Ensuring Data Consistency Over CMS Distributed Computing System

The 3 Stages of Maintaining CMS Data Consistency

CMS utilizes a distributed infrastructure of computing centers to provide access to data stored on disk only at Tier-2 centers and tape with disk caches at Tier-1 centers. Attached are CPU resources for organized processing and analysis. Data is organized in datasets which consist of files grouped in blocks for performance reasons. CMS uses it's data transfer system PhEDEx, to transfer datasets from site to site and its data bookkeeping service DBS to track location and metadata. Integrated over the whole system, even in the first year of data taking, the available disk storage approaches 10 petabytes of space. Maintaining consistency between the data bookkeeping service, the data transfer system, and physical storage is an important operational task which guarantees uninterrupted data availability.

1. During data transfer

All CMS sites are required to use two **PhEDEx** components to verify data consistency.

Upon data arrival, **FileDownloadVerify** is used to verify the file by comparing the on disk file size with the cataloged file size. If FileDownloadVerify discovers a variation in file size the file is removed and retransferred. CMS is currently investigating greater adoption of checksum based file verification. Additional details are provided below.

The second component, **BlockDownloadVerify**, accepts requests from a central PhEDEx agent at CERN that looks for blocks which have been stuck in transit for 3 days or more. This agent can also be invoked on demand to allow for verification of any block on any site at any time. BlockDownloadVerify manages access load to the storage elements of the sites to prevent overloading the systems.

For CMS Tier-1 sites a third component is required. **FileMSSMigrate** verifies that data has migrated properly to the Tier-1's mass storage tape system. If problems arise during migration, additional steps will need to be taken for resolution.

2. During analysis

CMS uses **ProdAgent**, a central production and processing infrastructure and **CRAB**, a user tool providing distributed access to datasets for analysis. CMS is currently expanding the functionality of ProdAgent and CRAB to report data inconsistencies to the **CMS Dashboard**, the central monitoring system of CMS.

3. Periodically on storage

All CMS sites routinely perform consistency checks on data. Several tools are available to site administrators for this purpose. Three utilities are distributed with PhEDEx.

BlockConsistencyCheck uses the central catalogs of the transfer system and bookkeeping system to find files missing at a site. It compares the catalog information with the local storage namespace. Missing files and file size mismatches at the site are reported. Inconsistencies identified by this tool are provided to central CMS administrators via a ticketing system for resolution.

StorageConsistencyCheck uses the local storage namespace of the site to find orphan files which are not registered in the central catalogs. The output from this tool is a list of files known to the local storage namespace but not the catalogs. Resolution of inconsistencies are again reported via a ticketing system to central CMS administrators.

BlockDownloadVerify-injector allows operators and shifters to perform consistency checks on the block level of datasets without direct access to the local mass storage. In contrast to the above mentioned site administrator possibilities, the injector uses the BlockDownloadVerify agent to verify the consistency of a specific block of a datasets. This is often used by non-local operators and shifters to invoke quick evaluations at remote sites if problems have been reported or noticed in the central monitoring.

Future Direction

The existing consistency tools are continuously improved to add more functionality. The goal is to provide an open namespace framework that generalizes the consistency tools and simplifies their usage.

Checksum vs. File Size

There have been isolated instances where data files have been transferred passing the file size check yet having invalid checksums compared to the catalog. This could have a negative avalanche effect where a file with an invalid checksum is transferred successfully to other sites therefore multiplying the problem. If noticed, a situation like this would create unnecessary work for site administrators who need to remove the corrupt data files and request retransfer of the data.

At present most sites verify the data file after transfer with PhEDEx using only the file's size and its stored value in DBS. For Tier-2s and Tier-3s, if the sizes match, PhEDEx marks the transfer as successful and the file becomes available for analysis at the destination site. For Tier-1 sites, PhEDEx uses an additional step (FileMSSMigrate) to verify that the transferred file was written to tape.

