Computing evolution of ATLAS and CMS

ATLAS: ***Alessandro De Salvo***
CMS: ***Tommaso Boccali, Daniele Bonacorsi, Claudio Grandi***
15-5-2012

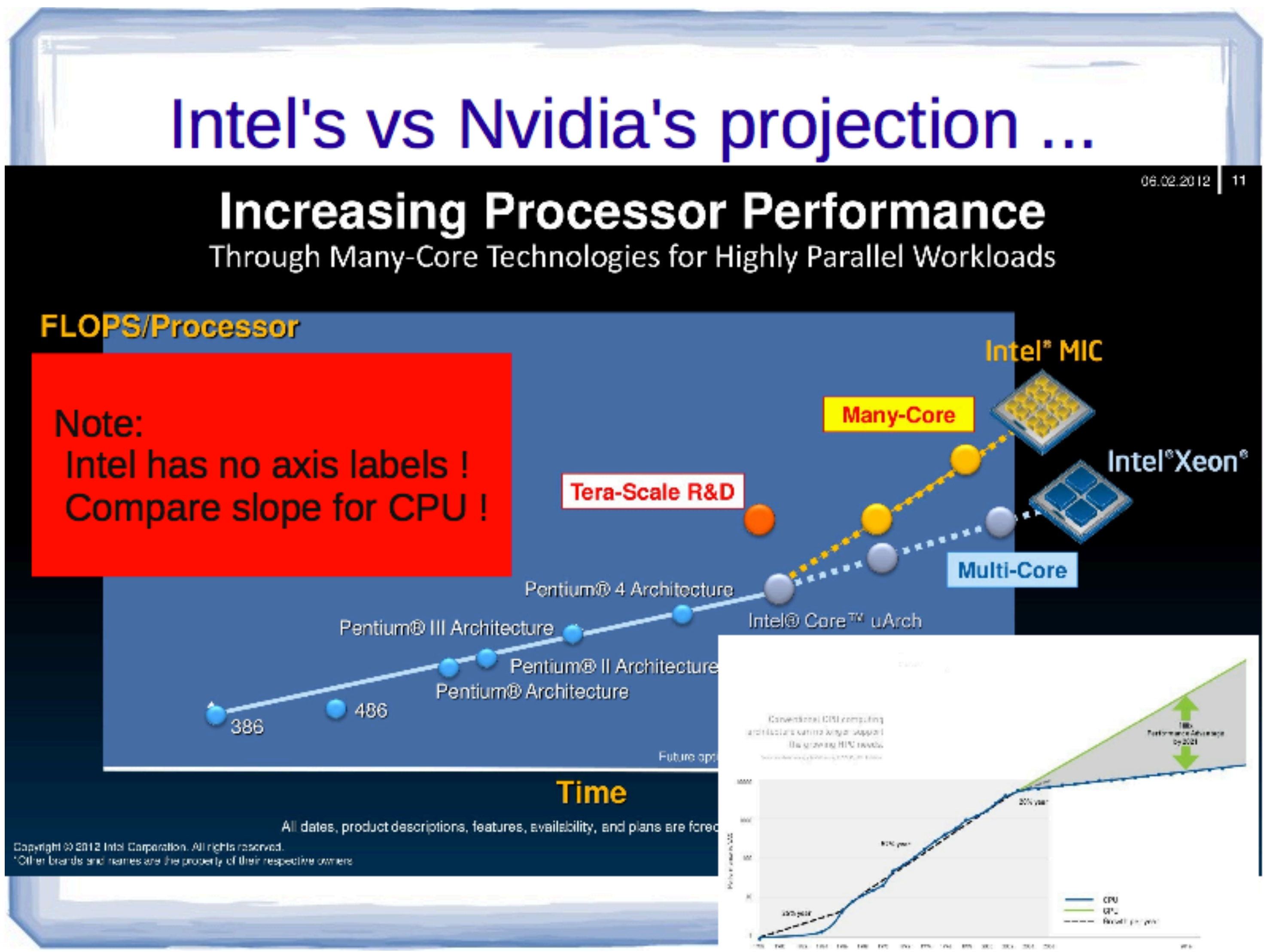
Outline

- **ATLAS computing upgrades**
- **CMS computing upgrades**

Why we need upgrades to our computing models?

- **We can identify at least 5 drivers for upgrade computing**
  1. consequences of increasing pile-up, event complexity and size
  2. consequences of new detectors and triggers
  3. consequences of increasing sample size
  4. consequences of new architectures
  5. consequences of new software technologies

New architectures of Processors



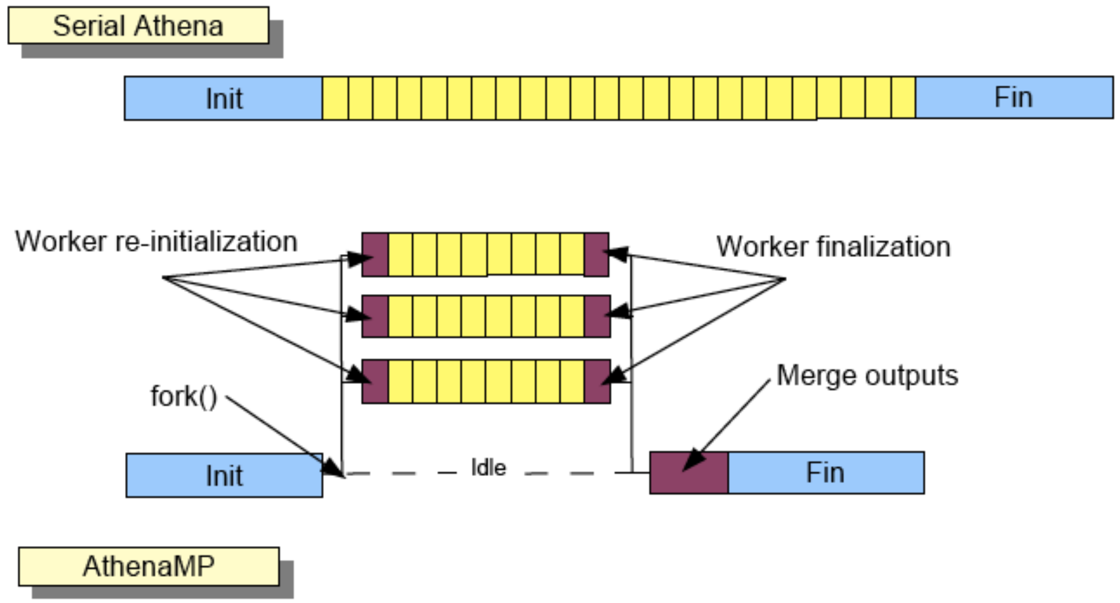
- **Going towards many-core architectures**
  - **Emerging consensus that to profit in future from performance increase code must utilize many core platforms**
  - **ATLAS and CMS software will have to be modified to fit new CPU platforms**
- **Technology may develop faster than we expect**
  - **Industry may require us for best performance to go to many/multi core already before Phase-1 (2014...)**

Implications of architectures' evolution

- **The frameworks are already adapting to new architectures**
- **Different approaches**
  - **WholeNode**
    - preferred by CMS (job handles the machine)
  - **Multicore**
    - generally preferred by site admins to keep the sites busy
      - we will probably converge on this one
- **Beyond this, the software algorithms must also adapt (reconstruction, simulation)**
  - **Concurrency, parallel programming**
- **The distributed computing must also adapt to make efficient use (e.g. whole-node scheduling)**

