Tenwen: The Re-engineering Of A Computing Environment

Rémy Evard – Northeastern University

ABSTRACT

In the summer of 1992, the computing environment at the College of Computer Science of Northeastern University was completely dysfunctional. Among other things, the network was down over 25 percent of the time, the computers and software were badly misconfigured, the users were confused, and it was nearly impossible to administer. It was on the verge of collapse.

Now, two years later, the situation is entirely reversed. The network is up well over 99 percent of the time, the computers and software are easily managed, and the users are (for the most part) satisfied. Many people have stated that it is the nicest and most functional computing environment they've ever used.

This paper is an overview of the key changes that we made and the methodology that we used in bringing about this transformation. I examine the lessons we learned as well as the mistakes we made, and offer advice to others starting with a similar predicament.

Introduction

The College of Computer Science at Northeastern University currently operates a network consisting of approximately 350 computers of various types. About 1200 people use the systems for education and computer science research. The Systems Group of the College, which consists of four full-time staff and a variable number of students, is responsible for the administration of the whole network.

We support several different types of UNIX workstations, as well as Macintoshes and PCs, all of which have hundreds of applications installed. The network as a whole has been up constantly for the last nine months. The user community is about as content as a group of 1200 people can reasonably be expected to be, and occasionally even compliment us on the way the computers and software run.

Two years ago, the situation was entirely different. The two main computing staff members, who had been primarily VAX-style operators, had just left the school. The facilities consisted of about 100 barely usable UNIX computers, several partially networked Macintoshes, a micro-VAX, and a convoluted network that was down more than it was up (or so it seemed to the users). By all reports, the environment was very nearly unusable.

The rest of this paper describes the process of moving from the dysfunctional system of two years ago to the current environment. I present a chronological overview in order to place the changes in perspective, and then summarize the key ideas, strategies and methods. This isn't a comprehensive list of all the changes made, nor is it a description of the final environment, both of which are beyond the scope of this paper. Instead, it's a look at the process of change and the lessons we learned.

Building the New Environment

The changes in the computing environment took place over a period a bit longer than one year. This section presents some of the important events in that process.

Formed a Team (July, 1992)

The Dean of the College made a decision to hire a UNIX administrator who was familiar with the Internet community. This position was to report directly to the Dean, not to the faculty, and was to manage the Systems Group. This person was given nearly complete autonomy over any matters related to the computing environment in the College. I was hired for this position.

Another position was created for a 'UNIX Systems Programmer', hired about a month later. This person was expected to provide the services expected of an Intermediate Systems Administrator.

These positions were in addition to two staff members, one of whom had been recently hired as a Technician. These four people formed the Systems Group, which consisted of three new hires.

This was a radical and important change for the College. Up until this point, technical positions had not been targeted for people with UNIX expertise, and had not been considered to have positions equivalent to the faculty. It was very clear who had what responsibilities and who reported to whom. By having one boss (the Dean), instead of twenty-five (the faculty), I could concentrate my efforts on fixing the environment rather than running to answer the requests of twenty-five people.

Examined and Marginally Stabilized the Computers (July, 1992)

At this point, we explored our surroundings and found all sorts of interesting things. No one knew what computers existed. No one knew quite how the network was arranged. No one was sure what software was installed. Hackers had infiltrated the network months earlier but were thought to be gone. We found seemingly endless technical nightmares, such as there being exactly one NIS [1] server for 100 UNIX hosts.

We fixed a few critical problems, such as removing routing loops, which undoubtedly cut down on the number of crashes per week. At that point we casually estimated that it would probably take all summer to get the environment in shape.

We were wrong.

Punted (August, 1992)

We spent a about a month that summer cleaning up the major problems, and gradually became convinced that we might have bitten off more than we could chew. Too many things were unknown, broken, or both. For example, a set of disk servers were missing critical parts of the OS (like /bin/mv, and half of the /usr tree). There were no copies of the OS on tape or CD, so what was installed was all we had to use. And, of course, the disks on those servers were going bad. The servers (all of them) rebooted frequently.

Nor were the servers the only problem... Security was still an open question. Critical network cables were far out of specified parameters. Every day brought new delights.

