The Corporate Software Bank

Steven W. Lodin
Delco Electronics Corporation

The Corporate Software Bank

Steven W. Lodin – Delco Electronics Corporation

ABSTRACT

The Corporate Software Bank is the implementation of an idea borne of many hours spent installing and maintaining public domain UNIX software tools and many hours spent trying to explain the process to other UNIX workstation administrators. The Corporate Software Bank is a company-wide computing resource which makes valuable public domain software available in working form avoiding the usual pitfalls of inexperience and lack of resources.

Delco Electronics Corporation is making the transition from mainframe based computing to workgroup computing utilizing UNIX client/server technology. This has created a need for many UNIX system administrators. Like any major corporation, these administrators can range in experience from someone with no public domain software experience to the guru class administrator.

A decentralized system administration model requires that each system administrator have all the skills necessary to meet their users' needs. Due to external factors, many system administrators are unfamiliar with Usenet/Internet related functions. Of these functions, retrieving, installing, and maintaining public domain software has become more requested from the user community. The Corporate Software Bank is an attempt to solve this problem by changing the answer from a lengthy explanation of the installation process to a simple answer of "retrieve a file and follow the instructions to get that program and many more already in working form".

This paper describes the motivation behind the Corporate Software Bank. It presents a site-dependent implementation discussing the problems encountered and solutions, when applicable. Finally, the usage history is presented and some future enhancements are suggested.

Environment Description

UNIX system administration at Delco Electronics Corporation (DE) is split up according to, but not necessarily limited by, functional, geographical, and business boundaries. The company is divided into strategic business units, each with its own end product or services, that spend their computing-related budgets to meet their needs. These units have three choices to meet their computing requirements: develop internal expertise, utilize an existing DE corporate resource, or utilize a General Motors Corporation (GM) resource. They may select one or more of those resources to meet their needs. Some groups have developed considerable internal talent. Other groups rely solely on a GM corporate resource. The third choice, sometimes known as Core Engineering, is the DE corporate resource of the Software/Computer Tool Development (S/CTD) department. The S/CTD department receives its funding from the other business units to create, administer, and distribute computer projects for DE, and in some cases, all of GM. The S/CTD department provides services in drafting, electronic document management, and software development and management.

A couple of years ago, the only corporate solutions for data management were IBM mainframe or DEC VAX based. Until recently, software development was done under MVS/TSO. Solutions are now being developed and installed that support the UNIX operating environment. These solutions manage corporate data for CAD, electronic documentation, and software development. In addition, other pockets of UNIX workstation use have arisen in niche areas like finite element analysis and manufacturing testing equipment. This has created a need for UNIX system administrators.

In some groups, system administration is recognized as a full time position. In others, it is just another part of the engineering duties. This has led to a wide range of UNIX system administration experience and skills. Because of the large number of different system administration groups, information dissemination can be difficult. This has been magnified by using a decentralized system administration model. See Figure 1 for some indication of the wide range of administration ratios.

The physical layout of the DE network is diverse. It is transitioning from a bridged model to a routed model. The transition started in early 1990 when DE acquired a Class B Internet Address range and installed many routers. As part of a continual upgrade process, existing bridges and repeaters are being replaced with routers. The DE network ranges from building networks of routers, concentrators, and hubs to campus networks of fiber and T1 lines to an international wide area network connected at basically every speed available.

Our link to the Internet is through a connection to the GM network infrastructure known as Infranet. Infranet was designed to connect GM sites worldwide for data sharing. Many other GM entities are on Infranet.

The GM Infranet is connected to the Internet through a firewall. This firewall went into pilot usage in December of 1992 during which there were a few select accounts. It went into production usage in July of 1993, at which point accounts were available to anyone presenting a business case.

To summarize, there is a large network infrastructure in place to support the growing community of UNIX workstations. In addition, there is a general lack of Usenet/Internet experience for system administrators. There has also been a lack of access to the Internet for a normal user or unsavvy administrator until recently. This has resulted in unfamiliarity with public domain software both from a user and administrator standpoint.

Motivation

Supporting a large engineering user community in a large heterogeneous environment has required the use of public domain software to maintain a productive work environment. In creating this environment, we feel we have made the most powerful computing environment for the users and the administrators in our company. This has created a situation where users on other systems have continually requested access to the public domain tools we had available. Some of these tools are the traditional ones like **emacs** or **elm**. Others are advanced programs like **xntp** or **amd**.

Also, some of our internally developed software depends heavily on public domain software such as **perl**. Initially, we expected the sites to which we distribute software to have the public domain software already installed, as a prerequisite. This was a mistake. After trying to explain the process of installing public domain software, we also realized this was a mistake and we started including the software in the distribution. There were no licensing issues since we do not charge a fee for our software.