Multiprocessing in ATLAS

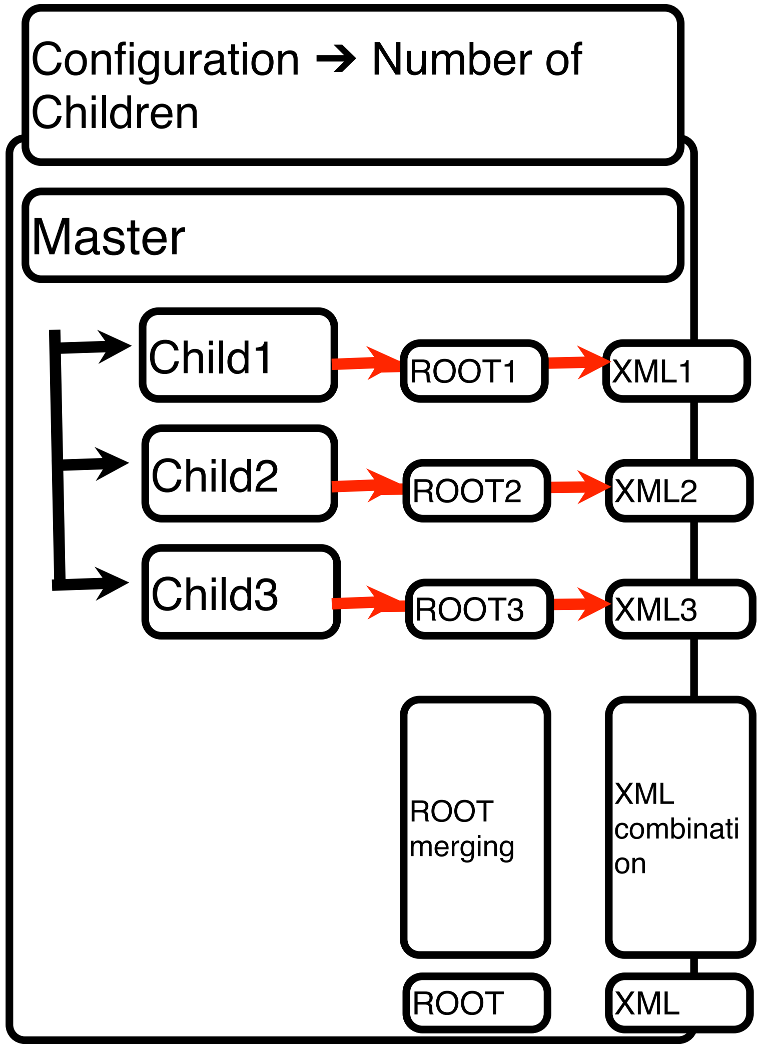
- **Current approach**
  - **A variation of the single-thread framework (Athena), called AthenaMP, is being rolled-out**
    - Athena uses N threads
    - Events are split into blocks
    - Each block is processed by a separate AthenaMP thread (workers)
    - A common thread collects the results and handles the output (finalize)
  - **Helps control the memory issues, but has limitations by ~32 cores**
  - **Synergies: AthenaMP being investigated for HLT**
  - **Extensive work on IO – new framework planned**
  - **A re-write of the Gaudi core is also likely**
  - **Process management migrated h Python multiprocessing to a custom C++ library**



- **Actively working to adapt the software algorithms too**
  - **Efficiency and architectural work is starting**
    - Optimization of existing code
    - Exploring GPGPUs, especially for tracking

Multiprocessing in CMS

- **Current implementation uses the cores via fork**
  - **Relying on the Kernel COW to allocate only once the shared memory payloads**
- **JobWrapper configures the number of children**
  - **Either via workflow settings (Manycore)**
  - **Or using /proc/cpuinfo to use the whole node**
- **JobWrapper executes a single CMSSW job producing master xml file and multiple FrameworkJobReport.xml and output files**
  - **Like MyAnalysis_0.root, MyAnalysis_1.root, … MyAnalysis_N.root**
- **JobWrapper merges all ROOT files and stages it out to MSS and also combines all xml into one**



Multiprocessing performance in ATLAS

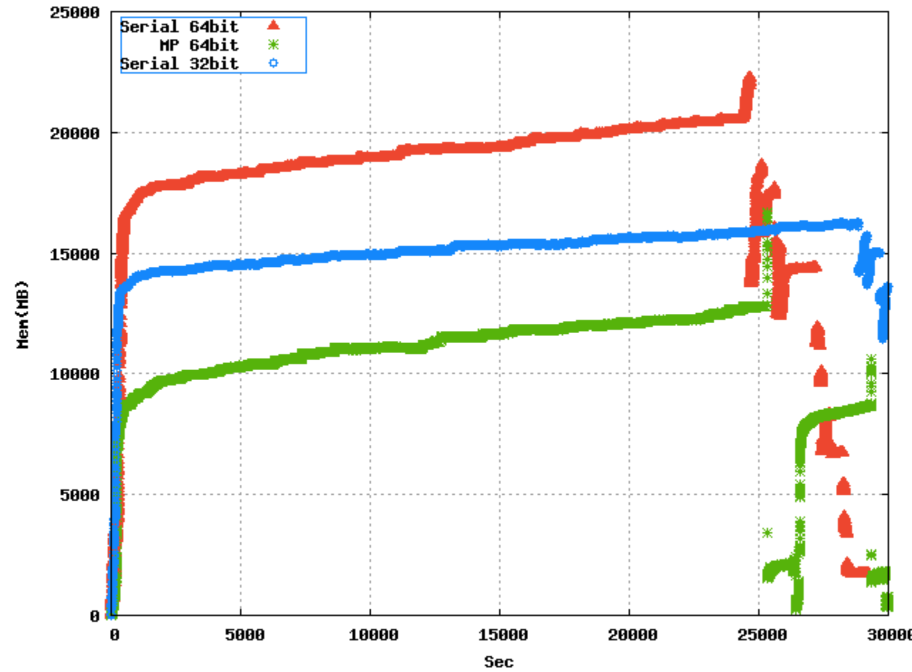
- **Saves up to 40% memory per core (RSS)**
  - **Scales well with cores, tested up to 32 forked processes**
- **A single AthenaMP job with N workers of course runs faster than the corresponding serial job, but not N times faster**
  - **Amdahl's law:**

$$S(N)=\frac{1}{(1-P)+\frac{P}{N}}$$

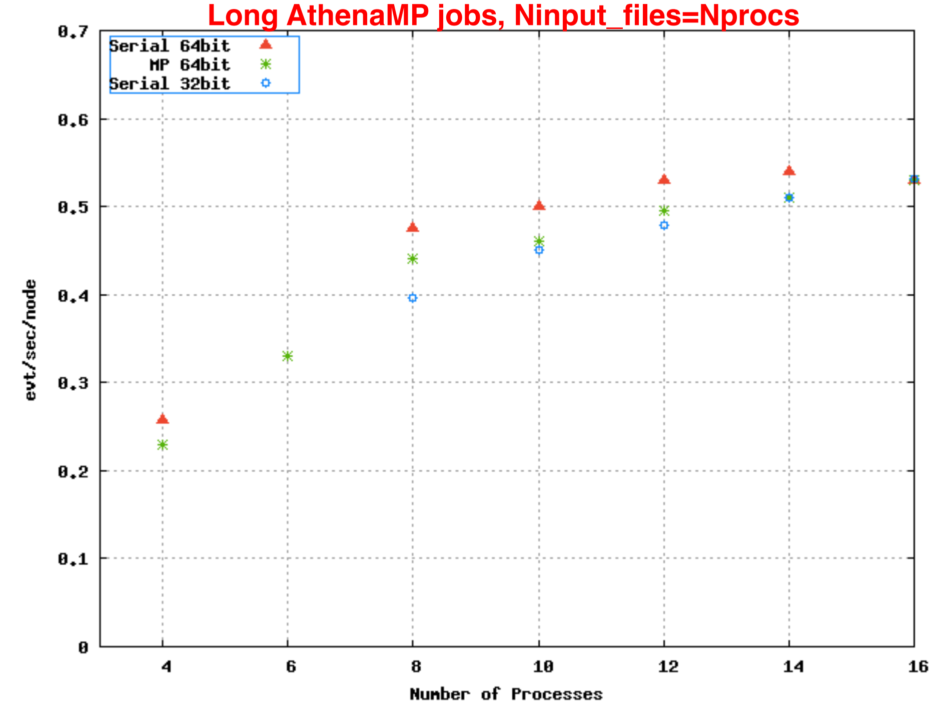
P is the portion of the program that can be made parallel
S(N) the maximum speedup that can be achieved by using *N* processors

- **Long reconstruction jobs result in large output files**
  - **With high number of workers it can become problematic to manage such big files**
  - **Also, the output validation of such files is slow**
  - **In order to address this issue we would need to have a mechanism, which allows single job make more than one output file of a given type**