At this point it became obvious that this wasn't going to be a simple fix and wasn't a standard upgrade. No part of the system could be relied upon. The whole network, from the cables all the way to the software, drastically needed to be rebuilt. In most situations, one must work within the existing structure to improve it. We didn't feel that would be possible here, since we couldn't trust any part of it... two years down the road, we might still be wondering what important part of /usr was missing.

This was a unique opportunity. We would build a completely new computing environment that had nothing whatsoever to do with the existing one. We would essentially be building a new computing environment from scratch, except that we would be doing it for a hundred computers and an existing user base. Unimaginatively, we named the project "Newnet".

Newnet Was Born (September, 1992)

We used funds from a research grant to buy a SPARC ELC. We also discontinued hardware support for computers that no longer needed support (some of which no longer even existed) to buy two additional SPARCs. Each of these came with their own copy of SunOS 4.1.2 on CD-ROM. It was quite refreshing to have a copy of an operating system that could be used to rebuild a computer if a disk crashed.

We installed these computers from scratch and connected them together. They shared nothing more than a piece of thin ethernet with the existing environment, which, for obvious reasons, had come to be called "Oldnet". (Fortunately for us, ethernet snooping hadn't become popular at the time we did this.)

Our goal at this point was to use these three SPARCs to build a working software structure that everyone could eventually move to. We didn't have have a formal plan in place. In hindsight, we should have analyzed our needs more carefully at this time.

Designed A Consistent Directory Structure (September, 1992)

The first step in building the network was to create the file system for the network. Rather than haphazardly mount file systems from our machines on each other, we designed a mounting and naming scheme that we felt would scale to several hundred computers and several different machine types.

The details of the plan are beyond the scope of this paper, but a few of the principles may be of interest:

- Local disks are mounted under /export if they will be exported.
- Network disks are mounted under a /net hierarchy.
- We created a global directory named /ccs to be used as a platform independent hierarchy. It has turned out to be enormously useful.
- The network file system structure is identical on every computer, which greatly simplifies navigation from a user perspective.

These assume a disk environment based on local disks and NFS [2], but the basic goals and the naming scheme for the user space of the file system would map to other disk implementations as well. The goals supported by this implementation include:

- A documented file system organization for user home directories and project space, which is consistent across all machines.
- A mechanism for locating files which are only useful only to a specific machine, files which are specific to a type of computer architecture, and files which are useful across all types of computing hardware.

As far as possible, the environment and operating system structure that each machine vendor supplied was left untouched. Other abstractions were used to allow users to quickly and easily move between machines from different vendors.

In order to create the global directory structure, we chose to avoid the automount program provided with SunOS and used the Berkeley 4.4 automounter, "amd" [3], instead. It had several features that we found to be extremely useful (the /home map, for example), was portable to all of our computer types, and was more reliable.

We completely documented the naming scheme for the local and global disk mounts. After two years of use, the scheme has undergone a few minor revisions, but has generally held up.

Designed A Cloning Process (September, 1992)

At this point, we had made fundamental changes to how the machines looked after a fresh operating system install. We felt that it was vital to record the changes that we made so that we could replicate them as needed.

We reinstalled one SPARC from scratch from the CD-ROM with the OS on it. We examined the changes that we had made to customize it, and wrote a flexible script to make (and verify) those changes for us. After a few iterations, the script worked quite well. This gave us a tool to take a completely clean SPARC and make it fit exactly into our environment with a minimum of effort.

Every time that we make a change on a computer that should happen on every computer on the network, we adapt this script to make that change for us. For example, we like to install a replacement for "inetd" on our SPARCs and DECs. When this script runs, it replaces the old version of inetd with the new. This means that every time we bring up a new computer, it immediately has all the modifications that the other machines on our network have. Installing a new computer usually takes about 15 minutes of person-time.

Over time, the script has grown and changed. It now consists of a hierarchy of flexible scripts, has a method for dealing with customized configurations of individual machines, and can do kernel installs. While it serves its purpose, we feel that we should move to a more robust and intelligent configuration management scheme in the near future.

Fixed Name Service (September, 1992)

We now turned our attention to becoming a network of computers rather than a bunch of computers connected by a network. The first step in this move was to get name service functioning correctly.

Up until this time, the College, and indeed the whole University, had been doing hostname lookup with a gigantic hosts file. The problems with this scheme are well-documented [4]. We can verify that those problems are real. In addition, reverse name lookup failed for all of the computers in the College.

