HEPiX Storage Working Group
- progress report 1.2009 -

Andrei Maslennikov

May 28, 2009 – Umeå

Summary

- Activities Fall 2008 – Spring 2009
- New test results obtained at FZK
- Plans for the next future
- Discussion

Activities Fall 2008 – Spring 2009

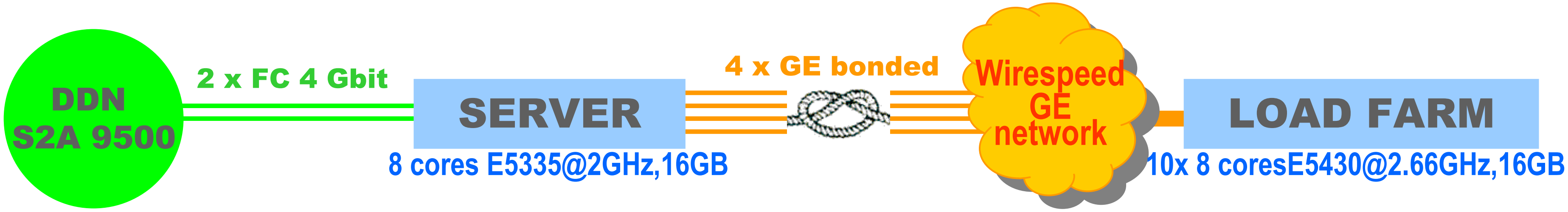
- These people took part in the meetings / discussions / tests:

CASPUR	A.Maslennikov (Chair), M.Calori (Web Master)
CEA	J-C.Lafoucriere
CERN	B.Panzer-Steindel, A.Wiebalck
DESY	M.Gasthuber, Y.Kemp, P.van der Reest
FZK	J.van Wezel, A.Trunov
FNAL	G. Oleynik, A.Kulyavtsev
INFN	G.Donvito
LAL	M.Jouvin
RZG	H.Reuter
SLAC	A.Hanushevsky, A.May

- In this period the group was mainly concentrating on the questions which remained open since the Taipei meeting. In particular, we wanted to cover GPFS performance for the CMS use case and compare it with others.

Where we rounded up last time..

In October 2008 we were using the following simple test setup kindly provided by Forschungszentrum Karlsruhe (special thanks to Manfred Alef, Artem Trunov, Jos van Wezel and Bruno Hoeft):

