STORAGE EVOLUTION AND TRENDS

G. Donvito
INFN-Bari

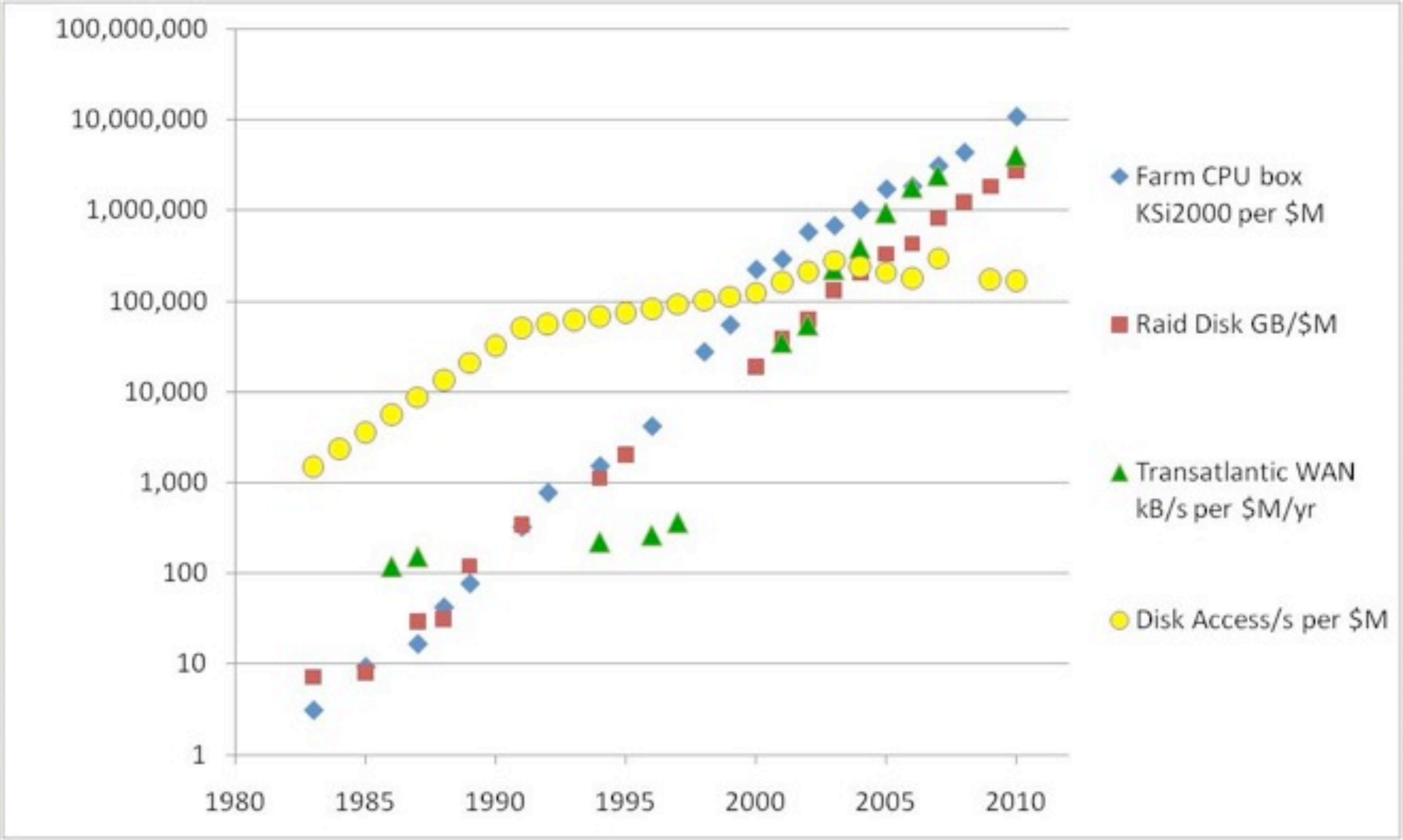
Outline

- Main requirements (problems) on accessing data
  - Trends and evolution on data management
- LHC use case
  - Requirements
  - Observations on firsts months of runs
  - New trends on LHC storage management
- On going storage activities
  - Hadoop
  - Test on going

Main requirements (problems) on accessing data

- CPUs are increasing constantly in computing power
- Storage devices are growing in size
    - but not in performance
    - packaging more TB into the same size is not the way to achieve better performances
- The network is not anymore the main bottleneck
- The CPUs are not efficiently used if the process are waiting for data

Performance/$M trends



Trends and evolution on data management

- The key is to parallelise the data access
- Using as much spindle as possible
- Pre-fetching could be the a solution
  - It is important to know the application and the access patterns
  - It is important to write data thinking on how they will be read
    - Physics data are "write-once-ready-many"
- Posix access is becoming a required "added value"

LHC Use case

* Huge dataset size
  * order of (tens of) TB for a single analysis
  * Moving a single dataset may require days
    * while an analysis should take few hours
* Hundreds of widely different sites
  * tens of thousands of CPUs
  * Petabytes of storage
    * It is important to optimise the usage of resource
      * both CPU and storage

LHC Use case

* Replicating data manually is a time consuming activity that should take input from usage statistics
* Huge physics community with widely different computing skills
* Smallest site (T3) could have difficulties to set-up and maintain a reliable storage installation
* End user interactive analysis is growing in size and requirements
* The analysis jobs are often I/O bounded already now
  * It could be worst as the amount of data increases

LHC: Observations on firsts months of run

WLCG Jamboree on Evolution of WLCG Data & Storage Management 16 - 18 June 2010

(http://indico.cern.ch/conferenceDisplay.py?confId=92416)

* "Started with a general concern about how we would support analysis access to users as we get additional data"
* Lots of choices on the LHC computing model were based on limitations ... or assumption of limitations (storage cost, network bandwidth, predictable utilization, etc)
  * a number of those limitation are not anymore valid
* "Improve the transparency of access."
* "Introduce less deterministic features to the system to improve flexibility and response"

LHC: Observations on firsts months of run

Ideas and problems

* Could we avoid using different access protocol for each site?
* Is there any protocol that allow a efficient CPU usage but provides the capabilities to access files from another site?
* Can we exploit fruitful a peer-to-peer system in order to transfer files among sites?
* Can we use "predictable data movement" ***only*** for T0-T1 flows?
* Is the Tape-archive model still valid?

LHC: Observations on firsts months of run

Feedback on first experiences of data taking

* The Monarc model for data transfer is often broken:
  * A full mesh is often used (=> transfers between T1-T2 belonging to different “regions”)
* The HSM model is not really used in production:
  * several system to pre-stage from tape or pinning on disk are always needed to cope with CPU request for data access
* There is the need to simplify the framework for accessing data to the final user, providing advanced capabilities such as:
  * intelligent defaults, file collection, load-aware replication, meta-data, etc

