

An Architectural Solution for Storage Management Problem

Helen Wu

FS Consulting, 9417 East Lemon Ave., Temple City, CA 91780

Swhelen21@hotmail.com

Abstract

An IDC analyst, Graham Penn [Rossi06], has recently pointed out that storage use has increased exponentially and many organizations go for a quick fix, i.e., buy more hardware. The concern is how to keep up with storage expansion through efficient data management. A technology writer, Sandra Rossi [Rossi06], gives her analysis: “Bad data the source of costly storage management problems.” In fact, the storage management problems have been pointed out as early as 2003 by a storage networking industry association, SNIA, presentation. It articulated the problems in the context of interpretability. In this presentation, relations among three trends, decreasing storage technology cost, increasing data volume and value, increasing storage management cost, were portrayed exhibiting a management gap. This paper attempts to analyze the problem and to propose an architecture solution. The paper follows an evolution of computer, the machine, and its development of I/O and the storage systems, and looks at the perpetual forces, the very reasons that have driven the evolution and the development. Through clarifying the storage management problem, an architecture solution emerges. The solution proposed here is similar to network attached secure disk, NASD [Gibson 96]. While NASD is motivated by off loading more of the file system's simple, expensive, and performance critical operations to the storage devices, this proposed architecture is motivated by minimizing storage management cost and to promote data storage efficiency, data sharing, and data protection. Learning from computer development history and making the analogy with library systems, current and historical development, the paper gives detailed rationales and outlines the design of the architecture. Ganek summarized the meaning of architecture in three aspects: 1) a direction-setting concept, 2) a set of products following that concept, and 3) a set of rules to which products must conform so that they may interoperate or present a consistent interface to the product user, or do both. [Ganek99] The intention of the proposed architecture in this paper is just that. Hope for the best, i.e., it will provide useful guidance in the future storage systems development.

Key words: Computer Systems Architecture, Storage Devices Technology, Cost and Manageability, DAS, NAS, SAN, NASD, OSD, SSD, LSD...

1. Defining the Problem

Figure 1 illustrates the storage management problem structurally. It shows the relationships between computer industry and end users. Compared to the components such as CPU, memory, network, and the storage, the manageability of the storage is the weakest spot in supporting end users. Figure 2¹ depicts the storage management problem analytically. It portrays a management gap resulting from decreasing storage technology cost, increasing data volume and value, increasing storage management cost.

Although computer storage vendors have taken the problem seriously and proposed various solutions, this paper attempts to analyze the problem from two orthogonal views in its own way. One is vertically, in learning from the computer systems development history. The other is horizontally, in absorbing current storage system technologies and inspecting the problem. The analysis ultimately leads to an architectural proposal at system level, which may be used as a problem solving approach to tackling this storage

¹ Both Figures, 1 and 2, are inspired by and adapted from a SNIA presentation given at MSST03.

management problem.

2. I/O is important!

Input/output (I/O) has been historically neglected by CPU enthusiasts [Hennessy 96] for computer systems performance is measured as throughput, more tasks per hour, rather than as response time. Michael Flynn [Flynn 95], while claiming, "I/O is important!" explains that the access time and transport rate of I/O devices frequently limits overall system performance. The execution of various input/output functions is quite time-consuming and usually involves a great deal of processor capability in managing various I/O processes. As a matter of the fact, the evolution of I/O systems organization, which was staged in milestones from programmed I/O, to Interrupt-driven I/O, to DMA-managed I/O, to system with I/O processors, to I/O system supporting multiprocessor, is all about making use of the processor capability efficiently and dealing with I/Os in order to improve the overall system performance.

