A Distributed Disk Layer for Mass Storage at DESY

Susanne Brand [a]

[a] *DESY Hamburg, Notkestr. 85, 22607 Hamburg*

For the mass storage system at DESY a disk layer is under development. Decoupling the client request queue and access to the mass storage system by means of migration, staging and prefetching it shall provide full utilization of robot and drive resources. Managing distributed disk resources in the heterogeneous computing environment of DESY optimized data access shall be given.

Key words: mass storage; disk layer; client-server architecture;

1 Introduction

At DESY 34 TBytes of Data are currently kept online in several mass storage systems. With the beginning of the year a new robot system with a capacity of 120 TByte has been introduced. Four large experiments are running at HERA producing raw data, reconstruction data, monte carlo data and user data in a heterogeneous computing environment. The current tape access (see fig. 1) at DESY runs under the Open Storage Manager (OSM) [1], which provides access to bitfiles via a flat namespace represented by bitfile ids. The Perfectly Normal File System (PNFS) [2] couples this via a multi-layer network file system to a hierarchical namespace, allowing normal namespace operations but no I/O due to the poor data transfer rates of NFS. Each layer of the PNFS allows a different view to a bitfile. One layer reflects the normal file attributes like size and permissions; a second layer, which is used by the application interface, preserves all information to retrieve a file via the OSM-API from the mass storage system. The coupling of PNFS directories to stores and storage groups, i.e. a storage hierarchy due to the mass storage reference model [3], is done by so called tags, which are assigned to a directory and inherited by subdirectories. Due to the highly heterogeneous and distributed computing environment at DESY there are about 60 OSM client installations on various platforms (Solaris, IRIX, HP-UX, Linux).

In this environment the further development of mass storage includes several aspects. Overall one wants to achieve a maximum utilization for the total

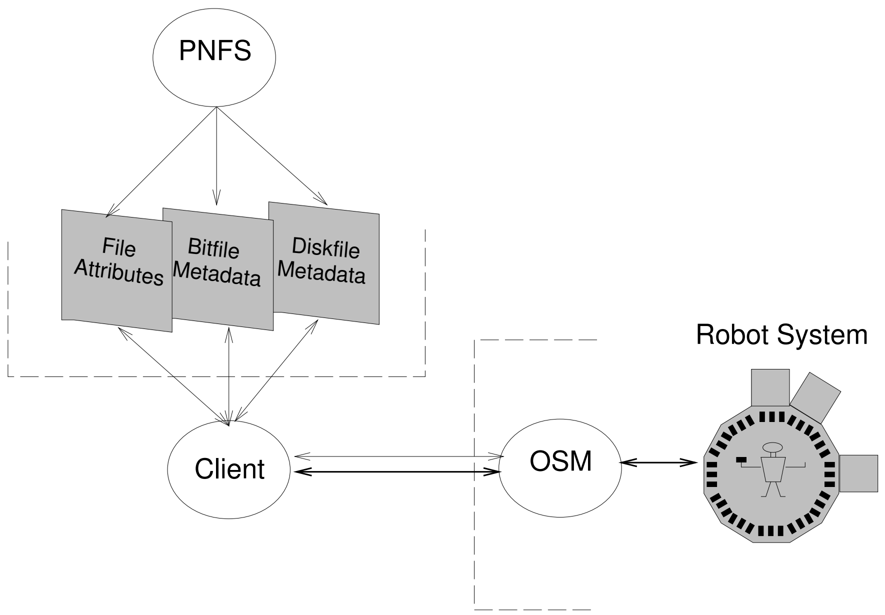

Fig. 1. Overview over current Mass Storage at DESY: bold lines indicate data transfer either via IP or locally, thin lines indicate control information transfer

system, including drives and robot system. All accesses should be optimized in speed and resource consuming, this is not only valid for mass storage itself but also for secondary resources such as network. Moreover the access should be as transparent as possible to the user.

To achieve these goals a disk layer is best applicative. In its basic functionality it provides decoupling of client requests and tape access. Without a disk layer the number of data streams is limited to the number of drives available in the mass storage system; with a disk layer this number is increasened proportional to the number of disk cache hits achieved by the layer (see fig. 2). Moreover a considerable amount of the overall data transfer time usually is the mount time; avoiding additional mounts by means of prefetching and prestaging will not only shorten access times from the users point of view but also optimize the overall data transfer rates. So even simple prefechting and caching mechanisms provided by a disk layer will result in an improved utilization, which otherwise could only be achieved by increasing system resources.

