Journal of Physics: Conference Series **396** (2012) 032090 doi:10.1088/1742-6596/396/3/032090

Data Storage Accounting and Verification at LHC experiments

C-H Huang[1], E Lanciotti[2], N Magini[2],N Ratnikova[3,8], A Sanchez-Hernandez[4], C Serfon[5], T Wildish[6], X Zhang[7]

[1] Fermi National Accelerator Laboratory, US
[2] CERN, CH
[3] KIT - Karlsruhe Institute of Technology, DE
[4] Centro Invest. Estudios Avanz. IPN, MX
[5] Ludwig-Maximilians-Univ. München, DE
[6] Princeton University, US
[7] IHEP Beijing, CN
[8] also at ITEP, RU

E-mail: Natalia.Ratnikova@kit.edu

Abstract. All major experiments at the Large Hadron Collider (LHC) need to measure real storage usage at the Grid sites. This information is equally important for resource management, planning, and operations. To verify the consistency of central catalogs, experiments are asking sites to provide a full list of the files they have on storage, including size, checksum, and other file attributes. Such storage dumps, provided at regular intervals, give a realistic view of the storage resource usage by the experiments. Regular monitoring of the space usage and data verification serve as additional internal checks of the system integrity and performance. Both the importance and the complexity of these tasks increase with the constant growth of the total data volumes during the active data taking period at the LHC. The use of common solutions helps to reduce the maintenance costs, both at the large Tier1 facilities supporting multiple virtual organizations and at the small sites that often lack manpower. We discuss requirements and solutions to the common tasks of data storage accounting and verification, and present experiment-specific strategies and implementations used within the LHC experiments according to their computing models.

1. Introduction

LHC physics data are produced at high rates, and are distributed to many sites for storage, reprocessing and analysis. Data are replicated at the sites to provide safe custody of raw data, to optimize the usage of the compute resources and to provide users with efficient access to the data of their interest. Data placement involves planning to meet these needs, as well as allowing for sporadic activities that cannot be easily foreseen.

The sites are allocating storage resources for the experiment's data which are formalized in terms of pledges. Within the limited storage resources available, the use of storage becomes very dynamic: while new data are constantly arriving, secondary data replicas are cleaned out. Information about the actual use of storage helps to make better operational decisions and more efficient use of the available resources. For the purpose of the following discussion we define *storage accounting* as the experiment's use of the space allocated to them by the sites.

In particular, we consider that storage accounting implies detailed enough information that an experiment can see not only their total usage, but the distribution of their usage with different granularities, typically at the level of an analysis group or by data type.

The experiments maintain central data catalogs representing their knowledge of the storage contents. These central catalogs typically store information in terms of the logical file names (LFNs). Due to the fact that storage elements and central file catalogs are completely decoupled entities, they are not necessarily consistent. One example of inconsistency is files registered in the central file catalog, but missing on storage. This may cause batch jobs to crash when they try to access the missing data. Another example is files found on storage but not registered in the central file catalog, perhaps due to failures in deleting files. If a large number of these errors accumulate over time, a significant amount of resources may be wasted. Both these problems are illustrated in Figure 1.

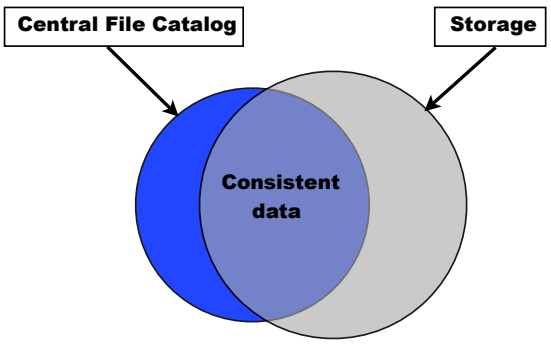

Figure 1. Examples of inconsistent data catalogs. Files outside the set overlap shown in blue color are listed in the central File Catalog, but missing on storage. Grey area represents files found on storage but not listed in the central File Catalog.

Accounting and verification tasks are common for distributed storage on the LHC Computing Grid [1], despite the different solutions implemented by the experiments for their central file catalogs, and despite the range of storage technologies in use throughout the Grid sites. The experiments, and many of the Grid sites, have already invested some considerable effort in providing local solutions to these problems, but until now no framework for a common solution existed. Now, with the ever-growing volumes of data the experiments have to deal with, and the wide variety of storage element technologies, it is becoming increasingly important to find common solutions.

2. Use cases and requirements

The main stakeholders in the storage accounting and verification are:

- *Grid sites.* Responsible for receiving, handling and storing the data, the grid sites monitor closely the storage contents and the available space. Regular data verification is important for timely detection of the lost or corrupted data and for the cleanup of the orphaned files [2]. Complete verification may also be necessary after a major transition or upgrade in the storage system.
- *Central computing operations* are responsible for efficient resource utilization. Space usage is monitored and extrapolated to estimate resource needs for all central computing tasks, including coordinated data production and processing, transfers to other sites for custodial storage or replication of popular data.