We designated one of the SPARCs to be the nameserver for the College, and installed the latest version of the Berkeley Internet Name Domain software. We worked with the network authorities for the University, who, fortunately for us, were happy to delegate name service for our domain name(s) and our network addresses to us. Name servers are handy tools, and having control of your own name server is a good thing.

Once name service was working, we modified the Newnet hosts to do hostname lookups correctly.

This fixed the naming problems for Newnet, but Oldnet was still broken. While we wanted to limit the amount of work we put into Oldnet, we also felt that having name service working correctly could only make it easier to administrate Newnet. We therefore modified Oldnet to use the nameserver on Newnet, and quit using host files on Oldnet as well.

This was the first instance of what was to become a regular and important tactic as Newnet grew: we used the work we were doing on Newnet to help support Oldnet.

Fixed Mail Service (September, 1992)

At this time, the College's mail setup was not homogeneous. Mail to different machines ended up in different locations, and outgoing addresses had different appearances. Mail bounced a lot.

We followed the advice of the O'Reilly books [5] and created one mail hub for the College. At the time, we put it on the same machine as the nameserver.

We made outgoing mail appear to come from the domain, and put in MX records that were used to direct all mail to Newnet to the mailhub.

Continuing the tradition of supporting Oldnet with Newnet, we eventually setup MX records for Oldnet hosts pointing to Newnet and forwarded mail to the appropriate Oldnet computers from Newnet. Oldnet computers were configured to send mail to Newnet's mail hub for delivery.

This was the most important thing we did that year for the users of Oldnet. Email, which is probably the most important function of the computers in the College, became reliable and much simpler. This showed the Oldnet users that progress was being made towards the eventual goal.

Defined a Server Strategy (October, 1992)

We now had a name server, a mail server (the same computer), several disk servers, and perhaps one client machine. It seemed like a good time to consider computer roles and uses.

We decided to designate a computer as a "server" if it provided critical network services. We made a rule that only members of the Systems Group were allowed to login to servers, and that those machines should not be used for general processing. This policy was approved by the faculty resources committee.

This policy has become something we rely on – when a server crashed, we could be sure it wasn't from user code. When a user's process started spawning across computers, it didn't affect the servers.

Designed a Coherent Software Installation Method (October, 1992)

Now it was time to start building the software base. We had been installing software by following unwritten rules. Knowing the role of servers and the directory structure, we felt that it was time to work out a consistent and reasonable software installation method. It turns out that we hadn't quite been following the unwritten rules, so formalizing the process was important.

The important aspects of a software installation plan included software naming, installation history, version control, documentation location, and architecture dependent and independent directories. Similar structures can be found in the LISA archives. LUDE [6] and Depot [7] are good examples of software structures. We chose not to follow these plans because they didn't fit into our plans or didn't fit our needs, but our solution is similar to those.

Over the next several months, we built a set of tools that eased software installation significantly. It was possible to build these tools because we had documented our directory organization and our software installation methods.

Enlarged the Team (October, 1992)

A number of students in the school had become very interested in what was going on. We formed a small group of volunteers who would help install software and design Newnet. All of them were relative UNIX novices, but had a lot of energy. Newnet development became something that sometimes went all night and carried on into the weekends. We picked a few machines from Oldnet, rebuilt them from scratch, and continued to install the basic software base. The students didn't remain UNIX novices for long.

Each volunteer was given root access to all of Newnet when they (and I) felt comfortable with the concept. At the time, Newnet was pretty small and the group worked closely, so the security risk was minimal. Change management was more of a problem, but one of the first things one of the volunteers did was install RCS [8], a version control system.

As a side note, we still have student volunteers, and several of them have root access, but we are a bit more restrictive with who may have root. For example, we limit some tasks using sudo, a publicly available program that allows one to specify who may execute what as root.

Oldnet Crashed (October, 1992)

Oldnet, which was the primary computing base for nearly a thousand users, was still struggling along pretty much as it had been in the summer. A hardware problem on the only NIS server caused downtime for the whole network for two days. A few hours into working on it, we realized it was serious problem and developed a new plan for Oldnet.

One of the client computers on Newnet was turned into a server and used as the NIS server for Oldnet. Other Oldnet computers were coerced into being NIS servers for each of their respective subnets.