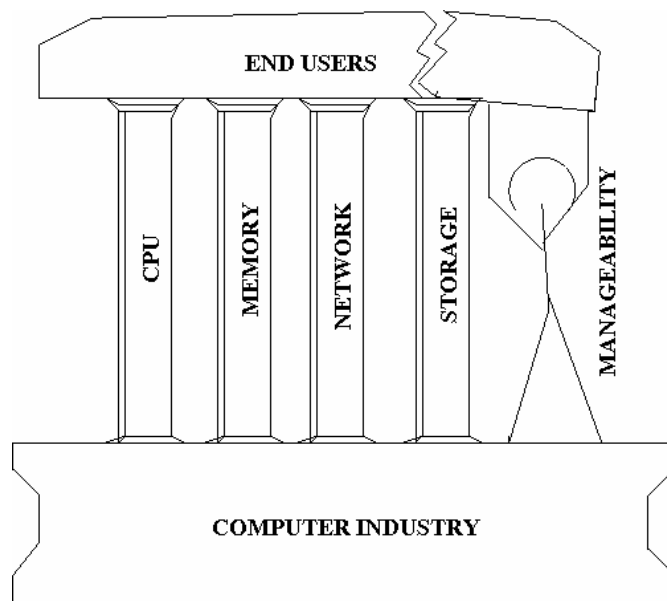


Figure 1: Storage Management Problem –an architectural view

Blaauw and Brooks [Blaauw 97] credited Kilburn and his colleagues for treating the drum and memory together (1962), and all other I/O separately. "The result was far cleaner and faster drum use as part of the one-level store." Furthermore, the access, data rate, and reliability requirements of storage devices are typically far greater than those of source/sink devices, so different storage technologies and control techniques are used as appropriate.

In Flynn's book [Flynn 95] he describes the model of I/O subsystem as one or more control/storage devices that interface between the processor memory bus and the I/O peripheral units, with which the analysis and design of storage subsystems are discussed. Given that data is distributed over many physical devices, the problem in accessing data is how to design a system. The system should then be efficient in

- 1) Finding the physical location of the required data;
- 2) Finding a path from the processor memory to the particular physical device that contains the information, and
- 3) Accomplishing 1) and 2) without significantly interfering with the central processor's execution of user programs. In this capacity, Flynn outlines the I/O systems storage management.

3. Turning Points in Systems Architecture

Having briefly introduced the storage system in the context of the computer architecture, let us look at the major systems architecture milestones, starting in the 1960s and spanning four decades, as chronicled in the IBM Systems Journal. [Ganek99]

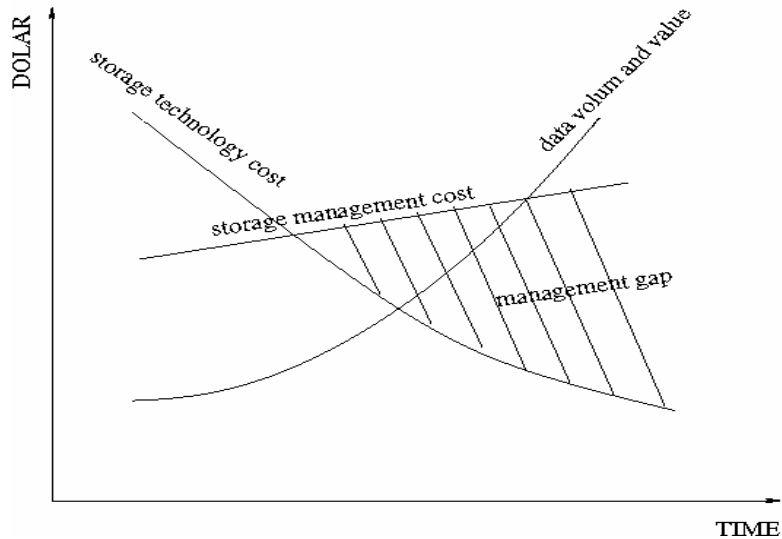


Figure 2: Storage Management Problem—an analytical view

3.1. System/360

The system is so designed that "The overall structure lends itself to programming-compatible embodiments over a wide range of performance levels." Seemingly simple, with the upwardly compatible, the principle is indeed the key to the computing system development success within IBM as well as to other vendors.