LHC: Observations on firsts months of run

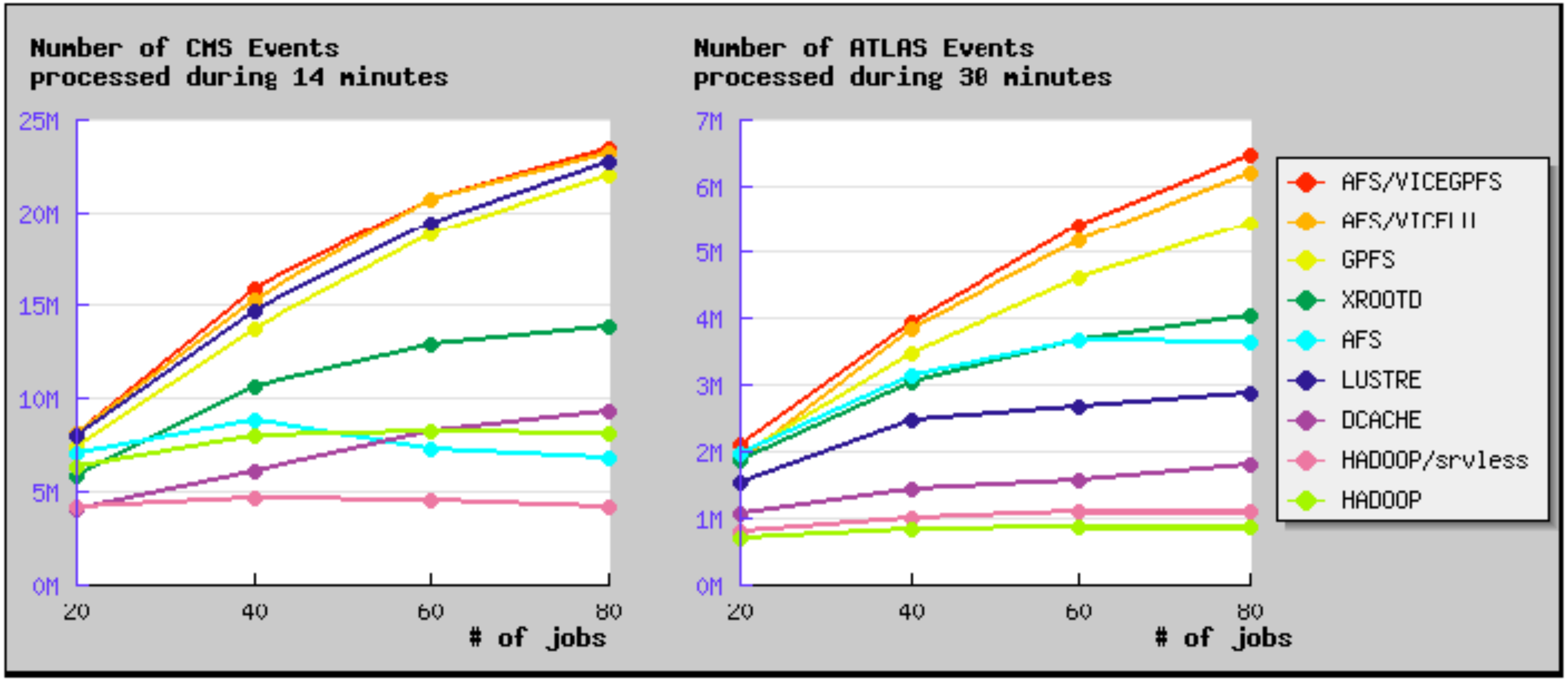
Feedback on first experiences of data taking

- Nowadays the network give the possibility to implement different caching policies in order to avoid the model: "Dataset scheduled transfer based on imagined demand"
- The main issue with the file catalogue is the consistency with the underling storage systems.
  - advanced features could be implemented into the catalog: ACLs, overall quotas, replica/cache management
- The scenario could dramatically change if the scheduler is organized on a "per node" basis
  - The memory footprint could be reduced, I/O could be optimized, etc
- It is evident the need of a "global home directory" for the output of the end user analysis

LHC: Observations on firsts months of run

Feedback on first experiences of data taking

* In the industry there is a trend to exploit “multi-tiered” storage in order to obtain the right balance between performance and TCO
* An HEPIX group is constantly testing new experiments software against the storage solution on the market in order to understand the performance:
  * at the moment it looks like posix files-system (GPFS, LUSTRE, AFS) are the best solution from a performance point of view
* NFS4.1 (PNFS) looks promising as “standard” protocols as it will be supported natively from several storage vendors.
  * dCache and DPM will provide NFS4.1 interface in the near future



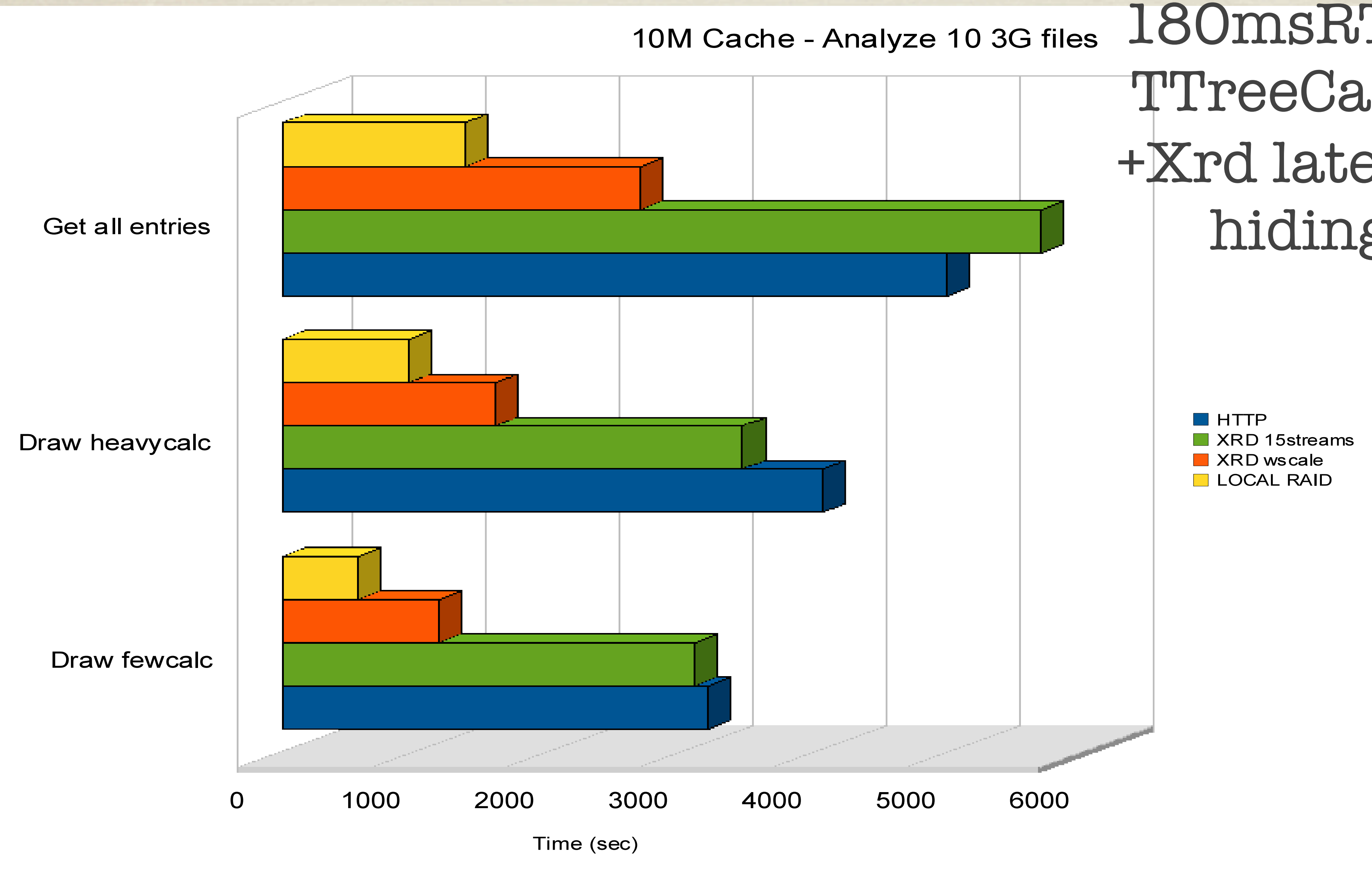
- Storage Efficiency (events processed / minute) may vary a lot from one solution to another. By simply changing the data archival technology on the same hardware base, as much as a factor of 4-5 in efficiency increase may be obtained
- Some of the solutions look universally good for both (very different) use cases
- Posix file systems in general look more efficient compared with the special solutions. They also require less tuning effort.