We took the downtime opportunity to trace cables under the machine room floor. We made the first version of a network map that we had seen for the College, and, while we were at it, removed over 150 cubic feet (in piles) of unused cable from under the floor.

Defined Newnet Clearly (November, 1992)

Having come this far, we understood the concepts behind Newnet better. We had been building an independent network with no real thought as to how to handle the conversion from Oldnet to Newnet, or how to manage the two networks simultaneously. We weren't exactly sure what a Newnet computer was, nor were we clear on who could use it.

We developed a document that defined a Newnet computer. It had to have been built from scratch from a clean copy of the operating system. It had to have been modified with our cloning script which included kernel modifications, security fixes, and OS patches, among other things. It had to be a member of the Newnet NIS domain, and was therefore limited to the (small) set of Newnet users. It would only use Newnet servers. For security reasons, Newnet users were not allowed to login to Newnet from Oldnet. Therefore, one could not have a login on Newnet without first having their workstation rebuilt to be on Newnet.

Having a clear, written definition served an important function. The process of writing it helped us understand what was important about Newnet. Once we had a functional definition, we could see what we had to do next, and could make estimates of how long it would take.

Addressed the User Issue (December, 1992)

Once we understood exactly what Newnet's goal was, we tried to project a timeline. It became obvious that it would be several more months. We had a few problems with this. First of all, the users on Oldnet were getting tired of being there and putting up with the problems. Further, it wasn't technically obvious how to move non-technical users from Oldnet to Newnet.

We made two important decisions:

1. We would update the user community relatively often as to the status of Newnet. This would let them know that progress was being made.

2. We would share home directories between Oldnet and Newnet because it was technically very difficult not to. We didn't have enough disk space for everyone to have two homes, which confuse most of the users, anyway. Unfortunately, our home disk servers were on Oldnet, so this went against our policy that Newnet computers shouldn't use Oldnet servers. However, we had no choice.

In order to keep Newnet reasonably secure, we continued to enforce the policy not to allow logins to a Newnet machine from Oldnet. The reverse was not true at all, however. In fact, the members of the Systems Group liked Newnet too much to use Oldnet, so we were almost completely administering Oldnet from Newnet.

Enlarged the Population (January, 1992 - March, 1993)

Newnet continued to grow, both in terms of the number of users on it as well as the number of computers on it. The users were limited to the Systems Group, the student volunteers, and certain key people who needed Newnet accounts for technical reasons or for morale. For example, the Dean of the College was moved to Newnet, as was the Assistant Dean. We watched these moves closely, noting the problems that they had, and working out scripts to help people move from Oldnet to Newnet. After two non-technical users had migrated, we wrote a much-needed (but minimalistic) user's introduction that described differences to expect. ("Emacs works. Your shell works. X works. Suntools doesn't. You can find software. ...")

Designed the User Environment (March, 1993)

On Oldnet, the primary means of setting up one's environment was to copy someone else's dotfiles and hope they worked. Most people used the ones developed by a knowledgeable professor. This caused all sorts of problems. (Without the professor's dotfiles, they would have been much worse...)

On Newnet, we built a set of reasonable default files that were installed in new accounts. We spent a lot of time on this. They were fully documented and contained lots of examples for users who wished to modify their environments. While the dotfile situation on Oldnet caused enormous trouble, we've had very few problems on Newnet with user environment configuration.

In addition, we designed an environment abstraction mechanism that allowed the users to select what sets of software they would like to use. The user's PATH, MANPATH, and related environment variables were built based on the user's selections. The mechanism has allowed maximum flexibility for users and administrators. We were able to do this because we had a consistent software installation scheme. This software mechanism is described elsewhere in these proceedings. The most important aspect of this approach is that it built an abstraction between the user's environment and the software installation environment. They wanted to modify one, we wanted to modify the other.

Changed the Domain Name (April, 1993)

Up until this point, the network domain name for Northeastern University had been "northeastern.edu". After months of politicing on our parts, it was changed to "neu.edu", which is considerably easier to type.

While the name change wasn't a vital part of Newnet, it certainly fit in. We were changing everything else about the network, so why not change the domain name?

The Newnet computers were moved to the "neu.edu" domain overnight, because administering them was easy. Oldnet computers were updated as they moved to Newnet. It became obvious from the name of the computer which net it was on. Users and administrators liked that.