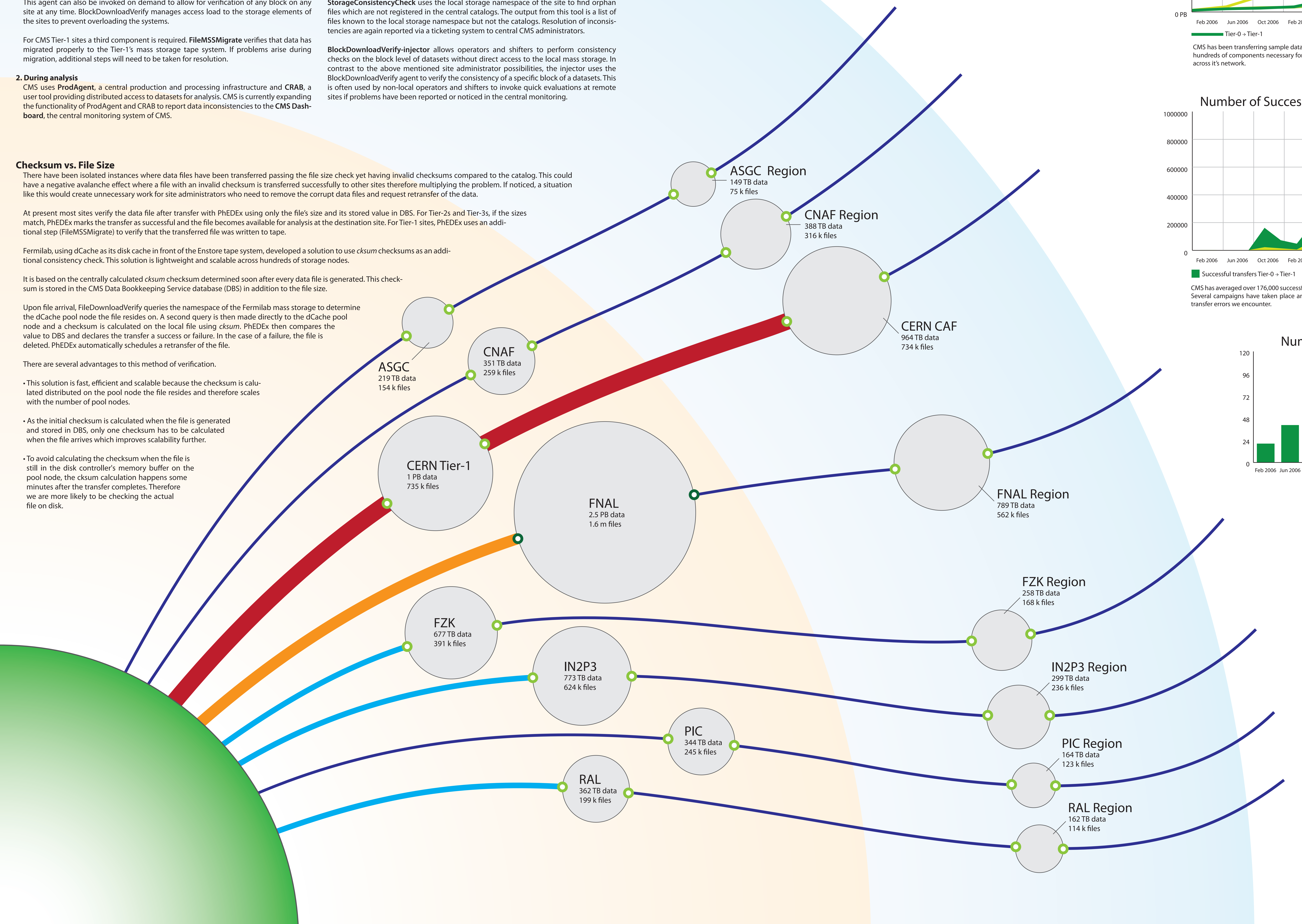
Fermilab, using dCache as its disk cache in front of the Enstore tape system, developed a solution to use *cksum* checksums as an additional consistency check. This solution is lightweight and scalable across hundreds of storage nodes.

It is based on the centrally calculated *cksum* checksum determined soon after every data file is generated. This checksum is stored in the CMS Data Bookkeeping Service database (DBS) in addition to the file size.