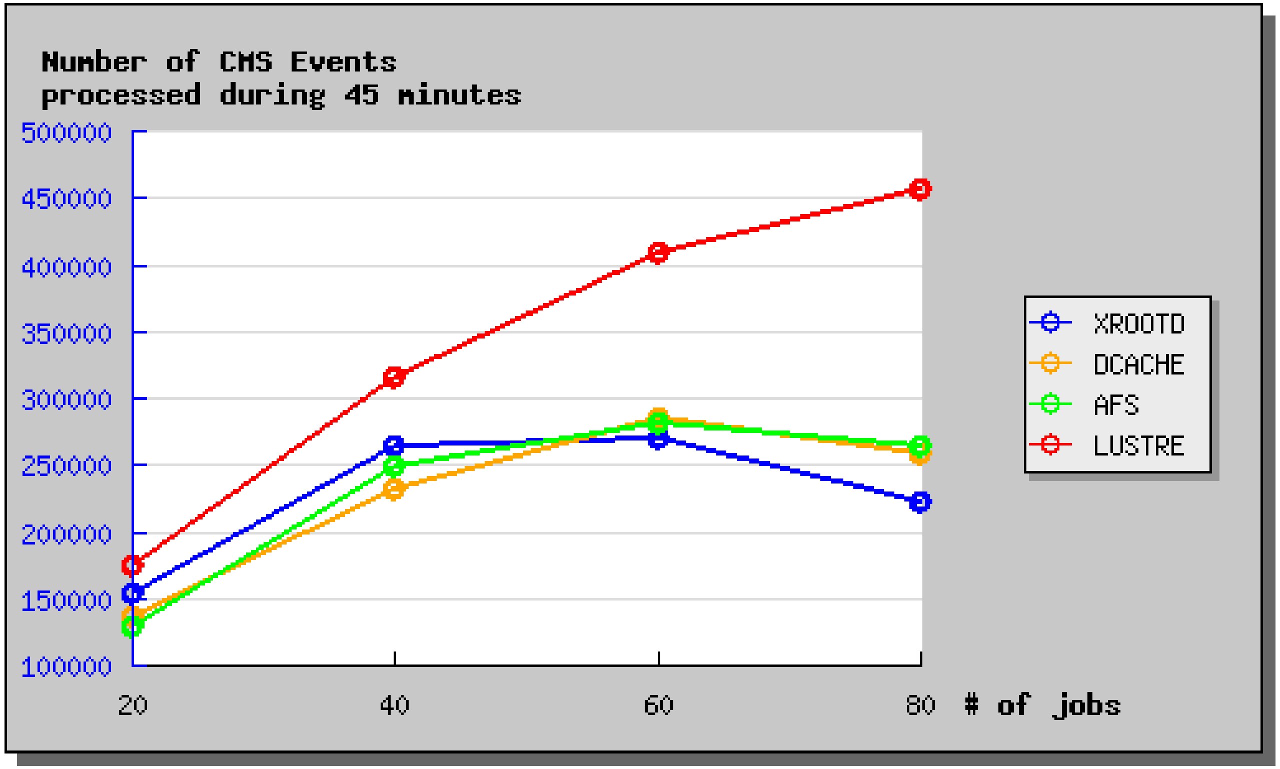


We used a realistic stand-alone use case (CMS data + analysis program).

The data sample was composed of 1500 CMS data files of 1.3-1.5GB stored under AFS, Lustre, XFS and accessed either directly via fs, or via Xrootd or dCache.

A single instance of the test program was capable to process the input data at a rate of 5 MB/sec. We were running 20,40,60,80 simultaneous jobs spreading them evenly over the load farm. The hardware used (disks and NICs) was capable to handle the data flow in excess of 400 MB/sec, so no hardware bottlenecks were present.

As a benchmark, we simply counted the aggregate number of CMS events processed by all jobs during a 45-minunte run period. Such a measurement has a clear advantage of being able to demonstrate an overall efficiency of the storage service as a whole. As the hardware base remained invariant, we were able to compare different solutions directly.



Lustre definitively demonstrated itself as being the most efficient solution for this particular hardware configuration and the test program used.

We were still missing GPFS numbers for CMS use case, so we decided to re-run the test in 2009, preferably on a different hardware.

New setup, sponsored by FZK and E4

- In 2009, we still could count on the hospitality of FZK. As well, we have obtained a new test machine from E4 which agreed to sponsor these test activities (and we thank them for this!). This unit was nothing else as a typical disk server identical to those that CERN and other sites procure in large numbers:

  - Intel E5405 @ 2.00GHz (2x4=8 cores in total)
  - 8 GB of RAM
  - 4 GigE outlets in bonding/alb (measured 450+ MB/sec aggregate)
  - 3Ware 9650SE, 16 disk spindles

  - RHEL 4.7+ /64bit,
  - Lustre 1.6.6 (no checksumming),
  - OpenAFS 1.4.8, 1.4.8/OSD
  - GPFS 3.2.1-7