3.2. AS/400

This is an application-solution-oriented system designed "to offer customers (i.e., users) business solutions for their data processing requirements." For example, it conserves existing application interfaces, integrates in the system of data description and manipulation facilities to a relational database, and consists of interfaces to all forms of work management including interactive, batch, office, and transaction processing. [Schleicher 89]

The system's layered architecture begins with a processor hardware core, a high-level machine interface implemented via microcode, operating system function, communications support, application enabler that integrated a relational database and end-user interfaces. The machine interface and layered structure allowed the system to make a transition from the original complex instruction set computer, CISC, architecture implementation to the PowerPC RISC architecture without requiring recompilation of applications. Unlike its predecessors with virtual storage organization, it is a true single level-store machine, that is, programmers do not know where their data reside, and references to files in the program implicitly cause a combination of hardware and software to make all of a programmer's data appear to be immediately accessible. [Ganek99]

3.3. RISC processors

A reduced instruction set computer, RISC, processor utilizes only simple instructions and optimizes hardware for performance, allowing low-level parallelism, i.e., more than one instruction can execute per cycle. More complex function is implemented via software, but because compilers generate it, no complexity changes for programmers. As Hopkins stated in his introductory paper [Hopkins 87] A perspective on the 801 /Reduced Instruction Set Computer:

In 1947, John Mauchly wrote about EDVAC, " a decision must be made as to which operations shall be built in and which are to be coded into instructions... Ultimate choice must depend upon the analysis by the designer of the character of the work to be performed by the machine, the frequency of the occurrence of operations, and the ease with which the non-built-in operations can be compounded from those which are built in." The 801 emphasis on simple instructions is just a restatement of this old wisdom.

3.4. Distributed processing

Both Microprocessor technology and networking have boosted the distributed processing carried out on distributed systems. Hardware of a distributed system is characterized with many interconnected CPUs whereas software is to allow many users to work together in applications inherently distributed. Tanenbaum [Tanenbaum95] differentiated such system from those that are parallel, in which multiple CPUs are dedicated to a single task, the research subject of the supercomputer.

Recognizing the unprecedented level of flexibility in designing applications in 70s, A.L. Scherr, an IBM researcher, listed motivation factors of distributed processing, [Scherr 78] as follows:

- 1) Reliability and availability, i.e., limiting the impact of failures through partitioning the workload among several processors and reducing the impact through providing redundancy.
- 2) Organizational needs, the vision of current applications such as enterprise resource processing, ERP, customer resource management, CRM, online transaction processing, OLTP, online analysis processing, OLAP, ...
- 3) Lower communication cost via networking and better price-performance ratio via simpler, smaller, inexpensive machines, or efficient use of powerful machines.

Technically, he categorized distributed processing into two configurations for two different types of applications. One is called back-end processor. An example can be a storage server in network attached storage, NAS or storage area network, SAN, which provide services for multiple applications accessible to multiple users. The other is called front-end processor. A client/server mode application with thin clients such as transaction-based applications, serves multiple users scattered globally.

These milestone events fundamentally reflect the thinking behind the proposed architectural solution to the storage management problem. Before laying out the solution, let us survey the needs of storage management for both applications and system administrations and examine the problem in detail.

4. What are storage management tasks?

IBM Tivoli Storage Manager, a client/server product providing backup, archive, and space management functions in heterogeneous distributed environments, performs extensive storage management after client data have reached the server. Kaczmarek et al in their paper [Kaczmarek 03] describing design points and functions of the software not only list the management tasks, but also raise the special concerns that the software addresses in storage management, namely,

- 1) The need for efficient, non-intrusive backup and proper recovery of data especially with enterprise configurations for applications such as DB2, Lotus Domino, Microsoft Exchange, and SAP R/3,

- 2) Data migration due to storage device migration,
- 3) Optimizing the restore operation and content management.

Simitci in his Storage Network Performance Analysis [Simitci 03] extends storage management responsibilities of storage systems (NAS and SAN) to keeping up with the storage network performance management requirements. That is, in addition to discovery, provisioning, monitoring, and automation, reporting of storage resource such as capacity and performance is also included. As he pointed out, problem detection and diagnosis, automated policy enforcement, performance and trend reporting, and capacity planning need policies and threshold values to be set. Policy should include application type, storage type, performance and availability requirements, cost, and security constraints. Both policies and threshold values are rather experimental and circumstantial, and they are difficult to select appropriately. Existing tools are specialized and require manual control and intervention.

