The Myth of MIPS for I/O

An Overview of Functional Multiprocessing for NFS Network Servers

Bruce Nelson,
Raphael Frommer, &
Auspex Engineering

Technical Report 1
Sixth Edition, Second Printing
August 1992

AUSPEX

Abstract

This report introduces a new computer architecture that dramatically increases the performance and scalability of network file servers. It begins with a discussion of the *I/O performance gap*, a widespread Unix-industry performance problem that has arisen because recent surges in workstation MIPS have not been balanced by similar increases in file server I/O capability. This report explains how I/O bottlenecks can be removed using *functional multiprocessing* (FMP).

FMP is a multiprocessor server architecture that divorces the critical network, file, and disk I/O functions from the Unix operating system and executes them instead on dedicated, high-throughput I/O processors. Complete user-level transparency and compatibility are maintained by executing Unix on a separate, general-purpose, Sun-compliant, peer processor. Though an FMP server may incorporate symmetric multiprocessing (SMP) CPUs to speed compute-intensive applications, its architecture is fundamentally different from the conventional SMP architectures common in compute servers, minicomputers, and some traditional file servers. This report discusses the major hardware and software elements of the Auspex server's FMP architecture.

Using FMP, Auspex has built an NFS file server that delivers 2,250 NFS I/O operations per second (IOPS) by connecting eight parallel Ethernets to 10–30 fully concurrent SCSI storage channels. NFS IOPS are a distinctly more relevant measure of file server performance than CPU MIPS because a file server's mission is to deliver files—not MIPS—to client workstations. In contrast to the Auspex server's 2,250 NFS IOPS, traditional NFS file servers peak at about 450 NFS IOPS. As a result, the FMP architecture has at least five times the NFS throughput of the best conventional architectures.

This report discusses FMP primarily in the context of NFS file service. Yet FMP readily incorporates compute server functionality, including SPARC-based SMP multiprocessor subsystems. For example, to increase database performance, an SMP subsystem attached to the backplane of an FMP server can take advantage of the server's high-concurrency SCSI-disk array-based storage subsystem.

With over 450 Auspex servers in production use worldwide, the FMP architecture has proven exceptionally reliable and scalable in practice. Typical systems range between 4–8 Ethernets, 10–60 GB of disk storage, and support 40–200 active workstations.

Systems of this capacity and throughput *do* represent a new class of Unix network server. That's why both IBM and DEC have joint development and system integration agreements, respectively, with Auspex for FMP server products.

Document 300-TC011, V6.1.14, 920819.

Auspex Systems
5200 Great America Parkway
Santa Clara, California 95054 USA
Phone: 408/986-2000 · Fax: 408/986-2020
Email:Info@Auspex.com

An edited version of the Third Edition of this report appears in *SunWorld (SunTech Journal)* 4(1):84–89, January 1991.
An edited version of the Fifth Edition is in the *Proceedings of the European Sun User Group*, Birmingham, U.K., 10–12 Sept. 1991.
An edited version of the Sixth Edition, Second Printing is in the *Proceedings of the Australian Unix User Group Conference*, Melbourne, Australia, 8–11 September 1992.

[Nelson88] Michael N. Nelson, Brent B. Welch, and John K. Ousterhout.
Caching in the Sprite Network File System.
ACM Transactions on Computer Systems 6(1): 134–54, February 1988.

[Nelson92] Bruce Nelson and Yu-Ping Cheng.
How and Why SCSI is Better than IPI for NFS.
Proceedings of the Winter 1992 USENIX Conference, San Francisco, CA, 22–24 Jan. 1992.
Also Technical Report 6, second edition, Auspex Systems Inc., July 1992.

[Sandberg85] Russel Sandberg, David Goldberg, Steve Kleiman, Dan Walsh, and Bob Lyon.
Design and Implementation of the Sun Network File System.
Proceedings of the Summer 1985 USENIX Conference, pp. 119–30, Portland, OR, June 1985.

[Schwartz90] Allan M. Schwartz, David Hitz, and William M. Pitts.
LFS—A Local File System for Multiprocessor NFS Network Servers.
Proceedings of the IEEE Systems Design & Networks Conference '90,
Santa Clara, California, 8–10 May 1990.
Also Technical Report 4, Auspex Systems Inc., December 1989.

[Wilson90] David Wilson.
Tested Mettle: The Auspex NS 5000 NFS Network Server.
Unix Review 8(8), August 1990.

7 References

[Auspex89] Auspex Systems Inc.
Benchmark Methodology and Preliminary Performance Specifications for the Auspex NS 5000 Network Server.
Technical Report 2, Auspex Systems Inc., October 1989.

[Auspex91a] Auspex Systems Inc.
Truly Balanced and General-Purpose Servers.
Auspex Benchmark Brief 6, April 1991.

[Auspex91b] Auspex Systems Inc.
An Auspex NS 5000 and Sun 690MP NFS Comparison.
Auspex Benchmark Brief 7, November 1991.

[Boggs88] David R. Boggs, Jeffrey C. Mogul, and Christopher A. Kent.
Measured Capacity of an Ethernet: Myths and Reality.
Proceedings of the SIGCOMM '88 Symposium on Communications Architectures and Protocols, ACM SIGCOMM, Stanford, CA, August 1988.
Also Digital Western Research Laboratory Research Report 88/4.

[Cheng91] Yu-Ping Cheng, Guy Harris, Bruce Nelson, and William M. Pitts.
Using Virtual Partitions for Disk Storage Administration in Network Servers.
Proceedings of the Sun User Group, San Jose, CA, 4–6 December 1991.
Also Technical Report 8, Auspex Systems Inc., July 1991.

[Hitz90] David Hitz, Guy Harris, James K. Lau, and Allan M. Schwartz.
Using Unix as One Component of a Lightweight Distributed Kernel for Multiprocessor File Servers.
Proceedings of the Winter 1990 USENIX Conference, Washington, DC, 23–26 January 1990.
Also Technical Report 5, Auspex Systems Inc., January 1990.

[Hitz91] David Hitz.
A System Administrator's Performance Monitor for Tuning NFS Network Servers.
Proceedings of the Sun User Group, San Jose, CA, 3–5 December 1990.
Also Technical Report 7, Auspex Systems Inc., May 1991.

[Horton90] William A. Horton and Bruce Nelson.
The Auspex NS 5000 and the Sun SPARCserver 490 in a One- and Two-Ethernet NFS Performance Comparison.
Performance Report 2, Auspex Systems Inc., May 1990.

[Howard88] John H. Howard, Michael L. Kazar, Sherri G Menees, David A. Nichols, M. Satyanarayanan, Robert N. Sidebotham, and Michael J. West.
Scale and Performance in a Distributed File System.
ACM Transactions on Computer Systems 6(1): 51–81, February 1988.

[LADDIS91] The LADDIS Group.
LADDIS—A Vendor-Neutral Standard NFS Benchmark.
Proceedings of Interop 1991 Fall Conference, "Wednesday" volume, 9 October 1991.
Free *updated* reprints are available from Auspex Systems Inc.

[Levitt91] Jason Levitt.
Gauging NFS Server Performance [using LADDIS].
Unix Today, 25 November 1991, pp. 30 and 36.

[Lyon89] Bob Lyon and Russel Sandberg.
Breaking Through the NFS Performance Barrier.
SunTech Journal 2(4): 21–27, Autumn 1989.

SPARC — Scalable Processor Architecture. SPARC International's specification for the Reduced-Instruction-Set-Computer (RISC) CPUs found in systems sold by Sun Microsystems, Auspex, Solbourne, etc.

SPEC — System Performance Evaluation Cooperative. A nonprofit corporation of vendors' technical representatives that develops and certifies accurate, vendor-neutral, computer-system benchmarks. As an example, popular SPEC CPU benchmark metrics include SPECint, SPECfp, and the now obsolete SPECmarks. See LADDIS.

SunOS — Sun's Unix operating system. The implementation of Unix used by Sun Microsystems. SunOS was originally based on the BSD 4.2 release. The NetServer Host Processor runs a licensed version of SunOS modified to accommodate the LFS file system and FMP software architecture.

UFS — Unix File System. UFS is the standard file system type in the BSD 4.3 kernel. See VFS.

VFS — Virtual File System. A generic file system interface defined in SunOS that allows convenient and transparent integration of different file system *types*. In the NetServer host processor, all file system requests are sent to the VFS, where they are mapped into NFS, LFS, or UFS calls. LFS calls are routed to the file processor, and UFS calls are routed directly to the storage processor. Since the host processor does not provide NFS service in the NetServer, file system calls are not normally mapped through VFS to NFS.

VME — A backplane bus conforming to a Motorola specification by the same name. The VME bus specification defines the mechanical and electrical properties of an interfacing system used to connect data processing, data storage, and peripheral control devices in a closely coupled configuration. It is comprised of four groups of signal lines: the data transfer bus (DTB, for high speed asynchronous parallel transfers), the DTB arbitration bus, the seven-level priority interrupt bus, and the utilities bus.

XDR — eXternal Data Representation. A Sun-defined ONC presentation-layer protocol for the canonical, serialized representation of programming-language data structures. XDR is used for convenient information exchange between heterogeneous computer systems that may have different byte orderings, floating-point representations, and so forth. XDR is used to marshal ONC RPC parameters and underlies NFS.

YP — Yellow Pages. An obsolete name for NIS. See NIS.

MTTR — Mean Time To Repair. Includes the time taken to diagnose the failure, replace or repair faulty component(s), and reboot the system. See MTBF.

NIS — Network Information Service. This is ONC's general name-binding and name-resolution protocol and service, previously called Yellow Pages. See YP.

NFS — Network File System. Both a SunOS file system type and a Sun-defined ONC application-layer protocol for peer-to-peer distributed file-system communication. NFS allows a remote file system (often located on a file server) to be mounted transparently by client workstations. The client cannot perceive any functional difference in service between remote and local file systems (with trivial exceptions). Using high-performance servers, there is not much response-time difference either. NFS is the most popular ONC service, has been licensed to over 300 computer system vendors, runs on 3,100,000 nodes, and is a de facto Unix standard. See VFS, ONC.

NFS IOPS — NFS I/O Operations Per Second. A vendor-independent measure of NFS I/O performance. NFS throughput, measured in NFS IOPS, depends directly on the relative *mixture* of NFS operations being executed because NFS operations have varying complexity. Typical NFS operations include: *lookup, read, write, getattr, readlink, readdir, create, remove, setattr, and statfs.* See IOPS.

NHFSstone — The generic name given to derivatives of Legato Systems' original NFS server benchmark of the same name. Numerous deficiencies in this single-client benchmark have caused it to be abandoned in favor of SPEC's LADDIS benchmark. See LADDIS.

ONC — Open Network Computing. The trade name for the suite of standard IP-based network services—including RPC, XDR, and NFS—promulgated by Sun Microsystems.

RPC — Remote Procedure Call. An RPC is an (almost) transparent subroutine call between two computers in a distributed system. ONC RPC is a Sun-defined session-layer protocol for peer-to-peer RPC communication between ONC hosts. ONC RPC underlies NFS.

SBus — SPARC International's limited-distance, 80-MB/s (peak rate) bus linking the central CPU(s) to small physical form factor controllers (e.g., Ethernet, FDDI, SCSI). Typical SBus systems consist of a "motherboard" containing the central processor(s) and SBus interface logic, a number of SBus devices on the motherboard itself, and some SBus expansion connectors. See MBus.

SCSI — Small Computer System Interface. An intelligent *bus*-level interface that defines a standard I/O bus and a set of high-level I/O commands. Each SCSI device has an intelligent SCSI *controller* built into it. SCSI is used for local data communication between a host CPU and an attached SCSI bus that contains intelligent peripheral devices such as disks, tapes, scanners, and printers. See IPI-2.

SMD — Storage Module Device. A device-level disk interface that predates IPI-2 and that was commonly used in traditional workstation-based servers. See SCSI and IPI-2.

SMP — Symmetric Multi-Processor. A computer architecture where processing tasks are executed in parallel on multiple, identical, general-purpose CPUs that share a common memory. SMP computer systems usually have modified operating systems that can themselves execute concurrently. The SMP architecture offers high computational throughput, but not necessarily high I/O throughput. See FMP.

SNMP — Simple Network Management Protocol. SNMP is a special TCP/IP protocol that is used for communication between simple, server-resident SNMP *agents* that respond to network administration requests from simple-to-sophisticated SNMP *manager tools* running on remote workstations.

6 Glossary

b — Abbreviation for bit, e.g., 10-Mb/s Ethernet.

B — Abbreviation for byte, e.g., 120-GB total capacity.

BSD — Berkeley Software Distribution. The major branch of Unix upon which SunOS and other variants are based.

FDDI — Fiber Distributed Data Interface. A new standard for local area networks that uses fiber-optic media capable of data rates up to 100 megabits/second over distances up to 100 km. An FDDI network is a token-based logical ring, and is often constructed as a pair of counter-rotating redundant rings (called dual-attach mode) for reliability. Ethernet, in contrast, is a bus-based, non-token, 10 megabits/second network standard.

