THE GENERAL PARALLEL FILE SYSTEM

The General Parallel File System

We have decided at this point to introduce a product of our employer, IBM, for once. The General Parallel File System (GPFS) is a shared disk file system that has for many years been used on cluster computers of type RS/6000 SP (currently IBM eServer Cluster 1600). We believe that this section on GPFS illustrates the requirements of a shared disk file system very nicely. The reason for introducing GPFS at this point is quite simply that it is the shared disk file system that we know best. The RS/6000 SP is a cluster computer. It was, for example, used for Deep Blue, the computer that beat the chess champion Gary Kasparov. An RS/6000 SP consists of up to 512 conventional AIX computers that can also be connected together via a so-called high performance switch (HPS). The individual computers of an RS/6000 SP are also called nodes.

Originally GPFS is based upon so-called Virtual Shared Disks (VSDs) (Figure 4.12). The VSD subsystem makes hard disks that are physically connected to a computer visible to other nodes of the SP. This means that several nodes can access the same physical hard disk. The VSD subsystem ensures that there is consistency at block level, which means that a block is either written completely or not written at all. From today's perspective we could say that VSDs emulate the function of a storage network. In more recent versions of GPFS the VSD layer can be replaced by an Serial Storage Architecture (SSA) SAN or a Fibre Channel SAN.

GPFS uses the VSDs to ensure the consistency of the file system, i.e. to ensure that the metadata structure of the file system is maintained. For example, no file names are allocated twice. Furthermore, GPFS realises some RAID functions such as the striping and mirroring of data and metadata.