Baker and Massiglia in their book, Storage Area Network Essentials, [Baker 02] frankly pointed out many challenges facing SAN after giving the recognition that SAN may be superior to both direct attached storage, DAS and network attached storage, NAS. What deserves to be restated here in their unique terminology is as follows.

- 1) Monitoring and analysis, i.e., managing logical connections is difficult for there are too many different devices connected in a SAN that are beyond human manual capacity. No one server has the entire picture and several servers may use the same physical path to access different logical devices.
- 2) Intra-gration, i.e., it is hard to correlate physical objects such as disk and tape drivers, redundant array of independent disks, RAID subsystems, switches, paths, and host bus adapters, HBAs, with data abstraction such as file systems and databases.
- 3) Inter-gration, i.e., storage management relies on managing components in the enterprise information processing environments such as computers, applications, user profiles, network components and facilities, printers and databases. Yet, there is no comprehensive tool for an integrated management.
- 4) SAN architecture, i.e., imposing the limitation of one SAN component vendor viewpoint. For example, optimizing switches may create disjoint and overlapping zones and redundant path configurations for performance and availability while RAID subsystem or volume manager may be ignored.

5. Proposing an Architectural Solution

First, let us look at three potential supports--hardware, management, and technology for this proposed architecture solution.

5.1 Hardware Potential

While enjoying the benefit of the magnetic disk storage technology advancement, [Grochowski03] end users expend their data collection with ease. Should the management cost have to be increased along with the increase of the data volume collected? With the ever-growing complexity of distributed systems and variety of software running on such systems, management cost seems to be running up inevitably. It is time to return to simplicity. Applying the principle that drives the RISC processor development, we ought to look at what can be done at hardware level, the storage devices. It is possible to decide on how much of management software can be implemented inside the storage devices (i.e., in embedded software and firmware) to make storage devices to be plug-and-play device, or as intelligent and automated [Horn01] as

necessary.

5.2 Management Potential

We all know, “all data are not created equal”, nor are applications. Therefore, with appropriate categorizations of data and applications, we ought to explore variation of system architectures and to minimize the ambiguity in the needs for management software. As a consequence, the unnecessary complexity of management may be eliminated.

5.3. Technology Potential

Remarkably by Chang et al [Chang 99] with a cost-effective means of designing, verifying, and building complex Systems on Chips (SOC), it is now the Golden Age of Electronics. An embedded microcontroller, sensor inputs and actuator outputs (A/D and D/A), network interfaces (both wired and wireless networks), and some embedded software will provide necessary functionality. With the latest process technologies and efficient SOC design (platform-based), many applications instantly become possible, including storage devices that are simply managed as plug and play device to any computing system.

5.4 An Architectural Solution

As shown in Figure 3, LSD is similar to the Network Attached Secure Disk (NASD) project at Carnegie Mellon University. [Ruwart02][Cipiroco03] However, the motivation of this architecture is to reduce storage management cost. Here the connection between node 1 and node 2 is via networks (LAN, WAN, etc.). The dividing line between node 1 and node 2 is drawn based on the needs for short-term storage or long-term storage, which differentiate applications and data by the timing requirements.

A short-term storage device (SSD) may be any storage device in current computing systems, i.e., direct attached storage, DAS, network attached storage, NAS, and storage area network, SAN. A long-term storage device (LSD), on the other hand, is the device that the proposal envisions. The connections to LSD and the operation behaviors of LSD should be completely specified for plug and play. Like object storage device, OSD, it is based on "the segregation of the storage technology from the applications/file systems, as well as the possibility of the storage technology development independently of the file system technology and/or the device technology itself." [Ciprico 03] Unlike OSD, it does not require a metadata server tightly coupled with LSD, but a network protocol at application layer, much like simple network management protocol, SNMP; call it SBMP, i.e., simple block management protocol. Like a RAID controller, it is an operating system, OS, independent entity, unlike a RAID controller, it does not depend on logic unit numbers, LUNs, but it requires a naming standard much like The international Standard Book Numbering, ISBN. ISBN system was introduced into the UK by J. Whitaker and Sons Ltd. In 1967 and into the US in 1968 by the R.R. Bowker Company. It is used by publishers, distributors, wholesalers, bookstores, and libraries, among others, in over hundred countries and regions to expedite such operations as order fulfillment, electronic point-in-sale checkout, inventory control, returns processing, circulation/location control, file maintenance and update, library union lists, and royalty payment. <http://www.ISBN.org>