LHC: Observations on firsts months of run

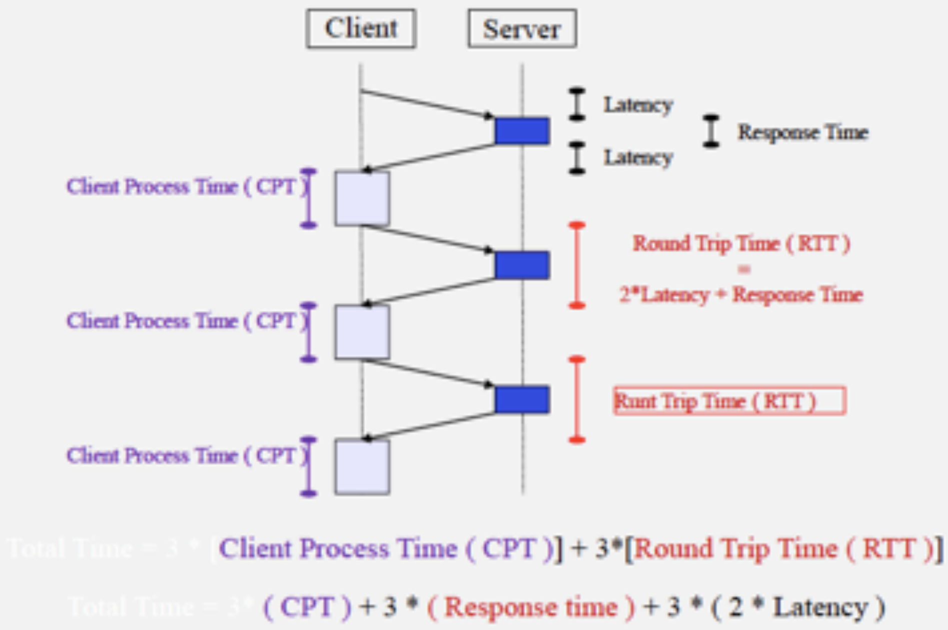
Feedback on first experiences of data taking

- Xrootd is a scalable and robust system born to fulfil the HEP community requirements and needs
  - A great work was carried on in oder to improve the performance on network with high latency
  - KIT provided a good feedback on the usage of Xrootd in production for both tape and disk management
- Root is providing new releases that increase the performance through using prefetch&caching
- SRM look like too complex and invasive for the end user
- Alien FC is providing a lots of feature needed from the final user:
  - Global unique namespace, Unix-like CLI, ACLs, input and output files, file collections, automatic SE selection, quota system, integrated with ROOT

CALTech 10GB/ 180msRTT + TTreeCache +Xrd latency hiding

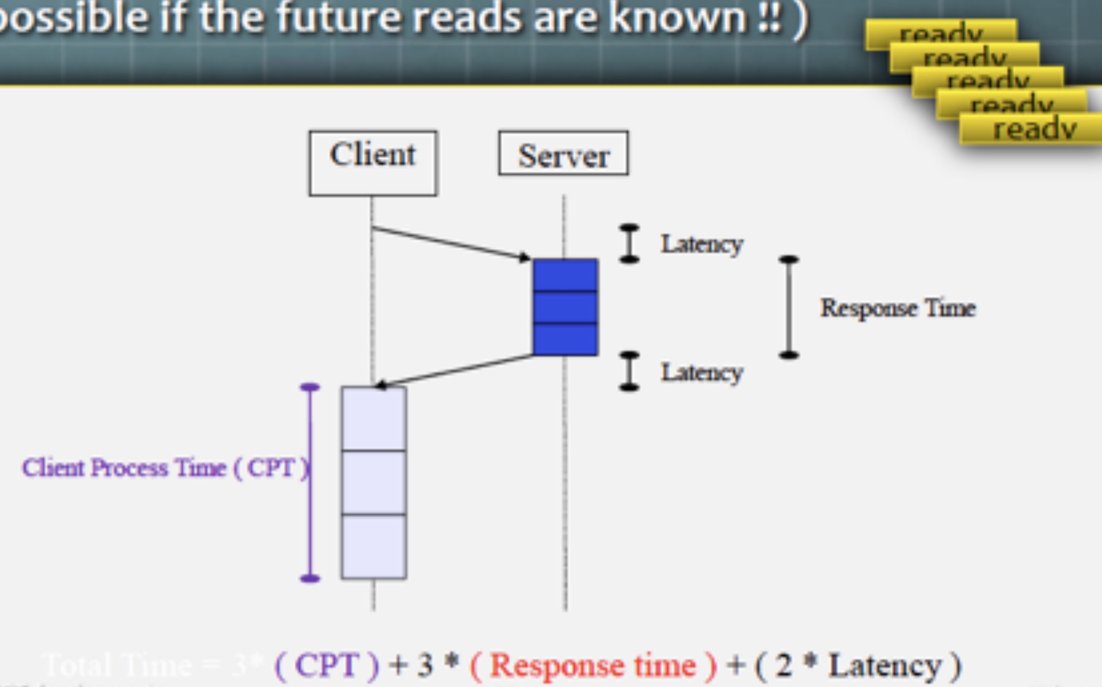


A major problem: network latency



client	latency (ms)	cachesize 0	cachesize 64k	cachesize 10 MB
A: local pcbrun.cern.ch	0	3.4 s	3.4	3.3
B: 100Mb.s CERN LAN	0.3	8.0 s	6.0	4.0
C: 10 Mb/s CERN wireless	2	11.6 s	5.6	4.9
D: 100 Mb/s Orsay	11	124.7 s	12.3	9.0
E: 100 Mb/s Amsterdam	22	230.9 s	11.7	8.4
F: 8 Mb/s ADSL home	72	743.7 s	48.3	28.0
G: 10 Gb/s Caltech	240	2800 s	125.4	4.6

Perform a big request instead of many small requests (only possible if the future reads are known !!)



Caching the same file

session	Real Time(s)	Cpu Time (s)
local	116	110
remote xrootd	123.7	117.1
with cache (1st time)	142.4	120.1
with cache (2nd time)	118.7	117.9

New trends on LHC storage management

Conclusion from the Jamboree (from Ian Bird)

* Storage:
  * Separate archive (Tape) and cache systems with different interfaces
  * Try to never read from tape
* Data Access Layer:
  * Need a combination of data placement and dynamic cache
  * Caches could optimize the disk space usage (or reduce it)
  * Can't assume catalogues are up-to-date, so it is needed a fall-back solution (remote access) in case of failure
  * Model of access is file-system-like

New trends on LHC storage management

Conclusion from the Jamboree (from Ian Bird)

* Data Transfer:
  * Need a reliable way to move data from/to an archive (or point to point)
  * Need a placement mechanism
  * Need transport for caching
  * Need remote access mechanism
* Namespace and Cataloques:
  * Want a dynamic catalogue (maybe it could be LFC+MQ)
  * The computing model should recognise that the information is only “best-guess” (not 100% reliable)
* Grid wide home directory
  * Is needed, but not already clear how to do it

New trends on LHC storage management

Demonstrator started

1. Brian Bockelman: xrootd-enable filesystems (HDFS, Posix, dcache + others) at some volunteer sites. Global re-director at 1 location, allow the system to cache as needed for Tier 3 use.
2. Massimo Lamanna: very similar proposal with same use cases for ATLAS. Also include job brokering. Potential to collaborate with 1)?
3. Graeme Stewart: Panda Dynamic Data Placement.
4. LHCb/Dirac – very similar ideas to 3).
5. Gerd Behrman: ARC caching technology: propose to improve the front-end to be able to work without the ARC Control Tower, also to decouple the caching tool from the CE.
6. Jean-Philippe Baud: Catalogue synchronisation with storage using the Active MQ message broker. a) add files, catalogue them and propagate to other catalogues; b) remove entries when files are lost if a disk fails; c) remove a dataset from a central catalogue and propagate to other catalogues.

