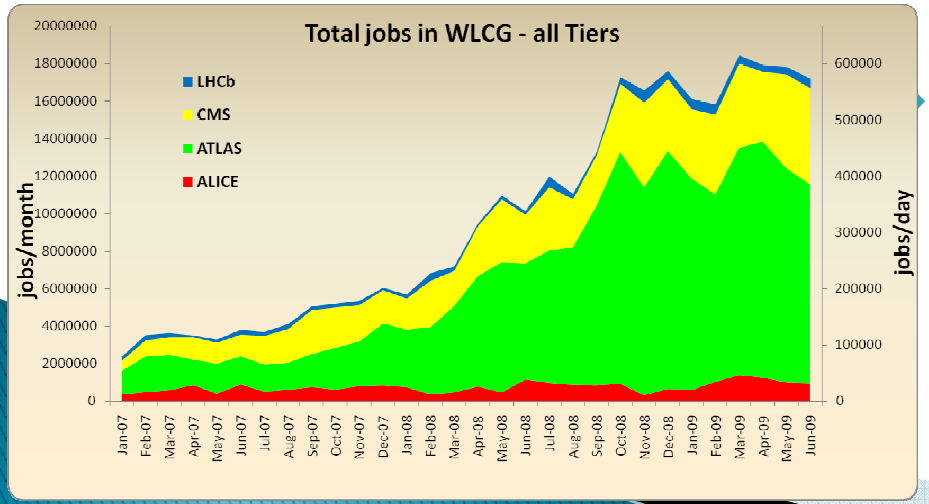
Offline Data Flow and Processing



- ✧Large and very complex system
- ✧Heterogeneous and irregular structure
- ✧Geographically distributed worldwide
- ✧Many specialists and non-specialist developing and operating it
- ✧Capacity grow continuously