Memory Profiling
Athena serial vs MP. Nproc=8

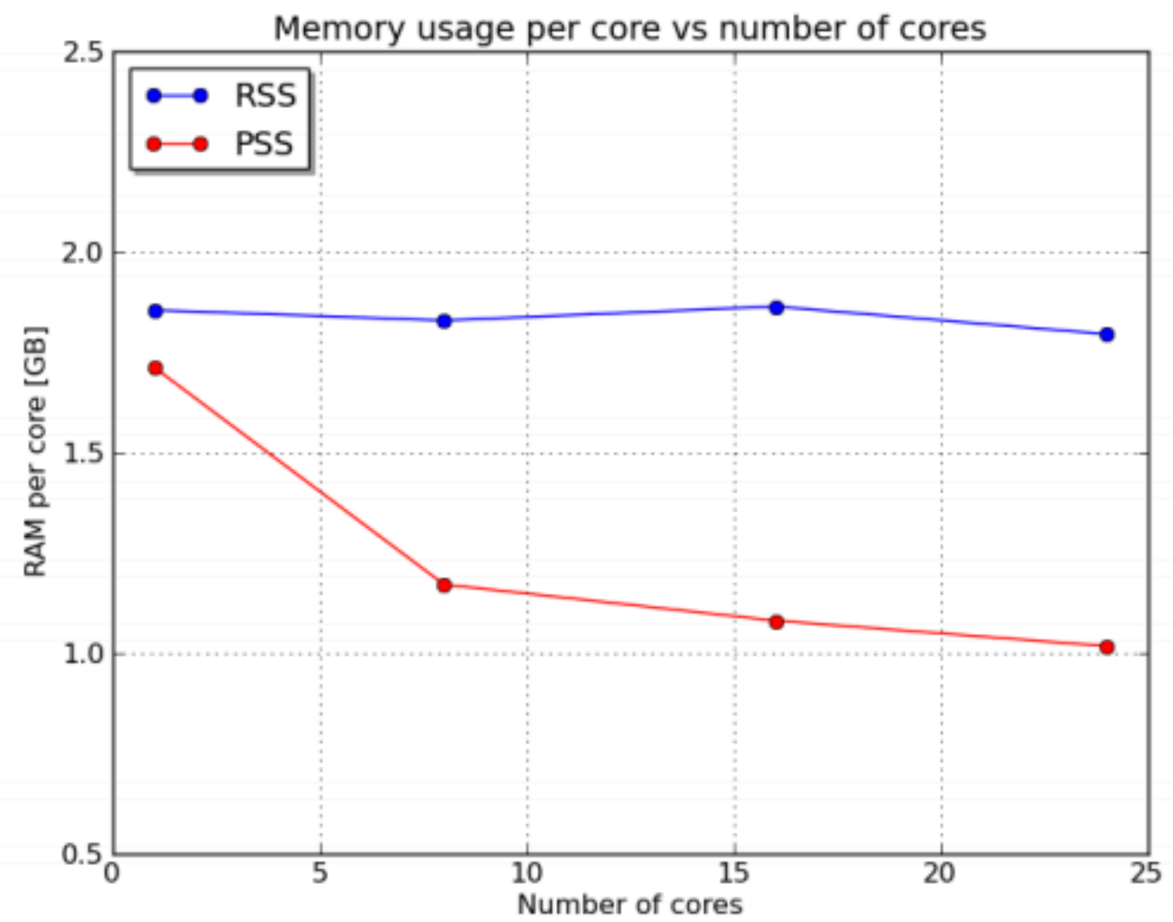


Nevents/wall-time vs Nprocs
Long AthenaMP jobs, Ninput_files=Nprocs



Multiprocessing performance in CMS

- Using a reasonable metric (VSIZE is not, and not even RSS), up to 40% memory gained per core
  - Scales well with cores, tested up to 96 forked processes
- Price to pay is a <5% CPU inefficiency
  - Not due to processing itself, which keeps all the CPUs at 100%, but
    - Skew between forked processes: they have to wait for the last before finishing
    - Time spent in merging results is idle for all the cores
  - Becomes ~irrelevant for jobs lasting more than 4 hours



- As of today:
  - Multicore approach simply works (it is in operation on specific T1s queues)
  - Not too much pressure to use it as standard, since CMS fits quite well with the 2 GB/core limit (after having spent 2011 to reduce the RAM footprint even in presence of PileUp ~ 40)

Is forking enough?

- Up to a certain point
  - It helps to reduce memory footprint by sharing common payloads
  - But it relies on processing N events in parallel
  - When events are big, the "not shareable event data" can choke systems

- After that
  - We need parallelism within the single event
  - Smaller memory footprint: a single event in memory
  - But: parallelism at event level→ we need to touch the algorithms

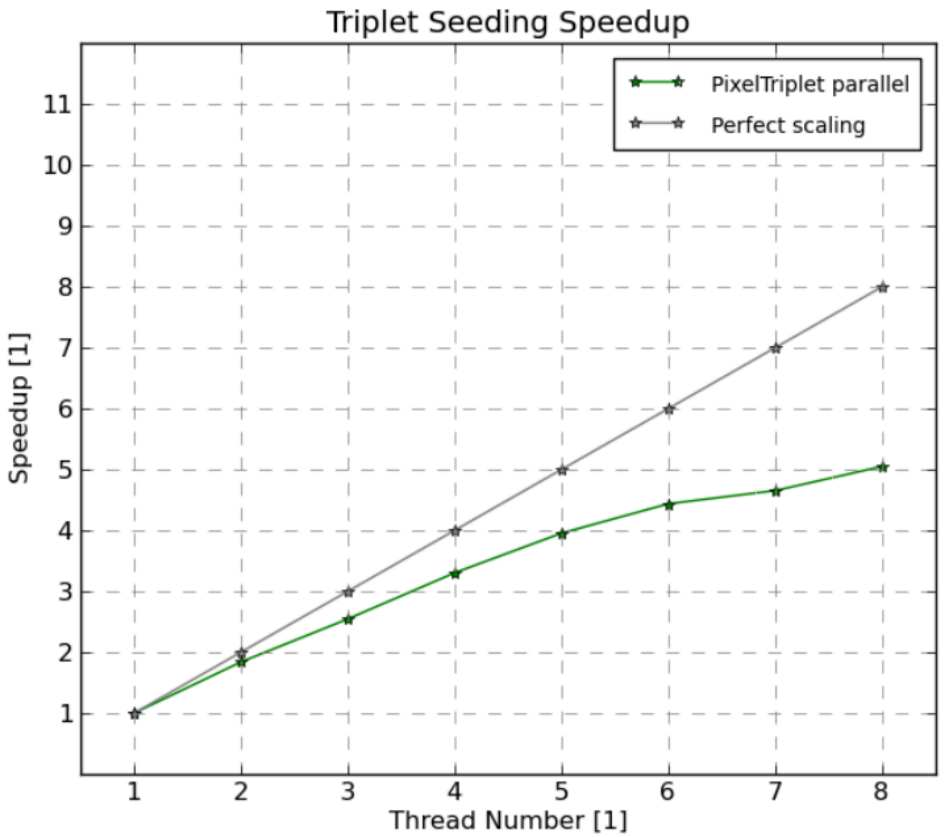
Work in progress for the software [1]

- **Short, medium and long-term work necessary**
  - **Common issues to be addressed**
    - Memory saving to be able to keep all cores in a machine busy (the usual, in principle trivial parallelism on event level)
    - Better usage of the resources

- **ATLAS**
  - **I/O framework, getting rid of POOL, etc - advancing well**
  - **Full usage of each core: vectorizing a few algorithms**
    - Tracking will be explored first
    - Likely influence on data model
  - **Parallelism on intermediate levels: on algorithm level, on sub-event data level. Both require refined communication mechanisms**
  - **Common work with PH-SFT strongly considered. Common tracking effort with CMS desirable**

Work in progress for the software [2]

- **CMS**
  - **libdispatch (Apple): send atomic tasks to a global queue, which dispatches it to the cores when available**
    - Close to a batch system handling thread dispatching in the system
    - Not (too) exposed to physicist: can be hidden at FW level, we do not need to be all multi-threading experts
    - CMSSW cores become tasks in the queue, FW to resolve ordering
  - **Use Intel Threading Building Blocks (TBB)**
    - Test application on a single algorithm: CMS tracker pixel seeding
      - Which is just a loop over hit triplets to see whether they fit an helix
      - Seed candidates are divided into blocks, and sent to different threads
      - With 8 threads, seeding goes from **14% of total reconstruction time to virtually negligible**
      - RAM penalty is very small (~ 1 MB/thread)
        - Much better than running pixel seeding on 8 different events as in forking



Work in progress for the software [3]