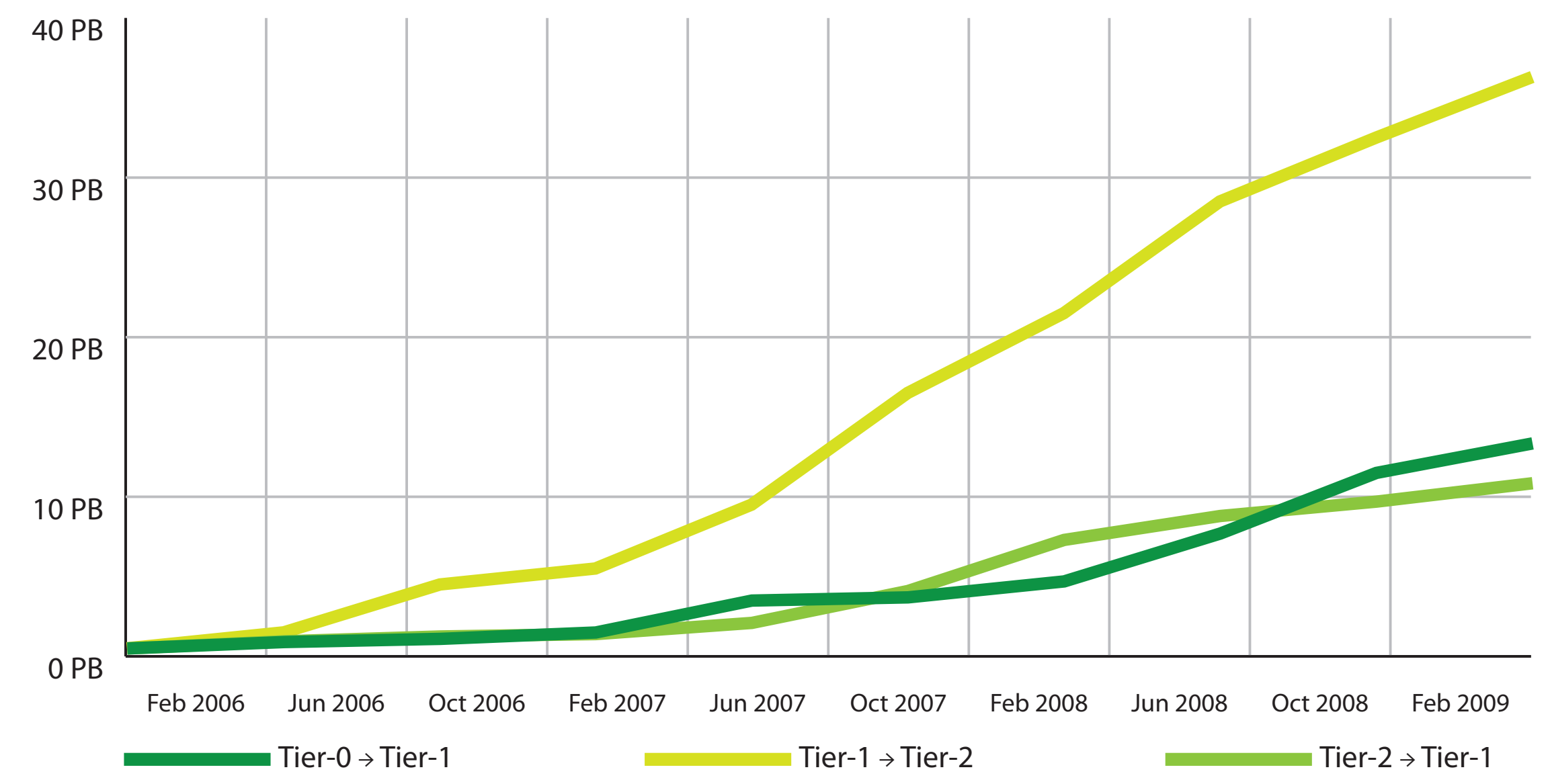
Upon file arrival, FileDownloadVerify queries the namespace of the Fermilab mass storage to determine the dCache pool node the file resides on. A second query is then made directly to the dCache pool node and a checksum is calculated on the local file using *cksum*. PhEDEx then compares the value to DBS and declares the transfer a success or failure. In the case of a failure, the file is deleted. PhEDEx automatically schedules a retransfer of the file.

There are several advantages to this method of verification.

- This solution is fast, efficient and scalable because the checksum is calculated distributed on the pool node the file resides and therefore scales with the number of pool nodes.
- As the initial checksum is calculated when the file is generated and stored in DBS, only one checksum has to be calculated when the file arrives which improves scalability further.
- To avoid calculating the checksum when the file is still in the disk controller's memory buffer on the pool node, the cksum calculation happens some minutes after the transfer completes. Therefore we are more likely to be checking the actual file on disk.

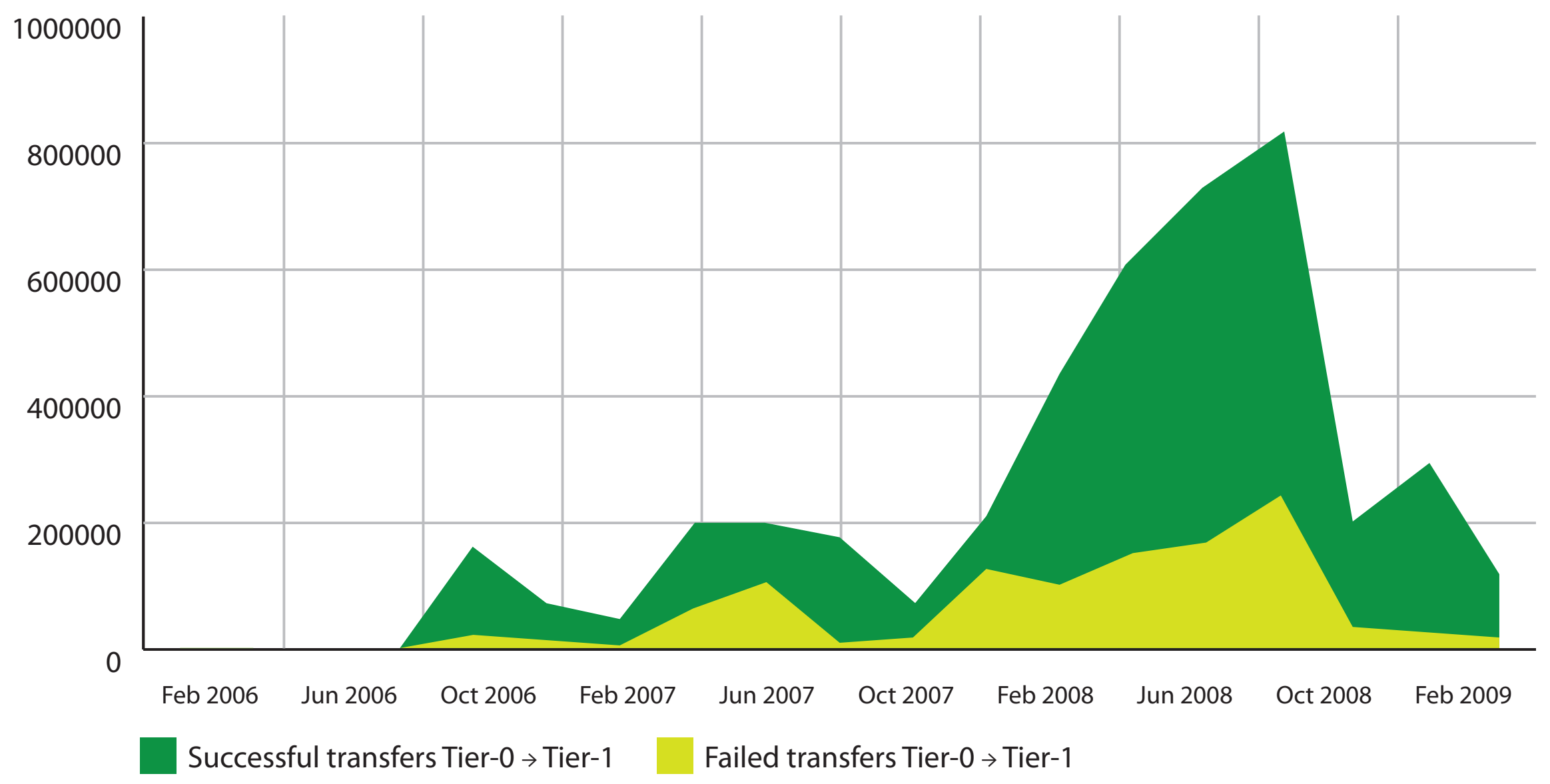


Cumulative Transfer Volume Since 2006



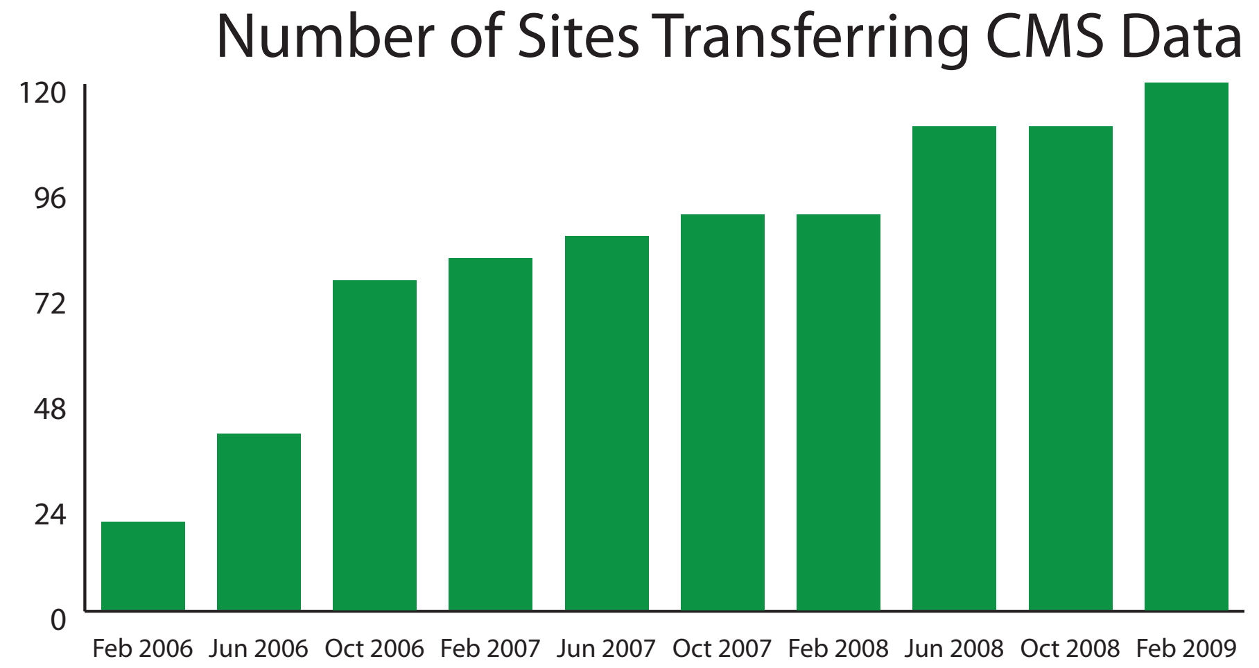
CMS has been transferring sample data for several years. We use this sample data to exercise and validate the hundreds of components necessary for successful data transfers. Since 2006, CMS has transferred over 60 PB across it's network.

Number of Successful and Failed Transfers Since 2006



CMS has averaged over 176,000 successful data transfers per month since 2006 from the Tier-0 to Tier-1 centers. Several campaigns have taken place and are being planned to develop solutions intended to minimize the transfer errors we encounter.

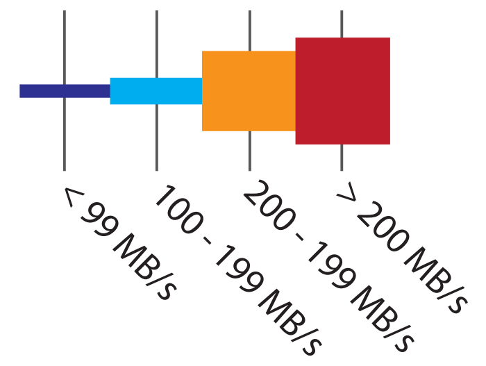
Number of Sites Transferring CMS Data



About this Poster

The CMS data model is built using computing resources at a range of scales, provided by collaborating institutes around the world. CMS uses a hierarchical architecture of tiered centers, with a single Tier-0 center at CERN, a few Tier-1 centers at national computing facilities and several Tier-2 centers at institutes and universities.

The data paths connecting the Tier-0 to Tier-1 and Tier-2 are size and color coded to the CMS nominal transfer rate.



Additional links exist between most sites, however those are not illustrated.

The Tier-1 and Tier-2 are scaled to size according to the current amount of data registered on disk in the CMS data transfer application PhEDEx. Except for the CERN CAF, Tier-2 regions consist of multiple sites.

Connecting each data path and site or region is an indicator of the technique used to validate all PhEDEx data transfers.

○ File Size based file verification

● Checksum based file verification

Paul Rossman, Fermilab
On Behalf of the CMS Offline and Computing Projects