LHC Computing Grid in Numbers



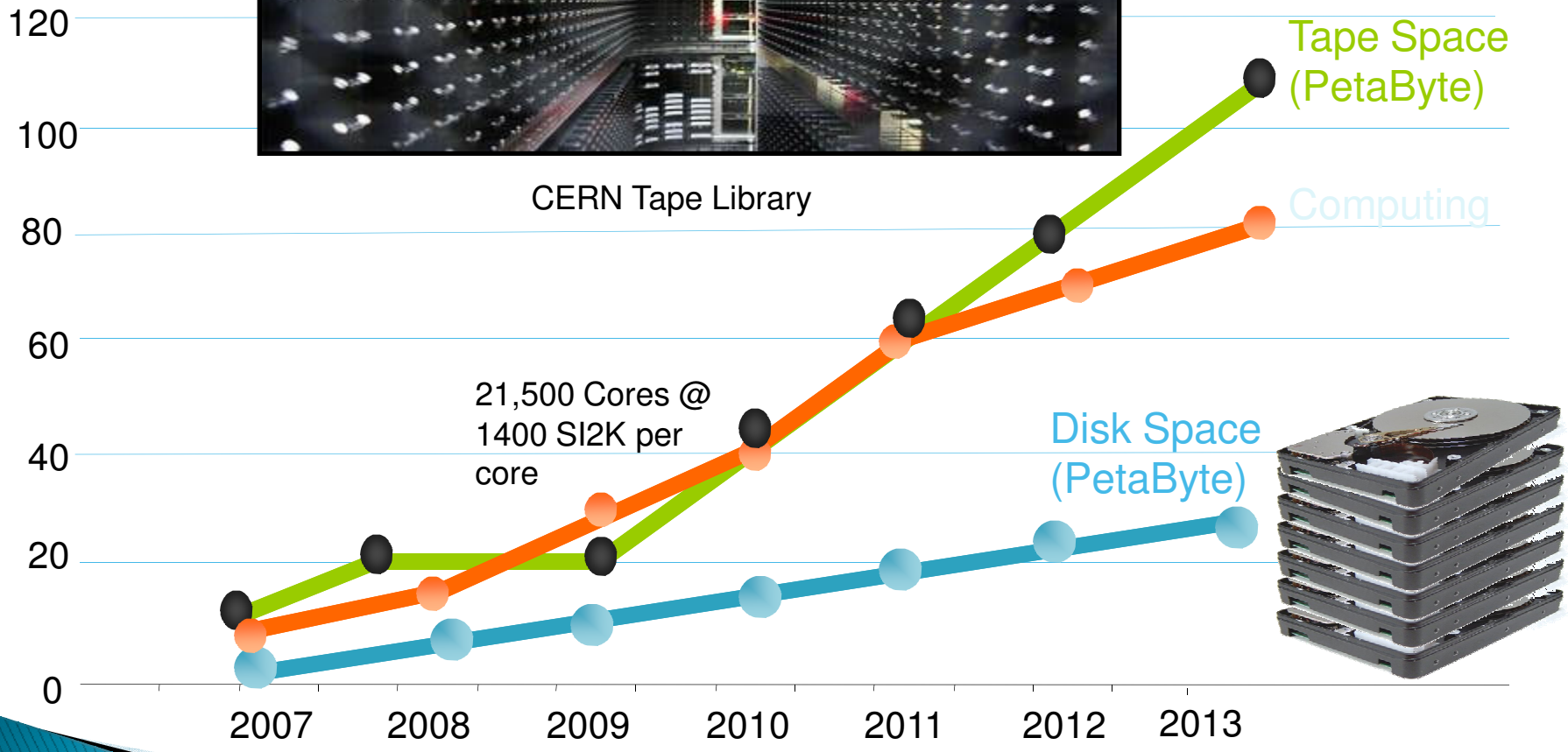


- Tier-0 (CERN): data recording, initial data reconstruction, data distribution
- Tier-1 (11 centres): permanent storage, re-processing, analysis
- Tier-2 (~130 centres): simulation, end-user analysis
- Installed Capacity (Sep-2009)
  - CPU: 78.600 kSI2K (~40.000 cores)
  - Disk: 31 PB
  - Tape: 49 PB
- 600.000 job/day
  - + 45% in a year

A look at CERN's Computing Growth

CERN Tape Library

Source: CERN, Jarp Sverre



Lots of computing (45% CAGR), lots of data; no upper boundary!

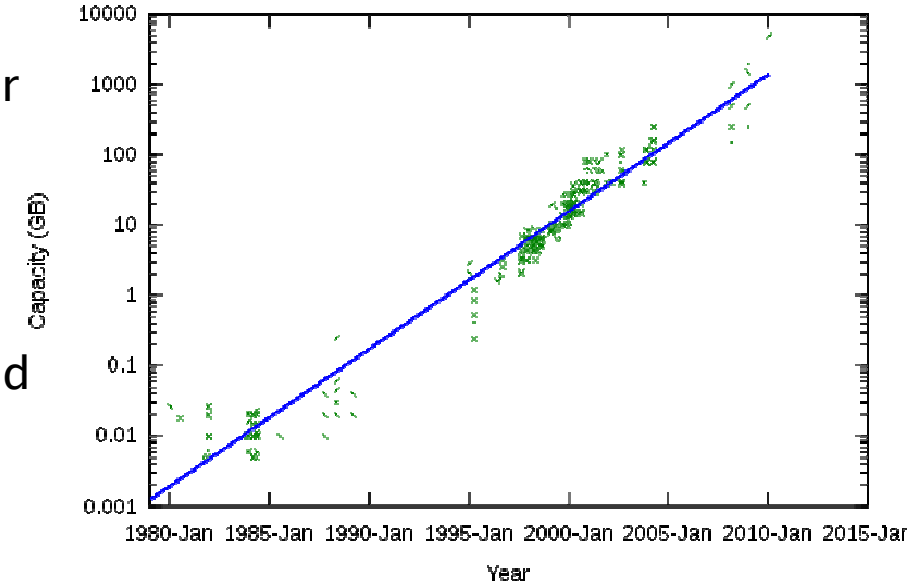
Challenges Ahead

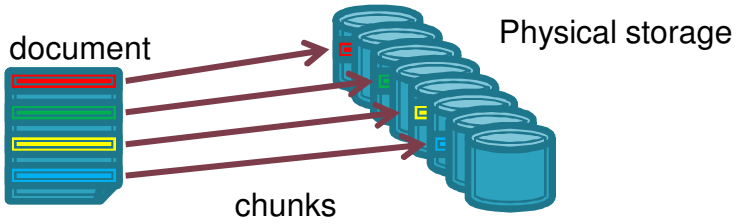
- Increase in Storage demands
  - HDD technology scales well (1.6X/year)
  - SSD very promising ((1.5X/year, expected 2X less power with 140x performance gain for 2018)
  - Major challenges: failures, data management, hierarchy of caches, etc.
- Networks
  - Expected that will follow (general public demand)
- Increase in computational demands
  - Scaling blocked by 'three walls' (memory, power, micro-parallelism)
  - Multi-core architectures, GPUs, etc.
- Deployment complexity
- Increased software complexity