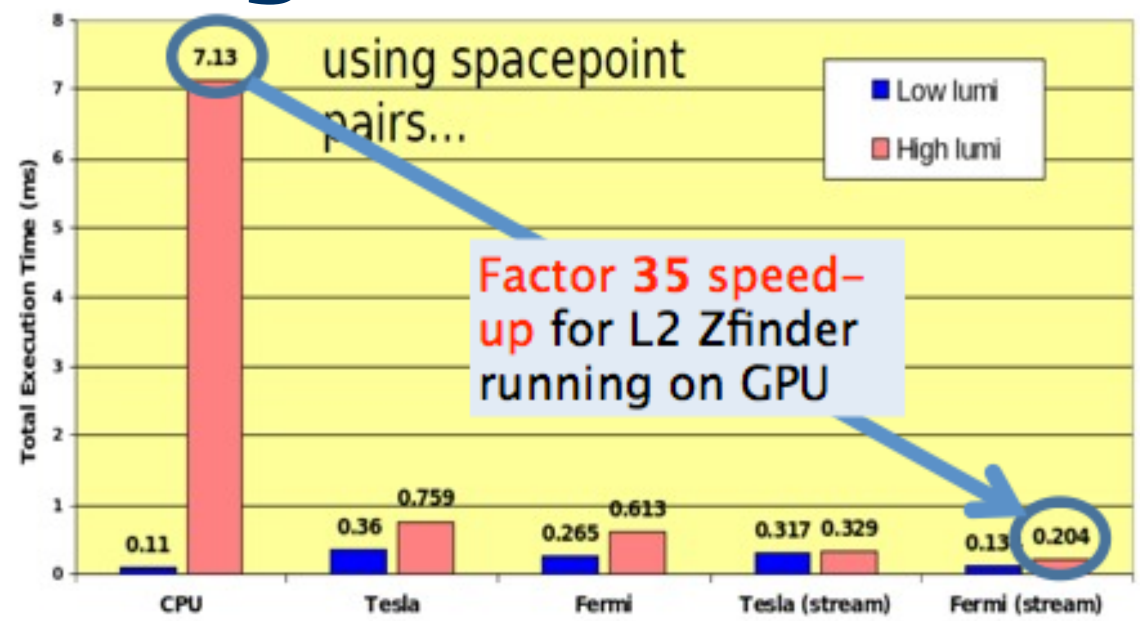
- **Vectorization**
  - Up to here we were just trying to use in a better way multi core CPUs
  - We can squeeze more performance by using the vector units which are paired to them (MMX, SSE, SSE2, … , AVX, …)
  - ATLAS and CMS are experimenting vectorization techniques for their software
  - Step #1 (CMS): use auto-vectorization in latest GCC; studies ongoing

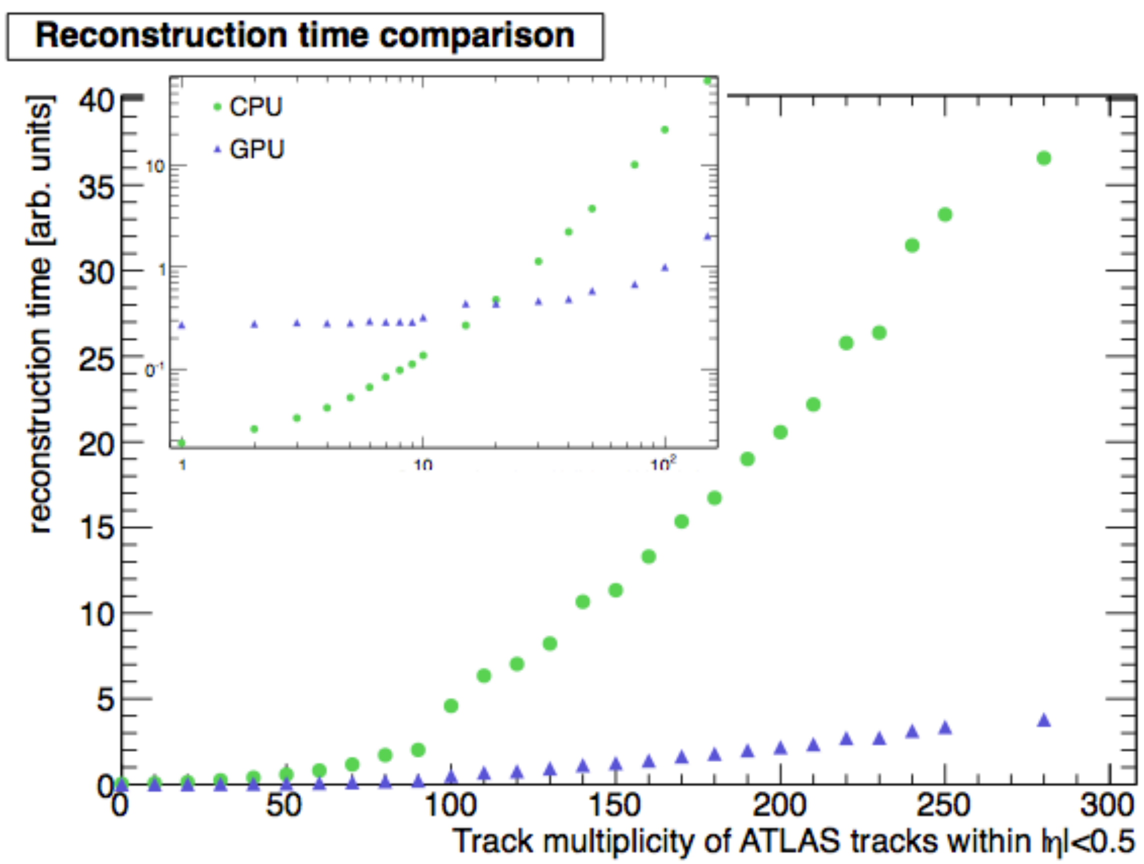
ATLAS GPU-based Tracking

- **First tracking prototypes for Level-2 track trigger and offline tracking**
  - **concentrate on aspects of track reconstruction chain**
    - z-vertex finder
    - track seed finder
    - Kalman filter
  - **early phase, still significant approximations**
- **very significant timing gains**
  - **but: lots of software development needed to obtain precise tracking**
  - **investigate mixed scenario ?**
    - e.g. combinatorial seed finder on GPUs
    - CPUs for serial processing steps to do precision calculations

~150 speed-up seen in other Kalman filter studies

Experience with GPUs can help with many-cores





Upgrading the ATLAS Computing Model

- The ATLAS Distributed Computing & the Grid are doing very well.
  - 1000's of users can process petabytes of data with millions or more of jobs
- But at the same time, we are starting to hit some limits:
  - Scaling up, elastic resource usage, global access to data
- What can we learn from external innovations?
  (without disrupting operations!)
- Various R&D Projects and Task Forces were formed one year ago
  - NoSQL databases R&D
  - Cloud Computing R&D
  - XROOTD Federation and File level Caching Task force
  - Event Level Caching R&D
  - Tier3 Monitoring Task Force
  - CVMFS Task Force
  - Multicores Task Force
  - Also Network Monitoring…
- Deploying and using LHCONE
  - Already done in some sites, no immediate gain but almost transparent migration
- https://twiki.cern.ch/twiki/bin/viewauth/Atlas/TaskForcesAndRnD

Upgrading the CMS Computing Model

- Job Submission
  - Going mostly to GlideIn WMS
- Storage
  - Differentiate T1D0 and T0D1 to help analysis @ Tier1s
    - No wild tape recall
- Remote (WAN) data access
  - US and EU level projects are effectively building up "federation" of centers
- Software
  - Redesign of critical parts to exploit multi core systems more efficiently; going beyond with parallelization
- Networking
  - LHCONE to be deployed soon (indeed, already partially done)

Short and long term work in LHC

- **Short and medium-term work well under way**
  - **Important technologies: Cloud computing, Hadoop basket**
  - **Common work ATLAS/CMS/IT-ES on job management with Panda+Condor**

- **Long term**
  - **Future role of middleware?**
  - **Try consolidate middleware flavors**
    - Possible consequences for systems we have on-top
  - **Or rather try to be independent of middleware**

- **Common solutions**
  - **With other experiments, with CERN (IT, PH), with other labs**
  - **Many of the TEG areas are chances for commonality**
  - **Concrete progress in a few areas so far**
    - ATLAS, CMS, IT-ES: common analysis framework based on PanDA+Condor/Glidein
    - Helix/Nebula: Cloud computing project involving CERN/ATLAS, EMBL, ESA, and 13 industrial partners
  - **Other opportunities**
    - Storage federations, network monitoring, data preservation

- **Common solutions are needed when manpower/funding shrink (EGI, EMI deadlines, OSG cuts)**