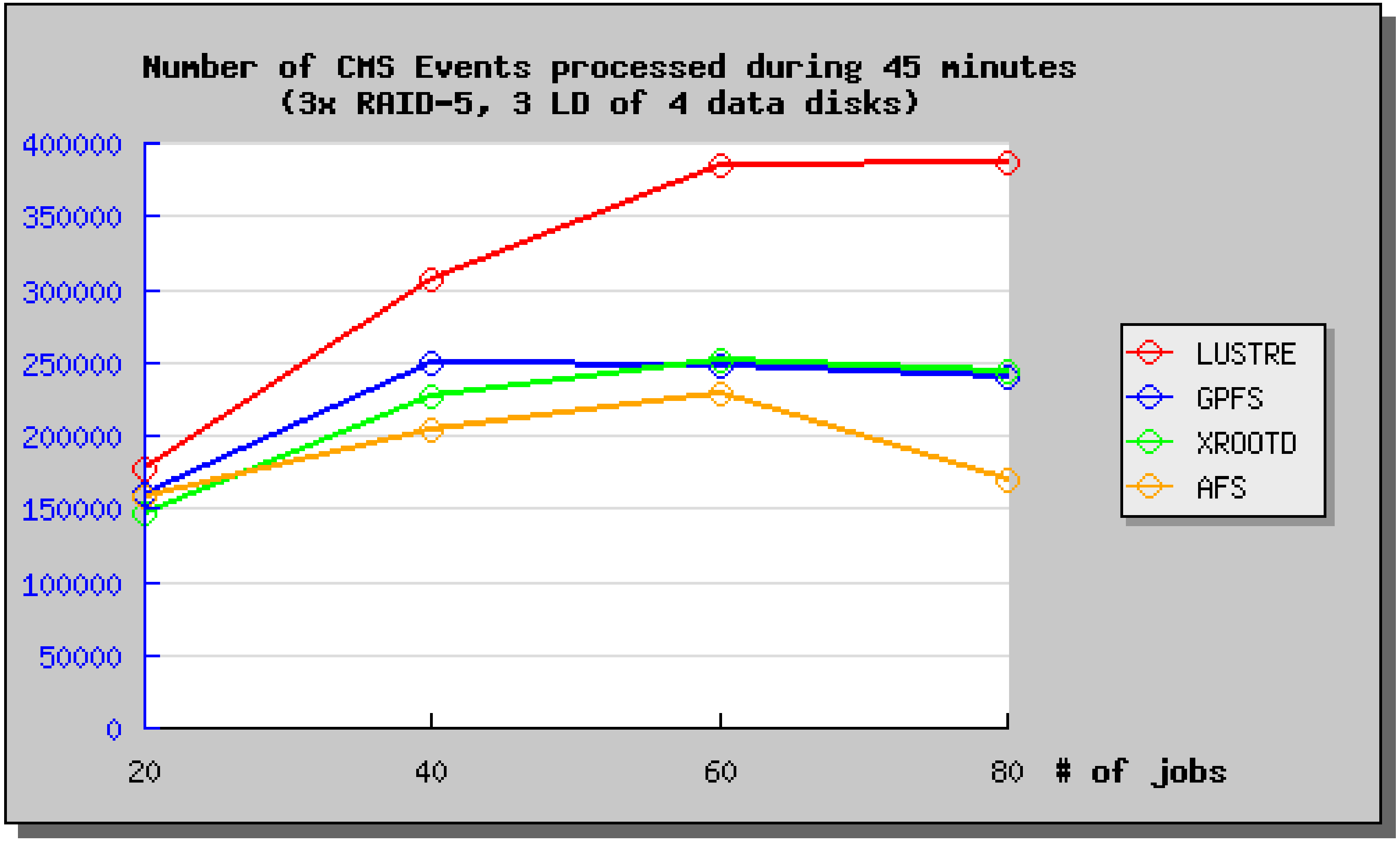
Test layout

- To make a fair comparison between Lustre and GPFS, we used three RAID-5 logical drives of five spindles each, rather than one single RAID-6 logical drive (at a loss of 2 data disks). This was recommendable since GPFS delivers best performance when its blocksize is proportional to the number of data drives times the stripe size.

- We tried to tune GPFS as well as we could; in particular, we varied the file system block size, pagepool size etc.

- We also used this very setup to re-run the CMS benchmark for Xrootd and AFS.