Data Storage and Managment

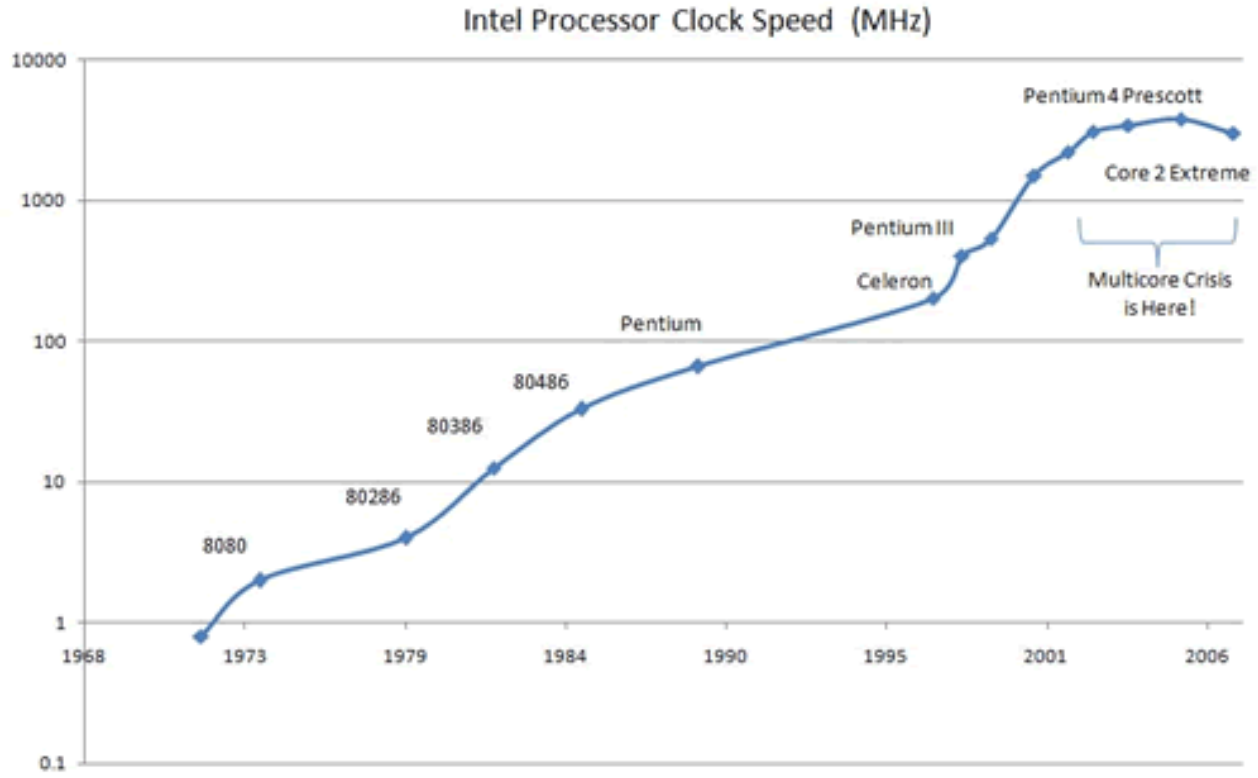
- Capacity increase should not be a problem (disks, tapes)
  - Scalability restored using bigger files
  - Latency is a problem
- Failures will be norm
  - Data replication/duplication
  - File split in chunks with encoded and redundant information
- I/O bandwidth can only be achieved with a hierarchical organization of caches
  - Similar to CPU memory caches
  - SSD -> (local) HDD -> (remote) HDD -> Tape
  - Cache coherency issues