- *Physicists.* Storage accounting helps to maintain users space and storage areas allocated to the physics analysis groups, especially on the storage systems where hard quotas are not supported. Data verification helps to insure that data needed for the physics analysis are intact and found where they are expected.
- *High-level computing management* uses storage accounting information for long-term resource planning and reporting. Verification results can be used as a general indicator of the stability and efficiency of the storage system.

2.1. Use cases

The fully fledged scenarios and use cases will differ from experiment to experiment, depending on the computing model, data organization, monitoring solutions and environment adopted by the collaboration. Here we name a few examples of typical use cases common for all experiments.

- *Monitoring of the resource health and utilization:*
  - total storage space used at the sites compared to the pledged resources;
  - detection of files not known to the central data catalogs, or leftover from partially failed removal of data, which could be removed to free disk space;
  - detection of files missing on storage but still present in the central data catalogs;
  - verification of consistency of the file attributes values stored in the data catalogs.
- *Storage usage accounting by users and groups:*
  - how much and what data are stored at the sites by data type, dataset, data owner, custodiality.
- *Data production monitoring:*
  - how many files have been produced, grouped by production type (MC, reco), by data type.
- *High-level resource management:*
  - e.g. study historical trends in resource usage and data placement.

2.2. Requirements

Detailed descriptions of ATLAS and CMS data verification systems and tools for checking data consistency have been presented earlier in [3] and [4]. Here we present the requirements for the storage accounting system.

- the system should provide information that can be easily converted into formats suitable for the end users;
- the collaboration policies, as well as site specific policies and general data privacy considerations should be taken into account;
- the system should give a general overview of the resource usage and allow to investigate the details interactively;
- the resource usage must be accounted in terms of the *occupied space in bytes* and the *number of stored files*;
- historical view of the storage usage metrics is required.

3. Overview of the current implementations

3.1. ATLAS

The central ATLAS data catalog keeps accounting of the used space according to ATLAS Distributed Data Management (DDM) [5]) and Storage Resource Manager (SRM)[6] systems,

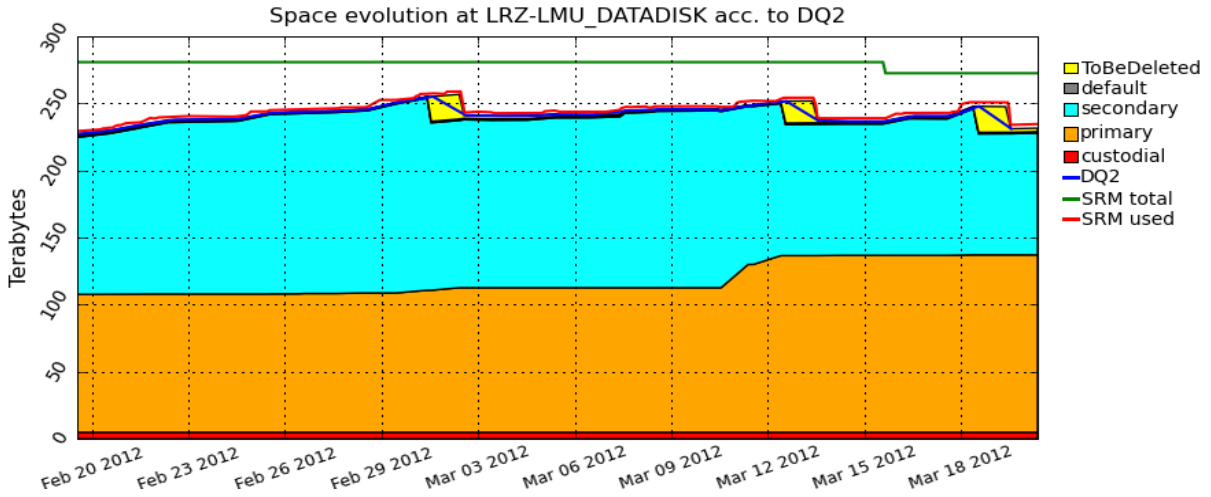

Figure 2. Historical view of space usage by ATLAS at one site

and total allocated space from SRM, for each *space token* (a special file attribute managed by SRM).

Plots in Figure 2 show historical space usage and also spot a possible discrepancy between SRM and file catalog. If used space gets close to the total available, automatic site cleaning systems identify secondary replicas to be removed. Plots in Figure 3 are being used to monitor

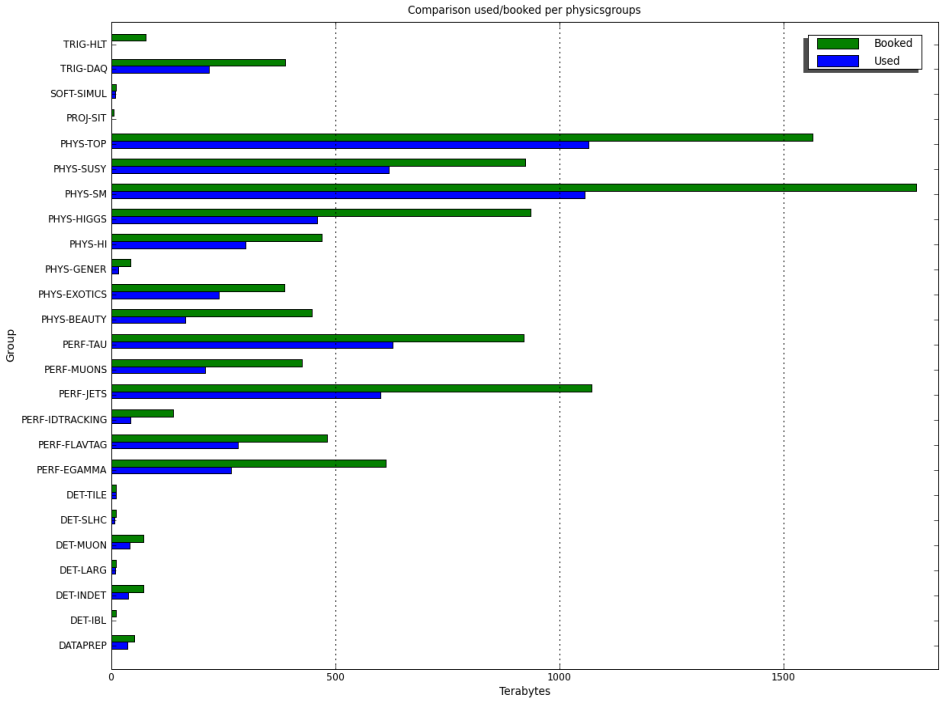

Figure 3. Comparison of booked and used space per ATLAS physics group

the physics groups space usage. One can see a comparison of the allocated and used space per physics group. Also break-down by site is available.

Development [5] is ongoing to completely change the backend, allowing more powerful and performing data accounting by project, data type and other relevant meta-data.

3.2. CMS

CMS [7, 8] stores data information in two databases:

- *Data Bookkeeping System (DBS)*[9] keeps records of produced data and production conditions;
- *Transfer Management Database (TMDB)* of the CMS Physics Experiment Data Export (PhEDEx) [10] system keeps information about the replicas stored at the sites.

Figure 4 shows an example of historical view of CMS resident data provided by the PhEDEx monitoring. Example in Figure 5 shows a pie diagram of the physics group data stored at

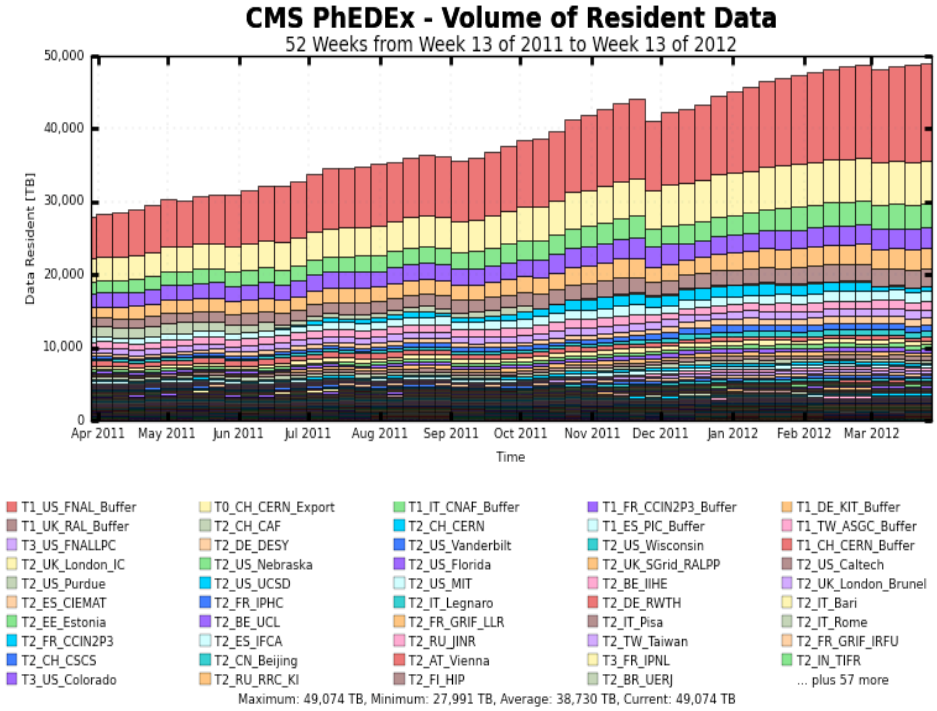

Figure 4. Historical view of volume of Resident Data at CMS Tier1 and Tier2 sites

the sites provided by the FileLight [11] storage visualization tool. FileLight is implemented as a part of the CMS Overview framework and uses the PhEDEx Data Service REST API to retrieve JSON data from TMDB, then renders it into a plot, serving a PNG image. In the new version of the Overview framework the rendering is done in the browser using the D3 [12] plotting library.

To enable accounting and monitoring for all CMS data, not only data registered in TMDB, a new Space Monitoring tool is being developed. It provides:

- accounting for all data on the SE;
- historical view of storage usage;
- auto-detection of orphaned files;
- possibility to interactively explore aggregated storage information.

Although the new CMS accounting tool reuses PhEDEx code base, it is not CMS-specific in any way. The concept, main components and implementation of this tool are discussed in section 5.

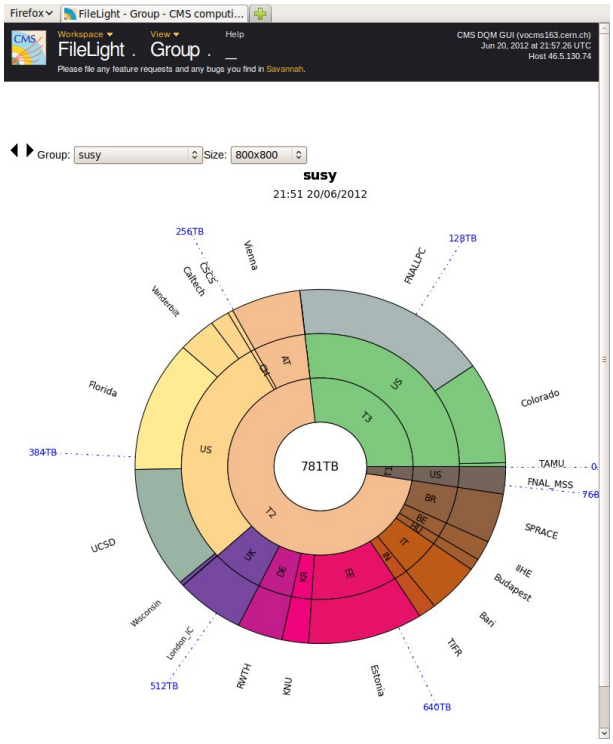

Figure 5. CMS SUSY physics group data distribution by data tier, country and site.

3.3. LHCb

LHCb data are managed centrally through the Data Management System (DMS) of the LHCbDirac framework [13]. The already existing tool for storage usage monitoring and accounting provided by the Service Level Status (SLS) [14] system is based on the information reported by the Storage Resource Manager (SRM) [6] system and shows the total amount of used and available space per site, split by SRM space token. To give a more detailed view of the storage contents a new system has been implemented under request of the experiment. It provides space usage accounting based on the file meta-data which are meaningful for the experiment (activity, data taking conditions, simulation conditions, application version, data type, etc...), so that the collaboration can easily monitor how much space is used at a given site as a function of these parameters.

The new accounting system is based on two catalogs:

- LFC replica catalog [15];
- LHCb Bookkeeping meta-data catalog for production files [16]

Figure 6 presents an overview of the storage accounting implementation in LHCbDirac framework. Some examples of accounting plots are Figure 7, which shows the number of files produced for a given reprocessing grouping by file type, and Figure 8, which shows the amount of space used at the sites where the processed output data are replicated. The two accounting tools are complementary: whereas the information displayed by SLS comes from a direct query to the SRM and shows the real space usage on the storage element, the new accounting system queries the LHCb file catalogs, thus giving a much more detailed information on the type of data. Of course a good synchronization between file catalogs and grid storage elements has to be ensured for the reliability of the new accounting plots.

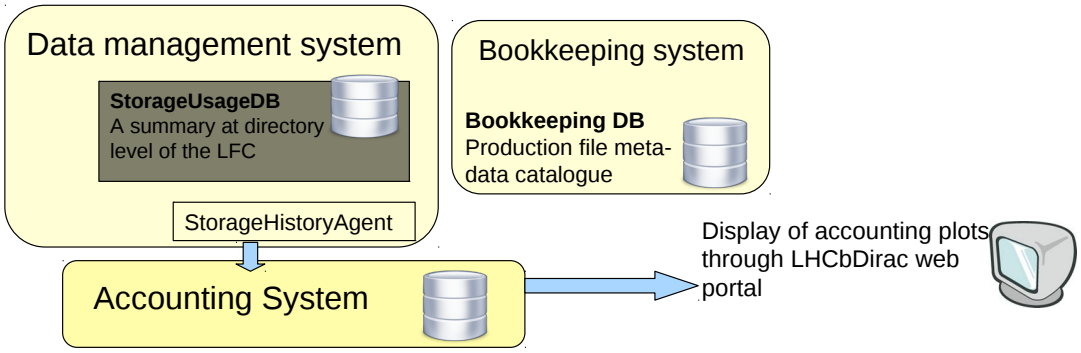

Figure 6. LHCb storage accounting system components and workflow

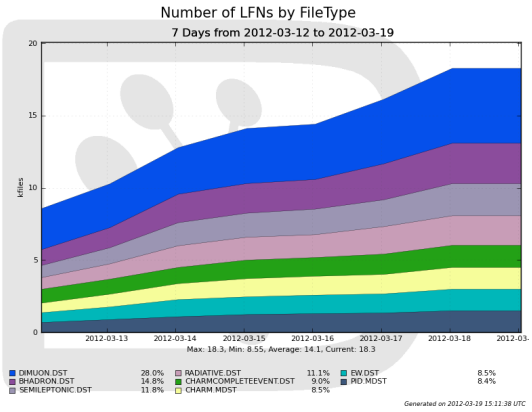

Figure 7. LHCb example of monitoring the progress of data reprocessing.

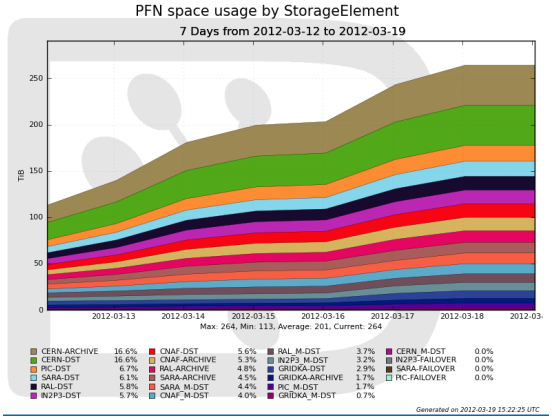

Figure 8. LHCb example of monitoring the progress of the datasets replication.

4. Common solution based on storage dumps

To verify the consistency of central catalogs, experiments are asking sites to provide a full list of the files they have on storage, including size, checksum, and other file attributes. Such storage dumps are being successfully used by ATLAS and CMS to verify the consistency between the central data catalogs and the storage element contents. Corresponding tools and operational

procedures have been presented earlier [3, 4].

In 2011 the CERN IT Experiment Support (IT-ES) group came with the initiative to define a common format suitable for all LHC experiments, and to coordinate requests to the LCG sites to produce storage dumps. Status and results of this activity in a context of data consistency checks have been presented at the WLCG Tier1 Service Coordination Meetings and published at the LCG Twiki pages [17]. It has been agreed among the experiments that the required information is:

- *Logical File Name*, or Physical File Name as addressed on the storage;
- *file creation time*;
- *file size*;
- *space token*, if used by the experiment;
- *checksum value*, adler32, optional;
- *exact time stamp of the creation of the dump.*

Storage dumps produced by the sites give the most complete view of the storage contents. If provided at regular intervals, they can be used for storage accounting and space usage monitoring of data not registered in the central data catalogs (so-called 'dark data').

In CMS it is important for monitoring disk space usage at Tier2 sites, which provide storage for user data not registered in the central data catalogs. Requirements and motivation for a common space monitoring tool based on storage dumps have been streamlined by the conclusions of the CMS Monitoring Task Force presented in [18].

Due to a common nature of storage dumps, which do not depend on the experiment computing model, a common space accounting tool can be suitable for any site or experiment.

4.1. Scheduling considerations

Producing storage dumps and running consistency checks may impose additional load on the system. The optimal schedule depends on the storage technology used, local site infrastructure, and other activities sharing the same resources. In many cases sites can dump their storage contents as frequently as twice a day. Analysis of the use cases and the operational experience by the sites and the experiments to date shows that weekly produced reports based on storage dumps satisfy most accounting needs.

4.2. Implementation example

The diagram in Figure 9 shows the general architecture and the workflow of the CMS implementation of storage accounting based on storage dumps.

Storage dumps are generated at the sites by a technology-specific utility. Information is extracted, aggregated into a compact record and inserted into the central database using a writable data service API. The monitoring application queries the database via a read-only data service API for an interesting set of data (a variety of convenient formats are available) and passes it to the visualization module.

5. Components of a common solution

5.1. Information provider

Depending on a given storage technology, local site infrastructure and administration policies, storage dumps are created using one of the two possible approaches:

- obtain storage element information from the database (either by query, or via full database dump);
- use the direct storage access protocol to traverse the file system.

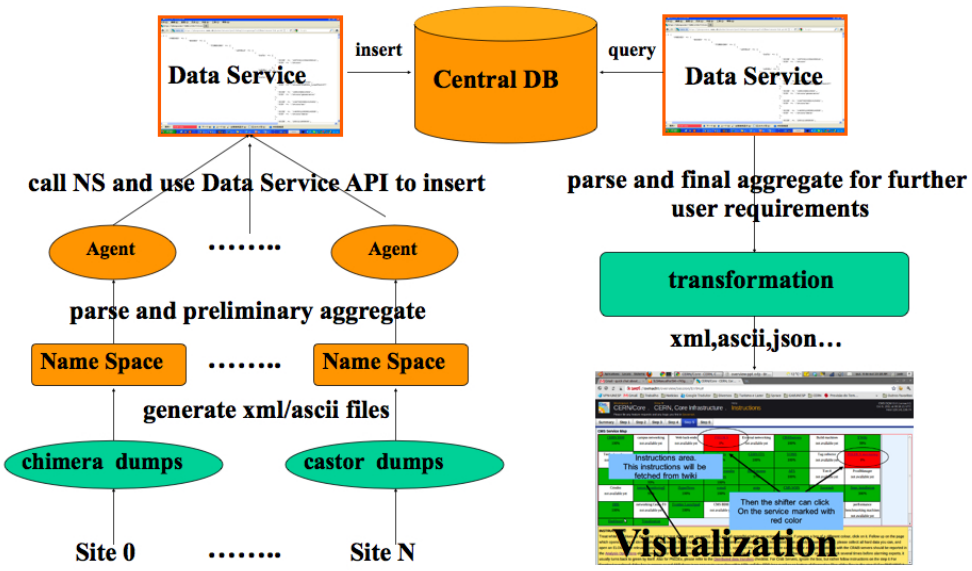

Figure 9. General Storage Accounting Architecture

Ideally, a uniform format, such as syncat [2], should be used for the storage dump. In practice output formats for different technologies will often differ. An additional adaptation layer is therefore introduced to mitigate the differences. We use the PhEDEx Namespace framework [19] designed to provide a uniform access interface to the storage element via technology-specific plugins. Namespace plugins are successfully used in the PhEDEx Consistency Tools suite for routine checks at all CMS Tier1 sites.

In early 2012 the CMS central data operations team launched a data consistency checking campaign at pilot Tier2 sites representing storage technologies used across CMS, with the aim to extend regular consistency checks to all Tier2 sites. The main goals were to validate the usability and performance of the available tools, and to enhance support for not-yet-covered storage technologies used at Tier2 sites.

The participating pilot sites have been asked to produce a storage dump following the WLCG recommendations [17] and to use it as an input for the PhEDEx StorageConsistencyCheck utility for detection of orphaned data. The sites were also asked to note the procedure they used for generating storage dumps and to measure performance.

Storage technologies covered in these tests were: *Castor*[20],*dCache*[21], *DPM*[22], *EOS*[23], *Hadoop/BeStMan*[24], *Lustre*[25], *L-Store*[26], *StoRM/GPFS*[27], *xrootd*[28]. The campaign was very successful. Particularly, CMS sites were able to reuse ATLAS tools for producing storage dumps with little or no adaptation.

The second phase of this campaign is ongoing at the time of this writing, with more than half of the CMS Tier2 sites involved.

5.2. Aggregation

A storage dump of the contents of a site contains a lot of information, down to the level of individual files. This is necessary for checking the consistency between the site and the central catalog of the experiment, but for monitoring the space used by groups of users, this can be excessive. Typically, experiments group data at some level in the filesystem hierarchy that reflects the organisational unit that owns the data. Thus raw data might be under */store/raw*, Monte Carlo data might be under */store/mc*, and user data might be under */store/user/username*. Knowing how much data belongs to a given user, group or category is then as simple as summing up, or *aggregating* the amount of data under a given directory root.

There are two places where aggregation can happen. First, it can happen when the storage dump is uploaded for long-term storage in a monitoring database. Second, it can happen when the user retrieves the information from that database, if that user wishes to have a 'higher level' view than the default. The application must support both possibilities.

Clearly, the level of aggregation applied to the storage dump when it is uploaded represents a limit for future exploration of the data, since a user cannot later drill-down below that level of aggregation. CMS, after some investigation, has settled on a limit of 6 directory levels deep for aggregating on upload. This captures almost all the directory-level structure that corresponds to the hierarchical organisation of the experiment. Levels below this correspond only to details of file-organisation within datasets, and are not of interest in this context.

The initial aggregation step also allows experiment-specific or site-specific security policies to be implemented. A site may have privacy policies in place which limit the rights of people to see details about files owned by other users. In this case, they could aggregate the information under the individual users' directories to show only the total per user, instead of recording details about lower level files and directories.

Since this initial aggregation happens at the site, before the storage dump is uploaded, the site is free to treat different parts of the filesystem namespace according to whatever policies they have in place. They could go one step further and anonymize the users names if they wished, but this makes it hard to correlate usage between sites. Site administrators with direct access to the storage element are able to monitor their own users directly anyway, so restricting the contents of the aggregated storage dump does not cost them any functionality.

Details of site policies, and of any global policies we need, are still being worked out. Initially we shall allow access to individual users information only to people who are registered as site administrators of that site, to be conservative.

5.3. Data storage

The new storage accounting application is modelled on the existing PhEDEx data service [29]. The application consists of an API which can store and retrieve information using standard HTTP protocols. The space usage record in POSTed to the server with the node name and the time stamp of the corresponding storage dump and is then written to an Oracle database. This information is kept separately from the PhEDEx transfer management database for the reasons discussed below. The schema allows to store records with different level of details, depending on the initial aggregation level applied to the directory structure. If desired, further aggregation to some common level can be done at the retrieval time.

The prototype was developed as a PhEDEx application, to re-use the data service framework. Before it is released to production, it will be re-packaged as a standalone product, albeit sharing a common codebase with the PhEDEx data service. This is because, despite being very similar to the PhEDEx data service in many ways, there are some significant differences:

- semantically, the space monitoring information does not belong to PhEDEx. PhEDEx maintains historical information about transfers it has performed, but is not concerned with

space used at the sites. We do not want to burden the PhEDEx database with information that does not belong there, so the space monitoring information will be maintained in a separate database.

- similarly, while PhEDEx maintains several 'instances', for production transfers, debug transfers, and so on, there is no sense in maintaining separate instances for the space monitoring information. Files either take space or they don't, regardless of how they got there.
- long term, this information may well be integrated with other sorts of information, from other sources. This could be harder to achieve if it is constrained by being implemented as a part of PhEDEx.

Clients use a command-line tool to prepare the storage dump in a manner that conforms to their sites' policy and then upload the information.

Space monitoring data can be retrieved for a range of time and a set of nodes in a single call, to allow historical profiles to be examined easily. The data can be restricted to a given directory-depth or a starting sub-directory, to minimise the volume of information returned. So one could inspect the top-level subdirectories of all T2s over the last year in a single call, or get finer detail about a specific groups' space without having to pull in information about the entire set of files owned by CMS.

5.4. Visualization

The information from the data store is visualized in a way most suitable for the end user. For historical trends, stack-charts are useful. These show how the data below a given root in the filesystem hierarchy varies over time, and can be used to clearly identify groups that are consuming space at above the expected rate. By allowing the user to drill-down into the data, they can explore the exact origins and behaviour of any feature they wish to investigate.

For a snapshot, pie-charts or the increasingly popular treemap charts can be used. Examples are shown in figures 10 and 11.

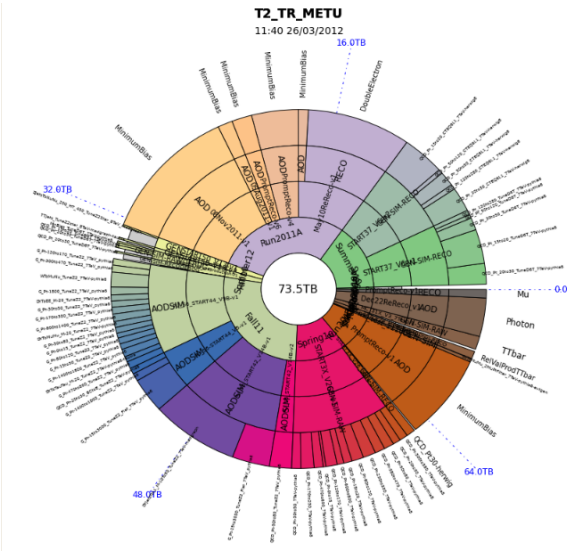

Figure 10. An example pie-chart showing storage usage at one of the CMS T2 sites. Four levels of directory structure are shown.

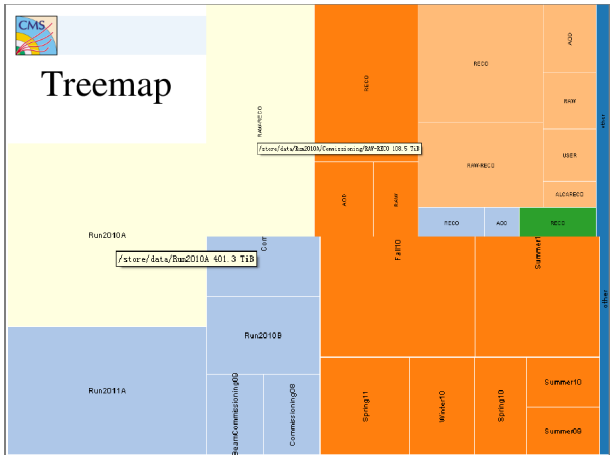

Figure 11. An example treemap. Space occupied by a rectangle is proportional to the space used on the storage element. Two levels of directory structure are shown, colour-coding denotes directories which share a common immediate ancestor.

Depending on visualization techniques already used in the experiment specific monitoring, a solution may be chosen for representing the storage usage results. Open source graphical libraries, such as protovis[30], D3[12] or highcharts[31] provide a rich set of data visualization features to allow the user to explore the data.

6. Summary and conclusions

Regular monitoring of the space usage and data verification help to improve the integrity and performance of the data handling system. Both the importance and the complexity of these tasks increase with the constant growth of the total data volumes during the active data taking period at the LHC.

In this article we have considered the requirements for the space monitoring and the most common use cases. We gave an overview of the current implementations for storage accounting by the ATLAS, CMS, and LHCb experiments. Particularly, we elaborated an approach to storage accounting and verification based on the use of storage dumps, which are agnostic to the experiments' data handling models. The use of storage dumps was extended from the data consistency checks to the storage accounting, for which a general four-step model of a space monitoring solution was presented. The main components of this model were analysed to identify the common grounds between the experiments. Furthermore we demonstrated the implementation of the proposed model using a CMS prototype of a space monitoring tool and described the ongoing deployment campaign at the CMS distributed sites as an example of fruitful collaboration between the experiments and the IT-ES group.

The experiments are constantly working on the improvement of the storage monitoring and accounting tools. Sharing the code and the expertise and adopting common solutions helps to reduce development efforts and maintenance costs at the sites.

References

[1] Knobloch J *et al.* 2005 "LHC Computing Grid Technical Design Report" CERN-LHCC-2005-024
[2] Millar P *et al.* 2010 "Dealing with orphans: Catalogue synchronisation with SynCat" *J. Phys.: Conf. Ser.* **219** 062060
[3] Serfon C 2010 "Data management tools and operational procedures in ATLAS: Example of the German cloud," *J. Phys. Conf. Ser.* **219**, 042053 (2010).

[4] N. Magini, N. Ratnikova, P. Rossman, A. Sanchez-Hernandez and T. Wildish, "Distributed data transfers in CMS," *J. Phys. Conf. Ser.* **331**, 042036 (2011).
[5] Garonne V, Molfetas A, Lassnig M, Barisits M, Stewart G A, Beermann T 2012 "The ATLAS Distributed Data Management project: Past and Future", *these proceedings*
[6] Shoshani A, Sim A and Gu J 2002 "Storage Resource Managers: Middleware Components for Grid Storage", Nineteenth IEEE Symposium on Mass Storage Systems'
[7] The CMS Collaboration 2008 "The CMS experiment at the CERN LHC" JINST **3** S08004
[8] Bonacorsi D 2007 "The CMS computing model" *Nucl. Phys.* B *(Proc. Suppl.)* **172** 53-56
[9] Giffels M and Guo Y 2012 "Data Bookkeeping Service 3 - A new event data catalog for CMS", *these proceedings*
[10] Egeland R, Wildish T and Metson S 2008 "Data transfer infrastructure for CMS data taking" *XII Advanced Computing and Analysis Techniques in Physics Research* (Erice, Italy: Proceedings of Science)
[11] Ball G 2011 PhD thesis, chapter 8 "Computing Monitoring Pages", https://workspace.imperial.ac.uk/highenergyphysics/Public/theses/Ball.pdf
[12] The D3 javascript library http://mbostock.github.com/d3/
[13] Stagni F *et al.* 2012: "LHCbDIRAC: distributed computing in LHCb", *these proceedings*
[14] Lopienski S 2008 "Service level status - a new real-time status display for IT services" *J. Phys.: Conf. Ser.* **119** 052025
[15] Baud J-Ph et al 2005 Performance analysis of a file catalog for the LHC computing grid HPDC 14 91-99
[16] Charpentier P *et al.* 2012 "The LHCb Data Management System", *these proceedings*
[17] Lanciotti E 2011 "Storage elements dumps and consistency checks versus file catalogues" https://twiki.cern.ch/twiki/bin/view/LCG/ConsistencyChecksSEsDumps
[18] Bauerdick L and Sciaba A 2012 "Towards a global monitoring system for CMS computing", *these proceedings*
[19] Sanchez-Hernandez A, Egeland R, Huang C-H, Ratnikova N, Magini N and Wildish T, 2012 "From toolkit to framework - the past and future evolution of PhEDEx", *these proceedings*
[20] Castor, http://castor.web.cern.ch/castor/
[21] dCache, http://www.dcache.org/
[22] DPM (Disk Pool Manager) https://twiki.cern.ch/twiki/bin/view/LCG/DataManagementDocumentation#DPM
[23] Peters A-J 2011 "The EOS disk storage system at CERN", *ACAT conference proceedings*
[24] BeStMan (Berkeley Storage Manager) https://sdm.lbl.gov/bestman/
[25] Lustre http://www.lustre.org/
[26] L-Store (Logistical Storage) http://www.accre.vanderbilt.edu/mission/services/lstore.php
[27] Cavalli A *et al* 2010 "StoRM-GPFS-TSM: A new approach to hierarchical storage management for the LHC experiments" *J. Phys.: Conf. Ser.* **219** 072030
[28] XRootD, http://xrootd.slac.stanford.edu/
[29] R. Egeland, T. Wildish and C. -H. Huang, "PhEDEx data service" *J. Phys. Conf. Ser.* **219** (2010) 062010.
[30] The Protovis javascript library http://mbostock.github.com/protovis/
[31] The Highcharts javascript library http://www.highcharts.com/