Implemented the Hosts Database (April, 1993)

The domain name change gave us the opportunity to finish a set of tools based around a hosts database. They were used to build any network or host configuration files associated with IP addresses, including nameserver files, bootparams, ethers, printcap, hosts.lpd, hosts.equiv, and xdm configuration files, to name a few. The scripts did sanity checking on all the files before installing them to make sure that the data was reasonable. They were also used to build the list of computers that various user groups can access.

Once again, we used these same scripts to maintain parts of Oldnet.

Host configuration files related to IP addresses had been an enormous problem on Oldnet. Changing IP addresses or hostnames of a computer had been done by hand. The frequency of operator error in the midst of Oldnet's chaos was pretty high, causing all sorts of interesting problems. By using these host file configuration scripts, we virtually eliminated the chance for human errors. We've had absolutely none of these types of problems since.

Our experience here has defined a technique that we try to follow as much as possible – we automate anything that we do more than once, and we do sanity checking on files and systems before we install them.

Continued to Support Oldnet (May - June, 1993)

By this point, Newnet was providing all of Oldnet's critical network services. We had three major steps to complete before replacing Oldnet:

- We needed to complete the Ultrix environment.
- We needed to write reasonable documentation for users about how to navigate Newnet.

- We needed to figure out how to migrate 1000 user accounts to Newnet and to write the tools to help us to that.

In order to keep users pacified while we continued to develop Oldnet, we exported the Newnet software environment to Oldnet, and explained to the user community how to modify their PATHs to get to it. We also moved several more faculty members and their graduate students to Newnet to use as beta-testers.

Wrote User Documentation (Summer, 1993)

Based on responses and questions from the users that had moved to Newnet, we enhanced our user documentation.

The first thing that we wrote was a Newnet users guide that explained the few differences that users would actually have to handle. This was really a more detailed version of the first handouts we had prepared. We turned a version of this into a Frequently Asked Questions file and posted it to various local newsgroups.

We developed a WWW-based help system that documented all of the software installed, organized by category. Again, this was possible because we had a well-defined software installation method. This was intended to help users locate software that might be useful to them, and to partially organize software documentation.

A comprehensive user's guide would have been useful, but we felt that most of the students on the network wouldn't read a 20-page document that explained how to print or how to access the local modems. Instead, borrowing an idea from the University of Oregon, we wrote several one-page documents, each of which were about specific topics. These papers, some of which talked about about dotfile customization, lab usage, simple unix commands, and emacs, were promptly named Clue Sheets. We distributed these in the public computing labs.

Developed The Account System (Summer, 1993)

In order to manage the creation of a thousand new accounts on our network, we designed a comprehensive account strategy. Each account had to be exactly one of these types: faculty, staff, grads, majors, students, or guests. Each of those had certain requirements that had to be met, including, for example, verification by the dean or with the registrar. In addition each user had to sign a form stating that they would abide by the stated reasonable use policy when using their account. These policies were approved by the faculty resources committee.

We wrote an "account" program that users had to run to request an account. (This could be done by users with no account by logging in as "account".) This program would perform certain accounting functions, such as making sure their requested login name was permissible and checking their password against a dictionary, and then would put their account request in the creation queue.

On the administrator side, we developed scripts that automated functions such as creating accounts, expiring accounts, changing passwords, and most importantly, moving accounts from Oldnet to Newnet. When an account was moved to Newnet, we added it to the correct NIS files on Newnet, enabled its shell, and put a trigger in the account that would cause new dotfiles to be installed in the account when the user first logged on to Newnet. We were reluctant to directly modify the user's accounts this way, but Oldnet dotfiles simply wouldn't function on Newnet. We compensated for this by notifying the user that it was happening, and storing their original files as backup versions.

Perhaps the most important aspect of this is that we have a documented account system, and we have an account policy that every user has signed. Converting them over to Newnet was a convenient way for us to reach all of the users on our net and have them agree to the policy.

Demise of Oldnet Predicted (September, 1993)

It was time to open up Newnet to the masses. We had a strategy for moving accounts, we had documentation for them, and we had a functional environment. We announced the upcoming move on local newsgroups and put pointers to those announcements in the message of the day.