Common Analysis framework ATLAS/CMS

- **Initiative from CERN IT-ES, ATLAS and CMS for a common analysis framework started March 13th 2012**
- **Assess the potential for using common components for distributed analysis, based on elements from PanDA and glideInWMS**
- **Initial plan**
  - **Feasibility study - Mandate: http://cern.ch/go/9mNC**
    - Analyze architectures of both experiment's analysis frameworks
    - Identify interfaces to external systems
    - Identify what can be reused
    - How much effort is it?
    - Identify show-stoppers
  - **Functionality study**
    - What do ATLAS and CMS gain and loose in terms of functionality by adopting a common framework
  - **Operations study**
    - What is the impact on the cost of operating various proposals
- **A common analysis framework could lead the way to further commonalities and collaboration between the experiments in the future**

Virtualization and Cloud R&D in ATLAS

- **Active participation, almost 10 persons working part time on various topics**
  - https://twiki.cern.ch/twiki/bin/view/Atlas/CloudcomputingRnD

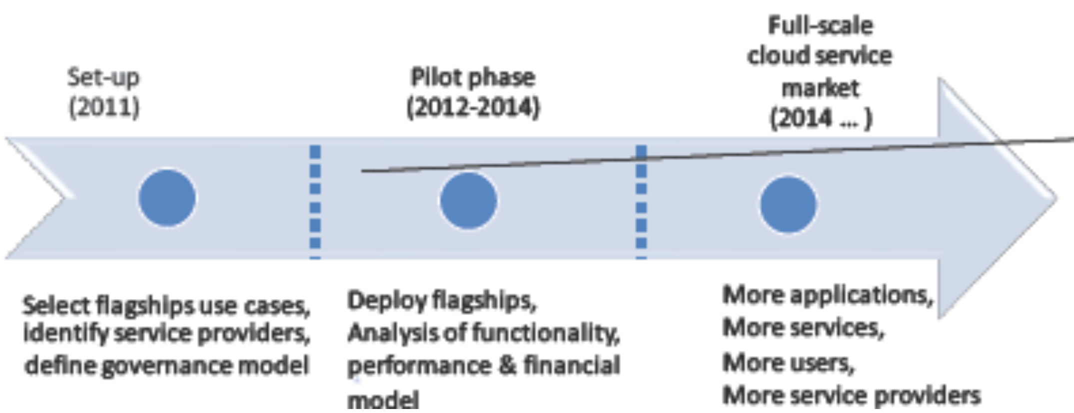
- **Data Processing**
  - **Panda Queues in the Cloud**
    - Centrally managed
  - **Tier3 Analysis Clusters (Instant Cloud Site)**
    - User/Institute Managed, Low/Medium Complexity
  - **Personal Analysis Queue (~One click, run my jobs)**
    - User Managed, Low Complexity (almost transparent)

- **Data Storage**
  - **Short term data caching to accelerate above data processing use cases**
    - Transient data
  - **Long term data storage in the cloud**
    - Integrate with DDM

Cloud Computing in ATLAS

- **Helix Nebula: the Science Cloud**
  - **European Cloud Computing Initiative: CERN, EMBL, ESA + European IT industry**
    - Evaluate cloud computing for science and build a sustainable European cloud computing infrastructure
    - Set up a cloud computing infrastructure for the European Research Area
    - Identify and adopt policies for trust, security and privacy at a European-level
  - **CERN/ATLAS is one of three flagship users to test a few commercial cloud providers (Cloud Sigma, T-Systems, ATOS...)**

HELIX NEBULA THESCIENCECLOUD

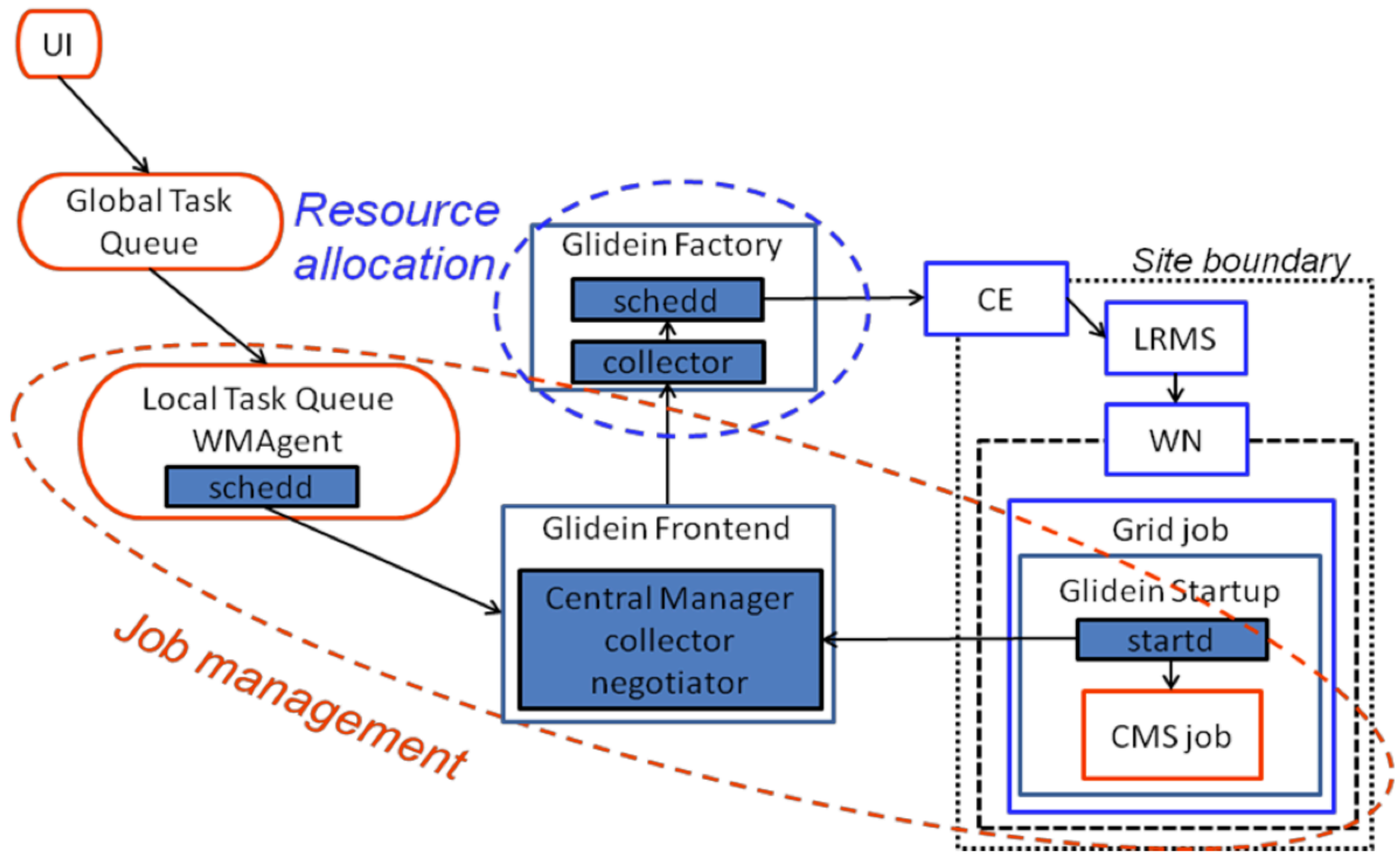


First step completed: Ran ATLAS simulation jobs (i.e. small I/O requirements) on Cloud Sigma AG

- **Simple approach**
  - **Use CernVM with preinstalled SW**
  - **Condor pool with master at CERN (one of the pilot factories)**
  - **I/O copied over the WAN from CERN (lcg-cp/ lcg-cr for input/output)**

Job Management in CMS

- Use of the glidein WMS factory to access WN slots.
- No reliance on BDII load information
- Has proven to be able to sustain > 50k jobs in production
- Having a single Global queue allows for CMS wide prioritization



Storage issues in CMS

- The main issue for hosting physics analysis @ Tier1s is the risk of having chaotic file recalls from Tape

- While CMS up to now has envisaged a flat T1D0 disk model, with all the files logically “living on tape”, a split with some files pinned to disk (T0D1) is the preferred solution for analysis use cases
  - Being investigated

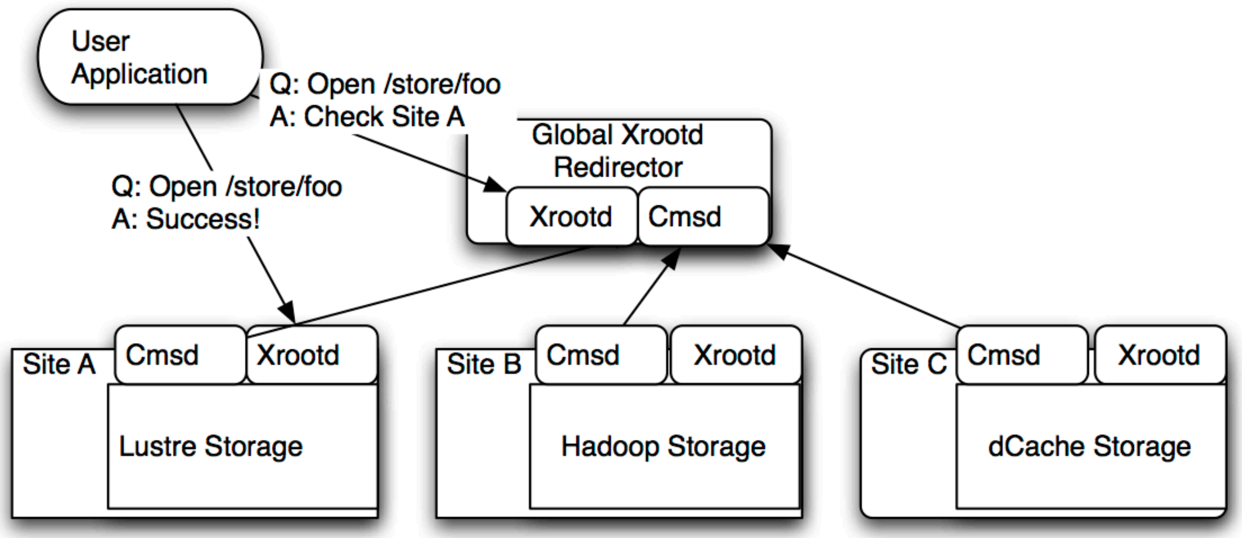
- Streaming access to data
  - One of the consequences of a next generation networking (GARR-X) is the possibility to at least partially overcome the “jobs go to where data is” paradigm
  - With current CMSSW on AOD events, it is typical to see analysis jobs reading < 500 KB/s from storage. With 10 Gbps, hundreds of job reading over WAN are possible (using vector reads)

Which storage model?

- **Common problem in ATLAS and CMS**
- **Driving ideas**
  - **We cannot avoid completely data placement – too early**
    - The current paradigm "jobs go where data is" stays valid in the vast majority of the cases
  - **But a number of use cases can be better served by streaming:**
    1. Fallback in case of hardware problems: In case of unavailability of the local copy of a file the user job can transparently access a remote replica, thus increasing the processing efficiency and decreasing job failure and resubmission
    2. Specific interactive use cases with low I/O processes, i.e. visualization programs.
    3. Address site congestion, when the available copies of a dataset are at overloaded sites (overflow)
    4. Increase the utilization of CPU power at sites where proper data management is not possible (for example, small University sites with no/small storage)

Streaming data: Storage federations

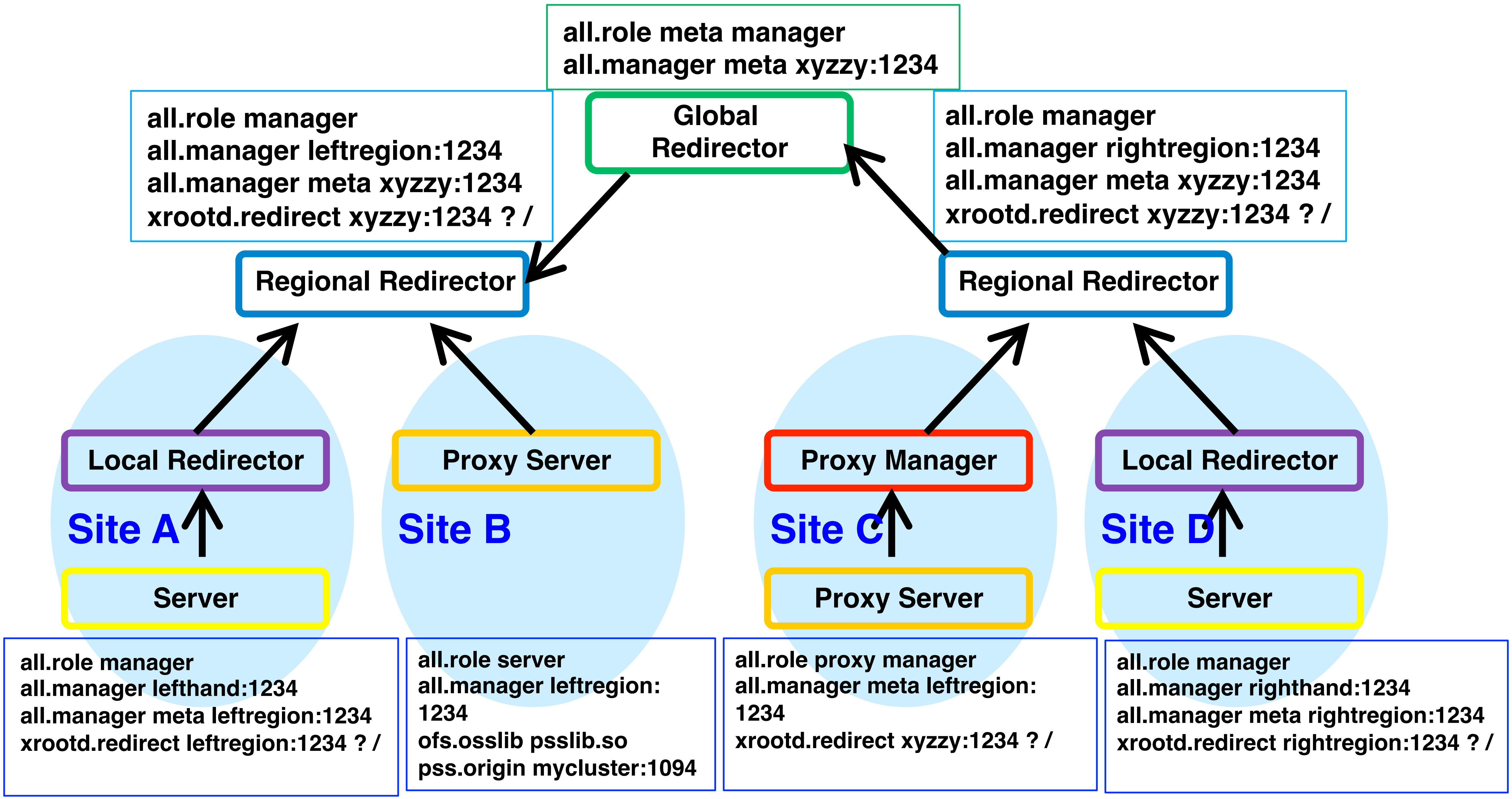
- **Why storage federations?**
  - **Fitting the experiments' needs**
  - **Little needed for code development**
    - Excellent access to Xrootd team if development is needed
  - **Mature product used for many years**
  - **Common technology with LHC experiments**
    - ATLAS, CMS and Alice
  - **Very efficient at file discovery**
  - **Works seeamlessly with the root data formats that we use**
  - **ROOT team collaborating with Xrootd collaboration on efficient wide area data access**



Example: storage federations status in ATLAS

- In US cloud, Tier 1 site and 80% (4 out of 5) Tier 2 sites part of Federation
  - Soon all Tier 2 sites will be part
- Using X509 authentication for reading
  - Plugin code looks for ATLAS VOMS extension to allow access
  - Deployed at MWT2, AGLT2, SWT2 and SLAC T2 currently
  - The rest (BNL, and NET2) shortly
- Prototype Midwest region redirector setup at Chicago
  - Prun jobs used to test regional redirector
- Work still needed
  - Integrate federated storage with job management system to combine the power of both
  - Change the Pilots to handle missing files at run time
  - Evolve Panda to understand federated storage and federated site queues
  - …

Configured regional redirectors in ATLAS



Andrew Hanushevsky (SLAC)