New trends on LHC storage management

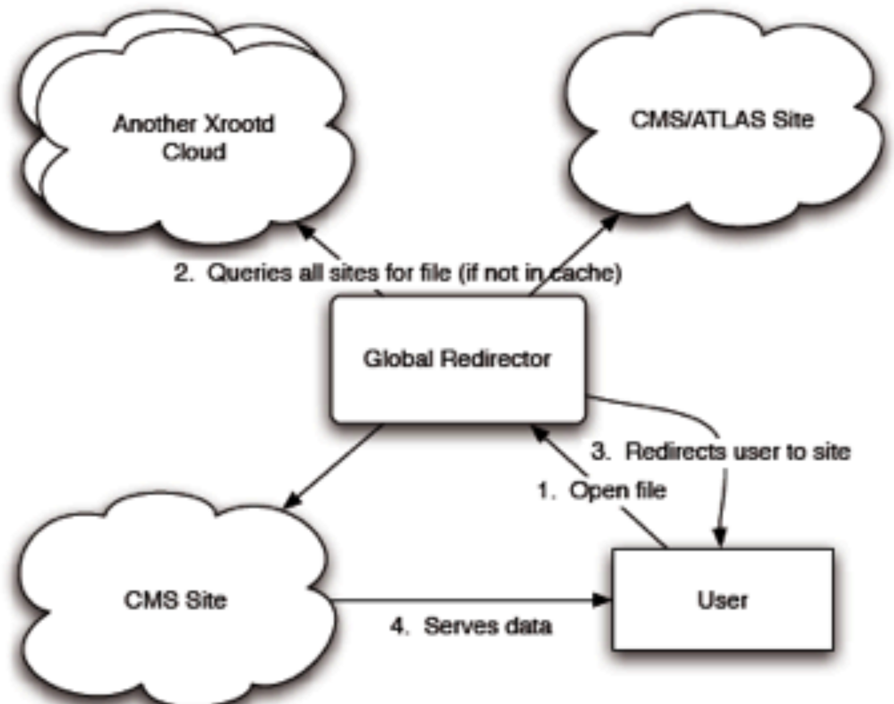
Demonstrator started

7. Simon Metson: DAS for CMS. Aim to have a demo in the summer.
8. Oscar Koeroo: Demonstrate that Cassandra (from Apache) can provide a complete cataloguing and messaging system.
9. Pablo Saiz: Based on Alien FC – comparison of functionality, and demonstration of use in another experiment.
10. Jeff Templon: Demonstrate the Coral Content Delivery Network – essentially as-is. Proposed metrics for success.
11. Peter Elmer: wants to show workflow management mapping to the available hardware (relevant to use of multi-core hardware).
12. Dirk Duellmann/Rene Brun: prototype proxy-cache based on xrootd. Can be used now to test several things.
13. Jean-Philippe Baud+Gerd Behrman + Andrei Maslennikov + DESY: use of NFS4.1 as access protocol.
14. Jens Jensen + (other name?): simple ideas to immediately speed up use of SRM and to quickly improve the lcg-cp utility

On going storage activities

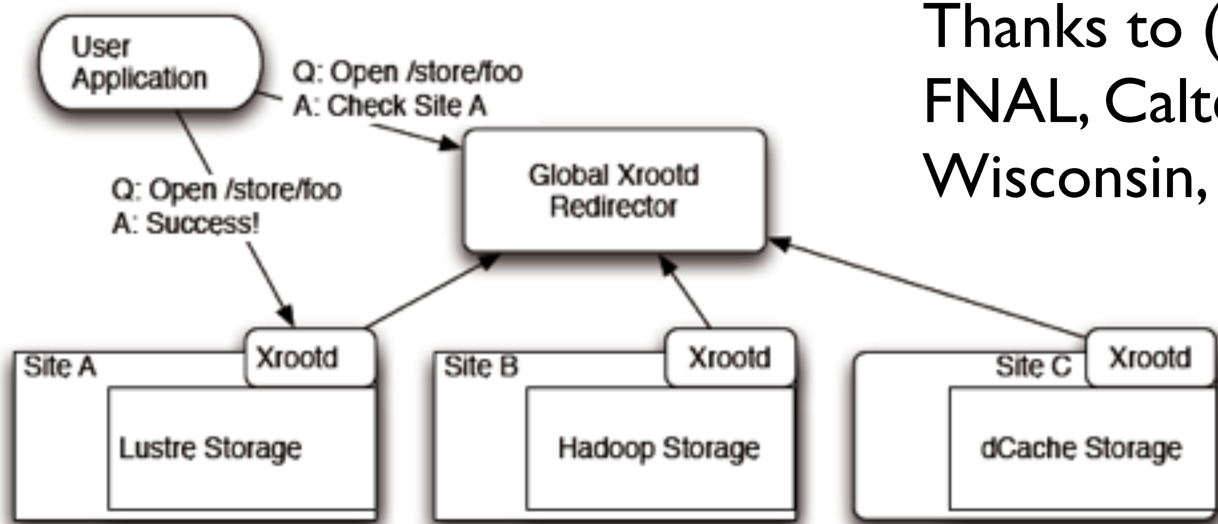
* One of the main interesting demonstrator is the CMS-Xrootd demonstrator (B. Bockelman)

Xrootd Architecture



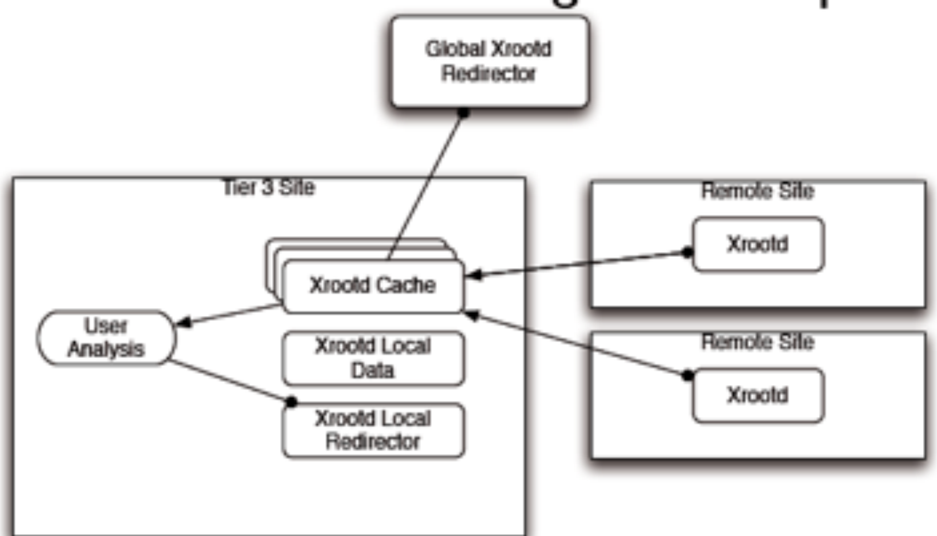
- Notes:
  - “Global redirector” can be up to 16 actual hosts (highly available)
  - Sites need to run at least 1 xrootd host, but can keep dCache/Lustre/HDFS/DPM/etc.
  - Each site exports according to their capacity - no distinction in terms of T0 vs T1 vs T2.
  - T3 is a special case; more later.