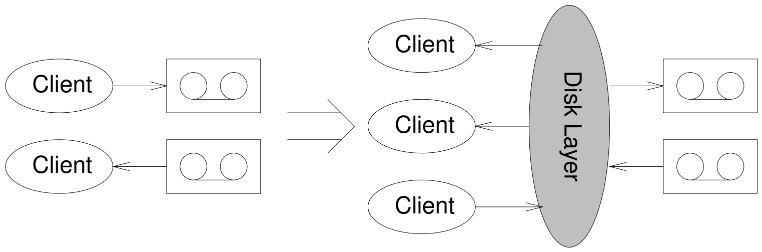

Fig. 2. Improvement of mass storage system utilization by a disk layer

2 Implementation

To get profit as fast as possible the disk layer will in a first implementation (see Fig. 3) run in a central server mode and is intended to manage local disk resources only. The PNFS serves as a transparent data repository not only to the disk layer components but also to the clients. Its third layer maintains information for each bitfile which is kept on disk. Besides the metadata for single files it keeps tags, which are attached to directories and used to configure migration and prefetching characteristics. Each request runs through the central server, which sets up the data connection to the client and provides staging and fetching of bitfiles. Migration and prefetching will be held simple; but already a simple prefetching of the next bitfile on tape will assure a high percentage of disk cache hits for certain data sets, which like raw data are generated as well as reprocessed in a serial manner.

In a later upgrade (see Fig. 4) the disk layer is planned to support the use of distributed disk resources. Then it will consist of three components mainly: a single master server providing sophisticated features, such as more elaborated migration and prefetching algorithms and the overall control of remote disks; a disk manager as a client server, running on each machine, which offers local disk resources to the disk layer; and again the PNFS.

The disk manager will now be the first instance a client is supposed to claim to. Only requests, which cannot be managed locally by querying the PNFS and local disk access, go to the master server. Each disk can be marked as secure or insecure, i.e. it may serve migration and staging or staging only. In this way one can make migration save, by choosing adequate disk resources such as a raid array for writing purposes.

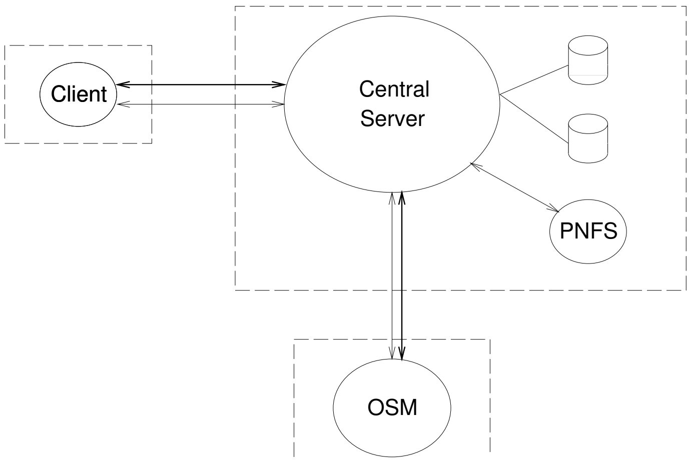

Fig. 3. Disk Layer Architecture in its first approach

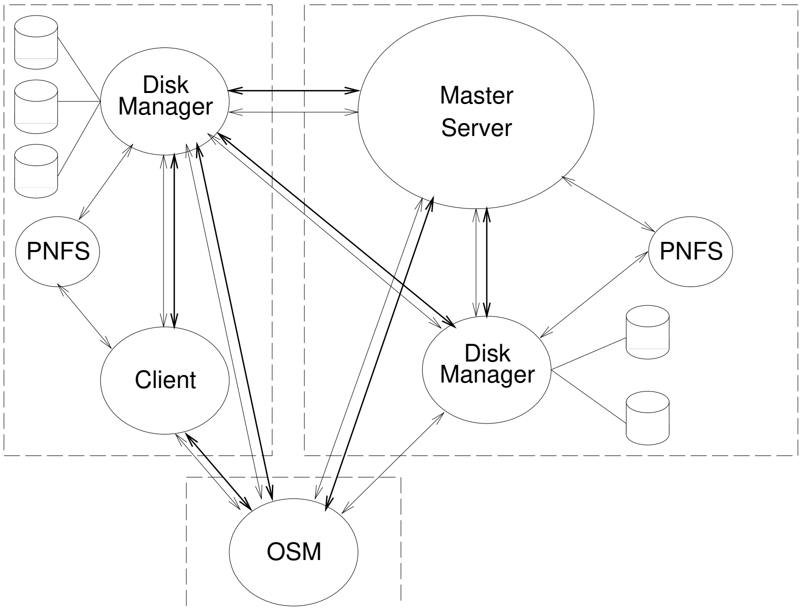

Fig. 4. Disk Layer Architecture in its final state

The master server only evaluates and handles all requests, which cannot be directly satisfied by one of the disk client servers. This includes staging and prefetch requests. It runs migration and prestaging cyclic or triggered by the client servers. He keeps overall control over disk resources, distributes files among available disks and establishes third party transfer between the client servers.

The final scenario is that each host offering free disk resources can be attached to the disk layer via a local disk manager. The disks will preferably and continuously be filled up with locally requested files, thus reducing needs of mass storage as well as network resources.

References

[1] Lachman Technology, Inc. : The Open Storage Manager.

[2] Patrick Fuhrmann: A Perfectly Normal Namespace for the DESY Open Storage Manager. CHEP 97

[3] IEEE P1244. Reference Model for Open Storage Systems Interconnection - Mass Storage System Reference Model Version 5, September 1994.