Results



Once again, Lustre outperformed all the others…

Improving AFS performance

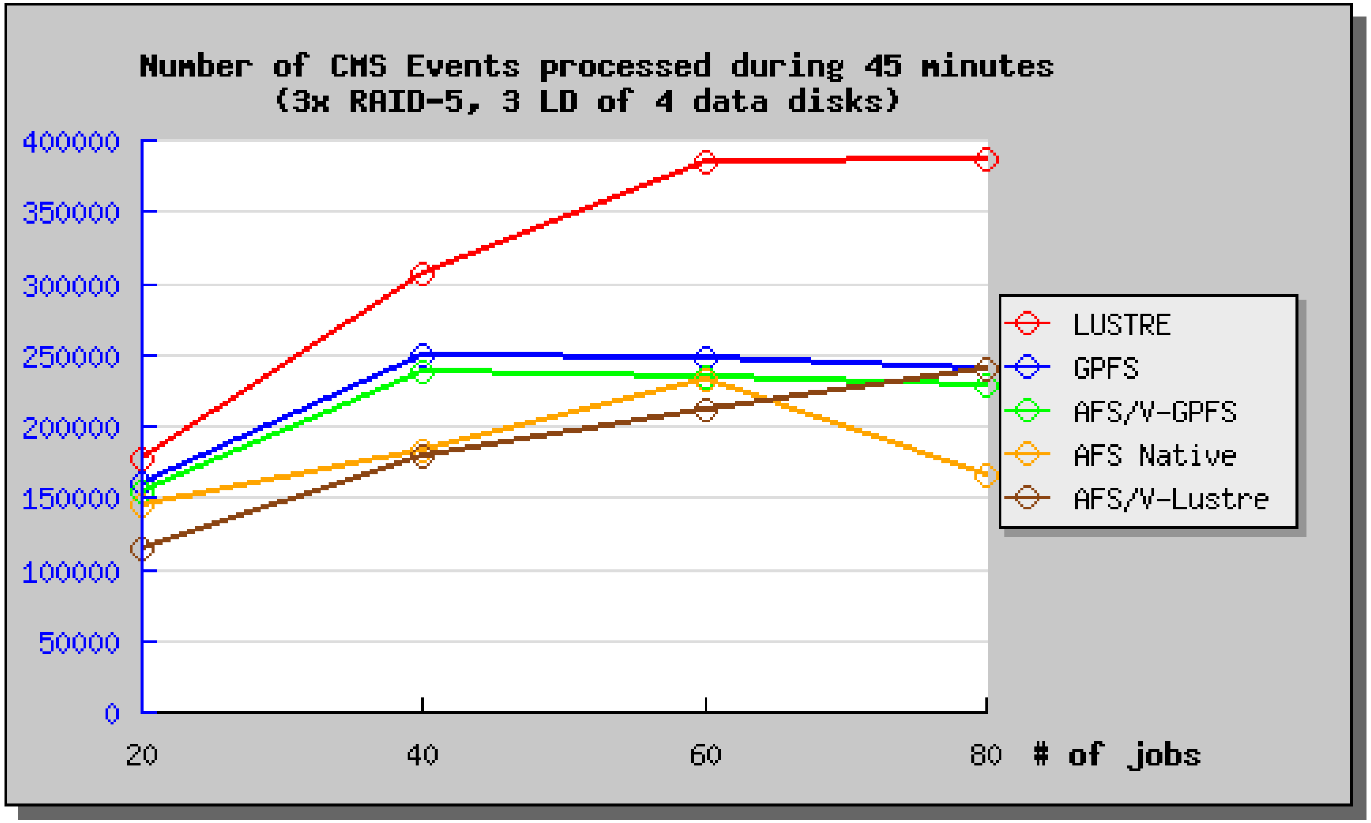
- AFS was doing visibly worse than all other solutions. We hence used an opportunity to try the newly available AFS/Vicep extension (it now makes part of the H.Reuter's AFS/OSD project).

- This extension allows, in case /vicepXX server partitions are mounted on the clients, to directly access the data stored on them.

  If GPFS or Lustre is being used to store the /vicepXX content on the AFS server, clients may be configured to make use of /vicepXX locally.