There are three basic functions that LSD should have: backup, archive, and space management. The configurations of these three functions bear no concern to end users and will be operated in a server environment. Again, it is a plug and play network device. Further, it is a scalable and flexible device and can be categorized in public, private, and specialized, just as current library systems. Each LSD is visible to its end users. The interconnection relies on ftp for block storage and retrieval. Current technologies in authentication, authorization, and accessing of data are transferable to LSD. Concerns with quality of service, QoS, in OSD can be transferred here as well. The levels of QoS in United States

Postal Services, USPS, (or FedEx, UPS, etc.) make perfect analogy with QoS of LSD, which are requested by end users. A fast delivery should be associated with a higher charge if applicable.

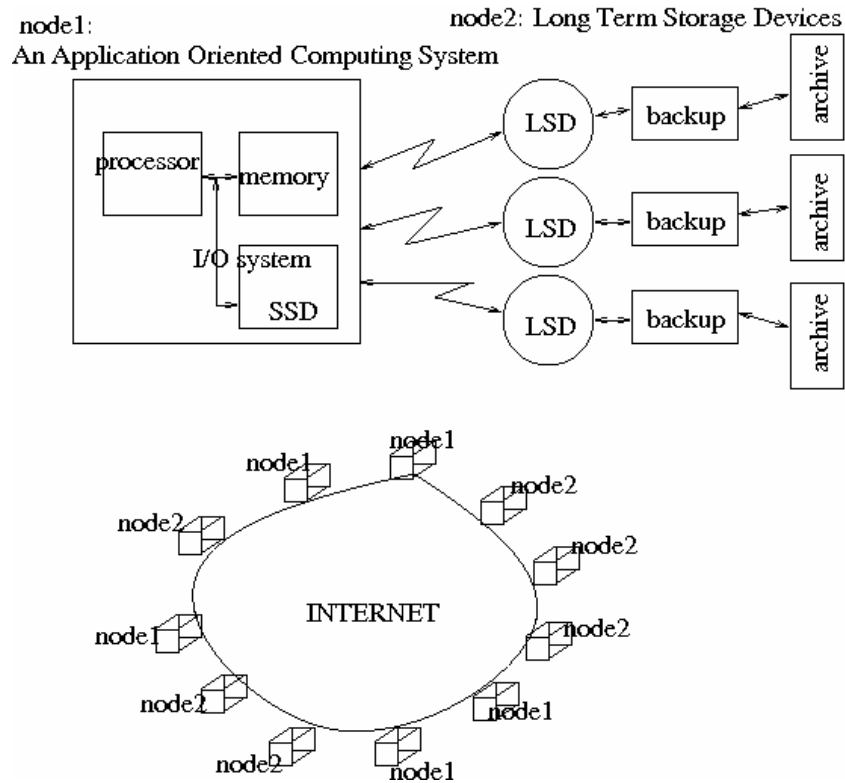


Figure 3: An Architectural Solution for Storage Management Problem

5.5 LSD Management Cost

Let's inspect the management cost from the challenges listed in the previous section under SAN:

- 1) Monitoring and analysis-connections between systems running end-user applications and an LSD are established in the Peer-to-Peer fashion. I/O performance of end user applications becomes loosely coupled with the block storage/retrieval of LSD. Complexity of I/O paths, hence, the performance analysis is dissolved.
- 2) Intra-gration-physical object of LSD is a black box to end-users. The space management and response time with degree of reliability and availability become features of LSD. Different data may require different quality of LSDs and different services of an LSD.
- 3) Inter-gration –multi-vendor components of SAN are no longer applicable to LSD, since switch vendors and disk vendors should converge together with one hardware entity, and one comprehensive tool to manage the LD. Further, the network stability is not tightly coupled with LSD performance. In other words, if LSD traffic is in a nicely routed and stable network (such as IP over ATM), the quality of LSD service will be better.
- 4) SAN architecture- zoning in SAN is no longer applicable to LSD. Design decision associated with different QoS of LSD is proprietary that may reflect the price-performance of that LSD.