Faculty members, their graduate students, and related workstations were the first large group to be moved. We converted about a hundred accounts and a hundred computers in the week between Summer quarter and Fall quarter.

Simultaneously, we converted all of the public laboratory computers and about half of the central computing facilities to Newnet in preparation for the next quarter. We left the other half of the central computing facilities on Oldnet for people who might not get around to moving to Newnet right away.

Once Fall quarter started, we allowed all of the remaining Oldnet users to register for Newnet accounts (assuming they met the requirements). Within a week, we created another six hundred accounts on Newnet.

We were expecting an onslaught of problems and complaints, but for the most part it went very smoothly. The primary difference between a Newnet account and an Oldnet account (beyond the fact that it worked well) was the computer that the user logged into. Their home directory was the same, and their dotfiles had been massaged by the account program, so their environment on Newnet was quite satisfactory. We believe that things went as calmly as they did because we tried to make as few obvious user-level changes as possible, and that we had a lot of documentation available explaining those changes.

These were the key points to our final conversion strategy:

- We were going to disturb a lot of users, so we made a lot of noise about it. (No matter how much noise you make, most of them will be surprised, but at least you can point at your signs.)
- Once we committed to it, we did it as fast as we could in order to minimize confusion.
- We left a way out. There were a few machines on Oldnet, just in case.

Settled Down (Fall, 1993)

Once the majority of users were on Newnet, we were able to stop supporting Oldnet (almost). This gave us time to finish off a few more projects which enhanced Newnet.

We installed new printers and a printer quota system. Both of these had been desperately needed for quite some time.

We rebuilt the physical network from scratch, using 10Base-T ethernet instead of the thick and thin net combination. We moved to SNMP manageable 10Base-T hub units, real routers, and a reasonable network configuration. This drastically improved performance and reliability. We could have made the shift to new wiring while still operating Oldnet, but it would have been nearly impossible to reconfigure the machines and the network addressing. On Newnet, it was relatively simple and resulted in about one day of downtime while we switched all the servers and reorganized the backbone topology. Once the servers were happy, we went back to our tried-and-true technique of incrementally moving machines from one wiring structure to the other while supporting the old wiring with the new.

And of course, we installed a lot of software needed by users that we had neglected. The users will always find things that you haven't done.

Matured (Spring, 1994)

The last Oldnet user machine was converted to Newnet in December. The last Oldnet server (which was being used for Usenet news) was turned off in March. Except for the network rewiring, the majority of Newnet has been up ever since it was conceived.

Newnet continues to be developed. Current projects include:

- Documentation for the systems administrators.
- A request tracking system.
- Proactive network management.
- A server for machine configuration files (eventually targeted as a replacement for "ccsify").
- Figuring out what to do next now that the big goal has been achieved.

Since it's no longer the "New" net, we refer to these developments collectively as "Tenwen".

Fundamental Concepts

While building Newnet, we followed several principles that we felt were essential to our success.

- Consistency and well-documented procedures are critical.
  By being consistent and documenting our design of important systems, we were able to create programs that automated important functions, and we minimized mistakes.
  One example of this is our directory structure and disk mount point scheme, which we relied upon when creating disk management programs.
  In addition, our account types and polices and our software installation methods follow the rules we established for them. We have been able to write programs that automate software installation and account creation which also rely on their being a consistent directory naming scheme.
- Automate anything you do more than once.
  Among other things, we've automated machine configuration, account creation, and the generation of files related to IP addresses. An essential part of all these automation tools are sanity checkers that minimize the opportunity for mistakes.
- When building a new system that will replace an existing system:
  - Create the new system separate from the old system.
  - If they will both exist for a while, support the old one with the new one.
  - Announce the demise of the old one well in advance of really discontinuing it.

  For example,
  - When Oldnet's NIS configuration died, we supported it with Newnet.
  - We supported the old wiring with the new 10BaseT wiring.
  - The Newnet software environment was exported to Oldnet.
  - The old domain name was gradually converted to the new domain name.
  - Newnet itself was built this way.

  We've found that this makes the old system more reliable while testing the new system. In addition, it eases the transition, and gives you the chance to move back to the old system when necessary.
- Have formal, written policies.
  We have policies about the uses of our servers, uses of disk and printer resources, who may have root access, what kinds of computers we support, and who may have

accounts on our network. The faculty resources committee has agreed with and supports these policies, and they are available for reference. In the case of the account and resource policies, users have to sign that they have read and will follow these policies.

- Build in abstractions that you control.
  Abstraction is one of the basic tools of computing. By creating a well-known interface and supporting that interface, one is given the freedom to change the underlying implementation. This is the core philosophy of everything from Turing machines to objects, and it applies to systems administration as well. There are several such examples in Newnet.
  Domain-based mailing, where one sends mail to a domain rather than to a host, was one of the first such abstractions put in place in Newnet. The advantages of doing this are widely documented.
  Our automount scheme mounts directories into a /net hierarchy, and then most user file systems are built with symbolic links into /net. We can change the /net mounts at any time as long as we update the interface, i.e., the symbolic links.
  The software environment mechanism that we use allows us to freely change the location and version of software packages. The user interface is a list of categories of software that they wish to have in their environment. That is expanded into a set of environment variables whose details we control.
- Support and communicate with the user community.
  It would not have been possible to put the effort into Newnet had the users not been willing to wait for it.
  As we built Newnet (and mostly ignored Oldnet), we periodically announced the progress of Newnet. We made the Newnet software available on Oldnet when it became obvious that Oldnet would be around longer than we hoped. Important and vocal users were moved to Newnet early on in order to test Newnet, keep them satisfied, and get them on our side.

User Reaction

User reactions during this whole process were mixed. While we were building Newnet, most users were curious about it. Most of the questions we had were along the lines of "What are you doing, again?", and "How's it going?". Towards the end there was a definite sense of impatience. Overall, the faculty and the user community were amazingly supportive of the whole process.

We expected a lot of negative reactions from users once we made the switch. After all, we were changing their whole environment, and they might not have agreed with our opinions about Oldnet. However, we only got two kinds of negative responses. The first were from the set of users who didn't qualify for accounts on Newnet, either because they were no longer students or had never been. They were understandably upset because they were losing their accounts. The second complaint type was related to software that wasn't installed on Newnet. We tried to fix these as soon as we could, although in some cases we slipped and installed things much later than we should have.

The only other complaint at all came from a user who didn't like the naming scheme we had chosen for Newnet. On Oldnet, most of the computers were named "sun3140a", "sun3140z", "dec5200a", and so on. We wanted something a bit more lively for Newnet, so we named all the Newnet SPARCs after mountains, and the DECs after features of the Northeast, to name a couple of our themes. This user complained that he could no longer figure out what computer to login to based on its name. We told him about the "computers" command, which listed all the available computers, their machine types, their load, and their location. He was satisfied.

Other than these complaints, the user community was very enthusiastic. We got a lot of positive email (which, in our profession, one learns not to expect), and one particularly wonderful faculty member dropped off a box of six dozen chocolate chip cookies for the Newnet crew.

Regrets

The transition to Newnet has been a major success, but we could have improved the process in several ways.

Without question, we should have written a lot more documentation while we were designing. We wrote a lot, but not nearly enough. Now we need to completely document the account system, the NIS methods, the mail system, and the security policies, simply to name a few.

Building Newnet took a lot longer than we expected, partially because we didn't know exactly what to expect. A clearer definition of the goals right from the start would have helped us focus and plan. On the other hand, we felt at the beginning that we could have installed Newnet in less than a month, and perhaps we could have, but it would not have been the solid and cohesive system that it is now.

Finally, supporting two environments for as long as we did was actually quite difficult for the administrators who weren't intimately involved in the design of Newnet. There were always two different ways to do things, and the potential for confusion was high. We should have had better documentation for procedures, or perhaps spent a bit more time trying to make Oldnet easier to administer.

Conclusion

In a bit more than a year, we moved from a dysfunctional environment to a very productive one. We did so by building the new environment separately from the old. The new environment, through the use of consistent naming, abstraction mechanisms, and documented policies, was designed to be easy to administer and use. While we developed the new environment we were able to use it to support the old one. We recommend this approach to anyone that has the option. Even where it may not be possible to follow our suggestions, we feel that the lessons we have learned will be valuable.

Author Information

Rémy Evard has been the leader of the Experimental Systems Group at Northeastern University for two busy years. He received his M.S. in Computer Science from the University of Oregon in 1992, where he developed many of the basic concepts for Newnet while working as a graduate student systems administrator. While leaving behind the bicycles and trees of Oregon was a traumatic experience, he feels that the in-line skating in Boston almost makes up for it. He may be reached electronically at remy@ccs.neu.edu .

Acknowledgements

Newnet was a long and exhausting journey that took place mostly after midnight and on weekends. It couldn't have happened without a lot of help.

Students who put everything they had into the design and implementation of Newnet include Brian Dowling, Ivan Judson, Robert Leslie, and Matthew Wojcik.

People from the University of Oregon and Argonne National Laboratory who had who helped in the initial design include Paul Bloch, Bart Massey, Bill Nickless, and Robert Olson.

The staff who coped with Oldnet while simultaneously contributing to Newnet consisted of Tom Coveney, Lorraine Gabrielle, and Jim Mokwa.

Finally, a major thanks goes to the faculty of the College of Computer Science and to Michele Evard, for hanging on while Newnet was developed.

Bibliography

[1] Sun Microsystems, "The Network Information Service," in System and Network Administration, pp. 469-511, Sun Microsystems, 1990.

[2] Sun Microsystems, "Network File System: Version 2 protocol specification," in Network Programming Guide, pp. 168-186, Sun Microsystems, 1990.

[3] Jan-Simon Pendry, "AMD – An Automounter", Department of Computing, Imperial College, London, May, 1990.

[4] Paul Albits and Cricket Liu, "DNS and BIND", O'Reilly & Associates, Inc, 1992.

[5] Hal Stern, "Managing NFS and NIS", O'Reilly & Associates, Inc, 1991.

[6] Michel Dagenais et. al, "LUDE: A Distributed Software Library", in LISA VII Proceedings, pp. 25-32, Monterey, CA, 1993.

[7] Walter C. Wong, "Local Disk Depot – Customizing the Software Environment", in LISA VII Proceedings, pp. 51-55, Monterey, CA, 1993.

[8] Walter F. Tichy, "Design, Implementation, and Evaluation of a Revision Control System", in Proceedings of the 6th International Conference on Software Engineering, pp. 58-67, ACM, IEEE, IPS, NBS, September, 1982.

Appendix A: Literature

We found many sources of information to be invaluable during the design and creation of Newnet. The articles, books and journals that we read and borrowed ideas from are too numerous to list. However, these were the main sources of inspiration and information:

- The SAGE News area of ;login:, the USENIX Association Newsletter, contains page upon page of good advice.
- The LISA Workshop and Conference Proceedings.
- The complete UNIX System Administrator series of manuals from O'Reilly & Associates.

Appendix B: Time Line

This has been included because some readers may find it useful to refer to while perusing the paper. It's also interesting to note what order various changes took place. In particular, one can see that we built a foundation of core network services, enhanced the environment, and then adapted it to the user population.

July, 1992	- Assessed the situation. Miscalculated.
August, 1992	- Decided to build an independent network.
September, 1992	- Started three SPARC with clean OSs.
	- Designed the global file system.
	- Wrote cloning scripts.
	- Built a name server.
	- Built a mailhub based mail system.
	- Built the NIS environment.
October, 1992	- Defined a policy for servers.
	- Defined a software installation strategy.
	- Recruited students.
	- Started supporting Oldnet with Newnet's NIS servers.
November, 1992	- Defined Newnet clearly.
	- Installed a lot of software.
	- Designed the WWW-based help system.
January, 1993	- Added the first non-systems user to Newnet.
	- Converted a few more machines to Newnet.
February, 1993	-
March, 1993	- Designed the basic user environment.
April, 1993	- Changed the domain name.
	- Created the hosts database and IP config file manager.
May, 1993	- Built the first Newnet DEC.
June, 1993	- Built the first Newnet Sun3.
July, 1993	- Wrote user documentation.
August, 1993	- Wrote the account system.
September, 1993	- Opened Newnet to everyone.
October, 1993	- Closed down Oldnet to users.
	- Installed software that we'd forgotten about.
November, 1993	- Stopped to breathe.
December, 1993	- Moved to 10baseT wiring and a new network organization.
March, 1994	- Shut off the last Oldnet server.

Throughout this whole period, we installed software. At last count, there were approximately 1500 programs in the default path for SPARC users.