- Application running outside grid
- Incomplete dataset at a site



Thanks to (no particular order): PSI, Bari, FNAL, Caltech, UCSD, Florida, Purdue, Wisconsin, and UCR for participating.

T3 Site - look! No data management responsibilities



The cache servers act as a client to the global system. Downloads from all possible sources as in bittorrent.

* Each site exports the global namespace, and translates the file open requests to the local namespace.
* Elapsed time is often around 100ms.

On going storage activities

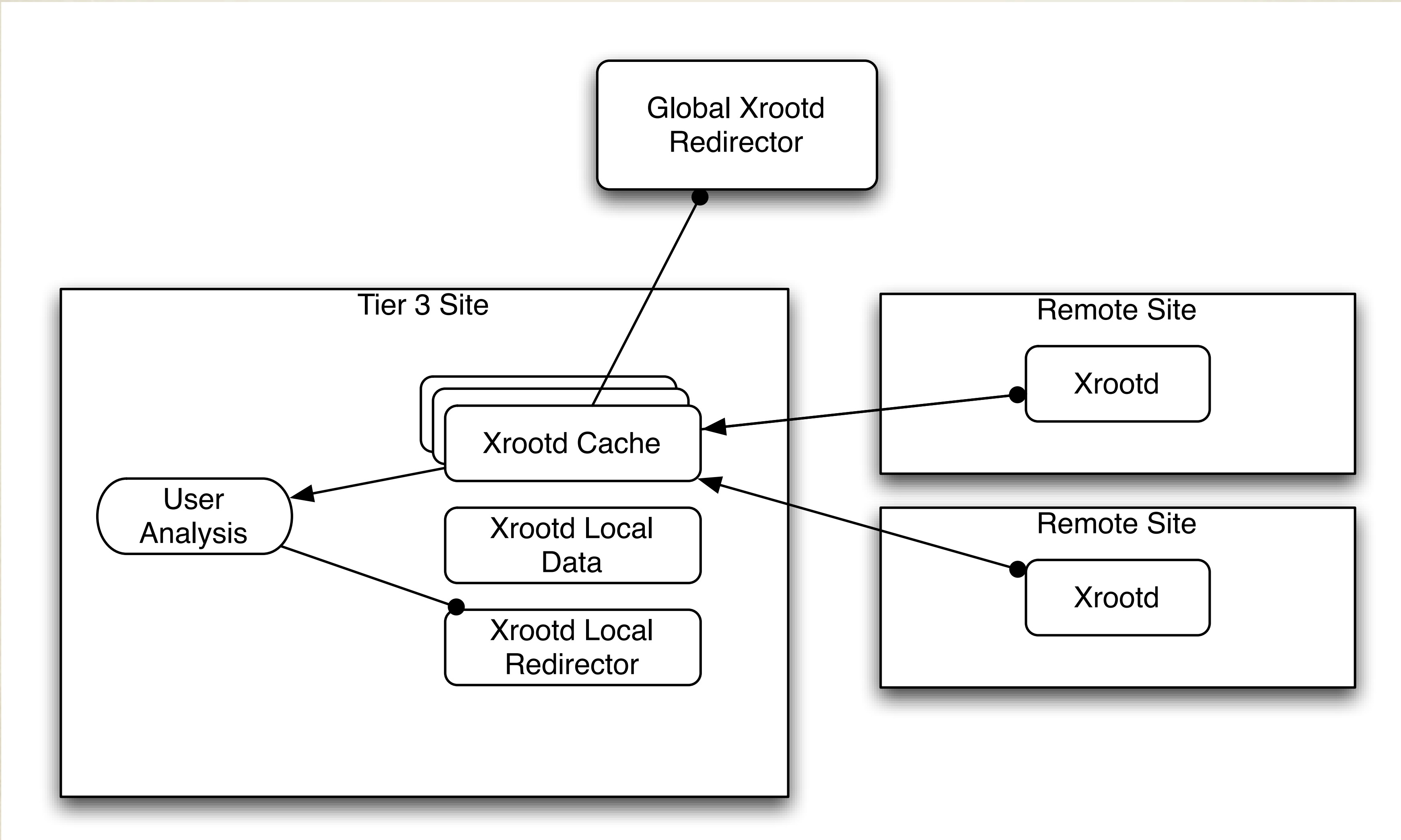
Future scenarios

* **Federating Xrootd:**
  * All data is accessed via a single global namespace (the CMS namespace)
    * No need to know location info
  * The system performs site selection.
    * Or you can use the bittorrent-like mode and download from all sites - this auto- tunes to select the best server.
* **Caching**
  * Xrootd can additionally act as a cache and bring the complete file locally.
  * In this case, the client will talk to a local redirector which will decide whether the file is local and download it from the global federation if not.
    * Once cached locally, the cache can be reused (both by local users and in the global architecture)!

On going storage activities

Future scenarios

* **Caching Architecture:**



On going storage activities

Future scenarios

- **Caching Downloads:**
  - The caching architecture can be combined with the bittorrent mode of xrdcp to optimize the performance of downloads.
    - Errors are only propagated if all sources error out.

- **Issues**
  - Namespace consistency is assumed.
  - Unsure about data integrity issues.
  - Authorization issues when redirecting.
  - Does not solve data archival/metadata issues.
  - Caching approaches have drawbacks thoroughly discussed by computer scientists.

On going storage activities

Few test

- **Evolution of CMSSW access patterns**
  - Version, ROOT reads, actual reads, commentary
    - 3_6_1, 13807, 11038, TTreeCache off (default for release)
    - 3_7_0, 13807, 6264, TTreeCache on (default for release)
    - 3_8_2, 14254, 6711, Increase probably due to construction of index into file
    - 3_9_0, 14014, 3371, Decrease likely due to more aggressive caching (Run and Lumi products are now cached).

A sample, I/O-intensive analysis of 60k evts reading data from FNAL dCache/Xrootd:

Site	Ping time	Wall time
FNAL	.1ms	80s
Nebraska	17ms	80s
CERN	128ms	161s
FNAL/dCap	.1ms	135s

On going storage activities

Xrootd as fallback solution:

* It is already possible to use global xrootd redirector in case of missing or corrupted file in a CMS site.
  * It requires a simple site configuration
  * and no reconfiguration needed as User level
* It is simple also for a site to participate to the global redirector:
  * Plugin is available and installable for dCache, Lustre/GPFS and Hadoop

On going storage activities

HADOOP

* It is one of the most interesting technology that could be investigating

Hadoop: concepts and architecture

* Moving data to CPU is costly
    * Network infrastructure
    * And performance => latency
* Moving computational to data could be the solution
* Scaling the storage performance, following the increase of computational capacity, is hard
* Increasing the number of disks together with the number of CPU could help the performance
* There is the need to take into account machines failures in a computing centre
* DB also could benefit from this architecture