- We have then tried AFS/Vicep for both /vicepXX stored in GPFS and Lustre. Out of the two, AFS/Vicep over GPFS demonstrated an improvement against the vanilla AFS. Instead, AFS/Vicep over Lustre was not doing very well.

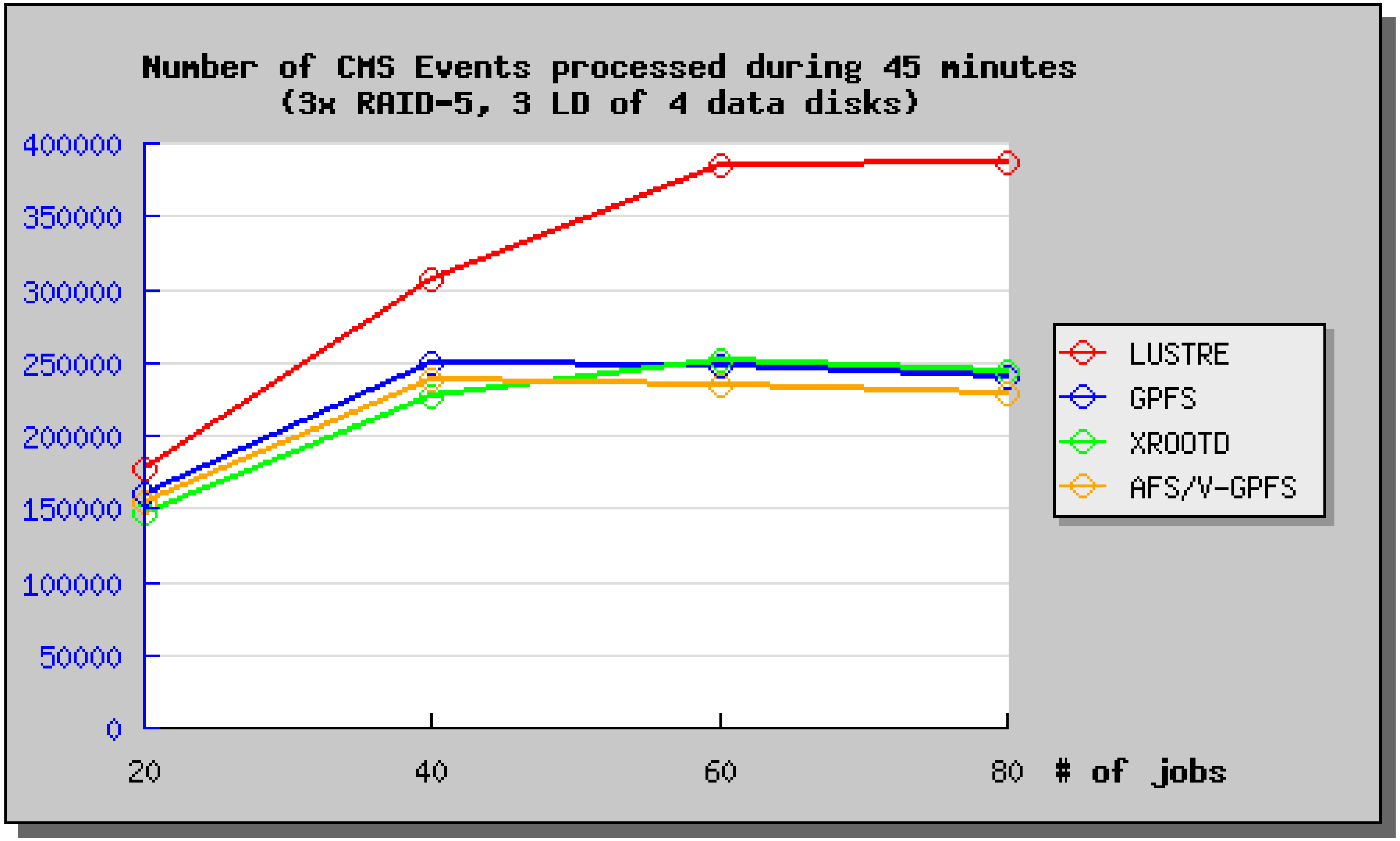
AFS/Vicep



As may be noted, AFS/Vicep-GPFS is very close to the native GPFS. This is definitively an improvement over native AFS. Instead, AFS/Vicep-Lustre is far away from native Lustre...