FMK — Functional Multiprocessing Kernel. Communication among the NetServer's multiple hardware processors and software processes is handled by the FMK, a low-overhead message-passing kernel executing on each FMP processor that communicates over the NetServer backplane.

FMP — Functional Multi-Processor. Auspex's multiprocessor architecture distributes I/O activities to parallel processors specifically optimized for particular functions such as protocol processing, storage management, host operating systems, and so forth. See SMP.

HIPPI — High-Performance Parallel Interface. A short-distance, unidirectional, extremely fast, 100 MB/s interconnect. It is usually used in pairs, providing bidirectional 100 MB/s transfers. HIPPI will be used for connecting supercomputers and minisupers to high-speed networks and storage devices. It is a draft standard formulated by Los Alamos National Laboratory.

IOPS — I/O Operations Per Second. A generic measure of I/O performance. To be meaningful, the *type* and *operation mixture* of I/O must be specified as well, such as *disk read* IOPS and *NFS write* IOPS. See NFS IOPS.

IPI-2 — Intelligent Peripheral Interface 2. IPI-2 is a *device*-level disk interface (in contrast to SCSI's *bus*-level interface). IPI-2 is an unintelligent but high-performance interface that can string multiple disks from a separate, intelligent disk controller. See SCSI.

LADDIS — An acronym formed by names of the group (Legato, Auspex, Data General, Digital Equipment Corporation, Interphase, and Sun) that has developed and popularized SPEC's vendor-neutral NFS server benchmark of the same name. See SPEC.

LFS — Local File System. A file system type developed by Auspex and used in the NetServer for file-system communication between the Ethernet processors and the file processor, and between the host processor and the file processor. LFS provides local file operations similar to NFS remote operations, but without the protocol processing overhead. See VFS.

MBus — SPARC International's 280-MB/s (peak rate) primary memory bus into which all CPUs plug (via so-called MBus *modules*). The MBus is one of several buses in Sun's hierarchical bus architecture. See SBus and VME.

MTBF — Mean Time Between Failure. A key component of the availability equation, AVAILABILITY = (MTBF – MTTR) ÷ MTBF. Example: A server which on average fails once every 5,000 hours and on average takes 2 hours to diagnose, replace faulty components, and reboot would have an availability rating of (5,000 – 2) ÷ 5,000 = 99.96%.

population growth. Another customer cites Ethernet bandwidth, not the server itself, as the gating factor in a 180-dataless-workstation single-NetServer configuration which is expected to grow to 250–260 clients ultimately.

- Some customers have discovered unusual FDDI cost savings with NetServers: using their multiple-Ethernet capability, they are delaying their migration to FDDI until costs are lower, throughput is higher, and compatibility between vendors is proven.
- Some system administrators report that they now lead "normal" lives, working 8–5 because they're no longer fighting multiple-server crossmounts, backups, and updates. They tell us, "I don't even know the server's there."
- Auspex's claimed dedication to "total customer satisfaction" is proving itself in practice. Especially among Fortune 500 customers, Auspex continues to win business because decision-maker reference checks with Auspex's existing customers show that Auspex's customer support—from sales, service, engineering, *and* manufacturing—is "better than any Unix vendor's I've ever dealt with."

Auspex calls this success because our customers tell us that Auspex is making *them* successful. Auspex is achieving this recognition because it is a focused *network server* company, not just a workstation company.

NetServer design features and functions that contribute to high reliability and easy administration are: hot-pluggable CD-ROMs, disks, and tapes; disk striping and mirroring via virtual partitions; NFS bulletproofing through FMP; NFS crossmount elimination by consolidating servers; single-point file-system backup; 15-minute replacement time of any component; dual power supplies; and the performance monitor. These topics are all discussed in companion reports listed in section 7.

Our customers' actual operating experience with NetServers has been extremely gratifying. With over 450 systems (primarily NS 5000s and 5500s) installed as of this printing, customer response is universally enthusiastic:

- Many customers, especially those in the Fortune 500, have multiple machines both throughout their companies and at single sites. One Fortune 100 semiconductor customer has a NetServer volume purchase agreement that essentially makes Auspex its exclusive worldwide corporate NFS server. This pattern in the United States is rapidly repeating itself in Japan.
- NetServer application environments are similar for both large and small customers: software development, VLSI and PCB design, mechanical CAD, oil exploration, and scientific research (at national laboratories and supercomputer centers).
- NetServer network environments are highly heterogeneous. While Sun Microsystems' workstations dominate the total, DEC, HP, IBM, Intergraph, NeXT, Silicon Graphics, and Sony News equipment appears frequently. Several customers have environments that are almost entirely HP 9000, IBM RS/6000, Silicon Graphics, or Next workstations—all attached to Auspex NetServers.
- Most systems have four or six Ethernets and average 15 disk drives (20 GB). Many are fully configured with 8 Ethernets, > 30 disk drives, and two to six 4- or 8-mm tape drives. Many systems have > 50 GB of storage.
- Heavily loaded systems experience exceptional uptime between scheduled shutdowns: an 80-workstation environment exceeded 100 days, a 40-workstation system exceeded 65 days, and a 60-client environment exceeded 45 days. NetServer uptime is typically governed by customer maintenance schedules—most reboots are voluntary and initiated by system administrators.
- NetServer disk reliability is exceptional. The Hewlett-Packard disks in Auspex servers have a manufacturer-guaranteed MTBF of 150,000 hours, but with over 4,000 disks in production NetServers , the observed MTBF exceeds 300,000 hours (over 34 *years*).
- Some users report that Ethernet access to data on a NetServer disk is *faster* than accessing the workstation's local disk. This seemingly implausible behavior is mostly attributed mostly to low-cost local disk drives whose seek times are dramatically below that of Auspex's drives, coupled with the unoptimized disk-arm movements of many workstation file systems.
- Some power users report that their workstations run faster during the day (sharing a NetServer with 40 other users) than they did previously in the middle of the night (using a conventional server as the only user).
- One uptime-conscious customer relocated his NetServer and its user community to a new building in record time. "In just two hours, versus two days, while our other vendors' servers were lying about in pieces on the floor, the NetServer was powered down, transported, reinstalled, and rebooted. Our engineers were up and operating in record time. Consolidating NFS service onto one NetServer meant there was just one server to move, not ten or fifteen. We had even tested our expansion from two to four Ethernets prior to teardown and transported the hot-pluggable disks by hand. We moved the NetServer ourselves; Auspex was the sole vendor we didn't have to pay to do the job. Auspex system engineers were on call, but all went smoothly without them." The speed and smoothness of the move earned the system administrator a standing ovation from his executive staff, as well as a performance bonus.
- One customer has 91 diskless workstations on six Ethernets, with more workstations planned. A large software development shop has over 100 diskless SPARCstation IPCs, SLCs, and ELCs on 5 Ethernets. A sixth network attaches eight headless workstations used as compute servers for software builds. A seventh subnet attaches the NetServer to a 100-Mb/s backbone connecting to a second NetServer, an optical disk jukebox, and geographically remote locations. An eighth net will accommodate client

In figure 13's tests, NFS load is increased in uniform increments using Auspex's Traffic Generator [Horton90]. These tests use the usual and workstation-preferred 8-KB *read* and *write* NFS block size, not the unrealistic 1-KB block size used by the popular NHFSstone benchmark. The standard Dhrystone 2.1 integer benchmark measures available CPU power for applications. Auspex's Benchmark Brief 6 has a more detailed comparison [Auspex91a].

5 Conclusion

The I/O performance gap is a tangible, daily problem for many Unix workstation users. The NetServer's functional multiprocessor architecture breaks this bottleneck for NFS clients. FMP eliminates the cost, performance, reliability, and administrative problems of the traditional-architecture, server-per-subnet NFS network. FMP is a demonstrably scalable network-server architecture with exceptional throughput and low latency.

5.1 The Future of Functional Multiprocessing

Many people suggest that database processing—not just NFS—is a natural complement for the NetServer's high-throughput I/O subsystems. Fortunately, as we have seen, FMP readily incorporates the SMP compute-server functionality needed for high-performance databases. By openly licensing their microprocessor architectures, Digital Equipment (with Alpha), Hewlett-Packard (with PA-RISC), and Sun Microsystems (with SPARC) hope to create PC-like clone industries. This gives Auspex several good options for a faster Unix host processor.

- *RISC microprocessor suppliers* such as Cypress-Ross, Fujitsu, LSI Logic, Texas Instruments, and Weitek are eager to supply companies like Auspex with the latest SPARC CPUs and chip sets. For SPARC, many vendors even offer complete, Sun-standard MBus modules for SPARC multiprocessor systems.
- *Microsoft-style software companies* actively compete for Auspex's Unix operating system business. Again using the SPARC example, SunSoft, Unix Systems Laboratory, Interactive Systems, and others all provide either uni- or multi-processor versions of SunOS/SVR4 for reasonable license fees.

The consequence is simple. Auspex has done the hard part—mainframe class I/O at Unix price/performance—and is quite happy to let the giants battle it out for MIPS. The CPU and operating system contenders are already telephoning to see if Auspex is ready to begin offering compute server functionality. (We will be when our customers are.) Benchmark Brief 6 discusses this further [Auspex91a].

A final sign of FMP's bright future is indicated by these business developments: IBM and DEC have joint development and system integration agreements, respectively, with Auspex for FMP-based server products.

5.2 Independent Evaluations

But the real test of a new computer architecture is not in the laboratory, or in future products, but in the marketplace. An independent assessment of the NS 5000 by David Wilson of *Unix Review* rated the system "outstanding"—the highest possible [Wilson90]. He said:

- "The Auspex NS 5000 is too good a system to ignore, and is likely to fit into a network environment with little problem. We like the Auspex system very much."
- "Based on our experience, working for the Auspex support group may be like working as a Maytag repairman—it's the loneliest job in town because there are so few problems."
- "For a customer that needs four or more Ethernets, and several gigabytes of disk storage, the NS 5000 is potentially the best buy of any system available today. At these levels of network use, the Auspex should be considered strongly even if there are no current problems in an existing network."

5.3 Operational Experience

While this overview report focuses mainly on architecture and performance, Auspex's customers say that the NetServer's high reliability and simplified system administration are just as important to them. The

4.3 Typical NFS Benchmark Performance

The next two sections summarize our benchmarking experience to date. *Unix Review* has conducted an independent operational and performance evaluation [Wilson90]. Auspex's own general benchmarking methodology is discussed in other reports [Auspex89, Horton90]. Auspex will shift its benchmark reporting format to SPEC's new LADDIS benchmark [LADDIS91, Levitt91] when LADDIS is released.

LADDIS is a standard, vendor-neutral NFS benchmark presently being certified by the System Performance Evaluation Cooperative (SPEC). The benchmark results in this report are, however, measured using Auspex's Traffic Generator [Horton90]. They are not LADDIS results because SPEC requests that vendors *not* publish LADDIS results until LADDIS certification is complete in late 1992. In the meantime, LADDIS will be evolving and its results changing. Fortunately, since both LADDIS and the Traffic Generator use a similar test methodology, their results will correlate well, but probably not match numerically.

Figure 12 shows NetServer performance with release 1.4.1ß software. The NS 5500 attains 2,250 average-mixture NFS IOPS at a 50-ms response time. This provides at least five times the NFS throughput of other high-performance servers. Further details (NS 5500 excepted) are described in a separate benchmark brief [Auspex91b].

In figure 12, the vertical axis shows server throughput in average-operation-mixture NFS IOPS. Recall that a typical client workstation doing moderate I/O requests about 10–20 IOPS depending on application and client configuration. The horizontal axis shows NFS response times measured at the workstation. Twenty to 40 milliseconds is ideal, while 50 ms is a tolerable upper bound.

4.4 Typical NFS versus Compute Throughput

Figure 13 compares the simultaneous compute and file service capabilities of Auspex NetServers with accelerated conventional servers like Sun's SPARCservers. It shows that Auspex NetServers provide compute capability that does not diminish under NFS load, whereas traditional compute servers like Sun's enhanced SPARCserver 490 CPU-saturate at modest NFS load levels. Thus, Auspex NetServers can provide truly balanced compute and file service while simultaneously offering at least five times the NFS performance of enhanced conventional servers (Sun's enhanced SPARCservers are the best tested so far, with NFS performance 1/5 of a NetServer).

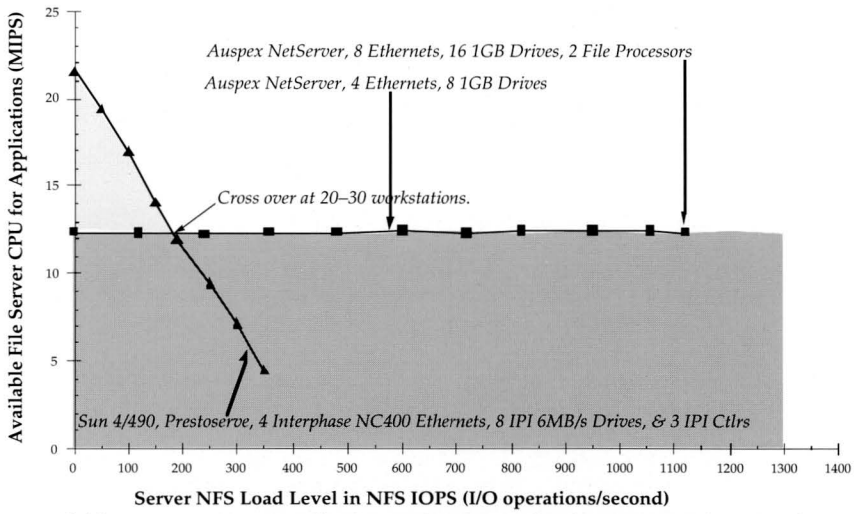

Figure 13. Comparing available compute power under increasing NFS I/O load. FMP network servers have compute power that does not degrade with NFS loading. This is because the FMP architecture separates NFS and file system processing from the Unix application CPU. Conventional architectures, however, suffer a linear CPU degradation as NFS load increases (the enhanced 4/490 is shown as an example). Neither NFS nor I/O is free, as these measurements show. At the crossover point of 200 NFS IOPS (about 20 diskless RISC clients or about 30 dataless RISC clients) an Auspex NetServer has more available CPU for user jobs than the 4/490—even though the 490 has more nominal MIPS. The NFS operation mix is the same as in figure 12.

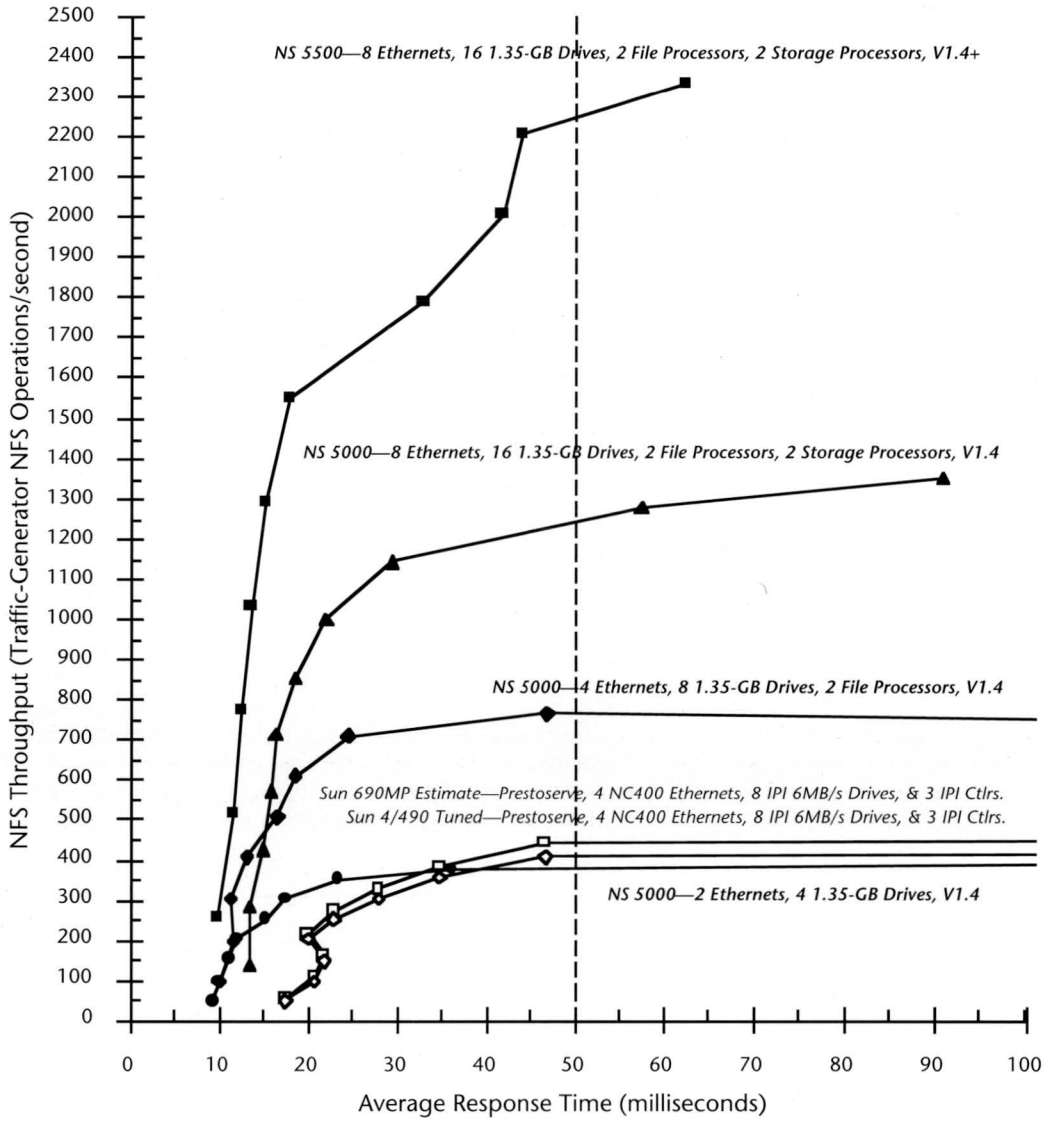

Figure 12: NetServer NFS performance measured by throughput and response time. This chart demonstrates the excellent scalability and maximum throughput of the Auspex NetServer. The NS 5000 is shown in three "doubled" configurations: 2 Ethernets and 4 drives, 4 Ethernets and 8 drives, and 8 Ethernets and 16 drives. For each doubling of networks and drives, the NS 5000's NFS throughput increases by an average factor of 1.9 (geometric mean). The NS 5500 (top curve), Auspex's latest generation NetServer, takes advantage of new Ethernet and file processors, as well as tuned system software. Sun SPARCservers 490 and 690MP—enhanced with a Prestoserve write buffer and intelligent Ethernet controllers—are shown for NFS comparison (the 690MP estimate is based on Sun's own, published 8%-improvement figure [Auspex91b]). Additional testing shows that the 490 curve does *not* improve when more disks or networks are added—disks don't help because of the serialized nature of compute-server-oriented IPI subsystems; Ethernet coprocessors don't help because of CPU limits. The 50-millisecond response time cutoff is a comfortable maximum: workstation users prefer 20–40 ms and notice a marked slowdown over 100 ms. The standard NFS operation mixture in these tests is: *Lookup* 34%, *Read* 22%, *Write* 15%, *GetAttributes* 13%, *ReadLink* 8%, *ReadDir* 3%, *Create* 2%, *Remove* 1%, *SetAttributes* 1%, *Statfs* 1%; *Reads* and *Writes* use 8-KB blocks. **Benchmark configurations:** The Auspex NetServer (the NS 5000 is running Auspex system software version 1.4; the NS 5500 is running 1.4.1ß; both are performing synchronous NFS writes) has 32 MB of I/O cache memory, 4 Ethernet Processors, 2 File Processors, 2 Storage Processors (each with Write Accelerator), and 16 Hewlett-Packard 1.35-GB SCSI-2 disk drives. The SPARCserver 490 (running SunOS 4.1 with 24 NFSDs, Prestoserve 2.0, and NC400 software 1.1) has 64 MB of memory with 4 Sun Network Coprocessors (formerly called Interphase NC400), Sun Prestoserve write buffer, 3 Sun ISP-80 IPI-2 disk controllers, and 8 911-MB 6-MB/s IPI-2 disk drives. The estimated 690MP hardware configuration is identical to the 490 configuration; software is SunOS 4.1.2.

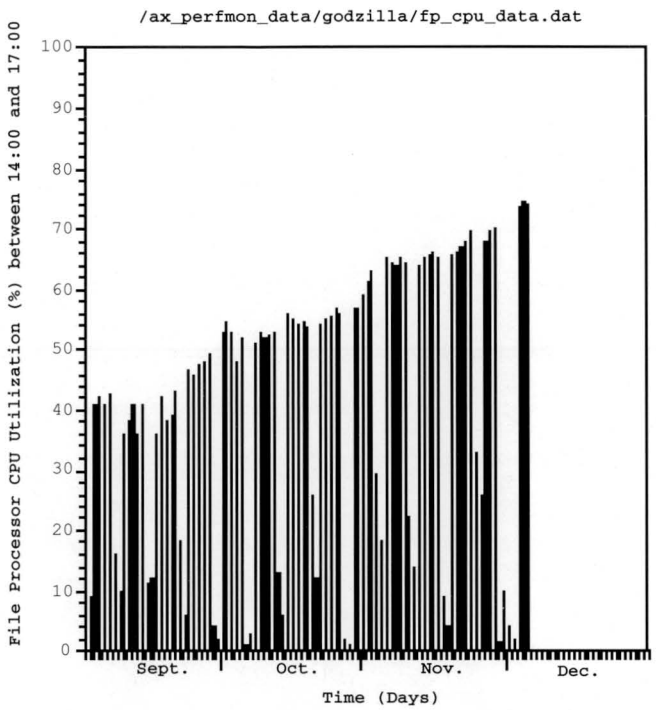

Figure 11: Histogram display of performance monitor statistics. As an example, this histogram displays a log of file-processor CPU data from the performance monitor. Each vertical bar shows average CPU utilization between 2 and 5 PM, rather than a single instantaneous sample. This 3-hour high-use period was selected, rather than 24 hours, to avoid deflating the values with low nighttime traffic. (Note how utilization drops substantially on weekends and holidays). The histogram shows that adding NFS workstations to the NetServer's networks over a three-month span has increased prime-time file processor utilization from 40% to greater than 70%. In early December, when this graph was made, the upward trend in user workload was expected to continue—file processor CPU utilization was projected to reach overload levels (above 90%) by the end of February. To avoid this overload condition, an extra file processor would be added in advance.

4.2 Performance Specifications

Table 3 contains a brief summary of NetServer performance specifications.

Specification	NS 3000	NS 5500
NFS throughput (average NFS mix @ 50 ms response time)	675 NFS IOPS (4 Ethernets)	2,250 NFS IOPS (8 Ethernets)
Peak storage-processor aggregate channel rate	25 MB/s (5 channels)	50 MB/s (10 channels)
Peak single-sector disk-array data transfer rate	20.5 MB/s (5 drives)	41 MB/s (10 drives)
Sustained sequential-access disk-array throughput	17.4 MB/s (5 drives)	35 MB/s (10 drives)
Random-access disk-array throughput (8-KB *disk* reads & writes)	510 Disk IOPS (10 drives)	3,060 Disk IOPS (60 drives)
Aggregate IP routing rate between all Ethernets	3,000 pkts/sec (4 Ethernets)	6,000 pkts/sec (8 Ethernets)
Enhanced VME backplane block transfer rate	55 MB/s	55 MB/s

Table 3: NetServer performance specifications. The NS 3000 uses 5 (of 10) SCSI channels on one storage processor and has a 4-Ethernet limit; the NS 5500 uses 30 channels on three storage processors and has an 8-Ethernet limit. These factors account for the general performance factor of two in the specifications. NS 5500 measurements are for Auspex NetServer software release 1.4.1ß. The average NFS operation mixture used to compute throughput is described in the caption accompanying figure 12.

Such a thorough view is especially important in a functional multiprocessor, where there is considerably more latitude for network, file, disk, and processor cache adjustment and load balancing than in conventional uniprocessor servers.

The NetServer's performance monitor tool has an extensive view into each NetServer processor type and provides both real-time display and fast-motion replays of significant server performance events. Figure 10 shows a sample performance monitor summary screen. The event-generated statistics include: per-processor CPU utilizations; network and disk I/O rates and I/O types; file system operation loads, mixes, and rates; and individual cache hit rates and age distributions for the network, file, and disk subsystems. The performance monitor's results are reported in two-dimensional windows at user-selected update times. The results can also be written to log files over a long time base for trend analysis.

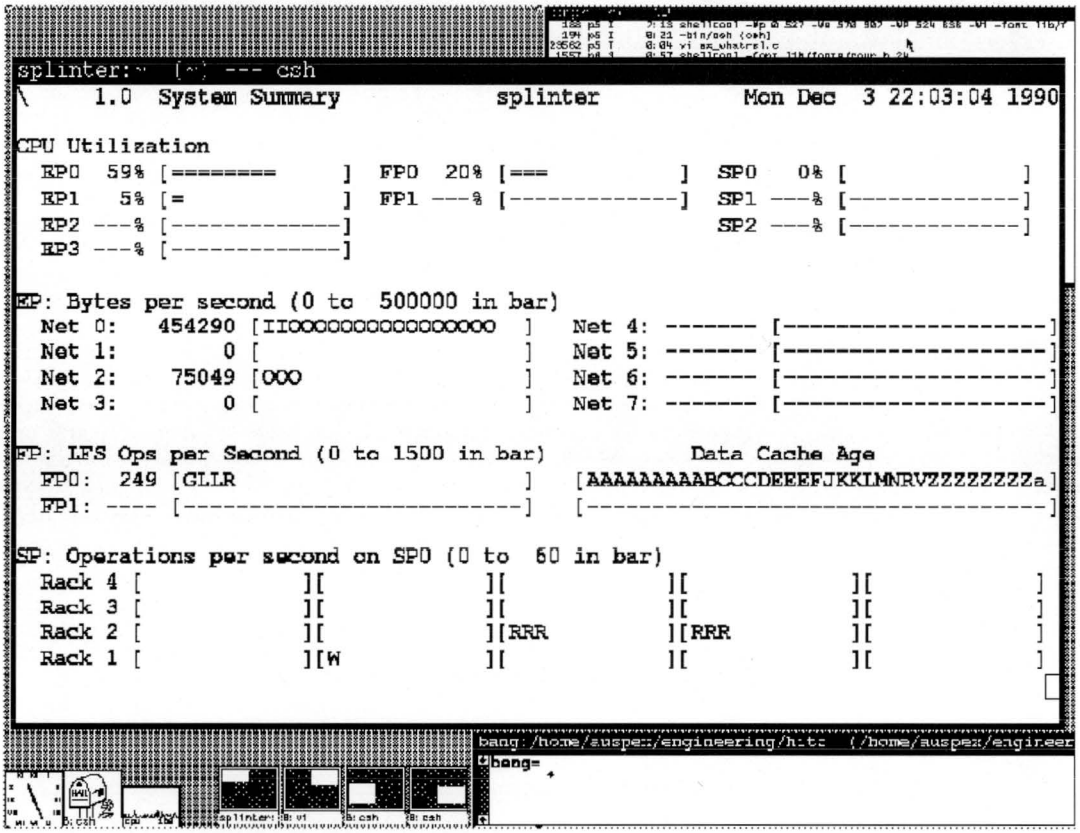

Figure 10: The Performance Monitor's *System Summary* screen. This screen summarizes the most important performance information about a NetServer. The *CPU Utilization* section shows that at 59% busy, EP0 is the most heavily used processor, and at 454 KB/s, Net 0 is the busiest Ethernet. The FP is handling 249 NFS IOPS ("LFS Ops per Second"). Only three disks are currently being used, and none heavily: *ad1* for writes; *ad7* and *ad8* for reads. The performance monitor has five additional, detailed screens covering the Ethernet, file, and storage processors as well as virtual partition and write accelerator activity.

Capacity planners routinely use the performance monitor to automatically log peak-hour performance statistics over long time spans. When this data is displayed in histogram form, as in figure 11, trends become apparent. Existing resources can be rebalanced, or, as the figure illustrates, a NetServer hardware upgrade can be predicted and accomplished well before increased user load demands immediate action.

The performance monitor allows system administrators to accomplish tuning that has heretofore been extremely difficult in Unix (as opposed to proprietary minicomputers or mainframe) environments. It encourages maximum use of existing hardware *before* any upgrade purchases. Hitz discusses the performance monitor in depth [Hitz91].

Unix vendors occasionally combine both intelligent network and disk controllers into an otherwise traditional server. To appreciate why this combined-controller approach typically fails to match the NFS performance of Auspex's FMP architecture, consider these points:

- NetServer Ethernet, file, and storage processors execute in a highly autonomous fashion. They communicate (through I/O cache memory or directly over the enhanced VME bus) without involving the host processor or its private data/instruction memory. In the combined-controller approach, the central CPU remains very much involved (e.g., in transferring data into the one and only sharable main memory).
- On NetServers, the speed of the NFS datapath—the VME backplane—linking the processors has been increased substantially to 55 MB/s. This gives both higher throughput and—equally significantly—reduced latency. In sharp contrast to the NetServer's *flat* organization of peers on a common bus, implementations of the combined-controller approach (e.g., accelerator-equipped Sun SPARCservers) employ multiple buses arranged in a *hierarchy*—throughput is throttled by the slowest bus in the chain.
- In the combined-controller approach, Unix retains responsibility for some NFS, some I/O caching, and *all* file system processing—the central CPU uses some of its compute cycles to satisfy NFS requests. While providing limited performance improvement over a purely conventional approach, intelligent controllers do not shield NFS clients from the performance variation and reduced uptime of a Unix CPU running a varied user-job mixture.

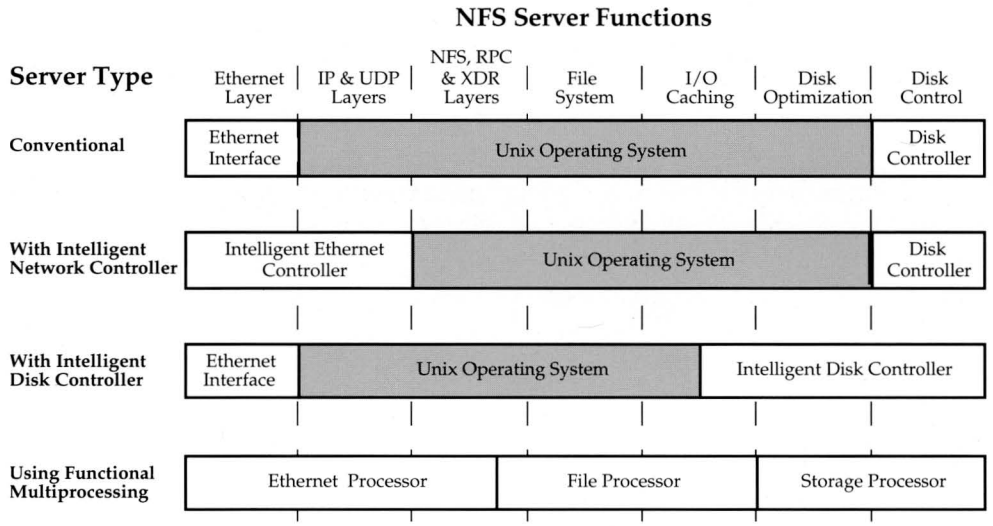

Figure 9: A functional view of file server evolution. The horizontal axis indicates important network, file, and disk storage functions. The vertical axis shows server architectural type. First, at the top, *conventional* servers (retrofitted workstations) use Unix for all but the lowest-level device control functions. Second, servers with *intelligent network controllers* offload some of the lower-level protocol processing from Unix to the network controller. Third, servers with *intelligent disk controllers* remove all disk and often some caching functions from Unix onto the controller. However, standard NFS benchmarks show that one popular disk write-caching scheme actually *lowers* performance by burdening the central Unix CPU with non-DMA cache-to-disk data transfers. Finally, *functional multiprocessing* servers remove *all* network, file, and storage functions from Unix. In the NetServer, each of these functions is handled by a 32-bit microprocessor.

4 Performance Tuning, Specifications, and Evaluation

4.1 Performance Tuning

Since NetServers support as many as ten processors, eight Ethernets, and 60 disk drives in a single system, a concise display of server I/O performance data is vital for system managers and administrators. NetServers come with a performance monitor that enables detailed investigation of site- and application-specific server performance behavior. This is possible because the monitor gives a clear and precise view of statistics not normally available to system administrators, and rarely viewed even by system programmers.

- *Write caching.* Not having to wait for completion of a physical disk write cuts the write operation from 20 ms to under 1 ms (see figure 8).
- *Data coalescing.* As mentioned above, NFS fractures a single client's write operations into successive 8-KB blocks. When several clients likewise access the same file system, the data presented to the storage processor takes on a random, not sequential, traffic pattern. By coalescing the 8-KB blocks in the write cache, the accelerator can reorder the eventual disk writes to be physically more sequential, thus greatly reducing disk arm movement and wasted time.
- *Inode collapsing.* Repeatedly rewritten metadata (inodes and indirect blocks) of frequently accessed files are long-lived in the cache. During that time, the cache accumulates metadata updates that otherwise would be *individually* written to the disk drive. Inode collapsing is particularly worthwhile when data is written to the end of a file with an *append* write, because each new append would normally write an inode to disk. Instead, the inode update is caught in the write cache—avoiding a disk operation. About 90% of all Unix file write operations are append writes, so inode collapsing is truly valuable.

These write accelerator performance benefits are user-transparent. For an average NFS operation mix, the system delivers 20% more throughput and 50% faster response time. This means that typical write-intensive applications complete 3–5 times faster. The write accelerator is easy to administer—write acceleration is selected on a per–file system, per–storage processor basis and its behavior is indicated in one of the performance monitor's display screens (see section 4.1).

Auspex's write accelerator implementation is a logical extension of FMP. Performance-enhancing hardware and software is positioned close to the resources (disks in this case) where they have the greatest impact, not in the Unix host processor (or its coprocessor). An ancillary benefit of the NetServer's storage processor–based implementation is disk write acceleration for *non*-NFS applications such as FTP, dump and restore, NIS updates, and sendmail.

At a conceptual level, the write accelerator is similar to write accelerator boards commonly retrofitted into conventional servers—Prestoserve is a classic example [Lyon89]. In comparing the two, observe:

- FMP write acceleration wastes no I/O bandwidth—a single VME transfer into the storage processor is all that's required. Prestoserve requires an extra bus round-trip (in SPARCservers, either over the SBus or the VME bus)—once to encache the data, once to retrieve it back to the CPU, and then to disk. This is three trips, not the NetServer's one.
- Write acceleration adds no Unix CPU driver overhead. All Prestoserve activity requires the intervention of the Unix CPU, and usually uses the CPU as a very expensive 4-byte by 4-byte DMA engine.
- The write accelerator daughter board consumes no additional VME I/O slots. Most Prestoserve implementations do (in the SPARCserver's case, a VME or SBus slot).
- Write acceleration scales with increases in storage (i.e., one per storage processor). Exactly one Prestoserve may appear in a conventional server.
- Write acceleration is compatible with all features critical for NFS. Prestoserve is incompatible with Online: DiskSuite features such as mirroring and concatenation.

3.5 Architecture Evolution Review

The NetServer streamlines NFS by explicitly eliminating Unix from all network, file, and storage processing and executing these functions on specialized processors. This contrasts with conventional server design, where all NFS software executes as a part of the Unix operating system on a traditional CPU platform. This difference is graphically depicted in figure 9, which shows several evolutionary steps of network server architectures. The seven server software functions play significant roles in processing NFS requests. Note how each server's reliance on Unix (gray) decreases as server power increases from top to bottom.

- *The functional multiprocessing kernel (FMK).* Dividing a network server's tasks among multiple peer processors creates the need for these processors to communicate. Recall that the NetServer's I/O cache is exclusively dedicated to NFS data and transient IP packets—not instructions. Consequently, I/O and host processors communicate directly via messages passed over the enhanced VME bus. Message passing is accomplished using a small, non-Unix microkernel—called the functional multiprocessing kernel, or FMK—that runs in each NetServer physical processor. As well as supplying message services, the FMK provides a low-overhead process mechanism, physical memory management, and other basic services. The FMK kernel is described by Hitz in a separate Usenix conference paper [Hitz90].
- *The virtual partition storage manager.* A *virtual partition* is an elaboration of Unix's traditional physical disk partition. Virtual partitions permit the *concatenation, striping,* or *mirroring* of disk data. These capabilities can be elected singly or in combination and apply to both file systems and databases. Concatenation permits partitions (and thus file systems and databases) to be larger than a physical disk, up to Unix's current 2-GB limit. Striping interleaves disk blocks across multiple physical disks, easily alleviating *hot spots* in busy file systems or databases. Mirroring duplicates file systems or databases onto two striped or concatenated virtual partitions on two sets of disks, permitting transparent recovery from drive or media failure (in conjunction with hot pluggability). Implementing virtual partitions on the storage processor, as close as possible to the disks themselves, injects lower overhead than conventional kernel-driver-level approaches and allows benefits to be applied to all disk processing, not just NFS. This is a cursory description of virtual partitions; they are described more deeply in a separate report [Cheng91].

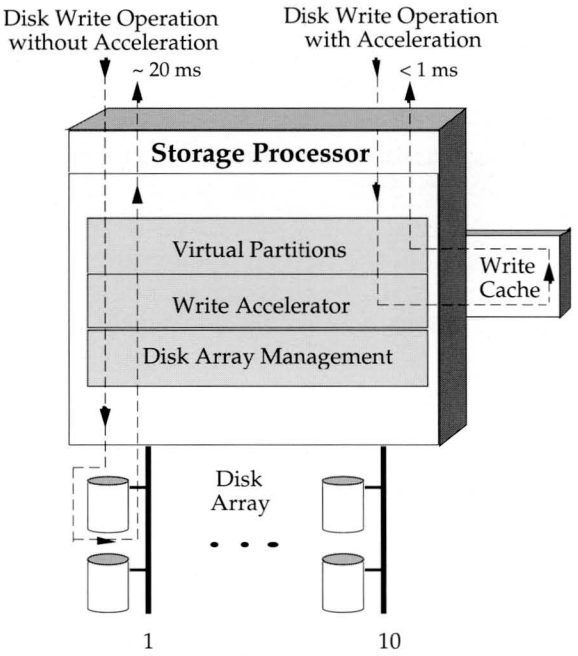

Figure 8: Write accelerator operation. The write accelerator achieves a 20 *x* speedup for all write operations for which acceleration has been selected. For NFS, these would include *Write, SetAttributes, Create, Remove,* and *MakeDirectory.* Host-processor functions that benefit include FTP, dump and restore, NIS updates, sendmail, and database access. The write-accelerator firmware is interposed between the storage processor's virtual-partition and disk-array management layers. It controls 1 MB of nonvolatile memory on the *write cache* daughter board. Each storage processor can have its own, independent write cache. Thus performance scales as the disk configuration grows beyond the initial 10 SCSI channels. The daughter board is removable, and can be switched to a new storage processor (to recover cached data) should an SP break.

- *Write accelerator.* NFS normally requires that all writes be *synchronous*, i.e., completed to a stable storage medium before sending an acknowledgment to a client. Servers that routinely perform *asynchronous* writes sacrifice data integrity for the sake of performance. The NetServer write accelerator provides asynchronous performance with synchronous data integrity. It accomplishes this in three ways:

Considering dynamic behavior, nearly all network traffic consists of NFS requests. The NFS *control path* follows the thick, dashed-line paths in figure 7. FMK messages flow between the Ethernet processor and the file processor, and—if disk activity is required (e.g., an NFS read of a file block not yet in I/O cache)—also between the file processor and the appropriate storage processor. This is subtly different from the *data path* (shown in figure 7 by medium dashed lines), which does not include the file processor itself. In an NFS read of a block not yet in I/O cache, the storage processor DMA-transfers the block directly into an I/O cache location assigned by the file processor. The Ethernet processor then DMA-transfers the block and, IP-fragmenting it as necessary, sends it on to the requesting client.

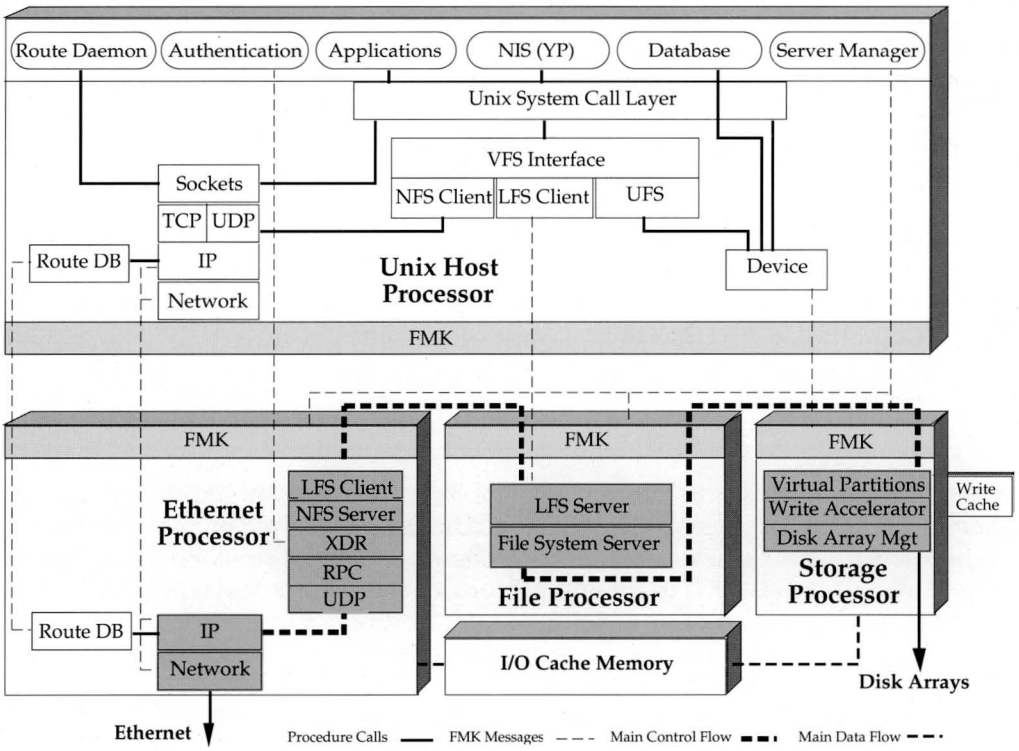

Figure 7: The NetServer system software architecture. The NetServer's FMK multiprocessing kernel (light gray) executes on each hardware processor (large boxes). Solid connecting lines represent control paths local to each processor. Light dashed lines denote remote control paths that use FMK interprocess communication. While the software structure is complex, the optimized NFS path is simple. This control path is indicated with heavy dashed lines through the Ethernet, file, and storage processors. The software modules involved are shaded in medium gray. Database processing on the host processor is accelerated by providing direct device-level access to virtual partitions in the storage processor. Disk write speed is increased by the write accelerator. Note the I/O cache memory's direct datapath between the Ethernet and storage processors (medium dashed lines).

A detailed discussion of the principal software components in figure 7 is beyond the scope of this report. Most of these elements are found in all networked Unix systems, but there are two novel components intrinsic to the FMP architecture: the LFS *local file system* and the FMK *functional multiprocessing kernel.* In addition, the NetServer's *virtual partition* disk storage manager eases large-capacity file server administration, and the *write accelerator* speeds disk write operations.

- *The local file system (LFS) interface.* For efficient file-system processing within the NetServer, Auspex designed a new internal file system interface, called the local file system or LFS. LFS is used for file communication between Ethernet processor(s) and the file processor(s), and between the host processor and the file processor(s), as shown in figure 7. The set of LFS operations is a superset of NFS operations, and these common operations share many semantic properties. LFS has been a strong unifying influence on the NetServer software architecture. But it is important to note that LFS is used only *within* the system; the external network protocol is precisely standard NFS. LFS is described in a separate IEEE conference paper [Schwartz90].

The NS 5500 can have up to three storage processors (supporting up to 60 SCSI devices); the NS 3000 has one (supporting up to 10 SCSI devices).

- *Write cache.* The NetServer's *write cache* is an optional, battery-backed, 1-MB nonvolatile daughter board that attaches to the storage processor—it uses no extra VME slots. It provides excellent, zero-latency write performance with absolute data integrity. The daughter board is removable, and can be switched to a new storage processor (to recover cached data) should a storage processor break. Section 3.4 describes the *write accelerator* software and firmware that executes on the storage processor.
- *Disk arrays.* The NetServer's disk arrays are organized in racks of five, 2.003-GB (formatted), 5,400-RPM, 11.5-ms average access-time, 5-1/4-inch form-factor drives. These 2.0-GB drives perform sector-to-sector data transfers at 4.1 MB/s, sustained cylinder-to-cylinder data transfers at 3.5 MB/s, and perform 51 random 8-KB I/Os/s. Disks—as well as tapes and CD-ROMs—can be inserted and removed *while the system is powered-up and running Unix* because they are hot-pluggable. Up to four racks, or 20 drives, can be attached to a single storage processor, so at most two drives are attached to one SCSI channel. Each SCSI drive has its own internal dual-microprocessor controller and 256-KB track buffer for autonomous, concurrent operation and data transfer. Disk-array storage organization is vital in high-performance NFS servers because these servers receive independent, largely random, 8-KB I/O requests from their 40–200 active workstations. Individual client workstations may exhibit sequential read and write behavior, but in aggregate the traffic will appear random and independent at the server. The crucial NFS performance role played by the NetServer's storage processor and SCSI disk arrays is described by Cheng and Nelson in a separate Usenix paper [Nelson92].

  The NS 5500 supports 60 disks x 2.0 GB = 120 GB maximum; the NS 3000 supports 10 x 2.0 = 20 GB.
- *Host processor.* The host processor is either a Sun-3 or Sun-4 architecture CPU board, and runs Unix (licensed SunOS). Auspex's NetServers are therefore compliant with *both* of Sun's application binary interfaces, and users typically select a host processor based on their desired binary environment. The Sun-3 architecture host processor is based on a Motorola 68020 board design licensed from Sun. The SPARC-based Sun-4 host processor provides increased compute capability and is a straightforward board-swap upgrade from the initial Sun-3 board. As the Ethernet, file, and storage processors are collectively responsible for satisfying NFS requests, the host processor is free to take on other functions. It is responsible for: system booting; ONC functions such as the automounter, Network Information System (NIS), and lock manager; system error reporting and logging; performance monitor event collection (section 4.1); SNMP agent coordination; backup and restore; and virtual partition administration (section 3.4). Even under heavy NFS loads, only a small fraction of the host processor's power is consumed by the above administrative functions. Consequently, a large and *predictable* fraction of the host processor's CPU remains available for services more compute-intensive than NFS, such as database processing. This balanced behavior is in sharp contrast to so-called general purpose architectures, wherein NFS performance is achieved at the expense of application processing (and conversely). Section 4.4 discusses this further.
- *I/O cache (primary) memory.* I/O cache memory is essentially a huge disk buffer cache. There is also transient buffering for IP packets being routed between different Ethernet processors (and future FDDI processors). There are no processor instructions stored in or fetched from this memory—the entire memory (and backplane) bandwidth of 55 MB/s is devoted to I/O. Memory can be configured from 16 to 96 MB in 16-MB increments with standard ECC redundancy.
- *Enhanced VME backplane.* The NetServer's backplane, using an enhanced VME bus protocol, performs block transfers at 55 MB/s. It is electrically compatible with the standard VME interface, though substantially faster. This increased speed provides both improved throughput and sharply reduced latency.

3.4 FMP Software Organization

The NetServer's software organization derives quite naturally from the functional multiprocessor partitioning described above. This static structure is shown in figure 7. The adjective "static" is important here because figure 7 indicates all functional relationships without any weighting for performance.

its resources are managed by a coarse-grained Unix kernel (i.e., whenever a single CPU enters the kernel, other CPUs are locked out of the kernel for relatively long periods).

- *Limited VME bus bandwidth.* Traditional Unix servers emphasize a fast *memory bus*, but not a high-throughput *I/O bus* that supports many concurrent networks and disks; data *and instructions* must share the same bus; hierarchical bus arrangements can further separate the network and storage (e.g., Ethernet and disk) resources that an I/O server must closely couple—increasing latency.

Attempting to address these bottlenecks, some vendors recommend the addition of higher performance, microprocessor-based, caching disk or network controllers to their systems. These are incremental steps toward a server with balanced disk and network throughput. But section 4 shows that these enhancements fall short of FMP's capabilities.

3.3 FMP Hardware Organization

Auspex's FMP architecture distributes performance-limiting I/O functions to multiple, dedicated processors. There are four processor types. As in symmetric multiprocessor systems, each FMP processor incorporates a 32-bit microprocessor and has multiple internal buses and memories. By storing processor-specific instructions and data in local memories, FMP dramatically reduces bus traffic and completely avoids the classic SMP cache-miss bus-contention problem. A base NetServer configuration includes one each of the four processor types, along with I/O cache memory and a rack for the first five SCSI devices. Here are further details, elaborating on figure 5:

- *Ethernet processor.* The Ethernet processor performs full protocol processing up to and including the NFS level by executing a licensed, ported copy of Sun's reference NFS release (the actual definition of the NFS standard). This processing includes IP, UDP, RPC, XDR, and NFS. The Ethernet processor passes the resulting NFS requests directly to the file processor through the LFS local file system interface, described in section 3.4. Each Ethernet processor supports two parallel Ethernets. Complete IP routing is performed between all connected networks without intervention by the host processor. Each Ethernet processor also contains an SNMP network-management agent that is coordinated by the host processor.

  The NS 5500 supports four Ethernet processors; the NS 3000 supports two.

- *File processor.* The file processor operates in parallel with Ethernet processors and runs the Berkeley 4.3 Unix Tahoe "fat" file system that has been totally excised from the Unix kernel. The binary representation of file systems on NetServer disks is identical to the representation in other 4.3 BSD (or SunOS) systems. NFS operations are processed directly by the Ethernet and file processors, bypassing the Unix operating system and thereby significantly reducing software overhead. The file system's buffer cache is partitioned into two pieces:
  - *User data* is kept in I/O cache memory, where it is directly accessible by the storage and Ethernet processors for rapid delivery to workstations.
  - *Metadata*—file system structural information like directories, inodes, and indirect blocks—is cached in the file processor's local memory for instant searches and low VME bus contention.

  The NS 5500 can have one or two file processors; the NS 3000 has one.

- *Storage processor.* The NetServer storage processor operates ten parallel, 5-MB/s, SCSI-1 I/O channels simultaneously. These channels are attached to disk arrays (described below). While SCSI disks were once maligned for poor performance in the personal computer and workstation communities, this was an artifact of drive and controller implementation, not of the SCSI standard. The NetServer storage processor, in concert with SCSI disk arrays, offers NFS performance that greatly exceeds that of conventional SMD and IPI disk organizations [Nelson92]. The storage processor is responsible for virtual partition management [Cheng91], write acceleration (described later), SCSI channel management, SCSI disk and tape device control, disk seek optimization, and DMA data transfers to I/O cache memory. For backup, SCSI tape options include 4- and 8-mm helical, 1/2-inch streaming, and 1/4-inch cartridge drives. CD-ROM drives are also supported—Auspex distributes its software releases on CD-ROM.

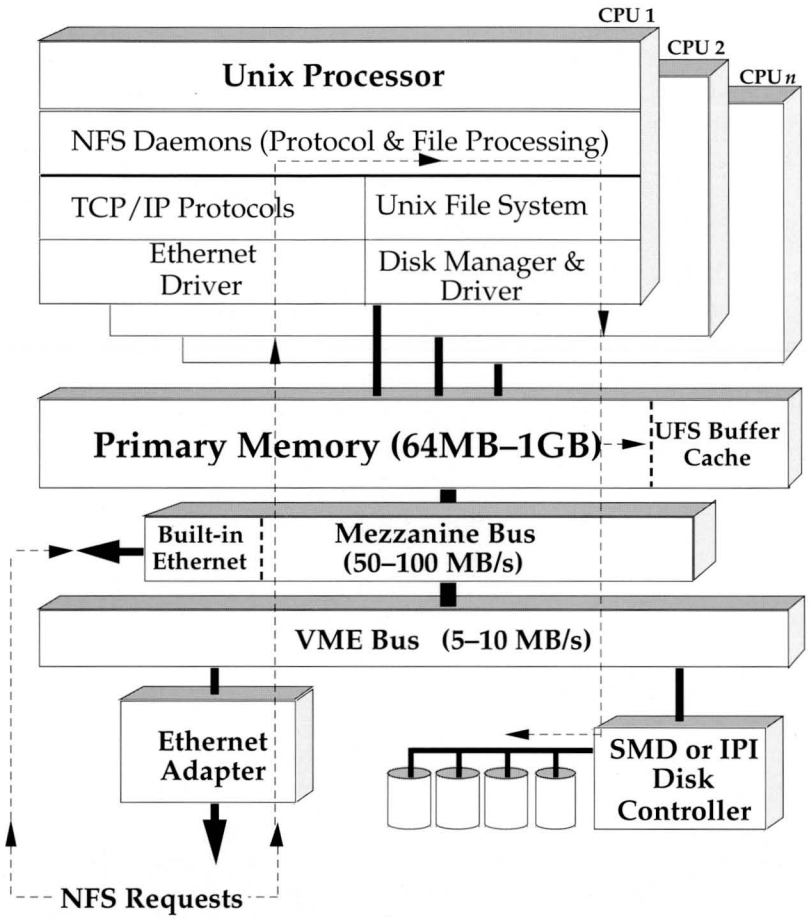

Figure 6: Conventional SMP file server architecture (compare with figure 5). Conventional servers share the execution of server functions with the Unix operating system. Frequently, insufficient kernel lock granularity throttles an SMP design, applying the processing power of just one CPU to NFS. The dashed line indicates the basic flow of an NFS request through the system. All network, file system, and most storage functions are performed by the Unix processor. Today, most of these processor boards—especially those using the latest RISC designs—are optimized for instruction fetch and execution, not I/O latency and bandwidth. Furthermore, many conventional file servers use SMD- or IPI-based disk subsystems. These compute-server-oriented disk technologies are decidedly inappropriate for high-throughput NFS file service [Nelson92]. These two conventional-server design decisions have exacerbated the I/O performance gap.

3.2 Conventional Server Architectures and NFS Bottlenecks

Many network servers today are symmetric multiprocessor systems, based as they are on workstation designs. Their conventional server architecture—illustrated in figure 6—offers a stark architectural contrast to the FMP system organization in figure 5. In a conventional SMP server, all network, NFS, and most of the disk processing takes place in the Unix host processor(s). The potential NFS and I/O bottlenecks in traditional uniprocessor servers—and many multiprocessor servers—include:

- *Limited network throughput and capacity.* Only a few Ethernets supported per server; single Ethernets have bandwidth of just 10 Mb/s (about 1 MB/s); and unintelligent Ethernet adapters waste Unix-CPU cycles.
- *Limited disk subsystem throughput and capacity.* Compute-server-oriented disk subsystems are often optimized for sequential, not random (multiclient NFS) access; insufficient or over-configured disk channels result in limited disk access and transfer concurrency; and servers support too little aggregate disk storage.
- *Unix operating system overhead.* User-job contention slows file system operations; protocol processing suffers the slow context switching of a Unix CPU; file data goes through Unix virtual memory unnecessarily; the full processing potential of an SMP hardware design is not realized when

3.1.1 Symmetric Multiprocessing

In an SMP architecture, each processor is a compute engine that usually executes all tasks, including kernel I/O processing. The goal of the operating system is to enable *n* processors to deliver as close to *n* times the throughput of a single processor as possible. It is a relatively straightforward task to achieve this multiplier effect for jobs executing *mostly in user* mode and *rarely in kernel* mode. A mature, optimized SMP implementation distinguishes itself by achieving a high degree of concurrency for *all* jobs, including especially those making regular use of the kernel. When such jobs are run on less sophisticated SMP implementations, the result is low individual CPU utilization and nonlinear scaling, rather than the anticipated *n*-fold aggregate throughput. Here are some performance optimizations typically found in the best SMP systems:

- An extremely high-bandwidth, cache-coherent memory system that can both feed instruction-hungry CPUs *and* pump large I/O blocks between the CPUs, the I/O bus hierarchy, and I/O devices.
- Fast hardware vectoring of interrupts to any available CPU, to provide low latencies and effective queued-interrupt service.
- Fine-grained multithreading in the kernel—including file system, protocol stacks, and device drivers—for maximum execution overlap.
- A high-speed I/O bus (or bus hierarchy) connecting the CPUs to all networks and peripherals, so that each CPU may be applied equally well to all clients and devices.

Even if these enhancements appear in an SMP design, a single task cannot execute any faster than the speed of a single processor. The theoretical best for *n*-way SMP is a linear increase in overall throughput (beyond what might be achieved with *n* conventional uniprocessors). Applications most readily benefit from even an un-optimized SMP implementation if and only if the applications are compute-oriented, receive few interrupts, and make few kernel calls. Neither NFS nor DBMS service have these characteristics. For a data server to enjoy the potential of an SMP design, a high degree of symmetry—though difficult to achieve—is mandatory.

The design decision to have NetServers use FMP stems in part from the realization that NFS execution overlap is greater and more easily achieved in an FMP design. In addition, since the NetServer's Ethernet, file, and storage processors are not obliged to perform general purpose Unix computing, they use simpler, much less expensive chip sets to accomplish their specific tasks.

By their symmetric nature, SMPs typically do not have special-purpose, processor-specific hardware to accelerate I/O. In a network server whose mission is to deliver data between disks and networks—not merely from local disk to local memory for processing by the Unix CPUs—special hardware I/O architecture and acceleration work superbly. An SMP network server will not easily achieve the higher throughput and lower latency of an FMP network server because most SMP servers lack a tightly integrated, high-bandwidth I/O bus and non-virtual I/O memory; special protocol-processing hardware; and NFS-optimized storage hardware. Furthermore, without FMP-style software partitioning, Unix adds significant overhead to NFS processing in an SMP server.

3.1.2 Master-Slave Multiprocessing

One kind of asymmetric multiprocessor, usually called a *master-slave* multiprocessor, has identical CPUs that all run user jobs. However, only one processor executes most kernel tasks, including I/O. These master-slave multiprocessors can easily bottleneck when executing simultaneous I/O-intensive user jobs—or when acting as network servers—because they must funnel all I/O through a single CPU. When a multiprocessor server lacks features to concurrently load and fully utilize *all* participating CPUs, its performance degrades to the level of a master-slave design.

To date, theory has been proven in practice. In benchmarking their SMP NFS servers, no vendor (e.g., Silicon Graphics, Solbourne, Sun) has documented performance approaching the NFS IOPS levels achieved by NetServers.

In summary, Auspex's design goal was to develop a network file server that could plug directly and transparently into existing NFS-based networks. It would replace five or more conventional servers, attaching to all of their existing networks and replacing their cumulative disk storage. The goal was to design a server capable of supporting the type of network configuration illustrated in figure 4.

3 Functional Multiprocessor Architecture

The NetServer achieves its five design objectives with a unique *functional multiprocessing* (FMP) architecture that maximizes efficiency of the data paths between client workstations and the NetServer's network interfaces, file system, and disk storage subsystem. As shown in figure 5, the FMP architecture achieves its performance advantage with highly intelligent processors that operate in parallel to optimize NFS throughput. A key FMP notion is that the Ethernet, file, and storage processors *do not run Unix.* This cuts unnecessary software overhead to a minimum.

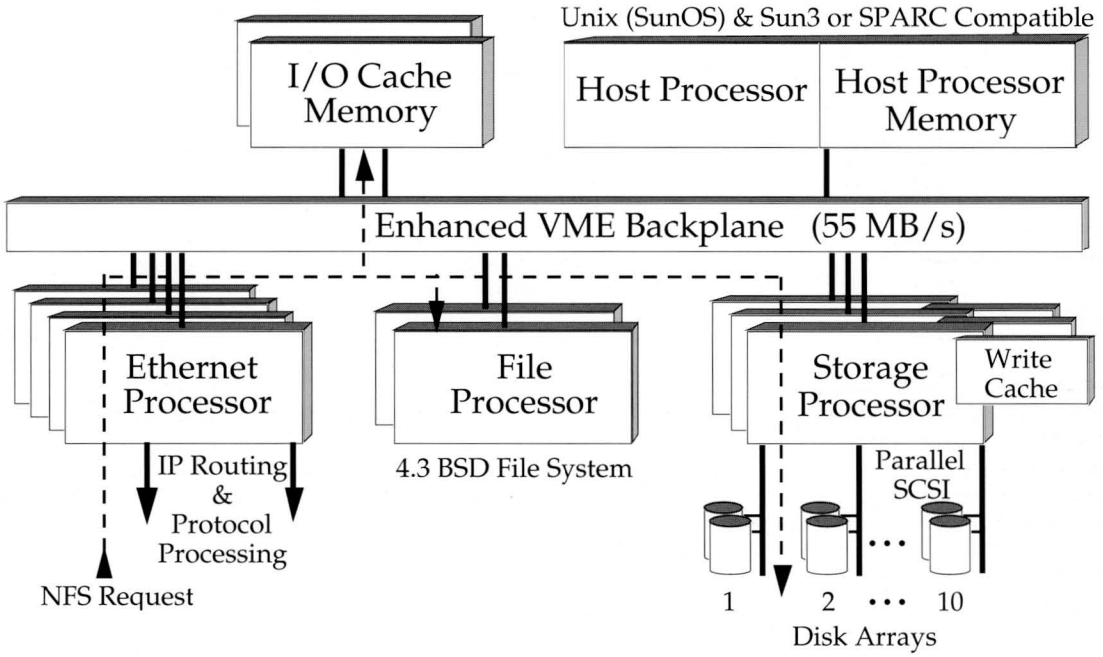

Figure 5: The NetServer functional multiprocessing I/O architecture. All network, file system, and storage processing is completely removed from the Unix host processor and performed instead by intelligent, dedicated processors. These non-Unix processors use 32-bit microprocessors with local instruction memories, optimized ASIC-driven datapaths, and a large I/O cache primary memory to achieve exceptional performance. Disk arrays are discussed in section 3.3; one notable feature is that SCSI devices are *hot pluggable*—i.e., CD-ROMs, disks and tapes can be replaced while NFS and Unix are running. Each storage processor is optionally equipped with a write cache, which reduces write latency and improves overall throughput by deferring, coalescing, and collapsing disk writes. A full NetServer is a 10-processor system: 4 Ethernet processors (8 Ethernets), 2 file processors, 3 storage processors (60 disks), and 1 host processor. There are two choices for the Unix host processor: a SPARC-based Sun-4 architecture CPU, or a Motorola 68020–based Sun-3 architecture CPU. Auspex's NetServers are thus compliant with *both* of Sun's application binary interfaces. The dashed line shows the basic data flow of an NFS read request through the system. The complete control flow is shown later in figure 7.

Because there are several kinds of multiprocessor architectures, a brief comparison of multiprocessor methods will be valuable before describing the NetServer's individual functional processors.

3.1 Symmetric, Asymmetric, and Functional Multiprocessing

The FMP architecture is a new architecture for optimized network servers. The FMP I/O subsystem is a collection of loosely-coupled peer processors, where each processor type contains different hardware and runs specific, tailored software. FMP is a strong counterpoint to symmetric multiprocessor architectures (SMP), which are becoming common in compute servers, minicomputers, and some conventional file servers.

Between Failure (MTBF) goal to 5,000 hours. Data is collected on a continuing basis (from over 450 production systems at the time of this edition) to confirm that this goal is regularly exceeded.

- *Software compatibility.* For comprehensive software compatibility, the NetServer needed to be compliant with NFS, other Open Network Computing (ONC) services such as NIS, Automounter, and SunOS Unix itself.

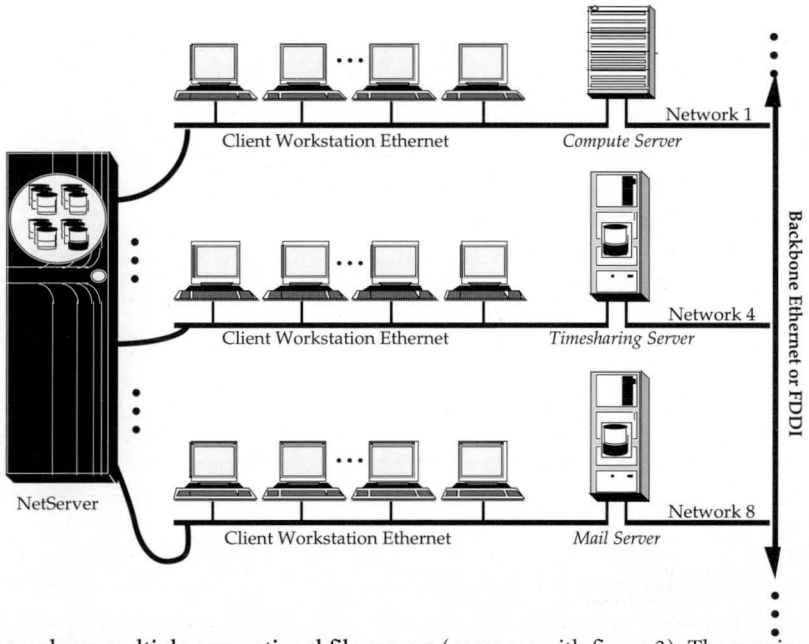

Figure 4: The NetServer replaces multiple conventional file servers (compare with figure 3). The previous servers are not retired but are given new tasks—in this case, as a compute server, a timesharing server, and an electronic mail server. While the NetServer now routes IP packets between all of the client networks, the existing backbone network is retained for redundancy. Previously replicated common files are now consolidated on the NetServer. NFS crossmounting is eliminated. This diagram shows only one conventional server per network, but it is common to find multiple servers per network in environments where partitioning and subnetting are difficult. This flat-network approach often overloads the network by increasing Ethernet utilization to overload levels.

- *Performance.* In 1987, Auspex designers set the target performance level for their first server to be 1,000 NFS 8-KB *read* IOPS. Of the 22 NFS operations defined, NFS *read* operations usually occur less than 25% of the time, but Auspex's performance goal was made more difficult by restricting the load to NFS *read* operations. This is a difficult, data-intensive goal for I/O hardware because it requires driving 8 Ethernets at 90% utilization each, achieving a total transfer rate of 72 Mb/s. Of course, excellent performance on the remaining NFS operations is important as well, and our NFS benchmark tests use an average NFS operation mixture (see section 4.3).

  Auspex picked 1,000 NFS read IOPS because we expected the NetServer would be about 5–10 times faster than any other server at the time of first shipment in 1990. The two assumptions behind this expectation were: (1) that normal product evolution would double the performance of comparable high-end servers to 100 IOPS (this proved to be an accurate prediction); and (2) a generous error factor of 2. Thus $1000 \div (100 \times 2)$ yielded our factor of 5–10. In practice, the NetServer's actual peak performance is 2,250 *average*-NFS-mixture IOPS (with the 1.4.1ß software release). In tests involving only NFS *reads* (as compared with an average operation mix), it has surpassed the 1,000 IOPS goal as well. Performance is discussed further in section 4.

- *Hardware compatibility.* To reduce unnecessary engineering and for hardware compatibility with the existing VME board market, the NetServer backplane needed to support VME boards in a completely compatible manner, at the same time providing the increased bandwidth needed for intensive, responsive I/O. This permits customers to select and use third-party hardware options—available in the de facto–standard VME format—when appropriate.

- *Administration.* System administrators must maintain many limited-capacity servers rather than a few powerful servers. This burden encompasses network administration, free-disk-space management, hardware maintenance, and user account administration.
- *File system backup.* System administrators or operators must conduct multiple file system backups, a time consuming task. It is also expensive to duplicate backup peripherals on each server (or every few servers if slower network backup is used).
- *Price per workstation seat.* With mid-range NFS-accelerated servers costing about $70,000 (for 1.3-GB secondary storage, 64-MB primary memory, NFS write cache, single Ethernet accelerator, and a 50-MIPS processor), the real cost of an entry-level Unix workstation is approximately 70% *more* than its apparent cost—a substantial increase. The straightforward assumptions behind this 70% calculation are that entry-level workstations cost about $5,000 and the client to server ratio is 20:1. Thus each client's share of a server is $3,500, giving a $5,000 + ($70,000 ÷ 20) = $8,500 price per workstation seat, or 70% more than the $5,000 workstation-only price. When client applications are more I/O-intense, system administrators prefer to see a more conservative 10:1 client to Ethernet ratio, in which case the per-workstation cost increase is 140%.

1.5 The Need for I/O Innovation

The widening I/O performance gap—in conjunction with the administrative and economic considerations of section 1.4—demonstrates a real need for higher-performance, larger-capacity Unix file servers. The Unix workstation vendors' traditional approach to server design—repackaging a workstation in a rack as a retrofitted compute server—is inadequate for large networks at current client performance levels.

Conversion of a display-less workstation into a server may address disk capacity issues, but it does nothing to address fundamental I/O limitations. As an NFS server, the retrofitted workstation must sustain 10–20 or more times the network, disk, backplane, and file system *throughput* than it was designed to support as a client. The addition of larger disks, more network controllers, extra primary memory, or even a faster processor does not resolve basic architectural I/O constraints. They do not substantially increase overall I/O throughput.

Closing the Unix I/O performance gap requires a new, server-specific computer architecture that can flexibly balance client data demand with server I/O throughput. The new I/O architecture must scale in throughput and capacity just as easily as client workstations scale in performance. The new architecture must focus on optimizing a Unix network server's most common actions—NFS and disk operations—just as RISC processors focus on optimizing a CPU's most common instructions.

The Auspex network server has such a novel I/O architecture.

2 NetServer Architectural Objectives

Five objectives dictated the architectural design of Auspex's network servers, called NetServers.

- *Scalability.* The server architecture had to support a system that was both flexible and scalable. Flexibility was desired so that FDDI networks, optical storage media, and multi-vendor Unix host processors could be included as appropriate. Scalability was important to accommodate server configurations with not just 2 or 4 networks, but 8, 12, or more networks; not just 5 or 10 disks, but 25, 50, 100, or more disks; and not just one host CPU, but multiple CPUs. A scalable architecture allows and encourages the consolidation of many individual server-per-subnet workgroups onto a single server that is more cost-effective, reliable, and easily administered. Significantly, scalability means more than configurability. To meet the growing NFS demand of an expanding client population, a server's ability to merely attach a large complement of networks, disks, and processors must be implemented with an architecture facilitating the *effective collective utilization* of these resources. The distinction here is between *theoretical* configurability and *practical* configurability.
- *Reliability.* Recognizing that the trend toward larger, consolidated servers must be accompanied by increased server reliability, Auspex set the base configuration (i.e., peripherals excluded) Mean Time

These NFS-specific findings correlate well with independent client-server I/O measurements in non-NFS distributed file systems [Howard88, Nelson88]. These statistics support the 10–20 dataless NFS client-to-Ethernet rule-of-thumb ratio practiced by many system administrators (this ratio depends more on Ethernet bandwidth than on server power per se). Further, this ratio is not increasing in the NFS environment because server I/O bottlenecks continue to throttle faster client processors.

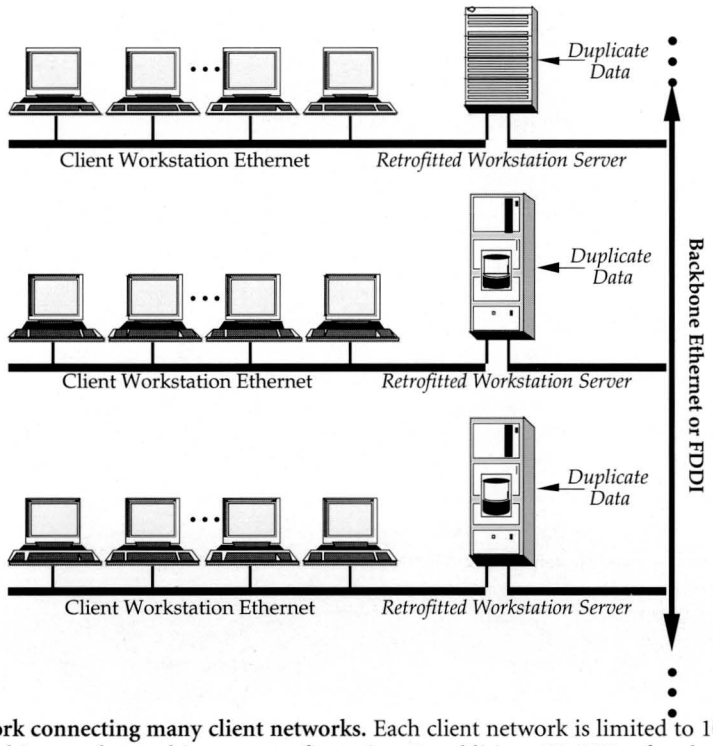

Figure 3: A backbone network connecting many client networks. Each client network is limited to 10–20 workstations by server performance, necessitating this complex multi-server configuration. In addition, 15–30% of each server's disk storage usually contains replicated common files, an expensive redundancy. The network configuration shown, where each server has two Ethernet ports—one for workstations and one for the backbone network—is often called the *server-per-subnet model.* Instead of using servers with dual Ethernet connections as shown, some networks use bridges or IP routers to furnish the backbone connections.

1.3 Complex Network Topologies

Low client-to-Ethernet ratios coupled with limited conventional NFS server power usually results in network configurations like the *server-per-subnet* network shown in figure 3. Each local client-workstation Ethernet isolates traffic for its own 10–20 clients and server (which is usually a workstation or compute server in a file server role). For overall connectivity, these local networks are usually joined together by a backbone network—typically Ethernet today, but occasionally FDDI. This backbone network is connected to the local workstation networks in one of two ways (or, occasionally, both for fault tolerance):

- with a second server network interface (shown in figure 3), or
- with IP routers or bridges (not shown).

1.4 Non-Performance Considerations

Performance aside, low client-to-server ratios create five additional problems:

- *Sharing.* Development groups of more than 10–20 people cannot readily share the same server, and thus cannot easily share files without multi-server file replication and/or NFS crossmounting.
- *Reliability.* A client's dependency on multiple crossmounted file systems reduces the reliability of the NFS service it receives.

File System and Description Workstation Type	root System Booting	swap Virtual Memory Paging	/tmp Temporary Files (never backed up)	/usr Applications, Documents, Etc.	/home /personal Personal files: Mail, News, Etc.	/home /projects Shared Project Files
Diskless	On Server	On Server	On Server	On Server	On Server	On Server
Swapfull	On Server	Local Disk(s)	Local Disk(s)	On Server	On Server	On Server
Dataless	Local Disk(s)	Local Disk(s)	Local Disk(s)	Local Disk(s) or Server	On Server	On Server
Diskfull (Standalone)	Local Disk(s)	Local Disk(s)	Local Disk(s)	Local Disk(s)	Local Disk(s)	Local Disk(s) and Server

Table 1: Client workstation type and file system organization. This table defines workstation type as a function of file system location: on local (usually SCSI) disks attached directly to the workstation, or remotely on one or more NFS servers. A *diskless* workstation mounts all its file systems remotely. A *swapfull* workstation locates only completely volatile swap and /tmp information on local disk. A *dataless* workstation is swapfull with a local root for booting, and optionally /usr as well; data requiring frequent backup is still kept on servers. A *diskfull* workstation is essentially standalone with all files local and requiring local backup; shared files are manually moved to a server as needed.

Workstation Type	Primary Application	Secondary Application	Moderate-I/O NFS Load (IOPS)	Main Advantages	Main Disadvantages
Diskless	Publishing; Secure environments; Low cost/user	Software development; X-windows terminal	15–25	Lowest cost; Easiest administration; Lowest noise; Best reliability	Higher network traffic because of swapping; Lower performance
Swapfull	Software Development	Large publishing; Small CAD	10–20	Up to 50% less network traffic; Almost no administration; Better performance	Disk cost; Disk noise; Disk unreliability
Dataless	Medium CAD, CAE; Large-scale software development	AI; Seismic analysis	8–17	Less network traffic than swapfull; Even better performance; Only shared files on server	Larger disk needed; Each workstation needs individual software updates
Diskfull (Standalone)	Large ECAD, MCAD; Seismic analysis; Visualization	Workstations acting as servers	0–10	No network traffic; No server disk traffic; Optimal workstation response with fast disks	Best disk needed; Each workstation needs backup and updating; Easy for shared files to become inconsistent

Table 2: Typical workstation applications, NFS traffic, and tradeoffs. The primary and secondary applications are typical, not fixed or comprehensive. The moderate-I/O NFS loads correspond to moderate workstation I/O, for example, compiling remote source files with included headers that must be fetched as well. The average NFS load would usually be lower. Diskless workstation load will decrease as primary memory is increased. The advantages and disadvantages columns are considered from several viewpoints: user, system administrator, and purchaser. The NFS load ranges are from first-generation RISC workstations (~25 SPECmarks).

Busy NFS file servers spend 75–100% of their CPU time dealing with NFS traffic. Considering NFS traffic only, traditional servers supply 100–450 NFS I/O operations/second (NFS IOPS) before saturation [Lyon89]. Additional NFS server capacities are presented in section 4. Client workstations, whose NFS traffic arrive at a server in bursts, request a wide range of 8–25 NFS operations/second, as shown in table 2.

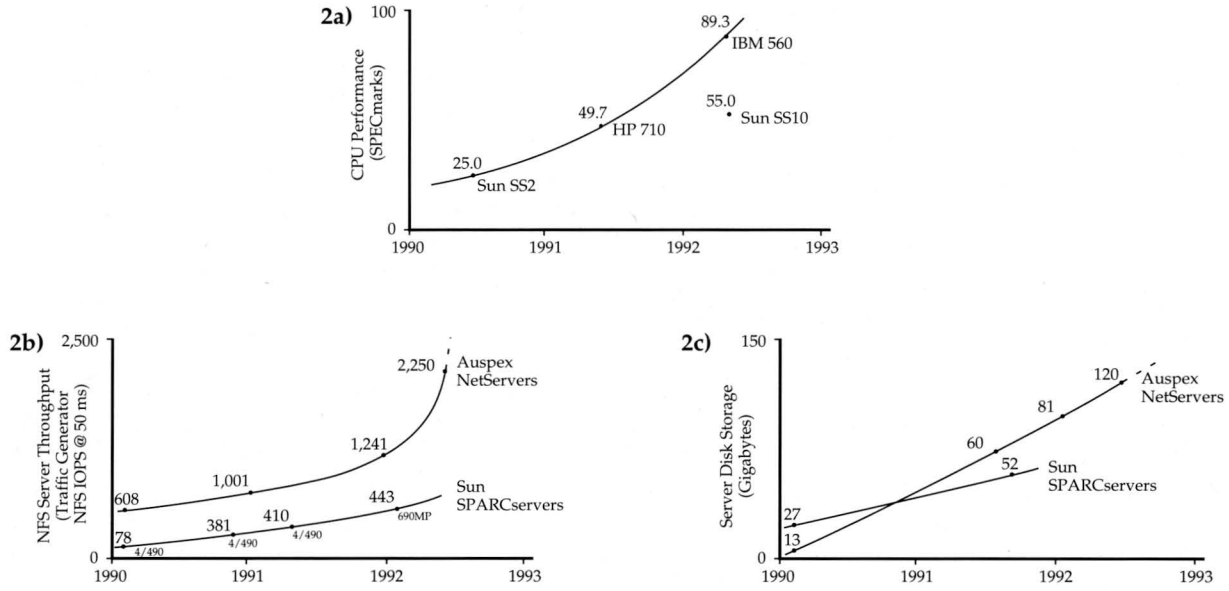

Figure 2a: Trends in workstation CPU performance. Over the past several years, workstation power has more than doubled each year, with desktop performance increasing from 25 SPECmarks in 1990 to nearly 100 today. This annual exponential boost in CPU power is what drives the workstation industry. (Data from vendor specifications.)

Figure 2b: Trends in NFS server performance. Auspex NetServer NFS throughput has quadrupled since 1990, from 608 NFS IOPS then to over 2,250 today. As will be discussed in section 3.2, NFS servers based on a *workstation* architecture (such as the Sun SPARCserver 600MP series) cannot double their I/O performance in pace with workstation CPU increases. (Data from figure 12.)

Figure 2c: Trends in NFS server storage capacity. Auspex also increases storage capacity in step with user demand. Again, this is a technically difficult pace for workstation-based servers. Notice the nine-fold increase in storage capacity since 1990—user storage requirements appear to increase *faster* than CPU power. (Data from vendor specifications.)

1.2 Workstation Configuration and NFS Traffic Characteristics

The most demanding NFS clients are diskless workstations with no local disks. *Diskless* clients, defined in table 1, depend on a file server for application binaries and virtual memory paging as well as data. Clients containing small local disks for paging and booting—*swapfull* or *dataless* clients, see table 1—are less taxing. Dataless clients generate about one-half to two-thirds the NFS load of diskless workstations.

The result of this increasing NFS demand is that the average number of *dataless* clients that a typical conventional high-end NFS server (e.g., Sun SPARCserver 690MP) can adequately support is between 25–50, depending on client power, local disk configuration, application workload, and Ethernet load. For *diskless* RISC clients, the limit appears to be about 15–25, again depending on client, application, and Ethernet load. As table 2 indicates, workstation NFS request rates (column 4) depend greatly on application (columns 2–3) and client workstation configuration (column 1).

Ethernet load plays a strong but little-understood role in minimizing NFS response times. Many network administrators consider Ethernet utilizations of 40–70% acceptable. This assumption is true with respect to fair, round-robin Ethernet access by the attached workstations. It is *false* if response times are also assumed to be *low* at high load. David Boggs, co-inventor of the Ethernet, extensively studied hardware-level latencies for packet transmissions on loaded Ethernets [Boggs88]. His study permits the conclusion that for NFS transfers in a data-intensive 15-RISC-workstation network with 37% NFS *read* and *write* operations, there is a hidden *extra* average response time delay of about 10 milliseconds as each workstation waits for all the *other* workstations' big packets to fly by (Boggs' figure I-4). This latency-causing CSMA/CD deference occurs in hardware and is *invisible* to software and various Ethernet analyzers. This 10 ms delay is significant because some frequent NFS operations—e.g., *GetAttributes*—take only *2 ms* on an unloaded Ethernet. Boggs' recommendation for reducing hidden Ethernet delays is simple: keep Ethernet utilization to a 10–20% maximum so workstations never wait to transmit.

1 Background

Over the past ten years, remarkable increases in hardware price-performance have caused a startling shift in both technical and office computing environments. Networks of PCs, workstations, and servers have rapidly displaced alphanumeric terminals attached to mainframes or minicomputers. As client-server computing enters the nineties, however, it is experiencing growth pains. Client workstation power is severely constrained by file server I/O limitations because dramatic jumps in microprocessor performance have not been matched by similar boosts in server I/O performance.

1.1 The Unix I/O Performance Gap

Driven by RISC and CISC microprocessor developments, client workstation performance has increased more than ten-fold in the last few years. Fifty-MIPS Unix workstations now cost less than $10,000 and 35-MIPS less than $5,000. However, these extremely fast clients have an appetite for data that conventional servers are largely unable to satisfy. The 50:1 RISC-to-CISC increase in workstation computational power has not been matched by a comparable increase in Ethernet capacity, in disk speed, or server disk-to-network I/O throughput. This disparity is summarized in figure 1.

Figure 2a's specific examples illustrate the 4*x* compute-power progress that the intensely competitive Unix workstation industry has made over a three-year period. Figure 2b shows that several NFS vendors have made substantial gains relative to their 1990 NFS throughput levels. But observe that it is much harder to increase NFS server throughput *at the same rate* that workstation CPU power has increased—exponentially. This is a tangible instance of figure 1's I/O performance gap. Finally, figure 2c shows a similar disparity in the ability to support exponentially growing server disk storage capacity.

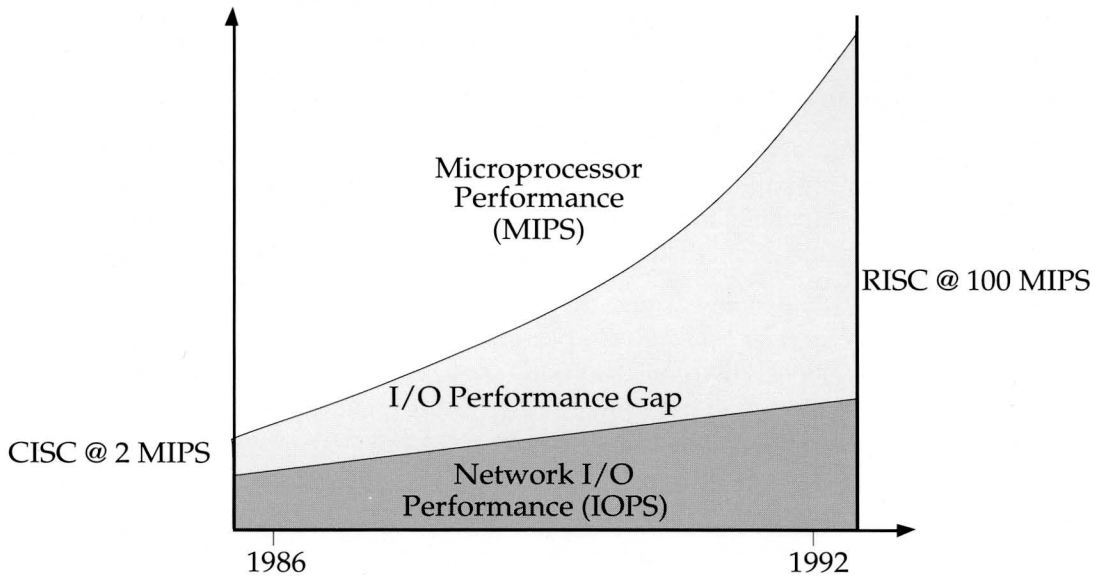

Figure 1: The I/O performance gap that results from mismatched advances in processor and storage performance. This gap has grown so wide that even a traditional file server can be overloaded by as few as ten client workstations.

In most Unix environments, clients and servers exchange file data using the Network File System (NFS) [Sandberg85], a standard distributed file system promulgated by Sun Microsystems and now widely adopted by the entire computer industry. While simple and reliable, NFS is neither sophisticated nor optimal. Clients using NFS place considerable demands upon both networks and the NFS servers feeding NFS data to them. Thus, to understand the I/O performance gap in the context of NFS, we must first examine the impact of client configuration and NFS network traffic on server performance.

Preface to the Sixth Edition

This second printing of the sixth edition of *An Overview of FMP* incorporates:

- *Charts showing quantitative trends in Unix CPU power, NFS throughput, and server storage capacity.* The three-year growth trend is exponential in all three measurements.
- *New performance data for the latest generation of NetServers.* A single, 8-Ethernet, 16-drive NS 5500 has achieved 2,250 NFS I/O operations per second at 50 ms average response time. This widens the performance gap between Auspex and its competitors.
- *Specifications for the new Hewlett-Packard 2.0-GB SCSI-2 disk drives.* These drives are the latest available for NetServers.
- *New installed-base statistics and customer-experience stories.* Currently, more than 450 NetServers are installed worldwide. See section 5.3.

— Nelson & Frommer, July 1992

This sixth edition of *An Overview of Functional Multiprocessing* describes NetServer technical advances incorporated in the past nine months, including:

- *Disk write acceleration.* This option consists of one *write cache* daughter board attached to each storage processor with corresponding write accelerator software. The performance advantages include 20% more throughput and 50% faster response time, resulting in typical write-intensive applications (e.g. restore) completing 3–5 times faster.
- *Hewlett Packard 1.35-GB SCSI-2 disk drives.* Relative to the previous HP 1.002-GB drive, the 35% increase in formatted storage capacity is accompanied by a 23% improvement in average seek time (down to 13.5 ms from 17.5 ms).
- *Hot-pluggability for all SCSI devices.* Disk, tape, and CD-ROM drives can be inserted and removed while the system is powered-up, running Unix, and providing uninterrupted service to users.

Section 4's throughput-response-time graphs show NetServer performance gains, due in part to write acceleration and new disks. A single 8-Ethernet, 16-drive NetServer has achieved 1,241 NFS I/O operations per second at 50 ms average response time. These performance gains keep Auspex well ahead of Sun and SGI, who are the best of the traditional Unix NFS competition.

On a business note unrelated to the technical content of this report, both IBM and DEC now have agreements with Auspex for FMP-based server products. This is a very real statement of the power, performance, and effectiveness of both the FMP architecture and NetServer products.

— Nelson & Frommer, March 1992

Table of Contents

Printed on recycled paper.