Single HDD capacity





Clock Frequency Crisis



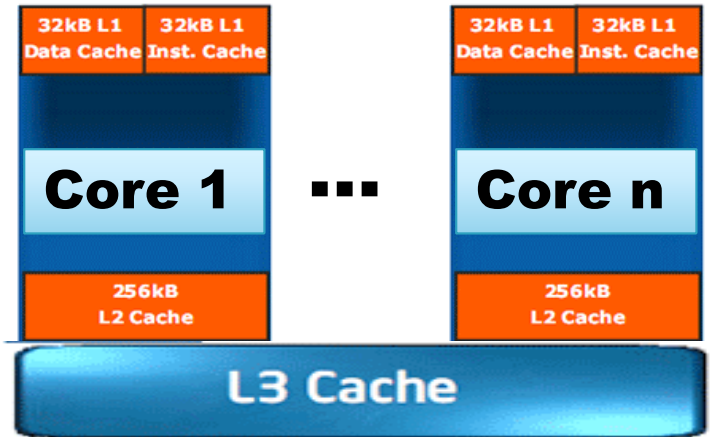
- Clock speed wins most of the time
- More CPU cores isn't automatically better
- Free lunch is over

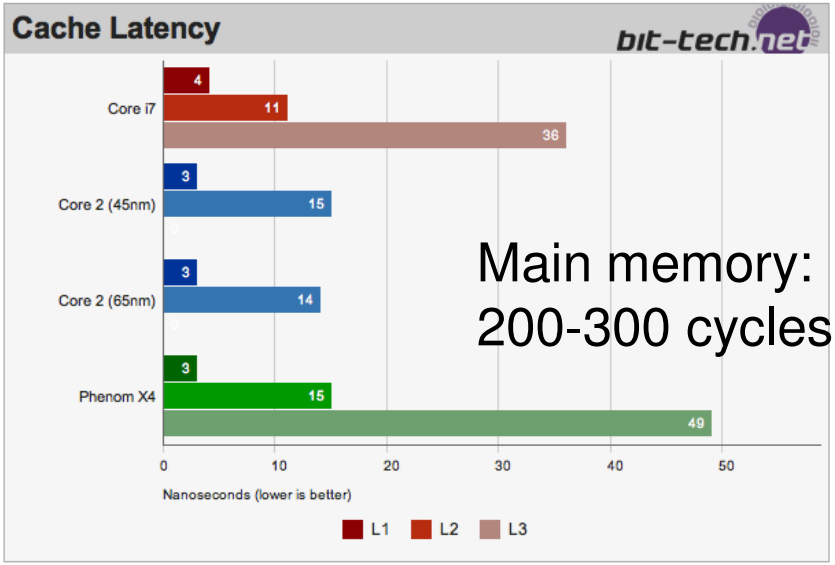
The 'three walls'

- While hardware continued to follow Moore's law, the perceived grow of the "effective" computing power faded away in hitting three "walls"
- The memory wall
  - Processor clock rates faster than memory clock rates
- The power wall
  - Reducing clock frequency by 20 percent saves half the power
- The instruction level parallelism (micro-architecture) wall
  - Longer and fatter parallel instruction pipelines. Data dependencies and mispredictions became very expensive
- A turning point was reached and a new paradigm emerged: multi-core

The ‘memory wall’

- Processor clock rates have been increasing faster than memory clock rates
- larger and faster “on chip” cache memories help alleviate the problem but does not solve it.
- Latency in memory access is often the major performance issue in modern software applications