6. Conclusion

Architecture of storage systems aimed at reducing storage management cost has been proposed in this paper. It

looks promising because it can not only reduce the management cost but also promote data storage efficiency, data sharing, and data protection. In retrospect of I/O development driving force, enabling technologies such as cache and memory residency database have made the shift from improving processor performance to end-user/system administrator friendliness, i.e., the ease of management. Major spaces of the writing are devoted to the rationale of the proposal. The author believes that, parallel with the current development of DAS, NAS and SAN storage devices, taking advantages of existing storage device development technologies, the development of LSD and SSD in this proposed architectural solution will make its significance in the future storage systems promoting the best of data storage efficiency, data sharing, and data protection while minimizing storage management cost.

7. Reference

- [Barker02] R. Barker and P. Massiglia, Storage Area Network Essentials, Wiley, 2002
- [Blaauw97] G. A. Blaauw and F.P. Brooks Jr., Computer Architecture Conception and Evolution, Publisher, 1997
- [Chang99] H. Chang, L Cooke, M. Hunt, G. Martin, A McNelly, and L Todd, Surviving the SOC Revolution, a guide to platform-based Design, Kluwer Academic publishers, 1999
- [Ciprico03] Does OSD have a role to play in the broadcast and media space? Ciprico's Technology White Paper, www.ciprico.com
- [Flynn 95] M. J. Flynn, Computer Architecture, pipelined and parallel processor design, publisher, 1995
- [Ganek 99] A.G. Ganek and E.H. Sussenguth. Turning Points in Systems Architecture, IBM Systems Journal, Vol.38, Nos2&3, 1999
- [Gibson 96] G.A.Gibson, D.F.Nagle, K.Amiri, F.W.Chang, E. Feinberg, H.Gobioff, C. Lee, B. Ozceri, E. Riedel, and D. Rochberg, A Case for Network-Attached Secure Disks, CMU-CS-96-142
- [Grochowski 03] E. Grochowski and R.D. Halem, Technological impact of magnetic hard disk drives on storage systems, IBM Systems Journal, Vol. 42, No. 2, 2003
- [Hennessy 96] J. L. Hennessy and D. A. Patterson, Computer Architecture, a quantitative approach, Morgan Kaufmann publishers Inc. , 1996
- [Hopkins 87] M.E. Hopkins, A perspective on the 801 /Reduced Instruction Set Computer, IBM Systems Journal, Vol. 26 No. 1, 1987
- [Horn 01] P. Horn, Automatic Computing: IBM's perspective on the state of information technology, <http://www.ibm.com/autonomic>, 2001
- [Kaczmariski 03] M. Kaczmariski, T. Jiang, D.A. Pease, Beyond backup, toward storage management, IBM Systems Journal, Vol. 42, No2, 2003
- [Rossi06] Sandra Rossi, Bad Data is the source of costly data management problem. Computer World, 2006.
- [Ruwart 02] Thomas M. Ruwart, OSD: A Tutorial on Object Storage Devices, MSST'02, College Park, MD
- [Scherr 78] A.L. Scherr, Distributed data processing, IBM Systems Journal, Vol. 17 No.4, 1978
- [Schleicher 89] D.L. Schleicher and R.L. Taylor, System overview of the Application System/400, IBM Systems Journal, Vol. 28, No3, 1989
- [Simitci 03] H. Simitci, Storage Network Performance Analysis, Wiley,2003
- [Tanebaum 95] A. Tanebaum, Distributed Operating Systems, Prentice Hall, 1995

Author: