YOU REALLY CAN GET THERE FROM HERE

SFU's Migration to UNIX from MTS

Edited by Ken Urquhart and Bill Baines

Abstract

The work undertaken at SFU to migrate to UNIX from MTS was significantly greater than the simple removal of one campus mainframe and the installation of several networked UNIX machines. Considerable effort was required to identify, analyze and replace the services and functions that were provided by MTS and SFUnet.

This paper is the result of a combined effort of the staff members of Academic Computing Services (ACS) and the Operations and Technical Support (OTS) group. It describes the migration process and provides an overview of the distributed UNIX environment now available at SFU. We begin with a brief description of the previous MTS computing environment and the mandate from SFU for our migration to UNIX. Subsequent sections cover our approach to the problem, the physical implementation of the new environment, staff and user training and acceptance issues, and the replacement of services such as printing, library/database access and BITNET traffic.

In retrospect, we were able to accomplish most of the change-over using "shrink-wrapped" products combined with a great deal of ingenuity and plain "hard work" on the part of ACS and OTS staff. Custom programming was required to implement some of the more important services such as campus-wide printing, POP clients, X.500 directory clients and MTS tape access.

While nothing is ever perfect, and you cannot please all of your users all of the time, we are now meeting or exceeding the level of service that was previously provided by MTS and SFUNet. The new UNIX environment allows us to provide valuable new services to our community that would have been difficult (if not impossible) under the previous system.

1. Summary[1]

In February 1991, SFU mandated a migration from MTS to a distributed UNIX environment. The process was to be substantially complete by December 31, 1991 when the mainframe was to be decommissioned. In order to facilitate this task, Computing Services was re-organized into two departments – each led by a single director. The two new departments were designated Academic

[1] Contributions by Ken Urquhart, Bill Baines and Peter Van Epp *(urquhart@sfu.ca, bill@sfu.ca, vanepp@sfu.ca)*

Computing Services (ACS) and Operations and Technical Support (OTS). Both reported to the Associate Vice-President Academic and were given direction by a hierarchical group of committees.

Working together, ACS, OTS and the committees decommissioned the central MTS system running on an IBM 3081-GX, the aging SFUNet, and migrated to a new environment built around:

- 7 UNIX timeshare hosts (SGI, Sun and IBM).
- 3 SPARC based servers for services like e-mail, name resolution, netnews, X.500 and Xerox printing.
- 1 Auspex NFS file server (18 GB and growing).
- 1 VAX server for BITNET, Datapac, library access and databases.
- 1 Novell server with 6 HP LaserJet IIIsi printers for campus-wide printing.
- 3 Annex terminal servers (192 ports).

Accomplishing the change-over in so short a time required a tremendous effort from both ACS/OTS staff and our users. In order to make this task as painless as possible, regular tutorial sessions on many aspects of the migration were scheduled and presented. Migration issues and the new UNIX services were described in a series of one-page *"How To..."* sheets that were widely distributed to the SFU community. The sheets were updated on a regular basis as the migration progressed. Departmental meetings between faculty and staff and ACS/OTS were held to minimize user anxiety and to bring potential migration problems to the attention of ACS/OTS.

CPU performance in the new UNIX environment has exceeded our expectations. Our general-purpose UNIX host (*fraser.sfu.ca*) can easily support 80 to 120 concurrent users for reading e-mail, accessing netnews and other low impact tasks. The rest of the UNIX hosts were put to immediate use by our research users. As each research machine was brought on-line, computationally intensive tasks soon consumed all available CPU cycles. In fact, time on our fastest UNIX machines *garibaldi.sfu.ca*, *skypilot.sfu.ca* and *tantalus.sfu.ca* was (and still is) rationed out by a special user committee!

Disk space has been at a premium since the migration due to ever increasing demands for our new services and the fact that binaries on UNIX take up more space than MTS binaries. It has been necessary to carry out two disk upgrades on our central Auspex NFS file server *catacomb.sfu.ca* and to enforce disk space quotas on our users. Additional local disks have been added to several hosts to improve the performance of specific applications like Sybase and netnews and to provide working storage for the disk-bound statistical packages.

Surprisingly, network congestion is not a major problem although it is expected to increase as the use of applications based on X-windows begin to find acceptance among the user community (we have site licenses for both the Macintosh X-server software and the NeXT co-Exist software in addition to a small number of X-Terminals).

Most policies governing the use of machines and the network are now in place. Some policies are still being written and re-written as we gain more experience. Security, remote access to services, and the use/abuse of accounts is still an area of intense interest.

Opinions about our success vary. Most users require little more than e-mail access and mailing-list maintenance. This group has basically "tolerated" the migration. Researchers and students who require raw CPU cycles and/or sophisticated visualization and analysis programs have been rather pleased with our new services (after the initial trauma of program conversion passed). Users who are less than satisfied with the migration tend to be restricted to people who continued to depend on MTS-specific services such as *TEXTFORM and program libraries like *IG.

One on-going thorn in our side is the lack of sophisticated tape support under UNIX. We were all spoiled by *FS (the MTS tape management system) and there appears to be no UNIX tape utilities with the versatility of *FS. Those who required a high-level of tape support under MTS have had to make substantial changes to the way they do their work. This includes AES/OTS!

2. The Initial Conditions

This chapter describes the "initial conditions" for our migration to UNIX. We begin with a review of the MTS and SFUNet services offered to users by Computing Services and how those services were utilized. We then describe several initiatives undertaken in previous years that were intended to decrease the dependence of our users on MTS. The last part of this chapter covers the "mandate for change" that resulted in the re-organization of Computing Services at SFU, the phase-out of MTS, and the introduction of UNIX – all in the space of 10 months!

2.1 MTS and SFUnet at SFU

2.1.1 The Hardware Base[2]

MTS at SFU was supported on an IBM 3081-GX (two CPUs) equipped with 32 MB of memory and sixteen I/O channels. Twenty four MTS volumes were maintained on IBM 3380A disk packs, providing approximately 15 GB of storage. One 3350 volume was maintained on a separate path for storage of the MTS e-mail message file *MESSAGES. Paging was handled by a STC 24 MB paging drum and a 3350 backup page pack. Tapes were supported by six 3420-type tape drives and four 3480-type tape drives (each driven from two channels).

Printing services for campus users was provided by a Xerox 4090 page printer (laserprinter) located at Computing Services and a smaller Xerox 3700 page printer located at remote computing facilities across the campus. Users of the remote printer had to release their print jobs manually by supplying their MTS ID and password to a release console attached to the printer.

The high speed SFUlan network was copper and fiber based. IBM type-9 cable was run to nearly every telephone jack on campus so that users could connect to SFUlan from almost any office or lab.

Low Speed serial connections were provided by low speed copper maintained by the campus telephone group. Most of these lines were connected to SFUNet which consisted of three Nodes, one HIM, and six NIMs providing approximately 1200 NIM ports. An additional 400 low speed lines were connected to a StarMaster data switch. 120 StarMaster host-side lines were connected to NIM ports.

[2] Contributions by Peter Van Epp *(vanepp@sfu.ca)*

The high speed SFUlan network was built around two main network centres located at opposite ends of the campus and connected by a fibre link. Over 50 wiring closets were connected to these centres via fiber. Local traffic was kept off the backbone by installing a bridge in each closet. Aside from reducing network congestion, the local bridges simplified maintenance (just pull out the bridge and replace it – no need to re-configure anything).

SFUNet was built upon a superb physical plant. Cabletron supplied nearly all of the network equipment and it has proven to be very reliable and delivers excellent performance.

2.1.2 Utilization of Resources[3]

MTS usage by the campus community could be broken down as follows:

E-Mail and Conferencing	28%
Statistical Packages	20%
Compilers	15%
Custom Applications	18%
Symbolic Mathematics	4%
Numerical Analysis	4%
Text Processing	4%
Utilities	2%
DataBases	3%
Graphics	2%

Nearly everyone with an MTS ID used e-mail. In fact, many users referred to MTS as "FSM". In terms of CPU utilization, the primary use of MTS was for "number crunching" by a relatively small group of students, faculty and staff. A further break down of utilization by user showed that a small group of less than ten individuals consumed the bulk of the CPU cycles. These research users eventually formed the "large-scale computing group". Several Silicon Graphics (SGI) mini-supercomputers were purchased for this group in the early 1989 in order to satisfy their computing needs. The other options of (1) purchasing a large super-computer like a Cray for the entire campus or (2) attaching a vector processor to the IBM 3081 mainframe were discussed in the late 1980's but dismissed on the basis of prohibitive maintenance costs. The SGI machines provided a much more cost effective solution. In retrospect, this group gave computing services it's first experience in managing networked UNIX machines.

The remaining areas of utilization such as text processing and graphics had low usage counts. This was due primarily to several initiatives implemented by Computing Services in the mid to late 1980's that shifted such work micro-computers. These initiatives, combined with the purchase of the SGI machines for large-scale computing, formed some of the groundwork for migration away from MTS towards a more distributed computing environment.

[3] Contributions by Lionel Tolan and Ken Urquhart *(lionel@sfu.ca, urquhart@sfu.ca)*

2.2 Previous Migration Initiatives[4]

For several years, SFU Computing Services had been systematically installing new services and facilities to reduce our dependence on MTS in anticipation of it's eventual phase-out.

- Administrative VAX Cluster (1986 to present) – all administrative computing was moved over to a cluster of VAX/VMS mini-computers in order to exploit the popular PowerHouse environment for developing and maintaining new administrative systems. This had the beneficial side-effect of removing the burden of administrative migration when MTS was finally phased-out.

- Macintosh and PC Labs (1987 to present) – the availability of inexpensive micro-computers and fairly sophisticated word processing, charting, and spreadsheet applications provided an obvious route towards reducing our dependence on the mainframe and an improvement in the personal productivity by students, faculty and staff. Computing services took the lead in introducing SFU to personal computing in 1987 by establishing two labs consisting of twenty Macintosh Plus and twenty Zenith Model 148 micro-computers. The labs allowed instructors to present "hands-on" tutorials and demonstrations to their students in an effort to show them how to use computers in their work. When classes were not in session, students could use the computers for general computing. The labs were open from 8:30 to 22:30 most days.

- WordStation (1989 to present) – in an effort to significantly reduce the user of *TEXTFORM on MTS, a special lab consisting of 75 Macintosh Pluses, 75 PCs, local dot-matrix printers and local laser printers was setup in the basement of the library in 1989. This student word processing facility was open during normal library hours and tended to be fully occupied during the day. It was a major success in that many students who would not have considered doing a paper using *TEXTFORM did use the WordStation and that overall *TEXTFORM use on MTS dropped sharply.

- Assignment Labs (1990 to present) – most of the undergraduate computing load was shifted off of MTS by establishing several "assignment labs" across the campus. The largest lab was equipped with 75 IBM Model 55sx (386sx) machines with VGA monitors and a number of Macintosh IIci colour machines all connected to the campus network and serviced by a Novell file server. Lab usage ranged from computing science students learning Modula-2 to Kinesiology students using HyperCard to create computer based training applications. Novell software written by Computing Services allowed instructors to book a fixed number of hours for each assignment. The scheduling software guaranteed that each student got at least the allotted machine-time per project. The programs required for each course were kept on the Novell server with students saving their data on floppy disks.

- UNIX Support Center (1988 to 1991) – an open laboratory was set up to allow people to explore UNIX. The centre was equipped with a Sun 4/280 and three Sun 3/50 machines networked together. Various other workstations (such as NeXT and HP machines) were made available under long-term loans from vendors. One of the Sun machines handled UNIX e-mail and services such as netnews for a limited user base. User ID's were available to all faculty, staff and graduate students who asked for them. The lab also provided support

[4] Contributions by Peter Van Epp, Lionel Tolan, Grant Dimock and Ian Reddy *(vanepp@sfu.ca, lionel@sfu.ca, grant@sfu.ca, ian@sfu.ca)*

services for the SGI mini-supercomputers already in use by the large-scale computing group as well as support for a small number of Sun workstations installed around the campus. Note that we did not support a large number of Sun SPARCstations installed in the Centre for Systems Science. Nor did we support the systems installed in Computing Science. These external systems were supported by their respective departments. See chapter 8 for a description of two of these systems and how they have been integrated into the new campus environment.

2.3 The Mandate for Change[5]

In February 1991, a special task force presented a report on the state of computing at SFU and made recommendations about how SFU should provide computing services to the campus community in the 1990's. Three basic changes to the computing environment at SFU were recommended:

1) Migrate from a traditional, mainframe-based central computing environment to a open, distributed computing system based on the UNIX operating system and a unified, high-speed campus network.

2) Shift control of the computing environment to the user community.

3) Improve the levels of service provided by Computing Services.

The re-deployment of services to meet these recommendations was significant:

- The personnel and the resources comprising the administrative development unit were distributed to their three major clients: the Office of the Registrar, Financial Services, and General Administration / Facilities Management. The three new administrative computing units were given the responsibility of providing maintenance and development support for all remaining minor administrative computing groups.

- The department of Academic Computing Services (ACS) was established. This group was led by a single director and reported to the Associate Vice-President, Academic. The mandate of ACS was to provide instructional and research computing services to the academic community and to service the needs of the administrative units in the various academic departments.

- The department of Operations and Technical Support (OTS) was established. Much like ACS, OTS had a single director and reported to the Associate Vice-President, Academic. The unit is charged with operating and maintaining the computing infrastructure, including the campus network and the hardware in common-use by both the academic and administrative computing units.

Four immediate goals were set by the task force:

1) Implement a UNIX-based distributed mail environment as the primary mail service. This was to replace MTS mail by August 31, 1991.

2) Implement a UNIX based distributed computing environment and remove the IBM mainframe from service by December 31, 1991.

[5] Contributed by Lionel Tolan, *(lionel@sfu.ca)*

3) Implement an electronic Campus Wide Information System (CWIS).

4) Convert the operation of the Microcomputer Store into an ancillary operation.

3. Our Approach to Migration

This chapter describes the steps taken by ACS/OTS to build the new UNIX environment. It discusses the philosophy of the new system and the important task of choosing the a centralized file server (the key to our system).

3.1 The Network Centered Model[6]

The model chosen for our new UNIX environment can best be described as "network centered". The idea was that all CPUs and peripherals are of secondary importance to a well planned network equipped with a high-performance, centralized file server. On such a system, user data could be NFS-mounted on any or all CPUs sitting on the network. Individual computers would then be tailored to provide specific services such as e-mail and conferencing, statistical analyses, or raw computing power for number crunching. Through the use of NIS, users would be able to login to whichever of these "compute-servers" met their immediate needs. This configuration also kept software costs down since licenses for the more expensive software packages would only have to be purchased for single computers instead of multiple machines. On the hardware side, adding new computers to the network would not require shifting userIDs and data around. If one computer or it's local hard disk(s) broke down, users (and software packages) could easily be shifted to other machines with minimal interruption of service.

While all this sounds wonderful in theory, the choice of this model was also dictated by two very practical limitations:

- Staffing: there were only three people within ACS/OTS who were capable of assuming the role of UNIX system administrator during the migration. They would not have been able to effectively manage a traditional UNIX machine network where volumes were cross-mounted "all over the place". A much simpler, centralized system was needed. This meant a centralized file server with identical NFS mount tables copied to the attached computers.

- I/O bandwidth: the mainframe running MTS was equipped with 16 I/O channels, each capable of moving data at speeds approaching 3 MB per second – and it was just keeping up with user demand! Since ethernet could not support moving this amount of data around, we had to get smart and choose a file server equipped with multiple ethernet ports. Each of our UNIX compute-servers could then be connected to the file server via dedicated sub-nets and the network traffic over any one of these links kept to an acceptable level.

3.2 Stalking the Elusive File Server[7]

NFS is the *de facto* standard for distributed file systems under UNIX and while there are some security concerns associated with it, our time and manpower constraints virtually dictated its use.

[6] Contributed by Lionel Tolan and Ken Urquhart *(tolan@sfu.ca, urquhart@sfu.ca)*

[7] Contributions to the rest of this chapter by Bill Baines, Ian Reddy, Peter Van Epp and Ken Urquhart *(bill@sfu.ca, ian@sfu.ca, vanepp@sfu.ca, urquhart@sfu.ca)*

The details by which a talented amateur could gain access to other peoples' files is beyond the scope of this paper, but suffice it to say that it is not outside the capabilities of a persistent "hacker".

While we would have been more comfortable with more secure systems such as AFS (Andrew File System), nearly every vendor we talked to couldn't (or wouldn't) deliver AFS systems. We therefore quit looking at AFS, accepted NFS, and carried on.

Finding a suitable file server was a non-trivial task. Some vendors selling NFS servers that might have met our needs had products ready for release "real soon" but none in the field. Most just stressed the power of their CPUs but almost totally ignored (or just plain didn't know about) the issue I/O throughput.

Only Auspex had large, centralized file servers installed in the field and their clients had only good things to say about them. Natually, this warranted a closer look.

Now, Auspex sells only one product: NFS file servers. As such, they didn't need to balance general purpose computing with the demands of NFS file serving like other vendors did. Further, they appeared to have studied and understood that not all computing is CPU bound; that is, some process are actually I/O bound.

After a visit to Auspex in June 1991 and informal chats with some of their clients, the choice of file server seemed obvious.

3.3 The Auspex Solution

When several hosts, each with several users, are served by a single NFS server the response time is typically limited by network bandwidth and head seek time. The Auspex NFS File Server overcomes (or at least stretches) these traditional NFS limitations. The principals of the company are former mainframe people (IBM and CDC) and they did seem to understand the disk I/O issues.

3.3.1 Architectural Overview

The Auspex file server distributes NFS over multiple ethernets (different IP subnets) and provides fast access to a common disk repository. The disk facilities support mirrors, stripes, virtual partitions, and "hot" replacement (ie, pulling out a bad hard disk while the Auspex server is up and running and sticking a new one in the drive slot to replace it).

The disk type is SCSI, with a maximum of 2 spindles per bus. A basic system comes with 10 SCSI buses. Auspex claims that by using the SCSI-2 command set, with qualified disks, it is possible to transfer commands to one disk on a SCSI bus while the other disk is transferring data. There is a great deal of concurrency and it appears that the architecture is more suited to the "typical" I/O of general purpose computing than to multiple IPI drives on an IPI controller. The IPI controllers that we looked at on general purpose Unix file servers apparently queued the requests at the controller so there was very little concurrency across a string of IPI disks. The 6 MB per second I/O rate quoted for IPI disks *just didn't matter* for the typically small, random I/O distribution seen via NFS on general-purpose UNIX systems. Most of the I/O response time was consumed in queuing the requests, seeking the head, and waiting for latency.

The other architectural feature that sets the Auspex above its competition is a "functional multiprocessor" design. Individual, specialized CPUs operate concurrently handling NFS transactions.

When a conventional UNIX system is configured as an NFS server, each transaction is handled by ethernet hardware, ethernet device drivers, daemon processes (which must be context switched), directory code, file and block code, disk device drivers, and the disks themselves. A single request may result in several context switches. In the Auspex, a loose pipeline of NFS requests are processed concurrently. Hardware processors are dedicated to:

- One to four ethernet subsystems (each ethernet processor handles two ethernets) handling all of the IP and UDP processing on board – including the NFS requests. This avoids the bus bandwidth and main CPU interrupt load required by a general purpose machine to deal with the ethernet interface (to say nothing of the load provided by several Ethernet interfaces) with respect to NFS requests.

- A cache manager and one or two file processors for the file system (Berkeley Fast File system on-board). A check is made in the external cache (32 MB in our configuration) to see if the requested block is already in the cache. If so, then the cache address is returned to the ethernet processor for DMA-out to itself. This reduces the amount of copy-traffic that fights for main memory and/or bus bandwidth in a general purpose processor.

- One to three SCSI storage controllers, each with 10 SCSI buses, one for for each 2 disk drives. Studies of I/O mixes show that there is a significant degradation after 2 drives per channel, but not after 1. Again data is DMAed between the controller and the cache.

- A Sun-4 series SPARC OEM board to load the system, run diagnostics, do backups, and run standard UNIX network daemons if desired. The SPARC is not required for NFS processing.

3.3.2 What Auspex does well

The Auspex does a good job at serving tens of gigabytes across multiple ethernets to client machines. It allowed us to present identical user directory environments to many users across many hosts.

Large sequential reads (such as `greps` through large files) are fast because the file processors will read ahead and place data in the cache, (32 MB on our system, expandable to 96 Mb). Subsequent I/O does not go to disk but is read from the cache by the ethernet controller.

Virtual partitions allow the administrator to set up complex configurations. Multiple virtual partitions may be striped or concatenated together up to the standard 2 GB UNIX limit.

Mirrors work well and can provide redundancy for critical data, improved I/O response for read-only data or, in our case, a mechanism to take a clean backup copy of a hot disk.

Hot replacement can be a real advantage for systems that require a high "up-time".

Note: It *is* possible to configure an Auspex file server with more than 100 GB of disk. That's a lot of spindles and the "hot change" feature would likely be required just to replace failed disk assemblies. Our Auspex has 18 GB of disk now, but it started out with 10 GB. Those 16 spindles have been on-line since August 1991 without a disk failure. We have only taken the system down for firmware and OS upgrades, never to replace a disk. Over the same period, we have had at least 3 failures amongst the 7 OEM disks externally connected to our various UNIX hosts. The Auspex disks are more costly but our experience seems to

indicate that Auspex is doing a good job of qualifying their disks and culling out the bad ones before they can be delivered to their customers.

3.3.3 What Auspex doesn't do well

The story breaks down when it comes to backup and tape handling. There just isn't good tape handling under standard UNIX. In the case of the Auspex, backups are done on 8mm tape. We write and check a tape label as the first record of the tape using a combination of shell scripts and C programs. Then we use UNIX to dump the file system after the label. This has saved us from overwriting dump tapes several times (although not from an operator accidently re-labeling an input tape during a fire-save copy). The weekly backup tapes are copied, via shell scripts and `dd`, to another set of tapes and sent to the vault.

We assumed (something you should not do in the UNIX market) that any vendor providing 100+ GB servers would have thought through the backup issues. A high performance 3480 or Exabyte 8500 interface with specialized backup/archive/recovery software would be a nice addition and is going to be required sooner or later.

4. Physical Implementation

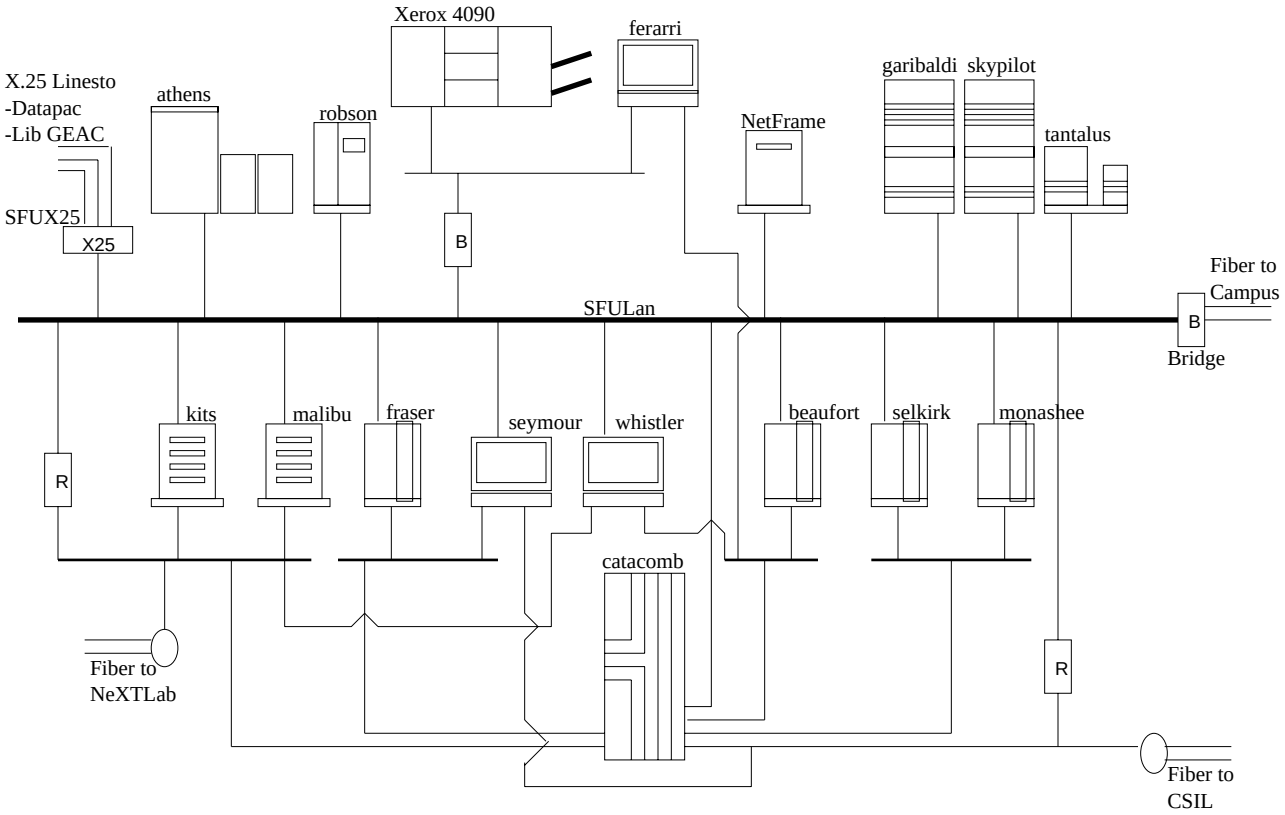
This chapter describes the pieces of the UNIX puzzle and how we fit them all together to form our new environment.

4.1 UNIX Systems and Configurations[8]

Prior to the migration we were supporting three dedicated UNIX processors. These were the three SGI multiprocessor systems used exclusively by the "large-scale computing group" (scientific users with numerically intensive applications) and the Sun 4/280 used for campus name-serving, news-services, and departmental UNIX training. All of the UNIX-specific support was provided by Ian Reddy with scientific applications being supported by Steve Kloster. Operations provided network connections, power, and cooling – but no OS support.

The following figure illustrates what is now in place. Each component is described in detail below.

[8] Contributions by Bill Baines, Ed Hargrave and Ken Urquhart *(bill@sfu.ca, ed@sfu.ca, urquhart@sfu.ca)*



4.1.1 catacomb.sfu.ca

Catacomb is an Auspex NS-5000 high performance file server. A detailed description is provided in section 3.3. Our NS-5000 is equipped with 18GB of disk, a 32 MB cache, three ethernet controllers and two 8mm tape drives. NFS is not exported to the SFULan backbone network for both security and performance reasons. All user home directories are kept on catacomb and provided via NFS to the other UNIX hosts. Mail spool files, large read-only datasets, and many commonly used applications such as `elm`, `nn`, `rn`, `CWIS`, `emacs`, TEX and `perl` are also exported.

4.1.2 fraser.sfu.ca

Fraser is our designated "general" UNIX machine. It is a Silicon Graphics 4D/320 equipped with 2 CPUs, 128 MB of memory, a 700 MB internal system disk and a 700 MB external disk for anonymous ftp services. There are two ethernet controllers and an internal CD-ROM reader for software distribution. It is currently running IRIX 4.0.1 and base level applications exported from catacomb.

4.1.3 selkirk.sfu.ca

Selkirk is a research machine for business, economics and general statistics processing. It is a Silicon Graphics 4D/320 equipped with 2 CPUs, 64 MB of memory, a 700 MB internal system disk, a 1 GB external disk, and two ethernet controllers. It runs the basic applications plus `bmdp`, `imps`, `minitab`, `shazam`, `spss`, `tsp`, and `maple`.

4.1.4 beaufort.sfu.ca

Beaufort is a research machine for business, economics and general statistics processing. It is a Silicon Graphics 4D/320 equipped with 2 CPUs, 64 MB of memory, a 700 MB internal system disk, three 1 GB external disks, and two ethernet controllers. It is running the basic application suite plus `Sybase`, `bmdp`, `imps`, `maple`, `shazam`, `spss`, `tsp`, and `maple`.

4.1.5 monashee.sfu.ca

Monashee is a research machine for scientific work. It is a Silicon Graphics 4D/320, equipped with 2 CPUs, 64 MB of memory, a 700 MB internal system disk and two ethernet controllers. It is running the basic application suite as well as most programming languages. Most of the user code is written in `FORTRAN`.

4.1.6 kits.sfu.ca and malibu.sfu.ca

Kits and malibu are instructional machines. They are Sun 4/470 machines with 64 MB of memory, a 700 MB internal system disk, a 150 MB quarter inch tape drive and two ethernet controllers. They run the basic applications plus languages such as `FORTRAN`, `C`, `perl`, `Pascal`, `COBOL` and `Modula-2` and instructional applications such as `bmdp`, `imps`, `shazam`, `spss` and `tsp`.

4.1.7 whistler.sfu.ca

Whistler is one of our servers. It is a Sun SPARCstation 2 equipped with 64 MB of memory, a 400 MB internal disk, a 1.2 GB external disk and three ethernet controllers. It runs `BIND`, `sendmail`, `popper`, the `gopher` server (for CWIS) and the `quipu` X.500 server.

4.1.8 seymour.sfu.ca

Seymour is also a server. It is a Sun SPARCstation 2 equipped with 32 MB of memory, a 400 MB internal disk, a 700 MB external disk and three ethernet controllers. It runs `NIS`, a netnews server, and `annex erp` (a login authentication server for our dial-in service).

4.1.9 ferarri.sfu.ca

Ferarri is a print server. It is a Sun SPARCstation 2 equipped with 32 MB of memory, a 400 MB internal disk, a 300 MB external disk, and two ethernet controllers. It runs *Soleil*, Xerox's PostScript to Interpress/XNS conversion software. It accepts `lpd` requests from some of the central hosts for ascii and PostScript printing on the Xerox 4090 laser printer.

4.1.10 athens.sfu.ca

Athens is a catch-all machine running systems we didn't know how to implement on UNIX. It is a VAX 8530 with 80 MB of memory, 3.8 GB of disk, a HSC50 disk/tape controller and one ethernet controller. It runs *P.S.I.* for Datapac and GEAC X.25 connectivity, *MultiLis* for the French library catalogue, and *BRS* for the Groliers and Wilson databases. It also runs *TGV Multinet*, *Jnet* and *MX* (from RPI) for BITNET services. Athens also has access to 9-track tape drives so it can be used for MTS *FS tape conversion (which can also be carried out on our administration VAX Cluster).

4.1.11 robson.sfu.ca

Robson is an IBM RS6000 Model 530 with 80 MB of memory, a 340 MB internal drive, a pair of 1.2 GB SCSI external disks for user data, and an 8mm tape drive for software loading and backup. It supports intermediate scale computing (less than 500 CPU hours per year) and currently has about a dozen or so people authorized to use it. User data is stored on the local disks which get backed up once a week.

4.1.12 garibaldi.sfu.ca, skypilot.sfu.ca and tantalus.sfu.ca

These are the original SGI mini-supercomputers (super-minis?) obtained prior to the migration for use by our large-scale computing group. Garibaldi and skypilot are 4D/380 machines while tantalus is a SGI 4D/240. There is a total of 20 CPUs across these three machines. The large-scale users consume *all* of the available CPU cycles on these machines with their `FORTRAN` programs (most Monte-Carlo simulations).

4.1.13 Netframe 100

Our Netframe NF100 provides a range of Novell Netware based services for the campus. It is equipped with one 80386 Ethernet I/O processor, 16MB of memory, and 1.7GB of SCSI disk. There are plans to add a CD-ROM drive and 8MM tape drive this fiscal year. We currently run a 250-user version of Novell Netware 3.11 with a 100-user Mac NLM and Netware NFS. NetFrame is to the Novell world what Auspex is to the Unix world – they have taken server performance and applied maximum technological effort to the I/O bandwidth issue. A proprietary 100MB/sec backplane links dedicated I/O processor boards that are responsible for disk and network I/O. The server has no keyboard or console. The only ways to configure it are by ethernet or by the built-in modem. The NetFrame currently provides printing services for PCs, Macs and Unix hosts to 6 LaserJet IIIsi printers installed around campus. A project is underway to provide read-only file services to serve Macintosh and PC software to faculty and staff. A CD-ROM service may be implemented in conjunction with the Library.

4.1.14 SFUX25

This is a Digital Equipment Corporation MicroServer: a hardware box with an ethernet controller and 4 synchronous ports. The system is not hardware configurable. It is loaded over the network (like a NIM) and DEC supplies three or four different software products that make the box do different things. We are running the `X.25 Router 2000` package which processes X.25 packets locally and sends the data over the ethernet to one or more client hosts. We route Datapac to the campus backbone through the DEMSA and athens.sfu.ca. Similarly, we route X.29 sessions from the GEAC OPAC Library system to telnet sessions on SFULan. Should athens be decommissioned, this box can be served by a DEC Ultrix host. A good deal of accounting and security logging is possible with this system when running to a VMS host. These features have already come in handy for controlling Datapac access to our system and getting a handle on who is running up the big Datapac bills.

5. User Issues

This chapter discusses how we got our users off of MTS and over to UNIX. Section 5.1 describes our user base and the kinds of problems they encountered during the migration. Section 5.2 covers the important tasks of training and documentation.

5.1 User Profiles[9]

Reaction to the announcement that we were phasing out MTS ranged from absolute delight ("you're finally getting rid of MTS!") to total shock ("you can't get rid of MTS!") with a frequent side-trip to confusion ("I don't care if they get rid of MTS, I only use FSM"). Some even refused to believe that we were actually going to do it.

ACS was charged with getting our users off of MTS and over to UNIX with as little pain as possible. Depending on the user, this task ranged from trivial to absolutely impossible. Everyone had an opinion on the subject. Some felt that we would have no trouble at all and we should leave the migration up to the users. Others saw literally hundreds of users moving over to UBC on January 1, 1992 where MTS would still be running. We had a gut feeling that most would migrate over to UNIX with minimal problems and that only about 20-30 people might have to go to UBC. In the end, the total number of people truly inconvenienced by the migration was less than 20 – with only 3 out of 10,000 users needing to go to UBC.

5.1.1 Users with Minimal Migration Problems

The undergraduates were the easiest. Most had absolutely no investment in MTS other than for e-mail and some electronic conferencing with *FORUM. They didn't have any files to speak of and had never used *TEXTFORM in their lives.

Most university staff, as well as many of the faculty members outside of the sciences, were in the same boat. Their only interaction with MTS was through the EASYMTS screens and the only applications they ran were *MESSAGESYSTEM and/or* FSM.

For these groups, changing over to UNIX was really a matter of showing them how to login to UNIX, set a password, and use one of the e-mail packages that we were going to be supporting (see section 7.1 "E-Mail" for details).

A second group of "easy migrators" were the researchers and graduate students who used one or more of the statistical packages running on MTS. Most of these packages existed under UNIX and migration became a matter of transferring these users' files to UNIX and pointing them at the documentation for the UNIX versions of their programs (see section 7.2.1 for more information).

5.1.2 Users with Moderate Migration Problems

This group included the researchers and graduate students who had come to depend on specific MTS services that were not available under UNIX. This included people who had made extensive use of *IG (Integrated Graphics) in their `FORTRAN` code or who had insisted on sticking with *TEXTFORM when we had been warning them for years that *TEXTFORM would be disappearing (even if MTS wasn't).

We helped those with coding troubles by either showing them how to convert their programs to run under UNIX or simply converting the programs for them. Steve Kloster and Ken Urquhart got very good at migrating `FORTRAN` code!

The *TEXTFORM users could either try and convert their files to Rich Text Format (RTF) and begin to use a word processor or convert to LaTeX and use TEX on one of the UNIX hosts (TEX was placed on all of our UNIX machines by Margaret Sharon).

[9] Written by Ken Urquhart *(urquhart@sfu.ca)*

5.1.3 Users with Severe Migration Problems

This last group ranged from the "power users" who had written extensive amounts of IBM/370 assembly code (for whatever reason) to faculty members who had 40-50 MB of *TEXFORM files that represented the culmination of years of research work.

There was little re-course for this group of twenty or so users.

The ones who depended on assembly code were faced with massive program re-writes. The only consolation we could offer was that UNIX allowed you to write many of the speed-critical routines in fast, compact C-code that could then be linked with `FORTRAN` routines. When they absolutely had to use assembly code, they could still use wrapper routines written in C and embed the assembler inside them. In this way if they had to migrate again, they wouldn't have to worry about calling sequences and register assignments.

The *TEXTFORM users were luckily nearing the end of their projects and could be moved over to MTS at UBC for the remainder of their projects (none of them anticipated spending more than another year on their TEXTFORM-dependent work). All new work would then be either in TEX or on micro-computer word processors like MicroSoft Word.

One last group that we encountered were the "tape-based" users. UNIX has notoriously bad tape handling facilites and users who did things like process tape-after-tape of Statistics Canada data faced real problems. One interim solution was to copy large chunks of the tapes over to 1 GB hard disks and mount the disks in special `/tmp/` directories on one of the research UNIX machines like *beaufort.sfu.ca*.

5.1.4 Summary of User Migration Problems

In the end, only 3 users could not migrate away from MTS without severe hardship (two were *TEXTFORM users and one used the LISREL program that is currently unavailable under UNIX). These people received MTS accounts at UBC.

There were five other users who needed to do massive code re-writes. This set them back about 3 to 6 months in their research efforts. Nearly all of them had made extensive use of *IG in `FORTRAN` programs.

Less than ten users were caught by the lack of UNIX tape handling facilities.

5.2 Training and Documentation[10]

My responsibility during the MTS-UNIX transition was UNIX training and documentation. Like most of the staff, I began knowing how to spell UNIX, and not much else. I started by reading extensively—some of the most useful books are listed in the bibliography.

5.2.1 Training for Computing Services

The first priority was to provide training for Computing Services staff, since we were the ones who would implement UNIX on campus.

[10] Written by Ellen Sangster *(ellen@sfu.ca)* and Margaret Sharon *(margaret@sfu.ca)*. All enquiries to Margaret Sharon.

I investigated outside training sources, but found that the time involved (3-5 days per course) and the cost ($1000-$2000 per attendee per course), eliminated these as useful methods of UNIX training.

Instead, we decided to look "in-house". Computing Services staff met in May 1991 to create a list of UNIX topics that we needed training on. Drawing on local expertise, I organized a series of talks and seminars for Computing Services over the spring and summer:

- Ed Bryant, our Sun sales representative, gave a well-received talk on UNIX history and "culture".
- Ian Reddy, Computing Service's system administrator, gave several seminars on UNIX topics, including file systems, permissions, networking (UUCP, Telnet, ftp), security, and NFS.
- Tim Devlin, also from Computing Services, gave most of us our first introduction to the vi editor.
- Hon-Man Wong, from SFU's Center for Systems Science department, gave two lectures on UNIX shell programming.
- Eric Kolotyluk, also from Computing Science, gave us a tour of their facilities and a demonstration in their NeXT lab.

In addition, the Computing Services Operations group bought a set of UNIX videotapes ("Fundamentals of the UNIX System" from the AT&T videotape library) for training its staff and computer operators.

With this in-house help, and many hours of reading and self-instruction, we had a basic foundation of knowledge for our transition to UNIX.

5.2.2 Documentation

The next focus of activity was documenting the new system for the campus. It was immediately apparent that our documentation would be added to and revised continuously for the first few months. I decided to create a series of one-page handouts on various topics. These would contain enough information to describe a topic, yet would be short enough to be revised easily.

We called our UNIX documentation "How To..." and divided it into several areas, each denoted by a letter of the alphabet:

```
A access (how to use the documentation, get an account, find public access)
B basics (how to log in, create a file, understand the file system)
C communication (how to use telnet, ftp, Kermit, Datapac, etc. with UNIX)
D database applications (Sybase, etc.)
E editing (how to use vi and emacs)
G graphics (how to use Disspla)
L languages (how to use FORTRAN, dbx, Modula-2)
M mail (how to find e-mail addresses and use Elm and Eudora)
N numerical & mathematical applications (Maple, etc.)
P printing (how to use the campus laser printer queues)
```

```
S statistics (how to use SPSS, BMDP, SAS, Shazam etc. on UNIX)
T text processing (how to use nroff, troff and TeX on UNIX)
U Usenet (how to use nn and Parti)
```

There are several or many single-sheet handouts within each area. A list of titles and an index for the whole set form the first two handouts (A-1 and A-2). Besides being distinguished by letter, the series of documentation is color coded (green for access, white for basics, yellow for communication, etc.).

I wrote many of the handouts myself (in the access, basics, editing, printing, and Usenet series), and other consultants wrote handouts in their areas of specialization (Frances Atkinson wrote the communication and mail handouts; Margaret Sharon the text processing series; and Reo Audette the statistics handouts). The Research Data Library and the Computing Science Instructional Labs groups have also contributed handouts describing their services.

A handout template was created for authors using Microsoft Word on the Macintosh (the most ubiquitous word processor in Computing Services) FrameMaker on the Macintosh was used for production of the handouts. (Frame opens Microsoft Word files directly).

Frame worked well for creating a consistent format across the series and it also allowed sharing of files across platforms. Titles of all the handouts were saved as Frame variables, ensuring consistency in cross-references. Frame's graphic tools and ability to import graphics files allowed illustrations to be added to some of the handouts. The only problem encountered with Frame was the spell-checker's habit of suggesting leper as the correct spelling for lpr.

Manuals for emacs, Eudora, elm, etc. are sold at cost at the Bookstore. Other manuals such as *A Gentle Introduction to TeX* are available as PostScript files for copying.

5.2.3 Distribution

Printed Copies:

The how-to handouts are distributed free-of-charge at Computing Services in a Plexiglas documentation rack. Copies are also available at some of our public microcomputer labs, and at our downtown campus.

On-line Distribution:

The handouts were very popular. Because we were occasionally out-of-stock, and because the handouts were not available outside of normal office hours, I decided to create an on-line system of distribution. Ian Reddy and I created a shell script called `print_howto`. The script presents a set of screens that allow users to select handouts and send them to the campus laser printer queue for printing.

When a new handout is written, it is saved as a PostScript file from Frame and transferred to a UNIX host with proper file permissions. When the new handout number is added to the menu screen, it is immediately accessible to everyone for printing.

CWIS Access:

The on-line printing mechanism described above allows users to print handouts in their original layout, complete with graphics. But these PostScript files are not particularly useful for people off-campus (who can't use the campus printing queue) or for people who just want to read the information. To meet these needs, an ASCII version of each handout is made available via CWIS (the Campus-Wide Information System). Using a series of menus, people can see the text of each handout, scrolling back and forward as necessary. Of course, ASCII format has many limitations, the most severe of which is the inability to include graphics with the text.

We are also working on making text-only versions of the handouts available for anonymous ftp.

5.2.4 Training for the Campus Community

The fall semester was approaching, and with it came multitudes of faculty, staff and students needing training in the new system.

SFU has provided free computing tutorials for several years, covering mainframe and microcomputer topics, but of course we had no classes for UNIX.

As a one-person documentation and training department, I knew there was not enough time to write courses from scratch. I was fortunate to find a set of PowerPoint tutorials on UNIX written by Cole Whiteman of the Information Technology Division at the University of Michigan. He very kindly allowed SFU to use and modify these courses, and this formed the basis for several lecture demonstrations.

Approximately 10 tutorials were offered in the first semester, including *Getting Started with UNIX*, *Using the UNIX File System*, *Using the UNIX C-shell*, *Using vi*, *Communicating with UNIX from your Mac and IBM PC*, *Introduction to Elm*, *and Introduction to Eudora*. Most tutorials were taught twice in the semester.

This original series has been improved and expanded on, and is now part of our regular tutorial offerings. We have found that the two introductory sessions have by far the best attendance.

5.2.5 Training and Documentation Bibliography

These and other books are suggested as recommended reading in handout B-22. We recommended that the SFU bookstore carry these books, and also placed copies of many UNIX manuals in the reserves list at the campus library.

However, the library's statistics indicate that the reserves collection is being used infrequently, so we will be considering transferring some or all of the books to the circulating collection.

1) *UNIX Communications*, second edition, Bart Anderson, Barry Costales and Harry Henderson (The Waite Group, 1991) pp. 736.

2) *The UNIX C-Shell Field Guide*, Gail Anderson and Paul Anderson (Prentice-Hall, 1986) pp. 374.

3) *The UNIX Operating System*, second edition, Kaare Christian (John Wiley and Sons, 1988) pp. 455.

4) *Life with UNIX*, Don Libes and Sandy Ressler (Prentice-Hall, 1989) pp. 350.

5) *Learning the UNIX Operating System*, Grace Todino and John Strang (O'Reilly and Associates, 1987) pp. 75.

6) *UNIX Primer Plus*, Mitchell Waite, Donald Martin and Stephen Prata (Howard W. Sams and Co., 1983) pp. 414.

6. Operational Issues

This chapter describes the role of Operations in the new environment and how we turned a department accustomed to maintaining a centralized mainframe into one capable of handling a system of networked UNIX boxes. The changing role of the operator and the task of operator re-training is emphasized. One very interesting section describes a rather novel UNIX *Console Station* implemented by Bill Baines and members of OTS to monitor the new group of machines.

6.1 Operators and Operator Training[11]

We were faced in the summer of 1991 with the prospect of running a distributed, UNIX-based computing environment with staff who had little or no knowledge of UNIX. Some were unfamiliar with the popular Macintosh and PC micro-computers.

Needless to say the task of getting 12 operators familiar with UNIX and with personal computing was daunting. I'm pleased to report that we and they were in large part successful.

Our approach was to provide time, some money, tools and a good dose of encouragement to the operators. The tools included instructional video tapes from AT&T on UNIX basics, a color NeXT workstation, a color Mac IIci and eventually a color VAX 3100 workstation to use as a master console for all of the UNIX hosts.

The tapes, while dry, had the effect of providing operators with fundamental UNIX knowledge to build upon. We found that the training process was enhanced by providing a terminal from which they could connect to our original UNIX host (*whistler.sfu.ca*, then a Sun 4/280) and try using the commands as they were explained on the video. We also had them write a quiz to test their knowledge at the end of the video sessions.

The best thing we did was to provide them with the color NeXTStation. This had the maximum amount of sex appeal and really turned the operators on to the possibilities of a high end UNIX workstation. Now the problem switched from one of teaching to one of access. They all wanted to use it and find out how it worked.

The color Mac and the color VAX station came later (we started out with a smaller, gray scale VAX station for the console machine). These tools provided them with access to Eudora (the Macintosh e-mail client) and with a central console location for access to the UNIX hosts.

The learning was largely self-directed at this stage. There was simply too much to cover all at once. What we found was people sharing ideas and knowledge amongst themselves to the benefit of the group. It was most encouraging to watch. As consultants passed through the computer room on their way to do something to a UNIX host they'd be accosted by the operators for information on

[11] Written by Bob Spratt *(bobs@sfu.ca)*

this or that topic. Some operators had taken it upon themselves to take a course or two to help them understand the new environment they'd be dealing with.

As the big day approached for the start of UNIX services we told the operators they'd be responsible for answering the Help line. While the operators had always provided phone assistance, the number of calls and the magnitude of the change to the client community was something quite apart from previous experience.

Operators who had been doing the same things in the same ways for twenty years were suddenly dropped into an environment foreign to them. They now had to answer calls from users who had a myriad of problems, many of a basic nature. Initially, we provided operators with a complete set of the *How To...* documents to learn about the specifics of the new services. While keeping the avenue open to the consultants to provide the bulk of the responses to help requests, we had the operators start to give direct feedback to the help requests in areas they felt confident of providing accurate information.

What started out as a constant phone ringing and answering service (the consultants had all forwarded their calls to the Help line) gradually over the days and weeks of the spring-92 semester settled into periodic flurries of calls when this or that problem arose and became noticeable to the community. The knowledge of the operators grew and continues to grow as they become more familiar with the UNIX/network services. They now answer many of the help requests directly and the number of calls has slowed to several a day from the overwhelming number it once was (it will probably escalate again in the fall when a whole new batch of users shows up and starts to use the services).

Its been a great learning experience for them (trial by fire, so to speak). We are starting a month long training program in mid-June on UNIX system administration for the operators and others. It's a formal presentation and should give the operators a much broader and deeper understanding of the workings of the UNIX operating system.

6.2 The HELP Line[12]

Computer operators are present 24 hours a day, 6 1/2 days a week. It was therefore decided to funnel all user help requests through the operators. Calls are accepted by phone (most of our unattended public terminal areas have a direct phone to the operators) or via e-mail to an e-mail alias *help@sfu.ca*. These requests at present get written out on a UNIX Help Request form and are then distributed to the person that supports the particular area the question is about. The standard is that the user should be contacted within two hours if at all possible. When the problem is resolved, the resolution is written on the form, and the problem is "signed off" along with the date it was resolved on. The form is returned to the operators to be filed. In this way, when a similar or identical question comes in, the operators may be able to answer it directly by referring to previous Help Requests or help summaries.

6.3 UNIX Administration and Operations Issues

This section covers the somewhat painful process of providing adequate backup services in a UNIX environment. If you haven't used UNIX before, you're in for a big suprise!

[12] Contibuted by Peter Van Epp *(vanepp@sfu.ca)*

6.3.1 UNIX Backups[13]

There are several hosts and servers in the cluster of computers that provide the campus-wide UNIX service. The Auspex file server provides the bulk of the disk storage with each host having it's own local disks.

Backup Strategy on the Auspex File Server:

Backup of the `/var/spool/mail` partitions and all the user partitions are done via disk mirroring (weekly for user data and always (weekly and daily incrementals) for `/var/spool/mail`. This gives us the advantage of a dump from a dismounted file system, but still allows the users to continue working on the on-line file system after the mirror is detached, so the users don't see the effects of backup.

Backup Policy:

Once a month a level 0 dump is done to a monthly backup tape for each partition on the Auspex. Four generations of these tapes are kept so we can recover a file up to 4 months old. The complete set of monthly tapes are copied to another set of labeled tapes (the firesave tapes) when the monthly backup is completed. This firesave set is then moved to the vault for protection against a machine room disaster.

There are two generations of firesave tapes (one set in the vault and one set in the machine room ready to be re-used at the next monthly or weekly backup). Since both monthly and weekly backups write to the firesave tapes, there is always one copy of the last level 0 backup in the vault and one copy of the level 0 tapes from the previous firesave backup in the machine room (except when a new firesave is running). This is in addition to the original backup tapes.

Once a week (except on the week the monthly level 0 is done) a level 0 dump is done to a weekly backup tape for each partition on the Auspex. Four generations of these tapes are kept so we can recover a file up to 4 weeks old from these tapes.

Every morning (except if either a weekly or monthly level 0 is being done), a level 5 incremental dump is done for every Auspex partition. These tapes are not fire-saved. We keep 14 generations of these tapes, allowing us to restore a file to the state it was in one day after the last level 0 backup (using the level 0 backup tapes from the previous firesave) all the way up to the state it was in yesterday. As well, if for some reason the weekly (or monthly) level 0 doesn't get run one week, we can still recover from the last level 0 dump (from 2 weeks ago) and the 14 daily tapes.

Tapes:

Each file system has its own set of tapes and all of the tapes used for backup are labeled. The first data block on the tape is an internal label that marks the tape as a labeled tape, gives the tape number for this tape, the date when the tape was initially labeled (and therefore how many times it has been used), the type of the last dump written on the tape (Monthly, Weekly, Daily or Scratch (not used yet), the machine name and partition name (real or virtual) of the dump on the tape, and the time that the label was last written (and therefore the approximate time of the dump).

[13] Contributed by Peter Howard, Peter Van Epp and Mike Dustan *(phh@sfu.ca, vanepp@sfu.ca, mike@sfu.ca)*

The backup keeps track of the tapes used by each generation and file system and will ask for them by number.

The Magic:

Peter Van Epp wrote a number of shell scripts and C programs to update tape directories, tape labels, `/etc/dumpdates`, and log files and to control the mirroring of partitions and to issue the dump commands. (These scripts are nowin use at other Auspex sites.)

Daily backups do not use the mirror volumes at all (except for `/var/spool`).

The weekly and monthly backups work identically (they of course, use different tape directories to get their tapes). The process is much the same as the daily backup, except that the virtual partitions are mirrored to the mirror disk to ensure dumps are restorable and correct before being backed up.

Backup Scheduling:

The backup is divided into two streams with each stream using one of the 2 Exabyte 8mm tape drives built into the file server. All backups and tape copies are done at night since file server performance is degraded too much if run during the day.

Daily backups are run every day. Weekly and monthly backups are run on 2 successive days since each weekly or monthly stream can take up to 8 hours. A weekly or monthly stream runs concurrently with the opposite daily stream. On days when no weekly is scheduled, both daily streams run concurrently.

Firesave copies are run after both weekly or monthly streams are complete.

If a backup has been aborted because of a failure then the backup stream is restarted and continues from where it failed.

6.3.2 Server Backups (BIND/DNS, X.500, NIS)

A Sun product, **Backup CoPilot**, is run daily for incremental backups and weekly for a full backup and a firesave copy for the SPARCserver 2's. It was decided to use this product because it allows the file systems to be backed up without quiescing the machines – the Backup CoPilot version of `dump` imposes locks against certain file system operations during various phases of the dump.

We're satisfied with this product even though there is an outstanding minor problem that prevents listing of labels of tapes used in the database. Sun's response to resolving this problem has been dismal.

IBM RS6000 Backup:

This system is backed up weekly. Currently we use backup by name rather than by i-node since OS-3005 has problems with i-node backups. We'll change to i-node backup when we upgrade to the next OS level in June-92.

Other Backups:

`Dump` or `bru` is used weekly to backup to unlabeled tapes.

Backup Futures:

All backups, except the two SPARCservers using Backup CoPilot, will be changed to use Peter Van Epp's scripts. This will allow for better tape management since tape directories and tape labeling is supported.

6.4 UNIX Console Workstation[14]

As the equipment was being ordered, Bob Spratt started planning changes to the machine room layout. For the fall term we would be running both MTS and UNIX, and with floor space being a tad on the tight side, the new UNIX systems were being delegated to corners, nooks and crannies. It also crossed our minds that every one of these boxes would come with a console terminal of some sort. The vision of more than a dozen VT100 clones scattered about the machine room, all of different shapes, and each with its own features (stupid little differences?) seemed like a retrograde step. Certainly someone, somewhere had some kind of a system that would multiplex a bunch of serial console lines to a single UNIX workstation. We found a write-up in DECprofessional about just such a system, but it was a new product, and it lacked many features.

Bob had seen a presentation at an "Operations Conference" of some kind in California in 1990 where the operations manager of a company that runs a 1-900 telephone service described his console multiplexor for a bunch of VAXen. He used a DEC product called VAX Cluster Console (VCS) which is tailored to VAXen. During the presentation he described an incident when the air conditioners failed and the temperature started rising. The air conditioners were on a different floor, so the operators neither heard, saw, nor smelled anything different. After describing the near meltdown, he went on to describe how he connected the environmental control computer to his VAX Cluster Console system so the operators would be alerted to environmental control problems.

We figured if you can hook an air conditioner to a VCS, then you can probably hook UNIX machines to it. The software was not a problem since SFU is a member of DEC's Campus Software License Grant program and the VCS software came on their CD-ROM distribution. We had an dormant VAXstation 2000 that would be at least good enough for testing the idea. All we needed was a terminal server and cables. We ordered a Cabletron (Xyplex) LAT terminal server and set things up to monitor the SGI and SUN hosts.

The VCS software is big. It can do lots of useful things but the learning curve to configure it is substantial. We proved the principal by operating with six of the UNIX timeshare host consoles connected to the VCS running on the VAXstation 2000. The 2000 had 14 MB of memory but it was just too slow, so slow that many of the operators were reluctant to use it. VCS is built on DEC's VMS version of X-windows and uses Motif-like UID files which bind at run time. This gives the end user lots of freedom with locally built icons and such, but lots of cycles are consumed whenever a new window program is started.

To speed things us, we eventually acquired a used VAXstation 3100/76 with a 19" Trinitron from DEC for half-price and moved the VCS software there.

6.4.1 How Monitoring Works

[14] Contributions by Bill Baines *(bill@sfu.ca)*

In VCS there is a master list of "events". Associated with each event is information like event name, event string, priority and a description of an action to be taken (such as the semi-legenday "Bad Thing – Call Peter Van Epp ASAP!" alert message). Specific events are grouped into "scan profiles". A scan profile can be called by a parent scan profile. For example, we found that the NFS errors coming out of the SUN and SGI consoles were identical while most other SUN and SGI console messages were different. The result is that we have a SUN profile named SFU$SUNOS and an SGI profile called SFU$IRIX – but both of these call the NFS profile SFU$NFS_ERRORS. The UNIX scan profiles are not very large, less than 25 entries while the NFS scan profile has less than 10 entries. The big effort was learning how all these lists worked together.

There are 3 VCS daemon processes. All three use the same binary configuration file which is created with a program called the "configuration editor":

- VCS$IODL is the data logger. On startup it opens and locks channels to the console ports using the LAT protocol.
- VCS$SCANNER scans all console data looking for events.
- VCS$ENS is the event notifier. It processes events and carries out appropriate actions.

The X-windows display is called VCS$C3 for Console Control Coordinator. Each machine being monitored is displayed as a colored icon. When anything out of the ordinary occurs, the icon of the machine in question changes colour to indicate the seriousness of the message (yellow for warning, red for serious trouble). To investigate the problem, operators can open custom terminal windows that allow monitoring and/or connection to the console associated with the machine. Depending on what they see, the operators can then acknowledge or dismiss the alarm(s).

All activities are time stamped and logged to disk.

There is also an interactive VT100 type interface which is sometimes used by the systems consultants to log in over the network and monitor or connect to the console lines. The operators can monitor the consultant's activity on a monitor screen.

Log files are managed by the VCS. They can be compressed regularly, printed daily, or flushed when free disk space reaches a trigger point.

6.4.2 Our VCS Configuration

We have built an iconic view of the UNIX hosts on the machine room floor and we actively monitor about 10 hosts. The host consoles are connected to the Cabletron/Xyplex LAT terminal server. All console activity is logged to disk and is available for review from either a terminal window or a remote telnet session. Scrolling backward and forward through each console log is possible.

All events are enunciated by coloring the icons and printing the event in an text window with the hostname and the event time pre-appended to the console message.

We are monitoring all of the SGI systems, the VAX, and the two Sun 4/470's. We are not yet monitoring the Auspex – mostly because we have not spent time resolving cable problems. We are also not monitoring the Sun SPARCstation servers – we just don't have the luxury of taking the time to figure out the cabling.

6.4.3 Is VCS Worth the Effort?

We think that installing and configuring the VCS was worth it – but it wasn't easy. We definitely under-estimated the effort required to set up an effective configuration.

On the positive side, it has allowed us to monitor most of our UNIX machines from one location and provides us with an X-server so we can implement other applications useful to the operation. We will probably display Nysernet SNMP on the VCS screen real soon (see chapter 9).

6.4.4 VCS Problems

The standard VAX DECterminal/X-terminal window is not appropriate for UNIX. For example, there's no ESCape key! We finally figured out how to work around this – and we recently heard that a UNIX keyboard is available. We also recently discovered a problem with memory de-allocation when terminal windows are closed. Apparently, the terminal server process dies after many terminal windows are created and deleted (DEC is aware of the problem). By logging out and then logging back on every couple of days things seem to work okay.

VMS is not UNIX, and that bothers some people. But until there is an equivalent UNIX version available, the VCS will do.

VCS does requires some horsepower though. A low-end VAXstation won't do. We are using a VAXstation 3100 Model 76 (8 VUP) with 16 Mb of memory. This is adequate under VMS 5.5, but more memory would is required if an earlier version of VMS is used. Color is useful, but the system will use pattern fills if color is absent (but we'd rather have the color since it's easier to pick out a change in color from light-gray to red rather than a change in patterns from sparse gray dots to closely packed gray dots).

We have slides of the VCS setup and they are available for viewing at the conference.

7. Specific Services

This chapter describes the more important services that ACS offers under the new UNIX environment. Implemention and user acceptance of each service is discussed – along with any on-going problems.

7.1 E-Mail[15]

At SFU we use Elm with emacs, Eudora, and POPMail-PC for e-mail. We have about 10,000 accounts in active use – including accounts for the teachers in the province of British Columbia.

Over 700 people use Eudora, mostly via campus ethernet or Appletalk connections, with a significant minority using dial-up Eudora from off-campue. About 100 people use POPMail PC. We expect many more IBM-PC users to be interested in using a POP product instead of `elm` when better POP clients become available (NUPOP/PC from North Western looks very promising).

Some administrative employees use VAXmail supported by their own computing people, a couple of departments use Pegasus Mail on Novell with the Charon gateway, and one department (self-supporting) uses Banyan-Vines mail.

[15] Written by Frances Atkinson, *(frances@sfu.ca)*

The SPARCstation *whistler.sfu.ca* serves as a mail hub machine that handles mail from all the other UNIX machines, POP clients running off of Macintoshes and IBM PCs, various gateways, desktop Suns, and NeXTs. The `/etc/sendmail.cf` file on whistler has been set up to re-write all addresses except those coming from gateways with a destination of "@sfu.ca".

All SFU faculty, staff and students, plus the B.C. teachers (ie, everyone owning a UNIX account) have e-mail addresses in the format `personal_name@sfu.ca` set up in the `/etc/aliases` file. People who use other mail systems re-route mail addressed to `personal_name@sfu.ca` via `.forward` files. Some users have had `.forward` files set up for them (typically users with their own computing support people such as the administration people on VAXmail) while others do it themselves.

We have a BITNET connection on a VAX, running MX and JNet. Outgoing BITNET mail from the main UNIX facilities is routed through this VAX, while incoming BITNET mail is routed back from the VAX to people's UNIX mailboxes.

The X.500 Quipu software is used for directory service. Eudora users can query the directory using the `Ph` menu item in Eudora. A modification was made by Ray Davison at the server end to query the X.500 directory rather than a Ph-server. Others can log into UNIX and use the `ud` command (from the University of Michigan) to query the directory. Rob Urquhart is about to release a Macintosh client application for the X.500 directory (see the X.500 section below).

For mailing lists (e-mail groups), Rob Urquhart produced a UNIX command called `maillist` that offers the range of function people had on MTS (create lists, add members, delete members, join public lists, resign from them, show existing lists, show membership of lists). Over 350 lists have been set up by users since January. The old MTS system had about 900 lists set up over 6 years, including many obsolete ones, so this facility appears to have met the need.

7.1.1 Making the Break

The basic infrastructure for the new e-mail facilities was largely in place by September 1991. By that time, accounts had been created and distributed, software had been installed, configurations set up, and user documentation written. Between October and December 1991, people were encouraged to migrate over to the new facilities from MTS. We wrote and taught tutorials (quickly revising them as needed), conducted faculty information meetings, distributed documentation by the thousands of pages, and talked to individuals.

Although many people tried out the new facilities, most stayed with MTS as their regular e-mail system until the deadline (in the fall of 1991 the number of messages being exchanged via MTS was higher than it had ever been). Thus, when MTS was finally shut down on Jan 2 1992, most people moved over to the new system cold-turkey.

For e-mail, it was generally successful, with a lot less mayhem than expected. Partly this was due to the fact that people had to change their ways overnight for all their computing needs (and not just e-mail).

7.1.2 Adapting to the New E-Mail

Relatively speaking, e-mail was accepted well. Some users temporarily turned their backs on e-mail (it's hard to estimate how many) but we have such a strong e-mail culture on this campus that they have gradually come back.

Most of the problems had to do with details, rather than with fundamental aspects of the service, and were mostly to do with breaking old MTS habits.

The MTS practice of typing a recipient's name in the `To:` line with a space in the middle caused much confusion when people did the same thing on UNIX, For example, they would send mail to Frances Atkinson according to their old MTS habit, with the result that mail would go to the account `frances` and account `atkinson` instead of the mail alias `frances\_atkinson`. This resulted in large volumes of mail going astray until people learned to replace blanks with underscores. Compounding the confusion, people could no longer look in an outgoing directory to track what had happened to their message, as they had long done on MTS.

Finding addresses was another major source of confusion. In the absence of a dynamic link between the e-mail system and directory system (like on MTS), people had no reliable way of finding addresses (the X.500 directory was not available until February 1992). Further, mail aliases had been set up from data from Registrars and Personnel, and this data used proper names (yielding aliases like William_Brown instead of Bill_Brown) or first initials (yielding I_Roberts instead of Ian_Roberts). Once it was clear addressing was becoming a very hot item, we produced a printed directory from the mail aliases file and distributing copies to every department. That defused things until the on-line directory became available. Over time, we altered mail aliases upon request so that people use more familiar forms of their name.

Elm was initially configured to use the `vi` editor. An early decision to install `.elm/elmrc` files with `editor=none` in all accounts proved wise in a campus that was largely UNIX-illiterate – the early users of `elm` using `vi` were going nuts. Elm itself was re-configured to use emacs as a global editor for situations where it must enter an editor. The goal was to allow people to get by using the on-screen prompts without being tripped up by such things as not knowing how to exit an editor. The documentation shows people how to set `editor=emacs` in their `.elm/elmrc` files if they wish. This approach seems to have worked well.

Eudora has become very popular. Most people have it configured for them by Computing Services staff when they install a new network connection in an office or by local departmental computing support staff. The main problem with Eudora is that it has sometimes been tricky to effectively troubleshoot problems because there are so many variables (such as the versions of Eudora, MacTCP and/or the Mac System, configuration of Eudora, switches set in Eudora, and the method of connection). For dial-up access, Ray Davison wrote new scripting facilities to manage the dial-in connection in a more powerful way – including dialing in via Datapac. People have used dialup Eudora successfully on portable Macs from hotels across Canada. The scripting enhancements to Eudora are available from `ftpserver.sfu.ca` if anyone is interested.

On the IBM PC side, the people using POPMail-PC like it. What we made available was a locally-enhanced version to provide a directory screen and improved memory handling, but work on that has been discontinued since NUPOP/PC seems to provide the functionality people are looking for. We never did actively promote POPMail-PC but made it available on an trial basis to various departmental computing support people, some of who installed it throughout their department.

We went to considerable effort to identify people in every department who would serve as front-line contact people for computing. These people are informed early about changes, problems, new versions of programs etc. They receive copies of all new documentation as soon as it is printed, including updates.

Our documentation scheme, being very modular, has worked well for e-mail in terms of keeping information up to date and producing new *How To...*'s to fill voids, as we become aware of the gaps in people's knowledge and of things that are causing particular difficulties.

7.2 Application Software

The applications most in demand under UNIX are statistical packages and the language compilers. Both are discussed in detail. This is followed by a short description of our implementation of TEX under UNIX.

7.2.1 Statistics in the '90s at SFU[16]

Most statistics users had little difficulty in getting off MTS due to availability of many popular statistics packages on UNIX. Microcomputers are powerful enough for most of them so that many have opted to take better advantage of their desktop computer rather learn UNIX. We were unable to support TROLL and LISREL.

The functional replacement for TROLL is SHAZAM, TSP, SAS/ETS, or LIMDEP. SHAZAM, one of the most heavily used packages on MTS due to student use, is now distributed free of charge for home use; SHAZAM-related class assignments are oriented towards using the IBM PC or Macintosh labs. LIMDEP, a recent acquisition previously unavailable on MTS, has more functional capability for econometrics than any of the other packages including TROLL.

The functional replacement for LISREL is EQS for the Macintosh, EzPATH for the IBM PC, or CALIS (Covariance Analysis of LInear Structures) available through SAS/STAT. CALIS provides the most functionality and is the recommended one for large models. EQS can also manage large models if you don't mind letting your Macintosh run for several days. EzPATH is the most limited in capacity but is very useful for small models and instructional purposes. The vendor for LISREL has announced their intention to make a SUN version available but they have yet to deliver. Since many journal editors insist on LISREL despite the fact that the other packages give identical results, we may support it once again.

We recently re negotiated our license with SAS Institute. Whereas we previously had only Base SAS and SAS/STAT on MTS, we now have a total of twelve SAS products available on UNIX: Base SAS, SAS/STAT, SAS/GRAPH, SAS/FSP, SAS/AF, SAS/ASSIST, SAS/INSIGHT, SAS/ETS, SAS/OR, SAS/QC, SAS/CONNECT, and SAS/IML.

Whereas we had only the SPSS Tables add-on product for SPSS under MTS, we now include SPSS Graphics and SPSS Categories under UNIX. We look forward to having SPSS for Windows, yet to be released. Many faculty have opted for SPSS for the Macintosh or SPSS/PC+ in lieu of the UNIX version. The microcomputer versions are more heavily used for instructional purposes with the demise of MTS.

We were one of the first to license BMDP for the SGI. We hope to convince them to make it available for the NeXT as well. Recently, BMDP Inc. has released BMDP 383 Dynamic under DOS which will analyze data requiring up to 16 MB. This will be the option of choice for the BMDP user wanting to avoid UNIX.

[16] Written by Reo Audette *(reo@sfu.ca)*

SCA is one of the best products available for time series analysis. We were the first customer to license SCA for the SGI. Unfortunately, the DOS version does not support the functionality of the UNIX version, but it is useful for instructional purposes at the undergraduate level. S-PLUS, unavailable on MTS, is new to us in our UNIX environment. Used extensively for research into statistics, it is growing in popularity in the social sciences in general.

Because statistics users traditionally generate large ASCII printouts, the ability to print two-up on our Xerox 4090 was initially missed. The acquisition of the public domain `mpage` for UNIX has saved the day. It prints two-up not only on the Xerox printer but on our distributed HP LaserJet IIIsi printers as well. Our users can double their throughput and half the cost of printing. It is evident the statistics software developers need to learn more about ASCII printers however. Many of them start their output files with a page eject command causing the first page of each printout to be blank.

Although we have yet to take great advantage of it, we have X-Window support for SAS, S-PLUS, BMDP and SPSS/Graphics. We expect this to be one of our most important contributions to the metamorphosis of UNIX at SFU in 1992. Perhaps our most important contribution to ease the shock of UNIX has been our extremely popular one-page *How To...*'s. Much effort has been put into writing these handouts which far surpasses any commercial text on how to use UNIX to its fullest advantage. Meant to be read piecemeal to make UNIX manageable to the novice user, they can be adapted for any UNIX shop. We will supply copies of them to anyone free of charge.

7.2.2 Compilers and Their Users[17]

We offered to help users convert their custom programs from MTS to UNIX, and we got lots of requests for help! We will describe our experiences, listed according the language.

In general, the conversion tasks were relatively simple with about ten or fifteen jobs that could best be described as time-and-labour intensive. These jobs made extensive use of MTS-specific routines such as IG (Integrated Graphics) or the MTS System Subroutine Library.

FORTRAN: This was the most common language. The programs that did not have MTS dependencies were straightforward to convert (generally just adding some OPEN statements). Programs which called IMSL and NAG were no problem either – these libraries were already available under UNIX. The biggest problem came from programs that called IG routines. In most cases, we were able to replace these with calls to DISSPLA routines. Some of these programs were converted to run on high-performance PCs. In these cases, we used the graphics libraries that were provided with the FORTRAN compilers, such as Lahey FORTRAN and MicroSoft FORTRAN. Ken Urquhart ended up writing an entire device-independent 2-D plotting package that allows users to create x-y plots with a few function calls on their PCs. It will soon be released into the public domain and is available for testing upon request.

Interactive FORTRAN: Many of the debugging facilities of *IF are provided by the UNIX dbx utility. However, dbx does not check for mismatched argument types or undefined variables. Steve Kloster found a program called Forchek which does check for these errors. It is available from netlib@ornl.gov by sending the message "send forchek from FORTRAN".

APL: Although Steve encouraged APL users to move to another language for some time, there were still some APL programs around. We helped some of them convert their programs to micros

[17] Contributed by Steve Kloster and Ken Urquhart *(stevek@sfu.ca, urquhart@sfu.ca)*

using APL68000 on the Mac, for example. Also, we bought Dyalog APL for Silicon Graphics and one of our users wrote a program to convert MTS workspaces into Dyalog workspaces. Terminal emulators had lots of problems with the APL character set. We advised users to switch to a system like S-PLUS or Maple V instead of APL if possible.

C: Both Sun and SGI have C compilers. We got C++ for the Sun. A few people are using Gnu C, but we don't encourage it, as there are always concerns about support. Good C compilers are available for nearly all micro-computers.

PL/1: We bought a PL/1 compiler from Language Processors Inc. for the Sun.

Pascal, Modula-2, COBOL: We bought these compilers from Sun.

LISP: We are using Ibuki Common Lisp on the Sun and Allegro Common LISP on the NeXT workstations.

BASIC: All BASIC users migrated to micros.

SIMSCRIPT: We got the PC version but it is not yet available. We need to run this over a Novell network and there are some incompatibilities which have not yet been resolved.

REDUCE: This was replaced by Maple V under UNIX and Mathematica on the NeXT, Macintosh and PCs – these applications have much better graphics as well as improved integration algorithms.

CSMP: We got ACSL for the SGI and converted users to this.

Graphics: The X-Window system is a great improvement over plain Tektronix emulators because users can display raster images and use color. We are currently using DISSPLA, S-PLUS, and Maple V to do graphics. We hope to expand our offerings in the future. DISSPLA users on MTS were able to convert to DISSPLA on UNIX without much trouble. TELLAGRAF users have converted to micros, using programs such as DeltaGraph, Kaleidagraph and Harvard Graphics. Die-hard TELLAGRAF users can make use of the DISSPLA CodeBook option that accepts TELLAGRAF-like commands and outputs FORTRAN programs whose sole purpose is to generate the desired plots.

7.2.3 Text Processing[18]

There were two problems here: converting TEXTFORM files to some form the a word processor or TEX could understand and implementing TEX on UNIX as a functional replacement for TEXTFORM.

To get a feel for who was still using TEXTFORM, I totaled the usage counts for four months for all MTS IDs and sent a message to those who had used TEXTFORM more than a few times in that period. As we suspected, there were only a handful of serious TEXTFORM users left—most had converted to word processors or TEX over the past few years as we began to encourage people to move away from TEXTFORM.

For people who were determined to stay with some form of "mainframe" text formatter, Margaret Sharon wrote two conversion utilities in SPITBOL to translate TEXTFORM source code into

[18] Contributed by Margaret Sharon, *(margaret@sfu.ca)*

Microsoft RTF and LaTeX. The converters and info on where to get SPITBOL for the Mac and DOS are available from Margaret *(margaret@sfu.ca)*. The TEXTFORM convertors will do an acceptable job on most TEXTFORM input files but give far less than perfect results when confronted with complicated MACROS and TABLES. We make no appologies. There are only so many hours in the day and so very few "power" TEXTFORM users. It was more cost effective in most cases to do a partial conversion and have the power-users finish their conversions manually.

In the end, there were only three people who had "extreme hardship" imposed upon them by the UNIX migration. They were all faculty members who had begun large research projects several years ago and had became "tied" to TEXTFORM. The best solution for two of them was to pay for MTS accounts at UBC and let them finish the work there (estimated completion times for both projects is under one year).

The third researcher accepted the results of the TEXTFORM to RTF convertor, moved the resulting files over to his NeXT, and finished conversion by hand using the FrameMaker word processor (which reads RTF formatted files).

In spite of many years of encouraging people to use the inexpensive yet powerful word processors on the Macintosh and PC platforms, we still have users who wish to use TEX for one reason or another. Some need it because their journal editors only accept TEX , other use it because they have collaborators at other universities who use it, and still others use it because they have absolutely no intention or desire to acquire a personal computer.

In any case, we have placed TEX on all of our UNIX machines. TEX comes free on the NeXT and we were able to purchase several different commercial versions of TEX for the Sun and SGI machines (contact Margaret for details). We have also installed the many popular macro packages for TEX such as LaTeX and AMSTeX. We will install others upon request if we can locate them somewhere out on the network or obtain them from suppliers (often free of charge).

Margaret Sharon and Ken Urquhart support TEX on campus. There are no tutorials on TEX although there are several of the *How To...* handouts devoted to using TEX at SFU. Users are encouraged to purchase one or more of the TEX books out on the market.

7.3 Printing

7.3.1 Campus-Wide Novell Printing[19]

The challenge here was to provide a laser printing service that could be accessed by *all* users, Unix, PC and Mac sessions and which would provide convenient pick up points across campus. In addition, a means was required to charge users for output since a high-level committee decision had mandated an end to "free" printing – which MTS and the 4090 printer had hooked us all on. Fortunately a solution was already in active service in the microcomputer labs that I oversee.

There had never been free laser printing in these labs – right from the start we had attached magnetic card readers to our laser printers – these cards deduct a preset amount from a user's laser printing card for each page printed and make the printer signal "out of paper" when the user is out of funds. Initially users would bring their document on a disk to the computer attached to the printer – we could not use spooling because there was no guarantee that the user would be at the

[19] Written by David Stratton *(david@sfu.ca)*

printer when their job turned up. The users had to stand in line waiting while people "just put the final touches" to their documents before printing them – we solved this problem by crazy-gluing the keys on the keyboard to prevent data entry!

What we needed was a way to make sure that User A was at the printer when User A's job was being printed and this led to the concept of the "Release Console". We wrote software that would run on a network-connected PC located at each printing station. The PC would present the user with a list of the jobs they had previously submitted to the Novell print spooler. The user would then select the desired job and *then* put their laser printing card into the attached reader.

This scheme, already in use in all our micro labs, was the solution to the Unix printing problem. The Novell Version 3.11 print services on which this was based were uniquely able to provide "native" print services to all platforms – `lpr` service for Unix (using the PLPD Network Loadable Module), redirection of LPT ports for printing directly from PC applications (a standard Novell "capture") and printing via the Chooser for Macintoshes (via the ATPS NLM). Having sent one's data to one of the print queues on a Novell server by one of these means, the user then goes to the nearest (or least busy) printing station and releases their job. Spool from anywhere, print anywhere is the name of the game.

This scheme will work with any type of printer, even a dot-matrix, but the big strength of what we have installed is the HP IIIsi printer. This dual-mode (Postscript/PCL), 17 page per minute, duplex, ethernet printer has the great attraction of one of the lowest per page consumable costs in the business – 1.5 cents/page for toner. This has meant that we can still make money charging 5 cents per sheet (we charge the same for simplex or duplex output which encourages saving of paper). The money that we make is ploughed back into expanding the printing service and maintaining the printers. Our 6 public access printers will have printed well over a million sheets by the end of their first year of service.

We have had two major problems:

1) Bugs (now fixed) in the PLPD NLM which caused Unix printing to go up and down like a yo-yo for a few weeks.

2) The inablity of the HP IIIsi's to be completely reset via their ethernet connection lead to problems when User A walked away without completing their job – when User B put their card in the reader, they would start to pay for the balance of User A's job. Result: unhappy users. The solution has been provided by the card reader company. The card readers now send a hardware reset to the printer if 20 seconds elapses with no card or funds – this kills User A's job stone dead.

7.3.2 Xerox Printing[20]

The Xerox 4090 Printer is only for large ASCII and Postscript print jobs or smaller print jobs with multiple copies. Other printing is to be done on lower capacity printers distributed throughout the campus. The 4090 prints from UNIX using a product called Soleil from Xerox. Soleil does the PostScript decomposition on a SPARCstation 2, creating an Interpress bitmap for each page of output. The interpress data is then sent to the ethernet card in the printer using an XNS protocol stack in the SPARC. If more than one copy of a document is required, the PostScript is decomposed once, and the printer prints as many copies as required at close to rated engine speed.

[20] Contributed by Peter Howard and Peter Van Epp *(phh@sfu.ca, vanepp@sfu.ca)*

Xerox Font File And Maintenance:

MTS was used as a repository for 4090 font files, patch files and user generated EBCDIC data files (forms, JSL) and an UBC program, `hostcopy`, was then used to download these file from MTS to the 4090. Simple and fast!

With the disappearance of MTS, fonts and forms were reduced considerably. Most fonts and forms are now primarily used for our Administrative VAX printing. Since there are few fonts and forms, Operations backs up these files onto 5.25 floppies using Xerox's 4090 antiquated floppy drive. All files, excluding Xerox patch and sysgen files, are on 3 floppies.

The Administrative group recently decided to create their own forms. Operations were reluctant to let them use the Xerox editor since it meant interrupting the 4090 printing whenever a forms file needed editing. Instead, the Admin. group edits form files on an IBM PS/2, then use a Xerox product, `Elixir`, to copy these files to the PS/2's 5.25 floppy drive. These files are restored onto the 4090 and compiled.

Postscript and Xerox Soleil:

PostScript printing with Soleil release 9.1.1 still needs some work:

- Adobe Type Manager (ATM) on the Macintosh is not supported.
- Some EPS files won't print completely.
- Appletalk time-outs result in jobs having to be resubmitted.

Xerox says most of these problems will be fixed in Soleil release 9.1.2 which should be available the time you read this. Stay tuned.

At this point, due to the requirement to charge back for printing on the 4090, and therefore the requirement to know who is doing the printing, it is likely that the native Soleil Appletalk support will be dropped and Mac printing to the 4090 will be done via the Novell print server (after the user logs in with a password to the Novell server). There are still several problems with the transmission of XNS masters between the SPARC and the 4090 resulting in lost jobs. The new release is supposed to fix some of these problems as well.

Printer Accounting:

As pointed out above, the Novell distributed printing service is charged back via Laser Debit cards and release terminals. This is not flexible enough for some types of printers (such as the 4090). As a result of a joint project with our Computing Science Department, we now have a database that runs under Sybase that can account for printing on various types of printers. The model is quite simple: if you want to print, you must have a positive balance. If you go negative, then you can't print anymore until your balance is positive. Money is added to the database by departmental advisors and can come from a departmental budget or from a cash payment made to the cashier's office. If the printing request is from an account on one of our ACS hosts (or from any "trusted" host that is using our NIS domain – ie, our Novell server), then that *login account* is charged. For departments that run their own servers,the account name that the database charges by is the *server-name* submitting the job. We accept and record the user name that the server supplied us, but we charge against the server's pot of money. At regular intervals we will provide the owner

of the server with a statement showing the breakdown of charges by the user IDs provided by the server. This means that the security and identity of the people printing are the responsibility of the server administrator – we only extend trust to their server, not their users. In this way, a security breach on their server does not allow the attacker to charge accounts that are under our control (ie, UNIX ids from our NIS domain), only the pot of money belonging to that server.

If another department has a printer that they wish to let their people print on, then it too can connect to this service. The database keeps track of which printer a given job was printed on, and charges the user at the rate set for that printer. On a set cycle, the money collected in the database for each printer will be credited to the owner of that printer to pay for supplies. This allows departmental printers to have restrictions as to which users can print on them without requiring the use of the laser cards. Under this scheme, the 4090 will be just another printer – probably a little cheaper than most.

ASCII Printing:

Users requested a font which included box characters so they could print BMDP jobs. Xerox added the appropriate font only to find line spacing incorrect (what should have fit on one page took 1.5 pages). Unfortunately with this release line spacing is controlled by the IP toolkit which doesn't allow for line spacing adjustment. This problem will be resolved with the next release of Soleil since DJDE pass through will be re enabled.

No provision to shift printing (it was documented but not implemented). This again will be fixed with DJDE pass through. We've installed a multiple-up printing glp called `mpage` on all the hosts. It will do 2-, 4- and 8-up in portrait or landscape on text or PostScript files, although it tends to choke on non-conforming PostScript (such as that produced by dvitps). `mpage` is especially popular with the stats folks for saving paper.

7.4 Electronic Conferencing – Life After *FORUM[21]

We have installed the PARTI (also called PARTICIPATE) conferencing system on UNIX as a replacement for *FORUM. A committee in the Education department recommended PARTI as the best choice of a conferencing system running under Unix.

PARTI seems to mainly appeal to students. One of the more active conferences is called "Farside", a sort of student electronic graffitti board. The education department also promotes it for their off-campus users and for classroom use.

PARTI does not at all have the activity that Forum did. Some people prefer to use netnews and join electronic mailing lists because of the greater global appeal.

PARTI Pluses:

- PARTI allows hierarchical discussions. One discussion can spawn sub-discussions, for example. PARTI offers most of the features that *FORUM offered, plus more (ie, one can scan for a particular word used in a discussion).
- Each participant is identified by Unix name (no more hiding under pseudo-names).

[21] Contributed by Wolfgang Richter *(wolfgang@sfu.ca)*

PARTI Minuses:

- Reference documentation is poorly organized. Remedied by a one-page *"How To..."* hand-out that we produced.
- Interface and display of discussion responses is somewhat ugly, even when compared to *FORUM. Takes a bit of getting-used-to.
- Bugs. Though most of the major bugs (such as those that made the system crash) are now fixed, we still get the occasional "segmentation fault" messages when doing a particular operation. Some of the problems have occurred because PARTI was developed for a Sun platform and we are running PARTI on a Silicon Graphics platform (ie, incompatibilities between BSD and SRV5).

7.5 Tapes[22]

There should be at least one other paper on this subject by Michael Betker, the student who worked on the *FS tape conversion project. See that paper for more details.

7.5.1 *FSTapes

Early on, I convinced Bill Baines that since VMS had 3420 and 3480 tape drives (and a SCSI adapter for the 3480s) then the VAX was the obvious machine to do tape support. We could easily copy from one format to another if we strung an 8mm, a 4mm, and a 1/4 inch cart off the SCSI port on the VAX. Having done the hard work of getting rid of tape support, I grabbed the UNIX *FS tape code from UM (by way of John Stevens at UBC. This looked a little too AIX specific so I filed the serial numbers off and did a little modifying. I freely stole ideas, read the MTS source to see what did what, and rewrote the whole thing. For portability, the code makes no assumptions about byte order on the target machine. I know the byte order on the tape and read the data, one byte at a time, into the low byte of a UNIX `int` or `long`. When another byte is needed to form an `int` or `long`, I just shift left eight bits. The result of this is that UNIX magically puts the data in whatever byte order it wants. The down side of course is that it is very inefficient.

Once the data was in a form that UNIX liked, it was necessary only to set up some structures to read the headers and data blocks and calculate the check-sums of the blocks. Again, this was reasonably easy and carried out over the course of a couple of weeks.

You do have to keep a few things in mind while you do this:

1) What do you do with the MTS line numbers?

2) How do you decide whether to translate the file to ascii?

3) How does UNIX read any of this after you write it to a tape format that UNIX likes?

These are all fairly easy. You start at (3) and work you way back You use `tar` as the format that UNIX understands and you don't translate the data at all – you assume the owner will know what it is. You simply write the data as a UNIX file (ie, a stream of bytes) exactly as it comes off of the

[22] Contributed by Peter Van Epp *(vanepp@sfu.ca)*

tape. The line number information gets written into a parallel file in the form of a series of MTS (line-number/ file-offset pairs), one for each line in the MTS file. The MTS line-number is an ascii representation of the MTS line-number (without the ".") and the offset is the number returned by `ftell()` just before you write the associated line of data into the UNIX file.

This data format preserves all the information contained in any kind of MTS file. Plain binary data, where the line numbers are not significant, is contained in the UNIX data file ready for use. A binary file indexed by line number can be accessed with a subroutine that uses the "line-number/offset" file to get the offset corresponding to a given MTS line number and then setting the file mark with `lseek()` prior to reading the data. Text files use a simple UNIX filter that we wrote. The filter opens both the binary data file and the line-number file. The line-number file is used to access in the binary data on a line-by-line basis. The filter then converts the data from EBCDIC to ASCII and then writes it a byte at a time to the output file. When all the data for one line has been written (as determined by the line-number file), a newline character is appended to the output file, creating an ASCII text file.

At this point, all the interesting parts of the project are done and the drudge work was "left as an exercise for a student" (ie, Mike "cannon fodder" Betker) to complete the grand plan.

This eventually got done it turned out (as these things tend to do) to be a lot more work than we had initially thought. The end result of all of this is a process that will take *FS tapes, read them on the VAX, and spit out a UNIX `tar` file with the data from the FS tape. The `tar` file is transferred to UNIX, untarred, and the data is used as-is or passed through one of the filters mentioned above.

7.5.2 The Final MTS Backup

Since a mistake once made is worth repeating, the next feat of do daring was to figure out how to get a final snap shot of the MTS file system before shutting down. Ray Davison made the mistake of not attending a morning status meeting one day and was elected to write "The MTS Final Backup (TM)". This consisted of taking most of the files on MTS (we skipped the system ids), converting them to the same format as the FS tape format (ie, a data file and line-number/offset file) and writing the whole mess out to a 3480 tape. This was done but the file format came out a little differently due to a technical hitch of some kind. This caused further problems with the VAX integration, but it was a good learning experience for the student and I didn't have to do it.

The final backup is not the most robust piece of code, nor the most efficient. The final backup of the roughly 15 GB of data took four days and died many, many times. There was much editing of files and restarting at tape boundaries, but it did work and we can recover users files off of the tapes without the use of an MTS system. As with all "two day" jobs, this ended up taking two weeks. But I'm sure the Professor from Biology that came back from a sabbatical year only to be told that MTS was no longer available has our thanks. His salmon population growth models are now safely in his UNIX account.

7.6 Datapac[23]

The removal of MTS and SFUNet meant that a new Datapac gateway had to be found to support the Datapac service. We had a VAX 8530 CPU available and had heard good things about Digital

[23] Contributed by Randy Raine *(randy@sfu.ca)*

Equipment's X.25 product offering, VAX P.S.I. (Packetnet System Interface) so the decision was made to implement the service on this platform. A piece of hardware called a MicroServer was purchased from DEC to off load the packetizing from the VAX 8530.

Our initial problem was to find a way to allow users to connect to the VAX without the need of a VAX ID. We didn't want to duplicate the thousands of UNIX ids which had been created. At the same time, we wanted our Datapac service to be restricted to Simon Fraser University students, faculty and staff. This was handled differently for incoming and outgoing calls.

7.6.1 Incoming Calls

When an incoming call is received, a 'captive' process is automatically started on the VAX. The user receives some informational messages and is left at a prompt requesting the name of a host to connect to. Users can only telnet to SFU hosts – connections to foreign hosts is not allowed. Once they have actually signed onto a campus UNIX machine, they are then free to telnet off campus, as we can now trace the origin of the call. This service has been heavily used with over 1500 incoming calls on some days.

7.6.2 Outgoing Calls

To place an outgoing call, users telnet to our Datapac host and sign on with a common ID that does not require a password. They are again placed into a captive environment which only allows calls to a select number of destinations. If they require access to destinations not on the list, arrangements have to be made to obtain their own ID on the Datapac host which they can use for calls. Demand has been surprisingly light - ten calls a day would be a heavy day.

The new service had some teething problems. Flow control was the most difficult to solve. Once the configuration problems were all ironed out things have worked very well. Judging by the Datapac bills now being received - they have tripled from the MTS days – the service could be viewed as being too successful!

7.7 Library and Research Data Library

Library access under MTS was a very popular service. Users could attach to the GEAC on-line library catalog via SFUNet and carry out numerous database searches through MTS and *SPIRES. There was a good deal of anxiety about losing these services under UNIX. Luckily, we have been able to provide functional replacements for nearly all of the former services.

7.7.1 GEAC[24]

The SFU library catalog runs on a GEAC system. In the MTS times it was accessed via an X.25 connection into one of the SFUnet Nodes. Since SFUnet was going, this service ended up being moved to the VAX doing Datapac (and therefore X.25 support). Some combination of the VAX TCP/IP support and DCL magic allows a user out on the network to telnet to *library.sfu.ca* (which is an alias for *athens.sfu.ca*) where the incoming call is converted to a Datapac call to the library GEAC. This same X.25 link provides access from the terminals connected to the GEAC to the French Library System running under BRS on the VAX.

[24] Contributed by Peter Van Epp *(vanepp@sfu.ca)*

7.7.2 Spires[25]

With MTS gone and subsequently SPIRES, what have we done in terms of replacement? In a nutshell here's what we did:

1) Most of the applications using SPIRES were public on-line databases such as the Grolier encyclopedia, Microlog, etc. We hired a co-op student who in a 4 month period converted these to run under BRS running on a VAX computer. Since BRS comes with a user-friendly, menu-driven searching tool called "onsite", we did not have to develop a front-end access to the databases as was the case for Spires.

2) Other applications (such as our Bulletin mailing list, and a billing system for the Print Shop) were easily converted to FileMaker Pro running on the Mac.

3) This left about 3 user applications, for which limited multi-user access via UNIX was required. Two of them I painstakingly converted to run under Sybase (of which we have a 8 user license). One of them I was able to easily convert to BRS.

7.8 X.500[26]

The shutdown of MTS meant that users would no longer have *USERDIRECTORY to look up the e-mail names of friends and co-workers. To supply an on-line directory service under UNIX, we decided to run an X.500 server using the QUIPU software that comes as part of the ISODE package. Why not use the `ph` system from the University of Illinois? Well, we were already experimenting with `quipu`, running a test directory service with faculty and staff data. Further, X.500 seemed to be catching on with the Internet community (in the US and Europe if not in Canada) and we are of the opinion that it has more of a future than `ph` (X.500 is likely one of the *better* ISO standards).

The initial service was run on a SPARC I with 8 MB memory. This service had poor performance, and was unable to handle the full database of staff, faculty, and student entries. The performance problems were due to excessive paging caused by `quipu`'s memory-resident database. This was apparently also the cause of frequent crashes. Starting the Directory was extremely slow – approximately 25 minutes to load the database (mostly due to paging).

The current X.500 directory service runs on *whistler.sfu.ca*, a Sun SPARCstation 2 with 64 MB of main memory. The machine also runs BIND/DNS and sendmail.

The X.500 software consists of:

- Quipu (ISODE 7.0 with patch #1).
- Dixie 1.3 server - provides lightweight access to the directory service for non-OSI clients (ie, Macintosh, PCs).
- `address\_of` – simple client which does a lookup of a name and returns department and e-mail addresses. Developed in-house.
- `ud` – UNIX directory client. Part of the dixie package. Some local mods which are being rolled into the dixie release. Mostly bug fixes and some patches for the SIG's.

[25] Written by Wolfgang Richter *(wolfgang@sfu.ca)*
[26] Written by Robert Urquhart *(robert@sfu.ca)*

- `FIND-500` – In-house developed Mac client. Uses dixie protocol.

Note that we have no DOS or Windows clients yet, but they are planned for the future – unless someone writes them first!

Quipu has been configured with the "Turbo" options, which replace the text database files with `gdbm` files. This has no effect on search, list, or read operations (since they don't access the files) but greatly speeds up modify operations.

It is also configured to build indexes for the following attributes – `commonName`, `surname`, and `userid`. This makes searching much faster at the expense of a longer load time and a larger process size. The database has entries for all staff, faculty, grad and undergrad students, and some external users – 27248 entries in all. The process size is approximately 42.5 MB.

Performance is quite good. The directory loads in 3 to 5 minutes, and has proven to be very reliable. Using `ud` to search for an entry gives sub-second response time. The `address-of` program takes only slightly longer.

User Agents:

So far, the choice of user agents has been poor. None of the user agents that come with `quipu` are acceptable for end-users – they require X.500 knowledge (DISH), they are buggy (especially the X-clients), or they are just ugly and complex (FRED).

`Address_of` was originally a shell program using DISH commands, but has been converted to a DIXIE client written in C. To use it, the user enters `address_of bill baines` and gets back the department and e-mail addresses of Bill Baines. It searches for full name, surname, and then by soundex. The user has no control over the search.

`ud` is one step up. It allows the user to move around the directory tree and search in other countries/organizations. It is reasonably friendly and based on MTS *USERDIRECTORY syntax for want of a better syntax. `ud` and `address_of` are running on all Sun and SGI machines.

`FIND-500` (`Find Is Not Dish`) has only just been released so I don't have any feedback on it. Initial response from some beta-testers has been positive. It is intended for a user with no knowledge of X.500 – you never have to enter an X.500 distinguished name to use it. It allows easy browsing through the directory tree, searching, and modify capability.

Problems:

There are no outstanding problems with the server software. The only problems are administrative (how to keep the data in sync with other databases). I get the data that is produced when accounts are generated and run it through a program that generates new entries. There is no link between the account creation and generation of directory entries. Periodically I ftp the accounting files and run the directory job. At this point I am just adding new accounts. When faculty/staff leave, their accounts are not deleted (yet) so the directory entries remain. When faculty/staff change departments there is no simple way for me to be notified, so the directory slowly gets out-of-date. These kinds of problems are not specifically X.500 problems. They would exist with any directory service.

7.9 BITNET

BITNET was used primarily for exchanging e-mail with other BITNET sites. File transfer across BITNET was used infrequently.

7.9.1 Moving BITNET from MTS to VMS[27]

In choosing a replacement BITNET service for Simon Fraser University, Computing Services considered two options: `urep` (which runs on a UNIX platform) and `JNet` (which runs on a VMS platform). Since many of our replacement services were moving to UNIX platforms, we would have liked to bring up our replacement BITNET service on UNIX as well. However, our our inquiries revealed that `urep` was difficult to install and did not have a good reputation for reliability. For this reason the decision was made to go with JNet on VMS, and "gateway" to our users' accounts on UNIX.

JNet (version 3.5) was installed on *athens.sfu.ca*, a VMS system attached to our campus backbone network. Since we wanted to bring up the BITNET service in parallel with the existing MTS BITNET service, a new BITNET node name was applied for from the BITNET NIC. Host athens acquired the BITNET node name SFUVAX. Since it takes some time for new BITNET routing tables to propagate throughout the network, we asked the PostMasters at adjacent BITNET nodes (UBC and Uvic) to add node SFUVAX to their routing tables to allow us to do preliminary testing. Once the entry had been added, we were able to test various JNet configurations until we were satisfied that the replacement service was reliable.

7.9.2 BITNET File Transfer

In order to replace MTS's BITNET file transfer service, we needed to provide a way in which our UNIX users could exchange files with their colleagues at other BITNET nodes. FTP provides some of the functionality, but does not address file exchange with those BITNET nodes not having TCP/IP connectivity. In addition, FTP does not provide a way to exchange files without logging in to the remote host.

Since FTP does not provide all of BITNET's file transfer capabilities, it was not possible to use athens/SFUVAX as a file transfer gateway in a suitable timeframe. However BITNET file transfer was not a heavily used MTS service, and so it was decided that those users who did need the service would be issued a VMS account on athens/SFUVAX where they could "stage" their files: FTPing between their UNIX account and their account on athens/SFUVAX and then using JNet's file transfer capabilities from there.

7.9.3 BITNET E-mail

We have been using the MX mailer for VMS (V2.3-5), written and supported by Matt Madison at RPI. MX interfaces with the JNet software and it was relatively simple to configure the MX mailer to act as a gateway between BITNET mail and our campus UNIX accounts. In turn, the UNIX hosts on campus had their sendmail configuration files changed to route all BITNET mail through the athens/SFUVAX mail gateway.

7.10 Campus Computing Laboratories

The best way to get the campus community involved with the new environment is to get them working with it – making it a part of their everyday activities. To this end, we have setup several computer labs across campus and continue to setup new ones as our resources permit. There are "open" labs and "assignment" labs. The former provide places for students, faculty and staff to

[27] Contributed by Michael Hayward *(hayward@sfu.ca)*

come in and use our computers for whatever reason they wish. The machines are all attached to the campus network and range from simple Macintosh Pluses and IBM PS/2 Model 25's, to color Macintosh IIci and IBM PS/2 Model 55sx machines, and (soon) X-Terminals and NeXT workstations.

The assignment labs provide computers and software tailored for specific course assignments. Special software written at SFU allows instructors to book blocks of time on these machines for their students. Each student is then guaranteed a certain minimum amount of time within which to do their assignment(s).

7.10.1 Burnaby Mountain Campus

Open Labs:

There are several labs equipped with: (i) Macintosh SE, (ii) colour Macintosh IIci, (iii) IBM PS/2 Model 25, and (iv) color IBM PS/2 Model 55sx (386) computers. All computers are connected to the network.

The labs operate from 8:30 to 22:30 (except on the weekends when they close at 17:30) and are open to the entire campus community. Some of the labs can be booked for tutorial use by ACS and by academic instructors when "hands-on" tutorials are warranted for a course.

The main open lab is connected to a Novell Server that gives students access to many of the popular micro-computer software programs including MicroSoft Word.

Printing for the labs is handled by the public print-servers that were described in section 7.3.1 under "Printing"

WordStation:

A special lab consisting of 75 Macintosh Pluses, 75 IBM PS/2 Model 25s, local dot-matrix printers and local laser printers. The lab was setup in the basement of the library in 1989. This student word processing facility is open during normal library hours and tends to be fully occupied during the day.

Assignment Labs:

There are several assignment labs across campus. The largest lab is equipped with 75 IBM Model 55sx (386sx) machines with VGA monitors and a number of Macintosh IIci colour machines all connected to the campus network and serviced by a Novell file server. Lab usage ranges from computing science students learning Modula-2 to Kinesiology students using HyperCard to create computer based training applications. Novell software written by Computing Services allows instructors to book a fixed number of hours for each assignment. The scheduling software guarantees that each student gets at least the allotted machine-time per project. The programs required for each course are kept on the Novell server with students saving their data on floppy disks.

Small X-Terminal Lab:

This facility will be installed by the Fall-92 semester. A DEC contribution will probably materialize as 8 mono and 4 color VTX2000 X-terminals served by an InfoServer 150 VTX. This will be our first public X terminal implementation. The DEC offering looks pretty good with the InfoServer

apparently providing paging services to the X-terminals. Other sites report 6 MB VTX 2000 systems performing equal to 10 MB systems served by conventional UNIX systems. There are also a set of management utilities which apparently reduce administration staff time. Security looks like it might be a problem: DEC X-terminals are reputed not to understand MIT "magic cookie" protocol.

40 Seat NeXTStation Lab:

A lab of 40 NeXTStations (105 MB disk, 8 MB of memory and a monochrome monitor) is in place and will be operational by Fall-92 semester. It will be supported in the same way as the Computing Science Instructional Lab (CSIL) computing Science lab which is connected to the Auspex file server via a dedicated fiber run, with a Cisco router to the backbone. The lab is intended for student drop in use and for "hands-on" tutorials. It is hoped that we will be able to give users access to the full suite of software that comes with a NeXT. This includes LISP, TeX, Mathematica, C++, objective C, and the NeXTStep development environment.

There is more room in the lab than there are NeXTStations. The rest of the lab will be filled with color Macintosh IIci machines setup much like the Macs in the other assignment labs.

7.10.2 Harbour Center Campus[28]

The Computing Facility, at Simon Fraser University Harbour Center, provides three teaching labs, a mobile lab of Toshiba laptop computers and a drop-in center that may be used by students, faculty, and staff of SFU. These labs are for the support of academic and professional development programs offered at the downtown campus. The labs may also be rented by off-campus groups for educational and training purposes.

Macintosh Lab:

Equipped with 16 Apple Macintosh SE microcomputers, and an instructors machine connected to an overhead LCD display. Each machine has 4 MB ram, two 800k floppy drives and a ram drive. The lab is connected to an Novell network server. This lab is also connected to the campus backbone, the Internet, and campus wide laser printing services.

IBM Lab:

Equipped with 16 IBM PS/2 model 55sx microcomputers and an instructors machine connected to an overhead LCD display. Each machine has a 386 processor with 4 MB ram, 40 MB hard drive, color monitor and a 1.44 MB floppy drive. The lab is connected to a Novell network server. This lab is also connected to the campus backbone, the Internet, and campus wide laser printing services.

NeXT Publishing Lab:

Equipped with 9 color NeXTStations, light table and layout benches. Each machine has an 68040 processor, 12 MB ram, 100 MB disk drive and is connected to a file server, the campus backbone, the Internet, and campus wide laser printing services.

Drop-In Center:

[28] Contributed by Mark Jutras *(mjutras@sfu.ca)*

Equipped with Mac IIx and IBM PS/2 model 55sx machines, as well as a 3.5" to 5.25" disk conversion machine. The equipment is connected to a Novell file server and campus wide laser printing,

Toshiba Mobile Lab:

The mobile lab is made up of 30 T1600 Toshiba Laptop computers, with 1 MB of memory, and a 20 MB hard drive. This equipment is used in the classrooms at Harbour Center and maybe connected to all the campus wide services.

8. Co-Existing With Other Campus Systems

The migration from MTS to UNIX had a significant impact upon the School of Computing Science due to their reliance on MTS for student computer accounts and for the many compilers and applications that came to reside on that system over the years.

In contrast, the Center for Systems Science (a collaborative effort between 12 university departments to stimulate leading edge research) has operated a self-contained UNIX network for several years and suffered little or no inconvenience as a result of the migration. The only "major" problem they had was reconciling the UNIX user ID's they had issued to their faculty and staff for their network with the campus-wide user IDs issued by ACS.

In this section, we present contributions from Centre for Systems Science (CSS) and from the School of Computing Services. The CSS contribution describes what CSS is and the network they maintain. The contribution from Computing Science is more ambitious. It describes the problems they faced during the migration and how they either resolved them or worked around them. Their observations provide an interesting counter-point to our view of the conversion process.

Both contributions have been reproduced without changes or comments except for some minor re-formatting to fit the style of this paper.

8.1 Centre for Systems Science[29]

The Center for Systems Science provides technical support and computer resources for its members, as well as managing the research computer network for the School of Computing Science, School of Engineering Science and the Department of Mathematics & Statistics. The joint research network facilitates collaborated research in the area of Computer & Communication Systems, Microelectronics and Intelligent Systems by faculty members, their research staff and graduate students come from 12 departments.

The joint research network consists of 195 SUN Workstations, sixteen NeXT, eight SGI, two HP, eight Bridge Communication terminal servers and numerous Macintosh and PC. The network is divided into a backbone and 11 subnets. Each subnet typically uses a SUN SPARCserver as the gateway machine, providing disk storage for users of between 10 to 20 workstations on the subnet. The backbone contains file servers providing storage for software, e-mail etc., and other servers providing CPU cycles.

[29] Contributed by Hon-Man Wong, Center for Systems Science, *(honman@cs.sfu.ca)*

8.2 School for Computing Science[30]

SFU's School of Computing Science has long been one of Computing Services' largest customers, and still continues to be. Consequently, the changes to Computing Services (now ACS and OTS), as well as the changes to MTS, (now the Campus Computer Network), have had a significant impact on the school's computer use.

Fortunately, while the necessary changes were significant, Computing Science began preparing for the change approximately a year before any other computing user group. This was more the result of chance, than warning, since Eric Kolotyluk left Computing Services in June of 1990 to accept a position in Computing Science. Although there was no prior indication of the dramatic events that were to unfold in 1991, Eric brought to Computing Science a strong impression that the move away from MTS was highly probable, and that Computing Science ought to prepare. This impression was based on ongoing efforts within Computing Service to migrate applications and services away from MTS.

8.2.1 Course Migration

The first step was to actively identify courses that were *not dependant* on MTS and could be easily moved to Unix or some other system (e.g., Pascal, LISP, etc.). It was a rather straightforward matter to move these course to systems with existing software.

The next step was to actively identify courses that were currently *dependant* on MTS based applications, but no appropriate software existed on other systems (e.g., Cobol, PL/1, custom simulation software, etc.). While not as straightforward[31], we did manage to find (or write) the necessary software for our courses.

For the most part we migrated 60% of our courses away from MTS before it was announced that MTS would officially be phased out in less than a year. In this regard, the demise of MTS was relatively painless for Computing Science.

8.2.2 Ahh, But...

Where the demise of MTS did affect Computing Science was in that it was necessary to move students from MTS to either Computing Science's own Unix facilities, or to the MS-DOS/Novell based Assignment Lab.

8.2.3 Computing Science Instructional Laboratory (CSIL)

In 1990, the CSIL Unix network comprised 3 Sun 4 CPU servers, 9 Sun 3/50 workstations, 60 ASCII terminals, and various odds and ends. By 1991, 29 NeXT computers were added to CSIL, as well as about 4 or 5 courses that were previously MTS based.

This added substantially to the Unix administration load on the Computing Science technical staff, and had almost totally eliminated any time for development (please, no violins, at least yet). In addition to the basic administrative load, there has been a second order load in trying to

[30] Contributed by Eric Kolotyluk, School of Computing Science *(eric@sfu.ca)*

[31] for example, try finding a decent Cobol compiler for Unix. If that's too easy, try finding a PL/1 compiler for Unix.

administrate and manage the new scale of things. For example, there is now so much software that needs to be set up in each user's environment, that new ways of managing .login and .cshrc files had to be invented.

8.2.4 ACS Assignment Laboratories

While it was hoped that using the PC-based assignment lab would be like using MTS, this has not been the experience. In fact, one of the Computing Science technical staff spends about 50% of her time managing assignment lab issues (further depleting existing staff resources). For the most part this is due to the fact that Computing Services have introduced a new service, without a corresponding increase in staff needed to support that resource at the same level MTS was supported. Another observation is that MS-DOS/Novell just does not have the same features as MTS (or even Unix) when it comes to managing large groups of transient users in a hostile (student rich) environment.

8.2.5 The End Result

The end result is that while the School of Computing Science was better prepared than most user groups for the demise of MTS, there was not adequate appreciation for the extra effort required from internal resources to support this change. Another tangible observation of all this is that *"distributed computing is more costly than central computing in terms of staff resources."*

8.2.6 From Mainframes to Madness

Given that Computing Science's technical staff were now faced with a new additional work load, with no hope for increased staff, there seemed few options for improving the situation. The answer lay in some rather radical thinking:

1) Since the additional work load on Computing Science was a direct result of ACS and OTS administering less systems for Computing Science, find a way to get ACS and OTS to take some adminstrative load back. This would not be easy as ACS and OTS were already cut to the bone in terms of staff

2) In order to get ACS and OTS to take back some administrative load, find some solution that was mutually beneficial.

Given the acquisition of the new Auspex file server, and the extremely high level of LAN systems at SFU, it was proposed that the CSIL Unix network be directly connected to the Auspex, and that the previously autonomous CSIL network, become a part of the new Campus Computer Network. Some of the motivations for this were that:

1) Since ACS and OTS were going to create and administer Unix accounts for everyone on campus anyway, it seems redundant to administer separate Unix accounts on CSIL.

2) Initially CSIL required each student have, and use, a separate UNIX account for each Computing Science course they were taking. Accounts were named like, c351123, where 351 was the course name, and 123 was just some magic number. Needless to say students were not much impressed with this system. As a matter of principle, it seemed that assigning students one, and only one, Unix account for the whole campus was conceptually easier[32] for new students.

[32] but not for students already use to old system, and who were already use to going to Computing Science for their course accounts

3) By increasing the collaboration between Computing Science, ACS and OTS, there would be better opportunities to share the development of new applications.

8.2.7 Computing Fusion

Implementing the proposed merge of CSIL with the Campus Computer Network was not without its frustrations, but in the end, CSIL was running in production mode ready for the first semester of 1992.

While conceptually all that was required was to connect the CSIL Unix network to an ethernet port on the Auspex, reconfigure the CSIL computers to mount their file systems from the Auspex, and get their NIS information from the CCN, in reality several frustrating issues came up:

1) Since CSIL was still technically managed by a department outside of ACS and OTS, there was a serious issue of which file system could be mounted from the Auspex. The solution was to only allow undergraduate files systems to be mounted. For others who needed to access CSIL (such as instructors and teaching assistants) it was necessary to create auxiliary home directories in the undergraduate file systems. This required a little smoke and mirrors with file system mounting.

2) While Computing Science previously had to administer much configuration information like, NIS maps, mailing lists, etc., now it is necessary to find someone in ACS and/or OTS to make such changes. While this hands off approach is often slower, the situation has improved with experience, and the net result *is* less work for Computing Science.

3) CSIL is at the mercy of ACS and OTS. When changes are made to systems on the Campus Computer Network, CSIL can be non-functional will little or no warning. This situation has improved also with experience.

The end result is that there is now less work for Computing Science's technical staff than before the merge (although there is still more work than prior to MTS's demise). The time freed up, however, has been spent mostly on refining the tightly coupled interoperation of two Unix networks managed by different departments.

8.2.8 Fusion Benefits

The fusion of CSIL with the Campus Computer Network has had some tangible benefits to Computing Science. While it is hoped that there has also been some benefits in return to ACS and OTS, these are generally less tangible.

8.2.9 Backups

OTS now handles the backup and restoration of all user files. For CSIL users, this was previously handled by the CSIL Administrator.

8.2.10 Documentation

Providing adequate documentation has always been a challenge for Computing Science. With all the new (and excellently written) *how to...* handouts provided by ACS, Computing Science has been somewhat relieved of this task (and worry). More importantly, because CSIL is now part of

the Campus Computer Network, the documentation is often just as applicable to CSIL as the rest of the campus.

8.2.11 Software Support

ACS maintains a set of generally available software on the Auspex under the directories `/usr/local/sun`, `/usr/local/sgi`, etc. Previously much of this software was also managed by Computing Science: installation, upgrades, troubleshooting, finding filespace, etc.

Computing Science still maintains a formidable amount of software on CSIL under the directory `/usr/local2`. This is a convention started on the CSS Research Network: `/usr/local` is for site-wide licensed or public domain software, `/usr/local2` is for department-specific licensed or more specific software.

8.2.12 Administrative Information

ACS and OTS manage systems information such as: NIS maps, Domain Name Server tables, mail lists, X.500 information, etc.

8.2.13 Ubiquity

ACS and OTS offer three systems for student use, fraser, kits, and malibu. From the students perspective these are not much different than using the CSIL computers: you just logon to a system with your campus id and it works.

8.2.14 Reverse Benefits

Although if you were to ask some people in ACS what benefits they have realized with the connection of CSIL to the Campus Computer Network, they might say "what, none, it's just a lot of trouble."

While there was much stress and hostility generated during the CSIL/CCN merge, this was more due to battle fatigue than any deep animosity. In the end, ACS and OTS really came through to make things work.

The benefit to ACS and OTS has mostly been validation of the ubiquitous distributed computing model they had been promoting. Additionally, there has been a closer working relationship with Computing Science, with increased motivation from Computing Science to 'pay back' some good-will (see Accounting Below).

8.2.15 What Next

While CSIL is in its second semester as an integral part of the Campus Computer Network, there are still a number of areas where development is continuing and needed.

8.2.16 Accounting

Prior to the merge of CSIL with the CCN, Computing Science began implementing its own accounting system as a way to manage PostScript printer usage. The system was implemented using the Sybase relational database system, and a substantial amount of custom application code.

Basically each student had an account that kept track of how many pages they were allowed to print. Each student who was enrolled in one or more Computing Science courses were given a quota of 100 pages. If they depleted their account, they could pay $5.00 for another 100 pages.

Given that Computing Science wanted a more flexible accounting system, and ACS and OTS were in need of some sort of accounting system, a cooperative venture was created to develop a more general accounting system. Computing Science would do the initial design and implementation, and ACS would take over the ongoing support.

Basically the system has been designed to be as general as possible as well as being designed to be implemented in stages. Some of the design goals were:

- The system should distribute the administrative load as much as possible among departments using the accounting services.
- It should be possible to account for anything. The design uses the concepts of devices and resources. For example, a printer is a device that may have a number of resources such as pages printed, processing time used, etc.; a computer workstation is a device that uses resources such as cpu time, connect time, etc.
- Users should only have to worry about one computer resource usage account for all of campus. For example, you might have several computing accounts on different Unix networks on campus, or even Unix accounts off campus, but still only have one resource usage account.
- Departments with their own devices and resources should be able to use the accounting system. For example, CSIL has its own printers that need to be controlled, in addition to OTS which needs to control access to the Xerox 4090.
- Department with their own devices and resources should be able to assign quotas specific to those devices. For example, a student may be given a quota of $5.00 for CSIL printing. Once that quota is depleted, computing charges are taken against their campus-wide computer usage account. The quota used for CSIL printing cannot be used for printing on the Xerox, however.
- Departments can design and implement their own resource charging and usage policies (which devices and resources to charge for, what rates to charge, what quotas to set, etc.). Each department that wishes to use the system gets its own Sybase account. Departments cannot interfere with the policies or accounting of other departments.
- The accounting system is designed more as a Computer Resource Control[33] mechanism than as a cost recovery system, although it can be used for cost recovery.

8.2.17 Login Environment

Setting up an effective login environment in Unix is substantially more difficult than in MTS for a number of reason:

1) MTS had one sigfile for each user. Unix has typically three (`.cshrc`, `.login`, and `.logout`) or more (`.profile`).

2) MTS had project sigfiles that could be used to implement, and enforce, department wide policy. Unix has no such concept.

[33] for this reason the database used is called the Computer Resource Control, or CRC, database.

3) MTS had a number of utilities for adaptively, and nicely, setting up a user's environment at signon time: *PROFILE, *SIGSETUP, etc. Even MTS's command macro system was easier to use and more powerful than writing C shell code.

4) Unix has a more complex environment to set up: environment variables like `PATH`, `MANPATH`, `LD_LOADER_PATH`, etc.

5) Unix services and applications have multifarious ways of handling user customization: `.XXXrc` files, environment variables, X/Windows files (`.xinitrc`, `.Xdefaults`, etc.), and so on.

6) Distributed computing often implies heterogeneous computing. It is even more challenging to create `.login` files that can work correctly on a variety of different vendor's Unix systems.

The end result is that you need to be an expert to effectively set up your Unix login environment. Given that most undergraduate students are not that expert, Computing Science maintains very sophisticated standard `.cshrc`, `.login`, and `.logout` files. This is still far from satisfactory, and more work is planned in this area, for example, an X.500-based preferences database and a program like *SIGSETUP to process the information and set up the user's environment.

8.2.18 Administrative Information

Currently ACS and OTS routinely uses information from the personnel office and the registrar's office to: create computer accounts, build mail lists, unix groups, netgroups, etc. Computing Sciences uses this information to control the access and security of the CSIL systems.

While it was hoped that ACS and OTS would do a more complete job of this, there is still much massaging of data by Computing Science. The problem is that Unix had no comprehensive way of managing such information, and so ACS and OTS have a formidable challenge to develop a better administrative system.

8.2.19 Current CSIL Unix Configuration

- 60 ASCII terminals
- 28 NeXTstations, 8MB, 100MB HD
- 1 NeXTcube cpu/file server, 32MB, 1.4GB HD
- 8 Sun IPX Colour workstations, 16MB, 200MB HD
- 2 Sun SS1 Colour workstations, 16MB, 100MB HD
- 2 Sun 4/320 cpu/file servers, 32MB, 300MB HD
- 1 Sun 4/360 cpu/file server, 96MB, 1.7GB HD
- 3 Printonix Line Printers
- 1 HP LaserJet IIISi

8.2.20 Conclusion

The demise of MTS had a significant impact on Computing Science, and continues to affect the support staff. Fortunately the School was prepared, and while adapting to the new world was very stressful, the worst is over. However, there is still much to do.

For the most part there have been three main benefits to the fusion of the CSIL Unix network with the Campus Computer Network:

1) From the user's perspective, mostly the undergraduate students, there is one computer network on campus, and they only need one computer account.

2) There is less duplication of effort between Computing Science, ACS and OTS. This has lessened some of the load, but not all, on Computing Science's technical support staff.

3) There is increased cooperation between Computing Science, ACS and OTS. In many cases, what one group develops is of benefit to the other.

The experience of this is that cooperative distributed computing *can* be made to work effectively. Given that the staff costs of distributed computing are typically higher than for centralized computing, effective cooperation between different autonomous groups is even more important.

The cold reality, however, is that there is still very much work to do in *properly* supporting distributed computing environments.

9. The Near Future

This is a bit of a "catch-all" section that describes some of the changes to our new system that we hope to implement over the next few months. The information is presented in no particular order.

9.1 Improved User Account Management[34]

We carry out user account management using a hodge-podge mix of scripts and C programs, mostly written on the fly in the heat of getting some 18,000 user accounts onto the system last August/September. Extensive modifications have ensued since then to keep everything running. We use Sun NIS with some configuration and programming changes (mostly in `/var/yp/makefile`) to make it possible to operate with 28,000 IDs in the database. If you get the idea that we need something better, you're right!

Richard is currently heading a project to find (or build) a replacement for this "system". We would be interested in hearing from the other big sites as to what works for them. We'd like to kerberos-ise, but it just isn't possible yet on some of the machines that we use. We would like to end up with a system that employs a real database like Sybase or Oracle to administer all of the accounts for all of the IDs and machines involved in the system. This would include operations to create/delete/alter IDs/passwords/groups/netgroups in a simple, straightforward manner. We would like to be able to delete accounts but reserve the IDs and names in case an individual returns to the University (students who are away for the summer should get the same ID back upon return

[34] Written by Richard Chycoski *(richard@sfu.ca)*

in the fall). We also need to be able to 'prepare' accounts – but not activate them – until someone shows up at our counter with picture ID. Right now, we create accounts that are disabled (password begins with a '*', shell is invalid), but this causes E-mail to be collected (uselessly) for people who haven't actually had their account activated.

If necessary, we will build another system from scratch, but we would much rather use (or extend) an existing system. As always, we need this stuff yesterday (Richard is drowning in user accounts) but we are willing to contribute a couple of month's work by a couple of people to help anyone build/complete their new system. Please feel free to send Richard Chycoski any information you might have about UNIX account management systems for large installations.

9.2 Increased Macintosh & PC Integration[35]

Shifting the central computing system on campus from MTS (not very network aware) to Unix (very network aware) is a major step toward providing a wide range of network services to microcomputer users on campus. In the course of the migration, we added some 300 to 400 IBM compatible PCs to the network by shifting them from serial connections to Ethernet. Sadly, most of these users are still just using character based terminal services (NCSA Telnet).

Some of the services we are currently looking at providing are:

- A license server for expensive or seldom used software (stats, high end CAD, expensive compilers, etc.)
- Information servers such as CWIS and CD-ROM services.
- Network based dial-in services based on Novell's IPTUNNEL and Apple's recently announced SLIP support.
- X-Windows clients (we currently have a right-to-copy license for Mac-X and are evaluating X-server software for DOS and OS/2 PCs)
- Client interfaces to services such as FTP, Mail and NetNews. Macs are much further along in this area than PCs but products like LanWorkplace from Novell and NuPOP from Northwestern University show promise.
- Direct desktop access to the users Unix files. There are a large number of issues and opinions on this one.

9.3 A True On-line Help System – Project Athena's OLC system[36]

While the paper help request system works, the paper being slung around is horrendous, and the answers that users get from the operators interpreting what is on the pieces of paper may or may not be consistent. The solution we are considering (and in fact working on right now) is to implement On Line Consulting (OLC) from Project Athena. The OLC package can function without any of the other Athena services (Kerberos, Hesiod, Zephyr, or Discuss) on everything from a dumb ASCII

[35] Contributed by Ed Hargrave *(ed@sfu.ca)*
[36] Contributed by Peter Van Epp *(vanepp@sfu.ca)*

terminal to a X/Motif server. One of the strengths of OLC that we wish to employ is called the "stock answer database", a database of common questions along with a "canned" answer that can be sent to the user with a couple of keystrokes. Another strength we like is that all OLC dialog is logged to log files, allowing management to review the questions and answers for accuracy, and to see which questions are being commonly asked and are therefore candidates for being added to the stock answer database. At present, when a question is answered and then marked "done" by the person answering the question. Unless the "discuss" conferencing system (or some custom interface to another system) is present, the question and answer are thrown away.

Right now we have a student hired to install OLC on a target machine and write the necessary code to cause the logs of completed OLC questions to be posted to an nntp server (which will be a private server running on a departmental machine in the production version). Netnews is the obvious choice since we already support a netnews server and several netnews readers. Most of our users already read netnews so there is no need for them to learn a new interface to read the OLC logs.

With OLC, our operators would submit the question and log the answer via OLC. After a while the common questions would be entered in the Stock Answer Database. Operators would scan this database for answers before considering passing the question along to a consultant. Assuming this works alright, in future we will probably do some additional work to the OLC e-mail interface and then allow the general user community to ask questions and receive answers via e-mail. We may even be able to set-up up on-line consultants, answering questions from the users in "on-line" mode (much like MIT).

9.4 Nysernet for the Operators[37]

We want to provide the operators with a broader – higher – view of the network. They already run LAN Traffic Monitor but the information that it delivers is too detailed in some ways and doesn't tell us much about IP traffic. We plan to run Nysernet SNMP on one of our Sun systems, and use X-Windows to display it on the VAXstation being used for our VCS Console Monitor. Licenses have been ordered for TGV Multinet, and Nysernet. We plan to use the Nysernet configuration file developed for BCNet.

10. Conclusions

It would be nice to say that we migrated from MTS to UNIX in just 10 months. However, that would be ignoring the many years of preparatory work by the MTS community and by people past and present at SFU. What we did in those 10 hectic months was to finish off 5 years of planning and infrastructure building done by a whole lot of people – some still with SFU and the others who are not.

Much of the knowledge concerning the pitfalls of conversion came from the discussions at various MTS workshops and from the WINE/INTERSYTEM project. Initiatives like the installation of the Vax Cluster for administrative use, the WordStation, and the assignment lab (all of which were in place prior to the migration) gave us a big head start towards reducing our dependency on MTS.

[37] Contributed by Bill Baines *(bill@sfu.ca)*

The heart of our new system – the high-speed network – would have been impossible without the fibre-optic backbone cable plant put in place by the old Computing Services with support from the campus administration. This was a remarkably expensive undertaking that is under-appreciated by the SFU community which has come to depend on it (whether they know it or not).

Now that the flurry of migration activity is over, users can now investigate the many new packages and services offered to them by UNIX and the Internet. X-Windows is slowly gaining ground as the preferred user interface as we begin to publicize it. In addition, research users are now in a position to share their source code with colleagues at other universities much more easily than before. There is no longer the MTS incompatibility issue.

This last chapter serves to review what we think we did well during the migration and where there could be improvements.

10.1 What we did well...

- The Auspex file server. One of our best moves. It basically sits in the corner and works – day in and day out.
- An athena-like "no matter where you go, there you are" file service where a user's home directory is available on all the machines that the user can log into (again, thanks to the Auspex files server).
- Support of the CSIL lab in Computing Science. The students see the same home directory structure in the CS labs as on our machines and the CS folks save the duplication of manpower needed to create and maintain duplicate sets of user accounts for their students.
- Put tape support on the VAX and write the utilities to allow continued access to *FS tapes.
- Mirrored backup on the Auspex so that we get consistent full backups without users losing access to the systems for a hours at a time.
- Distributing printing for our users, support for PostScript, cost recovery via lasercards, and reduced paper waste by having students spend their own money on their output. No more massive print jobs that go unclaimed.
- JNet on the VAX for BITNET support, came up in half a day and has worked ever since (just like the Auspex).
- The choice of Silicon Graphics (SGI) machines for the general login machine and the mini-supercomputers. The general login machine can support up to 120 users at a time as long as we limit them to e-mail and other light tasks. We had originally targeted for a machine that would support up to 80 logins. SGI has provided good software and hardware support.
- Using Cabletron for our network infrastructure vendor. The network gear works without giving us any trouble (although we can't be quite so positive about Cabletron's ethernet cards for PCs an Macintoshes).
- Eudora for e-mail on the Mac. Ray Davison has been adding function and improvements to an already good e-mail package for the Mac. SFU has a large investment in Macintoshes at

the departmental and administrative levels. Eudora proved to be just the ticket for gaining user acceptance of the new UNIX system. It provided a somewhat better user interface than FSM once the problems with finding usernames was elliminated by the X.500 directory.

- Support for free access to computing for every one at SFU. It presently looks like we have about 10,000 active accounts on the system out of some 17,000 full-time (equivalent) students. This does not include another 1000+ teachers around the province supported through our Department of Education. While many of these people may use UNIX only for e-mail, free access to computing can only result in more computer-literate graduating classes that will take that knowledge out into the "real world" with them.

10.2 What needs more work...

- TCP/IP support for the Xerox 4090. The Soliel product from Xerox is a good start, but it still has a long way to go in both PostScript support and in "engine performance". If we did not already have the 4090, a set of HP LaserJet IIIsi printers might have served our purposes admirably except for the truly huge print jobs.

- Tapes, and large read-only or read-mostly data sets. UNIX tape support is far from perfect (and that's being kind). As we have pointed out, conversion of IBM and other data tapes is currently done on a VAX since there don't seem to be the tools to do it under UNIX. While this works for converting data from tapes, the MTS users who used to run their program that took input from tape have been left out in the cold (or in most cases, blowing out their disk quota and filling up our disks). We are currently looking at optical juke boxes and 8mm tape robots and their accompanying manpower and support issues.

- SECURITY! Security in both the UNIX machines and the off the shelf UNIX solutions (NIS, NFS, and UNIX) is lacking. At present it looks like AFS is the solution of choice for distributed file services over the backbone to peoples offices in a secure manner. NFS is both too insecure and NIS with no `kerberos` means that we are a hackers heaven. Our first crack run netted 800 accounts with bad passwords. We ended up installing `npasswd` but we still have lots of broken accounts and people attacking other sites through us. It is very manpower intensive to do anything at all about this and without a network protection system (either `kerberos` or 10-base-T hubs that block traffic to other ports), all the good passwords in the world are useless against one undergrad in a PC lab with some public domain packet sniffing software.

- In some ways, UNIX just isn't MTS. The UNIX scheduler seems a lot worse than the MTS version, CPU bound jobs running in the background tend to kill interactive response much easier than on MTS. Control of permissions to files are basically non existent in UNIX. Part of our new accounting management package will split users up so that they can easily add and subtract people from their group (currently all users are members of the single group `users`). Ian Reddy has written and is testing a setuid program called `access` that allows a user to permit specific users read, write or execute access to specific files.

- There are too many ready-made tool kits and lots of expertise out there for breaking in to UNIX systems. A new security hole seems to show up in alt.security (and alt.hackers) almost every day! It may be reasonable to put in a tightly controlled firewall machine between us and the internet and then depend on backup tapes for internal security. Maintaining a vigilant watch is far too labour intensive.

- Support of workstations on peoples desks. If you buy a UNIX workstation right now it is up to you to find someone to administer it. ACS/OTS cannot take on the additional burden of workstation setup and support at the present time except as time permits. If users are willing to go without regular software updates and regular backups (as most seem willing to do), then some solution may still be found.

This paper has described several aspects of the migration project undertaken by computing staff at Simon Fraser University over the past year or so.

Thanks to some twenty-four authors and contributors: Ellen Sangster, Margaret Sharon, Frances Atkinson, Peter Van Epp, Ian Reddy, Richard Chycoski, Lionel Tolan, Grant Dimock, Ed Hargrave, Robert Urquhart, Bob Spratt, Peter Howard, Mike Dustan, Randy Raine, Reo Audette, Steve Kloster, David Stratton, Wolfgang Richter, Michael Hayward, Mark Jutras, Hon-Man Wong, Eric Kolotyluk, Ken Urquhart, and Bill Baines.

Editing and poetic license were exercised by Ken Urquhart and Bill Baines.

Work continues and we are interested in comments and suggestions anyone may care to offer. Please do not hesitate to contact any of the footnoted individuals for additional details.