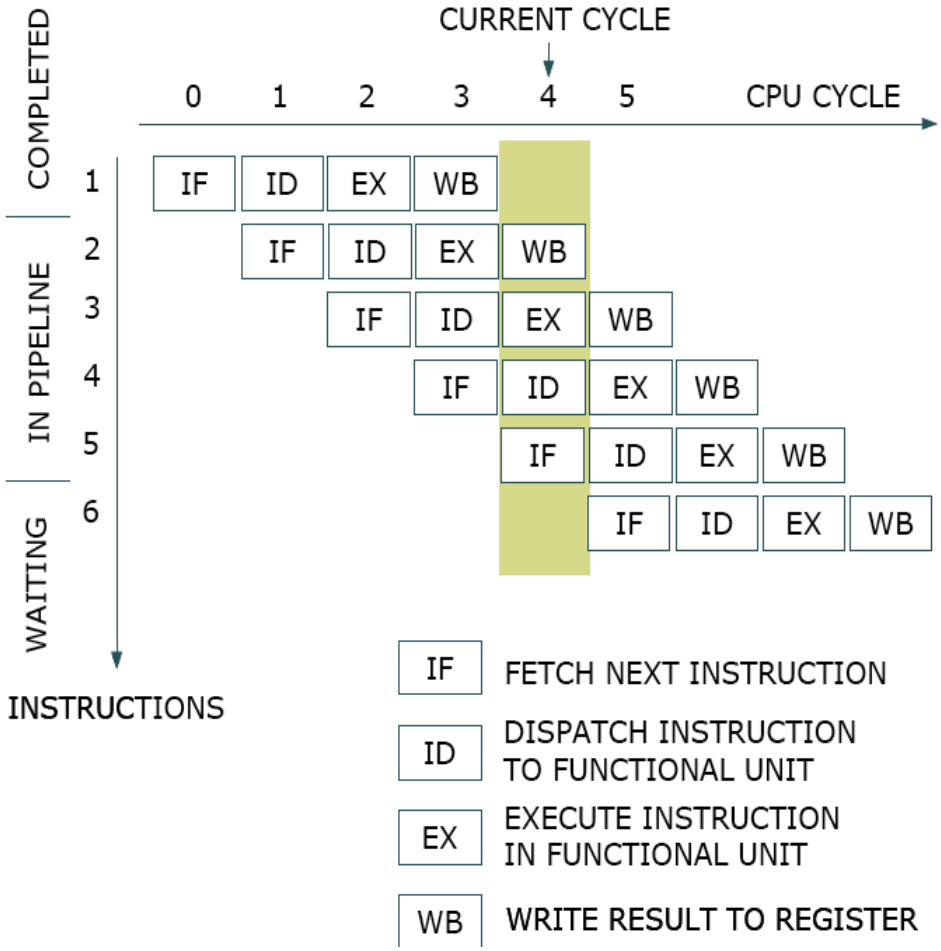


The 'power wall'

- Processors consume more and more power the faster they go
- This is not linear:
  - 73% increase in power gives just 13% improvement in performance
- Many computing center are today limited by the total electrical power installed and the corresponding cooling/extraction power
- How else increase the number of instruction per unit-time: ➔ Go parallel!

The 'Architecture walls'

- Longer and fatter parallel instruction pipelines has been a main architectural trend in `90s
- Hardware branch prediction, hardware speculative execution, instruction re-ordering (a.k.a. out-of-order execution), just-in-time compilation, hardware-threading are some notable examples of techniques to boost ILP
- In practice inter-instruction data dependencies and run-time branching limit the amount of achievable ILP



Where do we stand?

- HEP code does not exploit the power of current processors
  - One instruction per cycle at best
  - Little or no use of vector units (SIMD)
  - Poor code locality
  - Abuse of the heap
- Running N jobs on N=8 cores still efficient but:
  - Memory (and to less extent cpu cycles) wasted in non sharing
    - "static" conditions, geometry data and field maps
    - I/O buffers
    - Network and disk resources
  - Caches (memory on CPU chip) wasted and trashed
    - Not locality of code and data
- This situation is already bad today, will become only worse in future architectures

HEP software on multi-core R&D Effort

- Collaboration among experiments, IT-departments, projects such as OpenLab, Geant4, ROOT, Grid
- Target multi-core (8-24/box) in the short term, many-core (96+/box) in near future
- Optimize use of CPU/Memory architecture
- Exploit modern OS and compiler features
  - Copy-on-Write
  - MPI, OpenMP
  - SSE/AltiVec, OpenCL
- Prototype solutions
  - Adapt current software
  - Look for innovative solution for the future