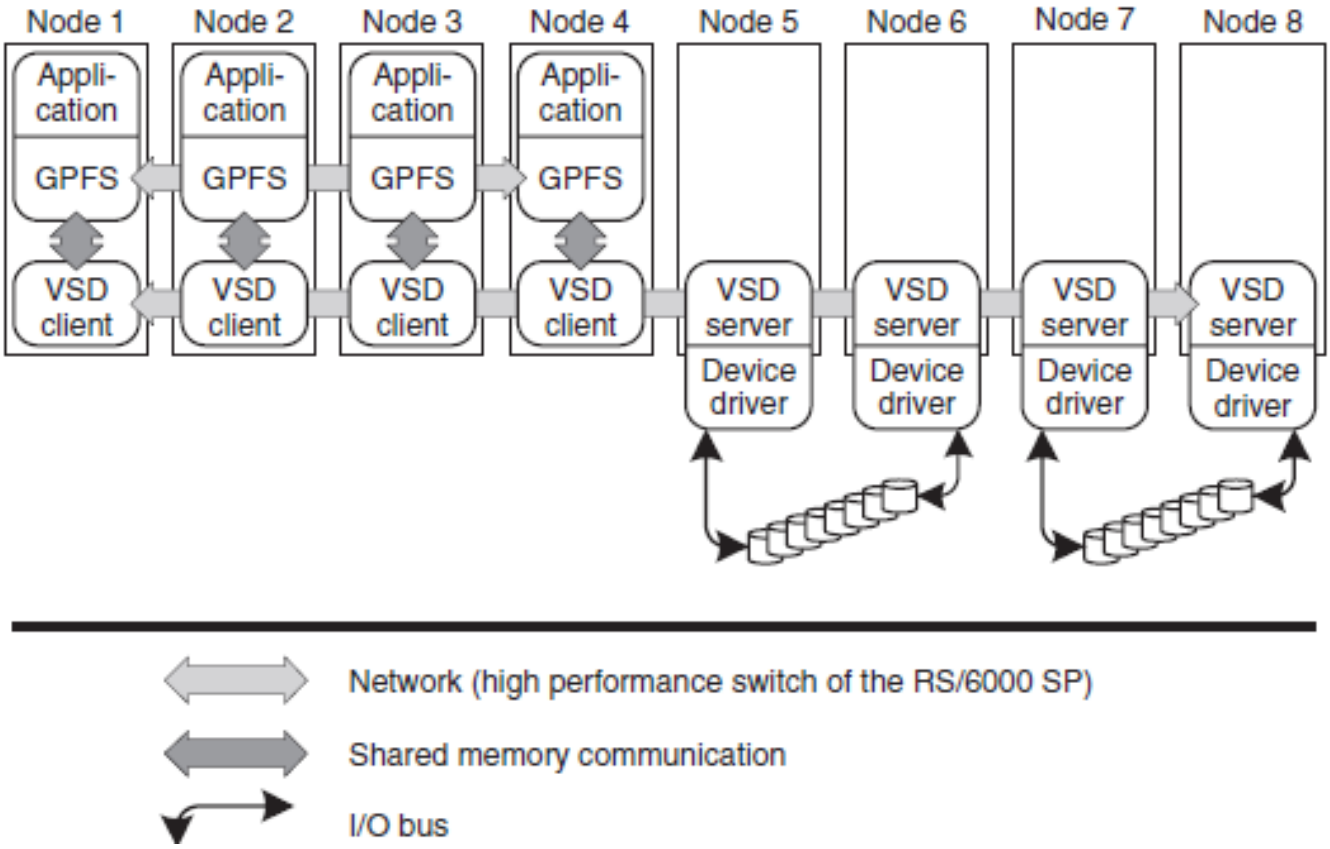

Figure 5.12 Applications see the GPFS file system like a local file system. The GPFS file system itself is a distributed application that synchronises parallel accesses. The VSD subsystem permits access to hard disks regardless of where they are physically connected.

Figure 4.12 illustrates two benefits of shared disk file systems. First, they can use RAID 0 to stripe the data over several hard disks, host bus adapters and even disk subsystems, which means that shared disk file systems can achieve a very high throughput All applications that have at least a partially sequential access pattern profit from this.

Second, the location of the application becomes independent of the location of the data. In Figure 4.12 the system administrator can start applications on the four GPFS nodes that have the most resources (CPU, main memory, buses) available at the time. A so-called workload manager can move applications from one node to the other depending upon load. In conventional file systems this is not possible. Instead, applications have to run on the nodes on which the file system is mounted since access via a network file system such as NFS or CIFS is generally too slow.

The unusual thing about GPFS is that there is no individual file server. Each node in the GPFS cluster can mount a GPFS file system. For end users and applications the GPFS file system behaves – apart from its significantly better performance – like a conventional
local file system.

GPFS introduces the so-called node set as an additional management unit. Several node sets can exist within a GPFS cluster, with a single node only ever being able to belong to a maximum of one node set (Figure 4.13). GPFS file systems are only ever visible within a node set. Several GPFS file systems can be active in every node set.

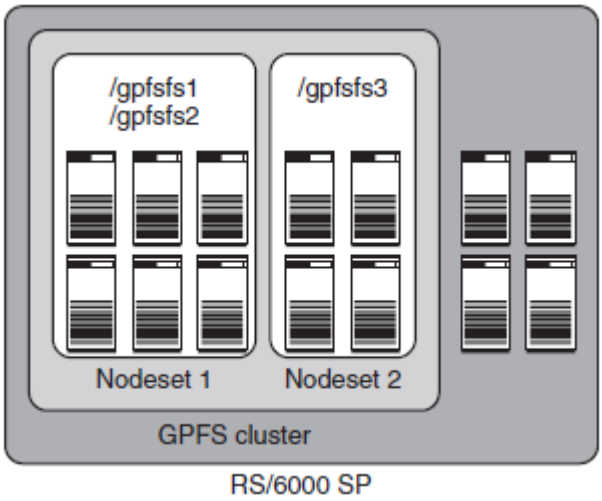

Figure 5.13 GPFS introduces the node sets as an additional management unit. GPFS file systems are visible to all nodes of a node set.