Xrootd and CMS Software

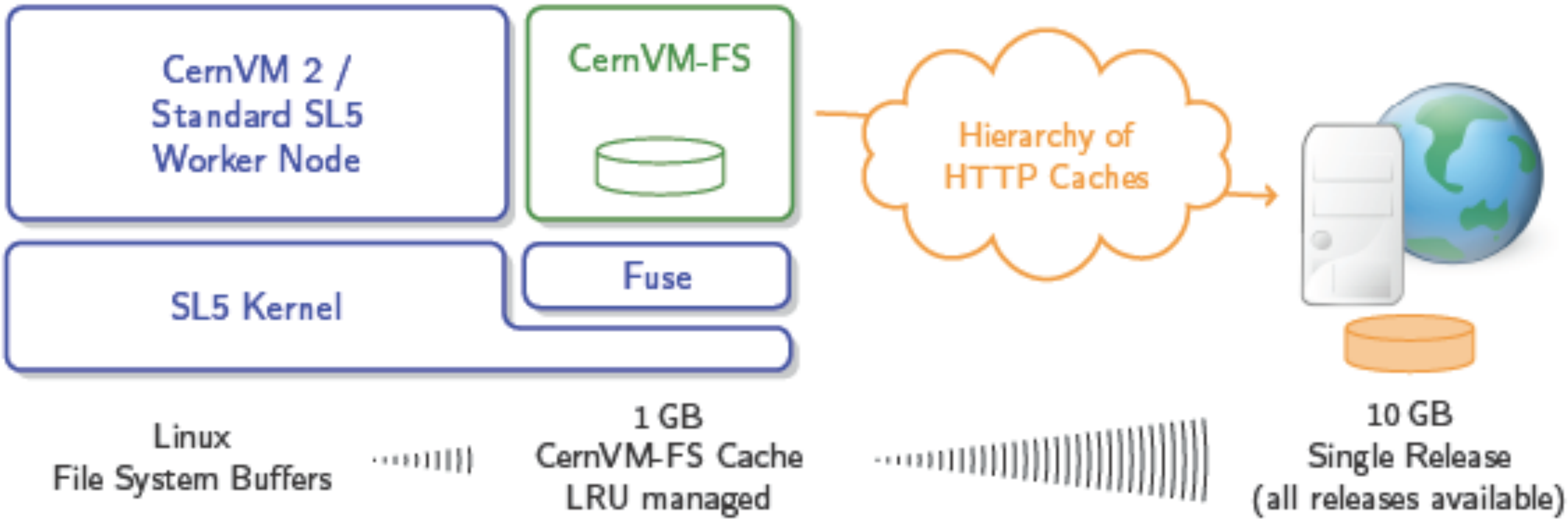
- Layered data access configured in CMSSW

- Regional redirectors already active
- UNL (Nebraska) for US sites
- Bari for EU sites
- CERN will soon publish EOS/CAF via Xrootd

I need '/store/foo'
- I try local access (via 'direct' protocol) … if not found
  - I try accessing a first level redirector (national, for example)… if not found
    - ….
      - ….
        - I scale up to the global redirector… if not found
          - Sorry file not accessible

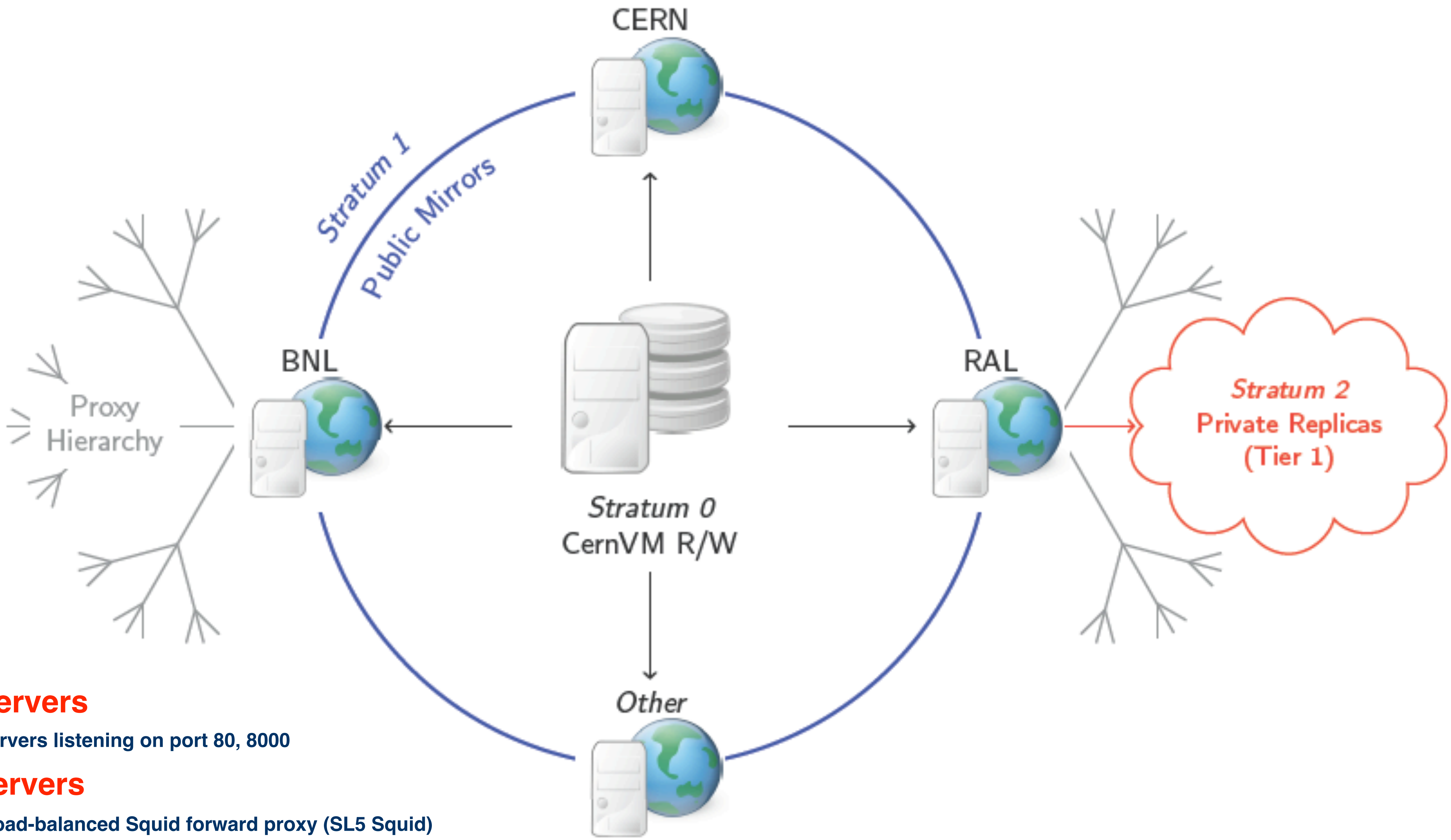
CVMFS

- **ATLAS is now fully using the dynamic software distribution model via CVMFS (CernVMFileSystem)**
  - **Virtual software installation by means of an HTTP File System**



- **Distribution of read-only binaries**
- **Files and file meta data are downloaded on demand and locally cached**
- **Self-contained (e. g. /cvmfs/atlas.cern.ch), does not interfere with the base system**
- **CMS is moving from training phase to real use (# of sites increasing fast)**

CVMFS Backends used by ATLAS



- **Mirror servers**
  - Web servers listening on port 80, 8000
- **Proxy servers**
  - Local load-balanced Squid forward proxy (SL5 Squid)

ATLAS and CVMFS

- **Migration status of the ATLAS sites:**
  - **> 60% of the sites are now using CVMFS**
  - **The rest should migrate by the end of this year**
- **/cvmfs/atlas.cern.ch**
  - **“Stable releases” software**
  - **New production server in CERN IT ready for production**
    - Populated, but not yet in full production
- **/cvmfs/atlas-condb.cern.ch — ATLAS Condition Flat Files**
  - **Release manager machine hosted by CERN IT**
  - **Automatic update several times a day**
- **/cvmfs/atlas-nightlies.cern.ch — ATLAS Nightlies**
  - **Tested on grid sites too**
- **Integrated with the current Installation System**
  - **CVMFS sites are used by the installation system transparently, aside of sites using a different FS**

Networking upgrades

- **Tier2 sites are ready to switch to 10Gbps connectivity**
  - **All of them have the appropriate hardware to handle that**
  - **We hope to have it soon, as in some cases we're suffering of link congestions (eg. Napoli), especially if the site already moved to LHCONE**

- **ATLAS in LHCONE**
  - **3 out of 4 Tier2 sites are already connected**
    - Napoli
    - Milano
    - Roma
- **CMS in LHCONE**
  - **3 CMS Tier2s out of 4 already had the transition to LHCONE**
    - Bari: April 26^{th}
    - Roma: May 7^{th}
    - Pisa: May 10^{th}

- **No problems so far**
  - **The transition was done very quickly**
  - **No performance difference detected with respect to the previous setup**