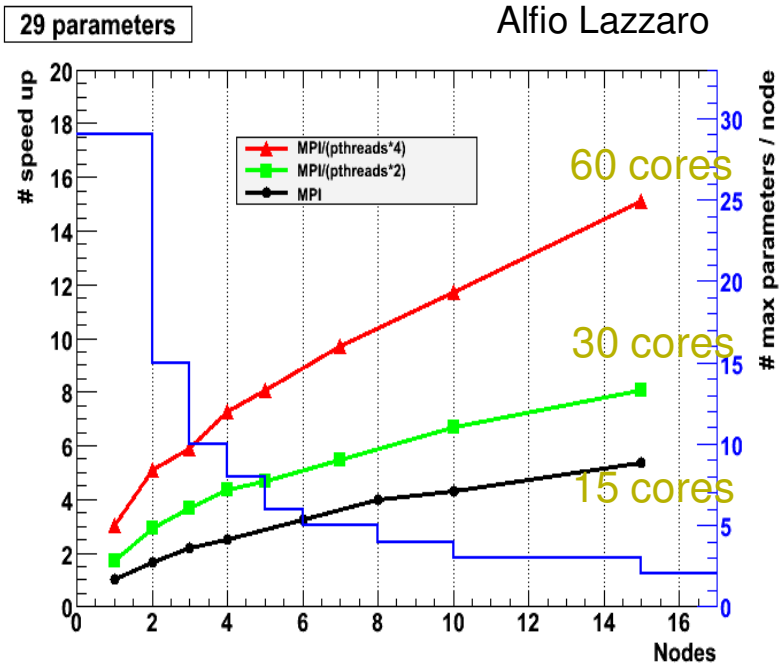
Current R&D Activities

- Parallelization of Data Processing Frameworks
  - The idea here is to get the benefits of multi-core architectures with minimal effort from developers
- PROOF-lite
  - Realization of PROOF in 2 tiers
  - Transparent to users to the analysis of TTrees
  - The client starts and controls directly the [NCPU] workers. No daemons
- Gaudi Parallel
  - Run Gaudi applications in parallel.
  - Transparent to the user (--parallel=N). The only noticeable thing should be that it completes in a fraction of the time
  - Same input and output files

Current R&D Activities (2)

- Progress toward a thread-parallel Geant4
  - Event-level parallelism with separate events on different threads
- Ultimate performance gain will come from parallelizing algorithms
  - Prototypes using posix-thread, OpenMP, MPI
  - Positive experience with Parallel Minuit
    - Waiting time for fit to converge down from a week to a night

➔ iteration on results back to a human time scale!



Using Multi-Core Architectures

- Recent progress shows that we shall be able to exploit next generation multi-core with “small” changes to HEP code
  - Exploiting these multi-core architectures (MT, MP) can optimize the use of resources (memory, I/O)
- Batch [Grid] capacity is realized as one job per core
  - Not ideal to exploit the new architectures
- Submitting a single job per node that utilizes all available cores can be an advantage
  - Efficient in resources, mainly increasing the fraction of shared memory
  - Scaling down the number of jobs [and files] that the Grid needs to handle

New Frontier of parallel computing

- Hardware and software technologies (SSD, KSM, OpenCL, GPU) may come to the rescue in many areas
  - We shall be ready to exploit them
- Scaling to many-core processors (96-core processors foreseen for next year) will require innovative solutions
  - MP and MT beyond event level
  - Fine grain parallelism (OpenCL, GPUs, etc.)
  - Parallel I/O
  - Use of "spare" cycles/core for ancillary and speculative computations
- Algorithm concept have to change: Think Parallel!

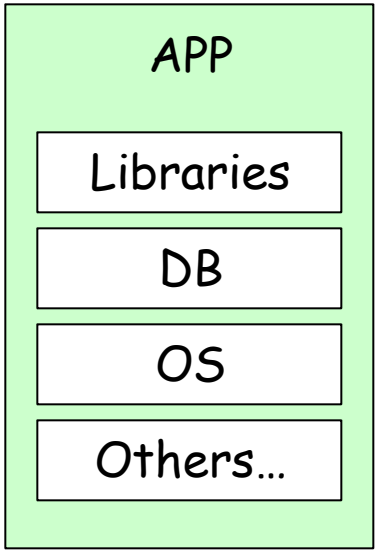
Deployment Challenges

- Installing the "complete software environment" in the Physicist's desktop/laptop [or Grid] to be able to do data analysis for any of the LHC experiments is complex and manpower intensive
  - In some cases not even possible if the desktop/laptop OS does not match any of the supported platforms
  - Application software versions change often
  - Only a tiny fraction of the installed software is actually used
- High cost to support large number of compiler-platform combinations
- The system infrastructure cannot evolve independently from the evolution of the application
  - The coupling between OS and application is very strong

Rethinking Application Delivery

- Emphasis in the 'Application'
  - The application dictates the platform and not the contrary
- Application (e.g. simulation) is bundled with its libraries, services and bits of OS
  - Self-contained, self-describing, deployment ready
- What makes the Application ready to run in any target execution environment?
  - e.g. Traditional, Grid, Cloud