The GPFS Daemon must run on every node in the GPFS cluster. GPFS is realised as distributed application, with all nodes in a GPFS cluster having the same rights and duties. In addition, depending upon the configuration of the GPFS cluster, the GPFS Daemon must take on further administrative functions over and above the normal tasks of a file system.

In the terminology of GPFS the GPFS Daemon can assume the following roles:

• *Configuration Manager*

In every node set one GPFS Daemon takes on the role of the Configuration Manager. The Configuration Manager determines the File System Manager for every file system and monitors the so-called quorum. The quorum is a common procedure in distributed systems that maintains the consistency of the distributed application in the event of a network split. For GPFS more than half of the nodes of a node set must be active. If the quorum is lost in a node set, the GPFS file system is automatically deactivated (unmount) on all nodes of the node set.

• *File System Manager*

Every file system has its own File System Manager. Its tasks include the following:

– Configuration changes of the file system

– Management of the blocks of the hard disk

- Token administration

– Management and monitoring of the quota and

- Security services

Token administration is particularly worth highlighting. One of the design objectives of GPFS is the support of parallel applications that read and modify common files from different nodes. Like every file system, GPFS buffers files or file fragments in order to increase performance. GPFS uses a token mechanism in order to synchronise the cache entries on various computers in the event of parallel write and read accesses (Figure 4.14). However, this synchronisation only ensures that GPFS behaves precisely in the same way as a local file system that can only be mounted on one computer. This means that in GPFS – as in every file system – parallel applications still have to synchronise the accesses to common files, for example, by means of locks.

• *Metadata Manager*

Finally, one GPFS Daemon takes on the role of the Metadata Manager for every open file. GPFS guarantees the consistency of the metadata of a file because only the Metadata Manager may change a file's metadata. Generally, the GPFS Daemon of the node on which the file has been open for the longest is the Metadata Manager for the file. The assignment of the Metadata Manager of a file to a node can change in relation to the access behaviour of the applications.

The example of GPFS shows that a shared disk file system has to achieve a great deal more than a conventional local file system, which is only managed on one computer. GPFS has been used successfully on the RS/6000 SP for several years. The complexity of shared disk file systems is illustrated by the fact that IBM only gradually transferred the GPFS file system to other operating systems such as Linux, which is strategically supported by IBM, and to new I/O technologies such as Fibre Channel and iSCSI.

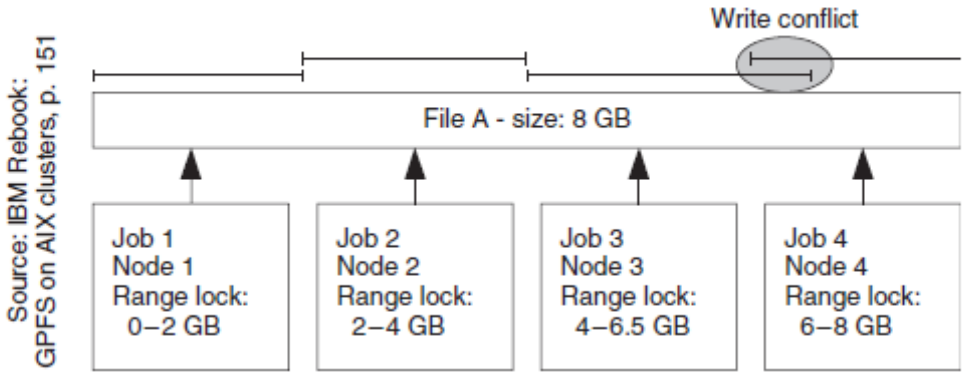

Figure 5.14 GPFS synchronises write accesses for file areas. If several nodes request the token for the same area, GPFS knows that it has to synchronise cache entries.

Source : http://elearningatria.files.wordpress.com/2013/10/cse-viii-storage-area-networks-06cs833-notes.pdf