The ATLAS Computing model changes

- **Move from a fully hierarchical model to less hierarchy and more Bandwidth**

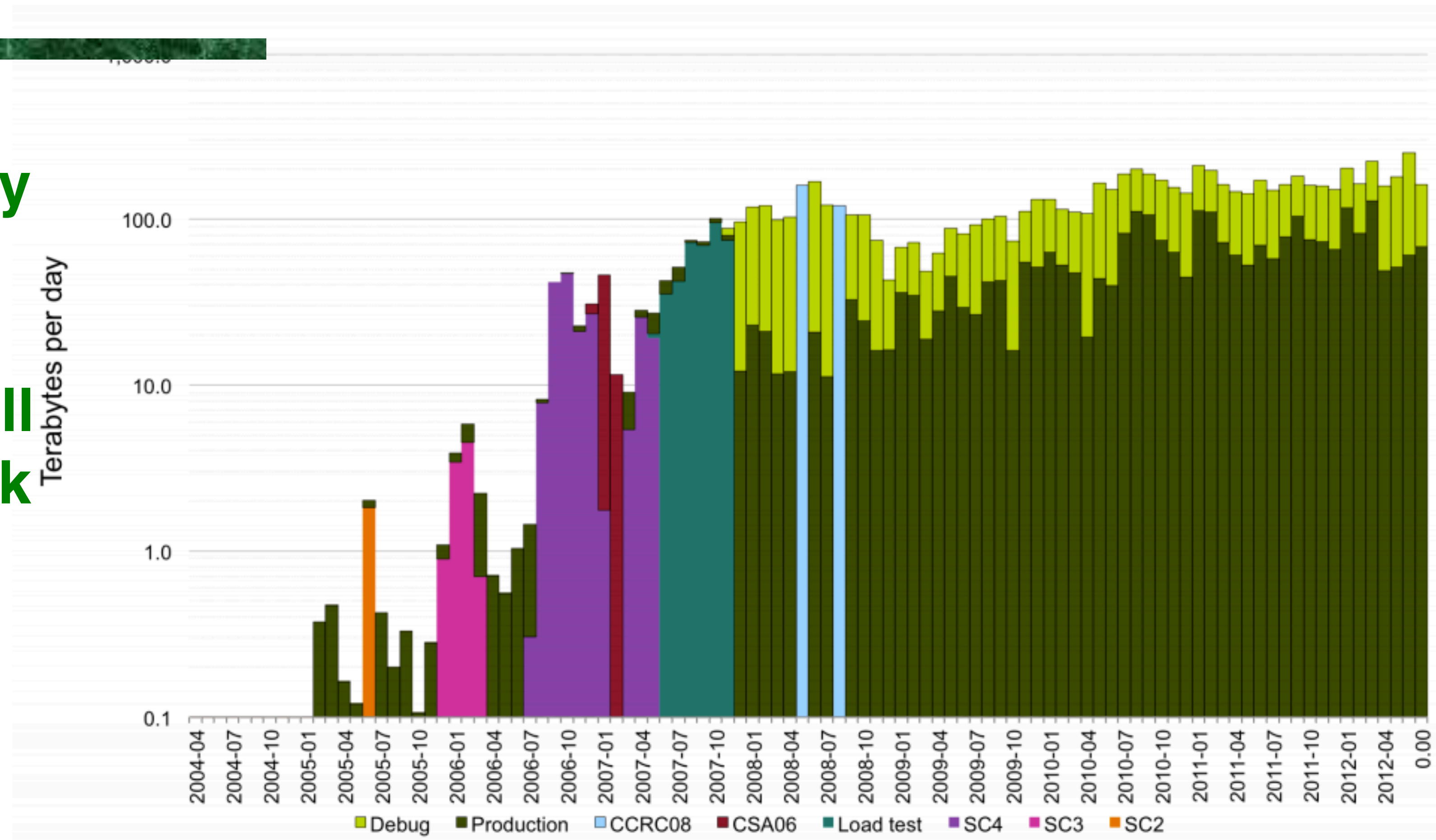
- **4 recurring themes:**
  - **Any site can replicate data from any other site**
  - **Multi Domain Production**
    - Need to replicate output files to remote Tier-1
  - **Dynamic data caching**
    - Analysis sites receive datasets from any other site "on demand" based on usage pattern
  - **Remote data access**
    - Local jobs accessing data stored at remote sites

- **ATLAS is now heavily relying on multi-domain networks and needs decent network monitoring**

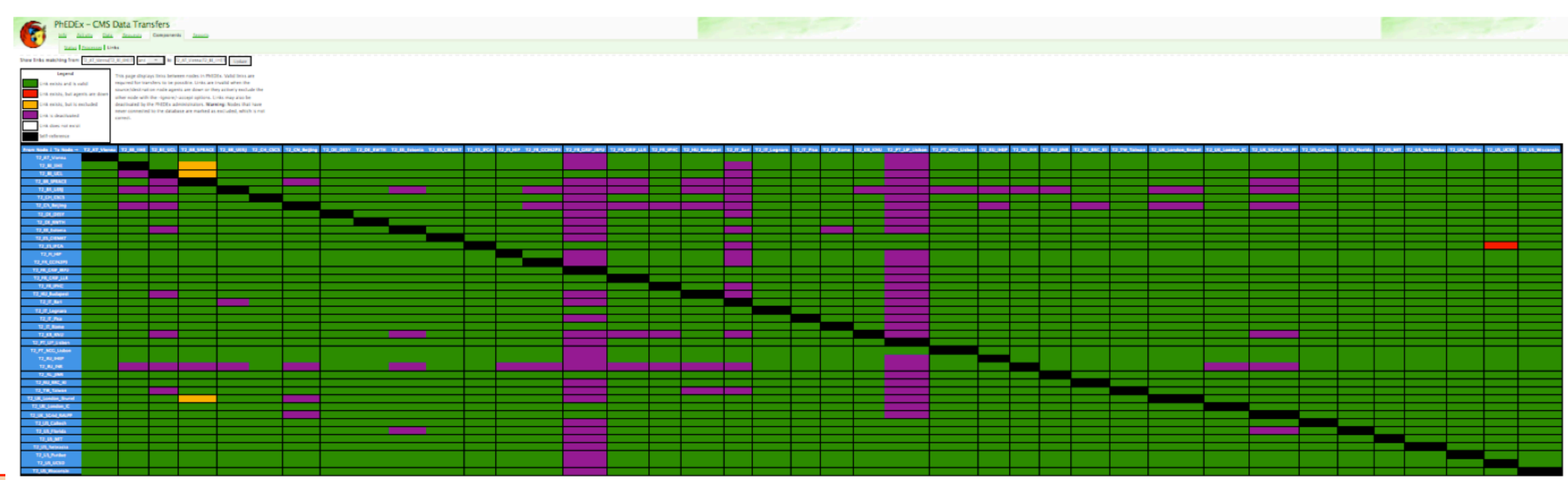
- **Work Ongoing on global access/Data Federation**

CMS – full mesh model

- Also CMS started data taking with a MONARC approach, fully hierarchical
- Already in 2010 this was changed to a model in which all the sites (most of …) can speak together directly
- Situation end of 2010:



A part of the full matrix (T0,1,2) vs (T0,1,2):
2550 active links



Common Distributed Computing evolutions

- **Database Scaling**
  - **Hadoop environment looks best**

- **Storage and data management**
  - **Maintain stable storage for placed data**
  - **Support access from experiment jobs**

- **Workload management**
  - **Pilots and frameworks**
    - GlideinWMS
  - **Whole node scheduling**

- **CPUs and I/O**
  - **Use of CPU affinity and pinning**
  - **Handling of CPU-bound and I/O-bound jobs**

- **Exploring Common Solutions among several experiments/WLCG**

Conclusions

- The ATLAS and CMS Computing models are about to evolve in the coming months in a variety of different aspects
  - Job handling and exploration of the Cloud Computing extensions
  - Storage models and data access
  - Data transfer model and data placement vs direct streaming
  - Usage of new CPU architectures and full exploitation of the multiprocessing
  - Network upgrades

- The Computing Infrastructure of ATLAS and CMS will continue working next year at full speed, taking advantage of the LHC stop to test and deploy new solutions

- "Common Solutions" seem more and more the way to proceed, while at the same time maintaining experiments' peculiarities

- Credits
  - Thanks to R. Jones, and D. Benjamin for the ATLAS material
  - Thanks to C. Grandi and D. Bonacorsi for the CMS material