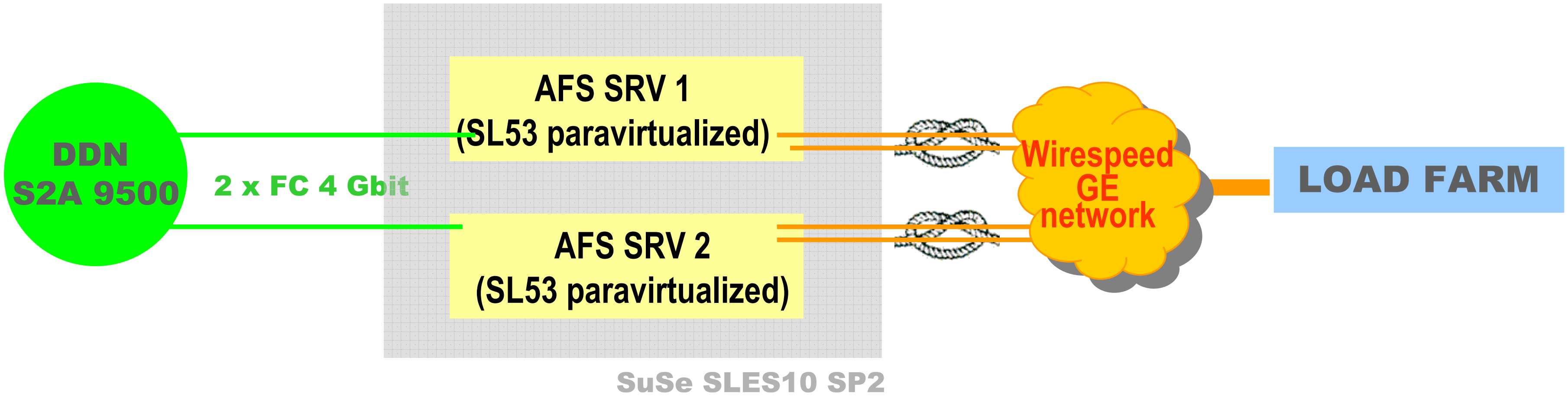
NEWS (27/05/2009): it appears that Felix Frank from DESY may have found a workaround for AFS/Vicep-Lustre code! We shall be testing it shortly.

Current (best effort) results



When last year we had seen that dCache, AFS and Xrootd delivered very similar results, we thought that some common blocking denominator, like XFS, could be accounted for this. However, GPFS is a totally different technology (and no XFS is involved), and it is still not doing any better than others… Thus it looks like Lustre is “simply” doing a better job.

May this improve the aggregate AFS performance?



A single AFS file server does not use all the available CPU, so the idea was to run two servers simultaneously in the same physical box and see what happens. Everything worked as it should, but at a first glance performance results are somewhat worse then those obtained without virtualization. Will still try to do some extra tuning…

Plans for the next future

- We plan to continue with performance studies, possibly employing new use cases and new hardware. Immediate plans include further evaluation of AFS/Vicep-Lustre solution and a closer look at AFS server virtualization.

- We certainly need to create a representative collection of use cases, the one that we are using now is clearly not enough. Ideally, we should be obtaining one recognisable standalone analysis job for each of the four major LHC experiments, and run tests against realistic data files stored in dCache, Xrootd, Lustre, AFS and GPFS.,

  Another use case to look at is a compilation process of a typical (giant) LHC analysis code inside Lustre, AFS, GPFS.

- Other field of activity will be the maintenance of technology tracking pages and yearly update and reporting of the storage situation accross the participating sites.

Discussion