Hadoop: highlight

- It is developed till 2003 (born @google)
- It is a framework that provide: file-system, sched capabilities, distributed database
- Fault tolerant
  - Data replication
  - DataNode failure is ~transparent
  - Rack awareness
- Highly scalable
  - It is designed to use the local disk on the worker nodes
- Java based
- XML based config file

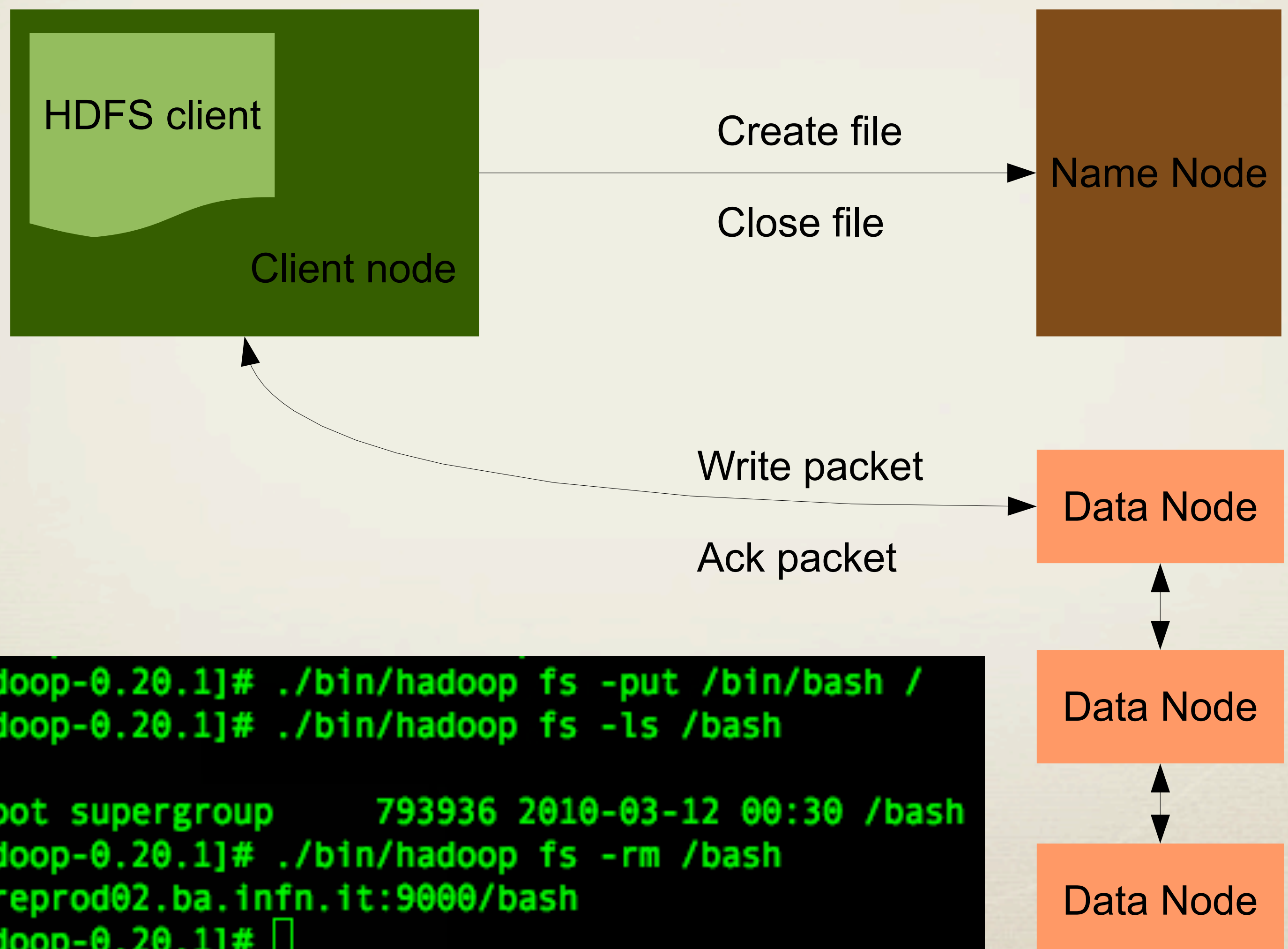
- A9.com
- AOL
- Booz Allen Hamilton
- EHarmony
- Facebook
- Freebase
- Fox Interactive Media
- IBM
- ImageShack
- ISI
- Joost
- Last.fm
- LinkedIn
- Metaweb
- Meebo
- Ning
- Powerset (now part of Microsoft)
- Proteus Technologies
- The New York Times
- Rackspace
- Veoh
- Twitter

Hadoop: highlight

* Using FUSE => some posix call supported
  * Basically "all read operation" and only "serial write operations"
* Web interface to monitor the HDFS system
* Java APIs to build code is data location aware
* CKSUM at file-block level
* SPOF: metadata host
* HDFS shell to interact natively with the file system
* Metadata hosted in memory
  * sync with the file-system
  * it is easy to do back-up of the metadata

Hadoop: concepts and architecture

Anatomy of a file write

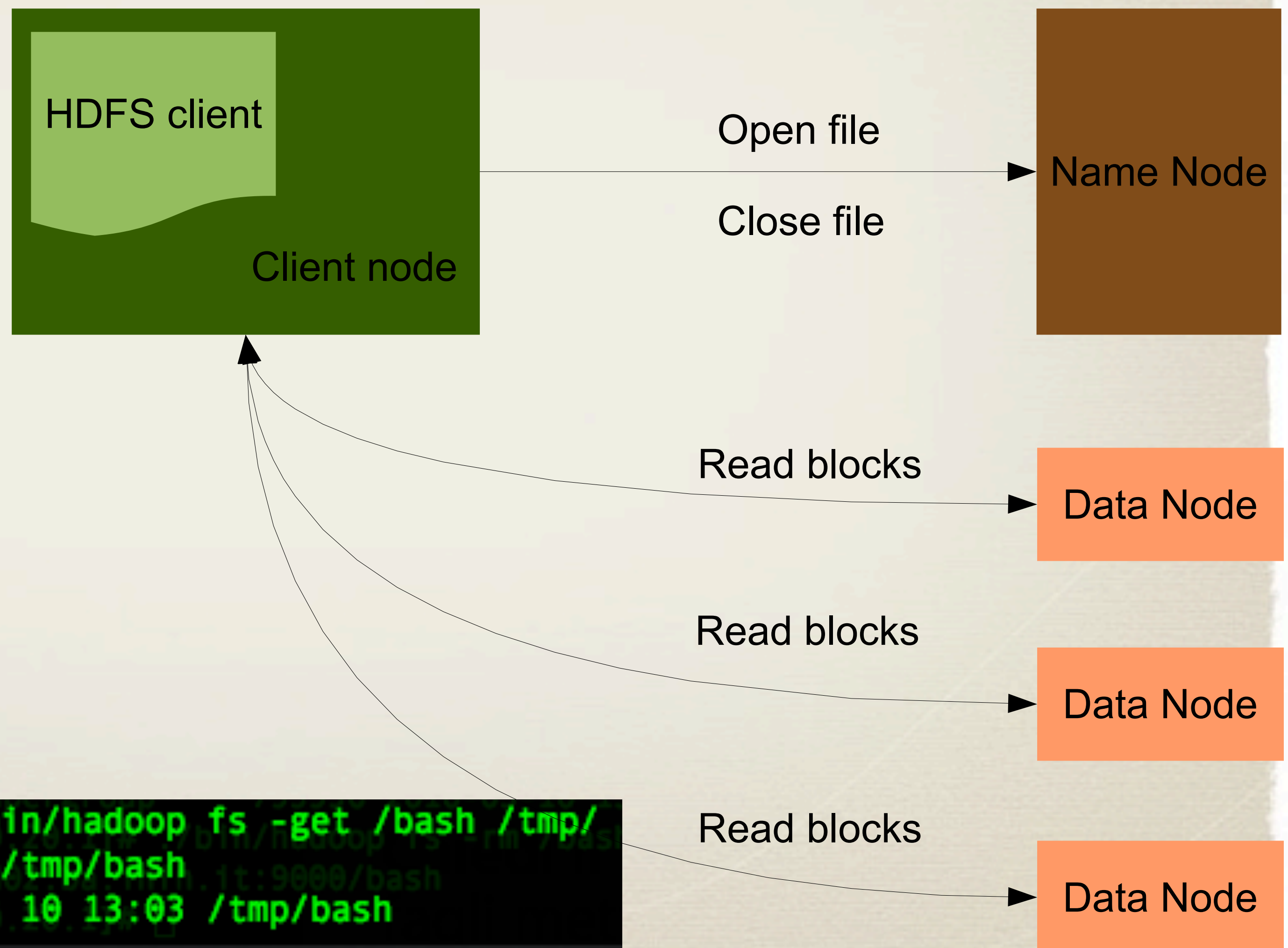



```
[root@pccms64 hadoop-0.20.1]# ./bin/hadoop fs -put /bin/bash /
[root@pccms64 hadoop-0.20.1]# ./bin/hadoop fs -ls /bash
Found 1 items
-rw-r--r--   3 root supergroup     793936 2010-03-12 00:30 /bash
[root@pccms64 hadoop-0.20.1]# ./bin/hadoop fs -rm /bash
Deleted hdfs://preprod02.ba.infn.it:9000/bash
[root@pccms64 hadoop-0.20.1]# 
```

Hadoop: concepts and architecture

Anatomy of a file read

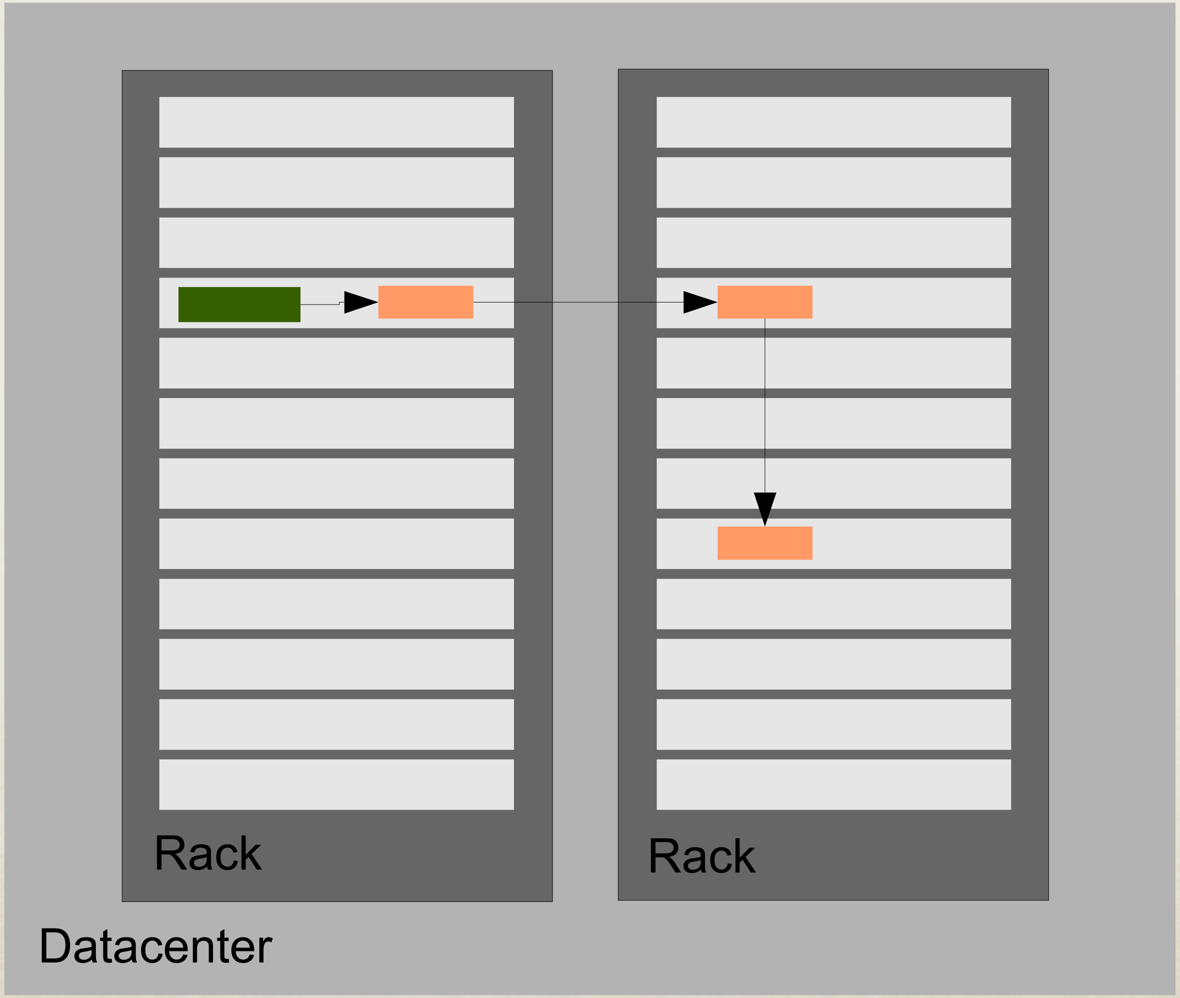
- Splitting files in different pools may give performance benefit when reading them back
- having the data replicated could be of help



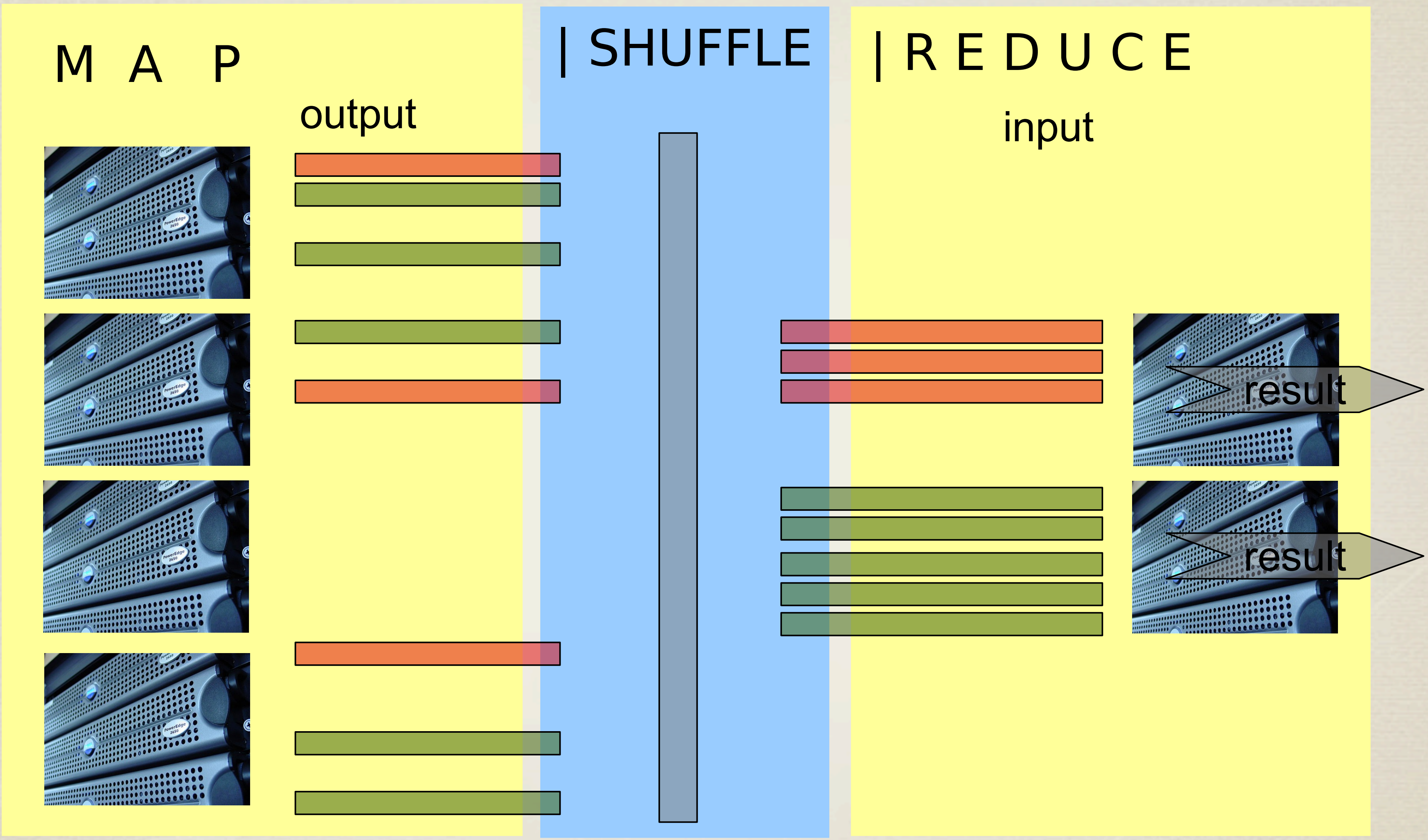

```
[root@pccms64 hadoop-0.20.1]# ./bin/hadoop fs -get /bash /tmp/
[root@pccms64 hadoop-0.20.1]# ll /tmp/bash
-rw-r--r-- 1 root root 793936 Mar 10 13:03 /tmp/bash
[root@pccms64 hadoop-0.20.1]# 
```

Hadoop: concepts and architecture

HDFS Replication Strategy



Hadoop: concepts and architecture



Local to data.
Outputs a lot less data.
Output can cheaply move.

Shuffle sorts input by key.
Reduces output significantly.

Hadoop: few examples

"Sort exercise"

Bytes	Nodes	Replication	Time
500,000,000,000	1406	1	59 seconds
1,000,000,000,000	1460	1	62 seconds
100,000,000,000,000	3452	2	173 minutes
1,000,000,000,000,000	3658	2	975 minutes

10x data
~6x time

Per node: 2 quad core Xeons @ 2.5ghz, 4 SATA disks, 8G RAM (upgraded to
16GB before petabyte sort), 1 gigabit ethernet.
Per Rack: 40 nodes, 8 gigabit ethernet uplinks.

Hadoop: few examples

"CMS example" (T2_US_Nebraska)

- **Numbers**
- 2.5TB < Each DataNode < 21TB
- ~260 servers
- 1.5PB of storage (700TB really usable)
- ~1600 Core
- SRM/gridftp layer provided by FUSE and BestMan
- Xrootd export

- **Reported Prod & Cons**
- Easy to deal with failures (file-systems, datanodes, racks, etc)
- Scalable
- Open Source
- Few monitoring tool already available
- Reliance on FUSE
- Real cost vs availability vs performance ?
  - CPU efficiency?

Hadoop: few examples

- **Geographical distributed Storage Element**

- Hadoop provides:
  - automatic replica management and storage distribution
  - rack awareness
  - advanced (and *pluggable*) placement policies
  - good monitoring features
- Why don't we try to use it on a WAN environment to see how it works?
  - The concept of rack is used to identify a Site
  - We need a performant WAN link between site
  - It could provide good reliability of data... also in case a whole site become temporarily unavailable

Hadoop: few examples

- **Geographical distributed Storage Element**

Live Datanodes : 10

Node	Last Contact	Admin State	Configured Capacity (GB)	Used (GB)	Non DFS Used (GB)	Remaining (GB)	Used (%)	Used (%)	Remaining (%)	Blocks
dbserv1	2	In Service	931.27	54.23	0	877.04	5.82		94.18	898
dbserv2	1	In Service	931.27	52.98	0	878.29	5.69		94.31	880
pccms31	1	In Service	43.28	0.1	2.39	40.79	0.24		94.24	1
superb01	0	In Service	213.42	29.74	11.02	172.66	13.93		80.9	494
superb02	2	In Service	225.54	31.7	15.65	178.18	14.06		79	390
superb03	0	In Service	213.42	23.73	11.02	178.67	11.12		83.71	371
superb06	2	In Service	96.9	21.11	0	75.79	21.78		78.22	343
superb07	2	In Service	96.9	21.45	0	75.45	22.13		77.87	350
superb08	0	In Service	100.62	23.43	0	77.19	23.29		76.71	382
superb09	0	In Service	100.62	23.02	0	77.61	22.87		77.13	376

Configured Capacity	:	2.88 TB
DFS Used	:	281.49 GB
Non DFS Used	:	40.1 GB
DFS Remaining	:	2.57 TB
DFS Used%	:	9.53 %
DFS Remaining%	:	89.11 %
Live Nodes	:	10
Dead Nodes	:	2
Decommissioning Nodes	:	0
Number of Under-Replicated Blocks	:	0

Bari (red)

Naples (blue)

Hadoop: few examples

- **Geographical distributed Storage Element**

HARDWARE IN THE NAPLES SITE FOR THE FIRST TESTBED WITH THE BARI SITE

3 SERVER R200 WITH 2 GigabitETH IN BONDING
250GB OF DATA DISK AVAILABLE

10 SERVER BLADE WITH 2 GigabitETH IN BONDING AND 100GB OF DATA DIKS AVAILABLE

THE SERVER ARE CONNECTED ON A 1Gbit/s SWITCH

OS - SL5.3

Hadoop: few examples

- **Geographical distributed Storage Element**

HARDWARE IN THE Bari SITE FOR THE FIRST TESTBED WITH THE BARI SITE

3 SERVER SuperMicro WITH 2 GigabitETH IN BONDING
5 disk in total from 50GB to 500GB

THE SERVER ARE CONNECTED ON A 1Gbit/s (non-blocking) SWITCH

OS - SL5.4

- Namenode are installed at Bari
  - SecondaryNameNode will be installed at Naples

Hadoop: few examples

WORK IN PROGRESS

* **Geographical distributed Storage Element**

* Few test:
  * Network bandwidth: ~600 Mbit/s
  * during a read operation the user do no see errors also if the whole Naples site goes down suddenly
  * Writing & Replicating data (2 clients): ~40MB/s sustained
  * Reading data (2 Client): ~100MB/s sustained

Conclusions

* LHC Community is trying to move away from a rigid and schematic data-management framework to a more flexible and dynamic one
    * It is important that the new framework is much more transparent and user friendly
    * It is important to look at already in place technologies as the time-scale is 2013
* It is important to work on the experiment software framework as this could lead to great improvement in performance and efficiency:
    * The framework should cooperate with the storage system as much as possible