➔ Virtualization is the enabling technology



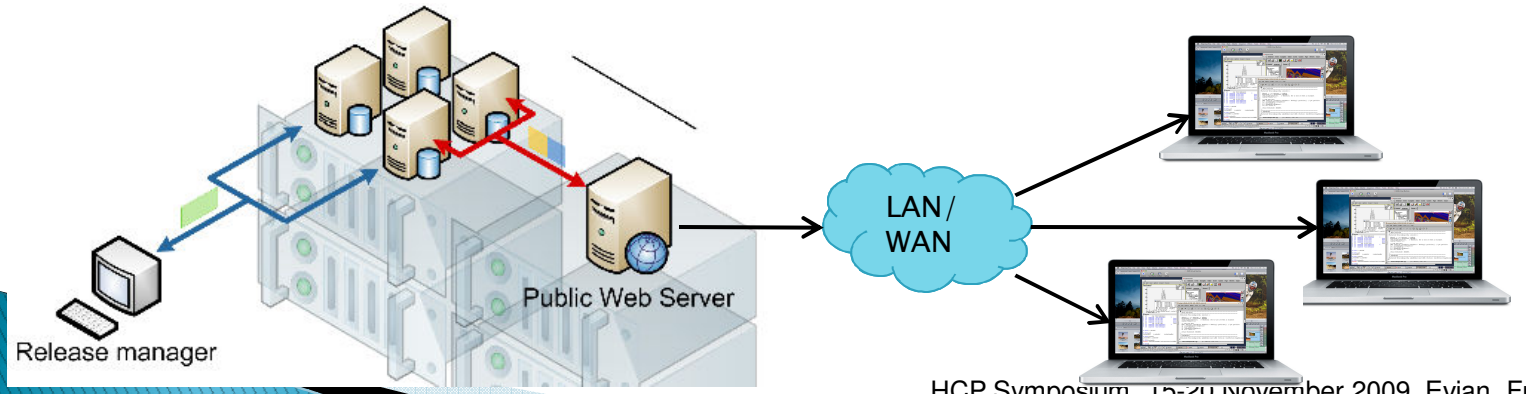
The CernVM Platform

- Provides a complete, portable and easy to configure user environment for developing and running LHC data analysis locally and on the Grid independent of physical software and hardware platform (Linux, Windows, MacOS)
  - Code check-out, edition, compilation, local small test, debugging,…
  - Grid submission, data access…
  - Event displays, interactive data analysis,
  - Suspend, resume…
- Decouple application lifecycle from evolution of system infrastructure
- Reduce effort to install, maintain and keep up to date the experiment software

http://cernvm.cern.ch

Application Software Delivery

- CernVM comes with the read-only file system (CVMFS) optimized for software distribution
  - Very little fraction of the software is actually used (~10%)
  - Very aggressive local caching, web proxy cache (squids)
  - Operational in off-line mode
- Moving towards using an existing Content Delivery Networks to remove single point of failure (e.g. Amazon CloudFront)



Cloud Computing

- The convergence of three major trends
  - Virtualization – Applications separated from infrastructure
  - Utility Computing – Capacity shared across the grid
  - Software as a Service – Applications available on demand
- Commercial Cloud offerings can be integrated for several types of work such as simulations or compute-bound applications
  - Pay as you go model
  - Question remains in their data access capabilities to match our requirements
  - Good experience from pioneering experiments (e.g. STAR MC production on Amazon EC2)
  - Ideal to absorb computing peak demands (e.g. before conferences)
- Science Clouds start to provide compute cycles in the cloud for scientific communities

Virtual Clusters

- Cloud computing (IaaS, Infrastructure as a Service) should enable us to 'instantiate' all sort of virtual clusters effortless
  - PROOF clusters for individuals or for small groups
  - Dedicated Batch clusters with specialized services
  - Etc.
- Turnkey, tightly-coupled cluster
  - Shared trust/security context
  - Shared configuration/context information
- IaaS tools such as Nimbus would allow one-click deployment of virtual clusters
  - E.g. the OSG STAR cluster: OSG head-node (gridmapfiles, host certificates, NFS, Torque), worker nodes: SL4 + STAR

Summary

- Very high overview of the DAQ and Computing systems for LHC – unusual in a single talk
  - Trying to identify, in my personal view, what will be the next challenges ahead of us
- The common impression is that technology evolution is in our side (networks, storage, computing, etc.)
  - Certainly not driven by our requirements
- Highlighted two technologies: multi-core architectures and virtualization
  - For the first we must to adapt our software/applications otherwise we are in big trouble
  - For the second we can exploit it to drastically simplify our software deployment and jump on the the Cloud-wagon