Increasingly, more time was spent making public domain software available, explaining the process for compiling and installing the software, and occasionally, getting an account on a remote system and installing the software. It was realized that unless something was done that would streamline this process, the effort for public domain software would soon be unmanageable. We analyzed the situation and came up with the following possible problems that needed to be addressed:

- Make programs available in working form
- Install software consistently and securely
- Make sources available
- Support multiple architectures
- Reduce disk space utilization

Based on the range of experience of the system administrators we had been dealing with, the solution would have to be a quick step-by-step method. The solution to these problems is the Corporate Software Bank (CSB).

Making programs available in working form would save time because the program would be installed once, in one location. The build and installation process would not need to be explained many times to many different administrators. In many previous instances, users would get their system administrators to call to get a program installed on their machine. The users and system administrators would expect something simple like copying over one executable. Most of the time, they did not feel the effort was worth the benefits of the program and the user needed to find alternative methods for solving his problem. To be fair, for someone who has never installed public domain software, it can be daunting. Once you do it a couple of times, it's just like following a recipe. The **configure** program included with GNU packages and the usage of **imake** in X11 programs has made the build process of public domain software much easier.

It was considered important to have a consistent methodology for compiling and installing the public domain software while still making it easily

Site – Admins	Seats	Accounts	Active Users
Site A – 4	250	830	500
Site B – 5	56	150	60
Site C – 2.5	87	122	80
Site D – 1	3	66	26
Site E – <1	4	60	40
Site F – 4	275	360	60
Site G – 2	40	550	450

Figure 1: Administration ratios

available to others. This implied a single hierarchy for installation instead of installing programs all over the filesystem like **/usr/bin/X11** or **/usr/local/bin**, or having to install in **/package_name** and using links. Another concern was to have a methodology for installing the software in a fashion that eliminates any security risks. The traditional guidelines for checking out public domain software are followed.

Making sources available is not as important now that there is an Internet connection that is accessible by everyone. It is nice, however, to have a single location known to have up to date sources on the local network. In addition, for those administrators that wish to have certain software running locally, this allows them to utilize the CSB for testing the software, then later copy the software and compile and install it locally.

Supporting multiple architectures is a problem that the CSB tries to solve to a limited extent. The list of currently supported platforms happens to coincide with the list of platforms supported by our system administration group. However, all software that is installed in the CSB for UNIX platforms is installed for as many of those platforms as possible. Obviously, some public domain software is not built to be compatible on all platforms, and it is not worth our time to try porting it. Generally, the more popular (which is usually proportionate to useful) a piece of software is, the more platforms it is supported on.

Reducing disk space utilization was a consideration for implementation that is based on our administration model. Our workstations are set up as dataless so the only uses of the local hard disk on a workstation are operating system, windowing system, and swap space [Nelson92]. As a result, we did not want to have a solution that required adding software to the local workstations. Also, having a good network infrastructure for the DE Kokomo campus allowed us to decide to make the CSB available to everyone else utilizing the network.

One of the other problems that we encountered was having multiple copies of sources and executables scattered both in system and user directories. Granted, this problem could have been worse if we had an Internet connection that all users had access to. Having one central location has helped eliminate this problem. Users now know that there is a location and methodology for making public domain software available to them.

Attempting to solve those problems would hopefully provide the following benefits:

- Manpower savings
- Enhanced productivity
- Material savings
- Quality improvement

It saves the installer time by only requiring the installation of the software once, not once for our workstations and once for each of the other workstations that want the program. It also saves time for all the other administrators because they do not have to individually install the software. Once they do the initial installation of the CSB, they have access to all the programs.

In the end, the user benefits from enhanced productivity because high quality software is available to them at little or no cost in terms of disk space or administrative time. It also gives users working at remote sites access to the same tools as the home site.

The implementation and the network infrastructure allow disk space savings to be realized since the CSB can be accessed across the network. This may not seem like much at first, but when you start to consider the tens to hundreds of megabytes on hundreds of workstations then it starts to add up.

Finally, there is an implicit quality improvement. In most cases, the most recent version of a particular piece of software is available. For public domain software, a newer version usually includes bug fixes and program feature upgrades. In the case where a vendor delivers the software with the operating system, like HP with elm, the CSB offers a newer enhanced version.

While working on the initial concept, it was decided to try to make this as general as possible to appeal to as many users or systems as possible. This meant that no proprietary or strategic data or proprietary programs would be installed or accessible. This also meant that our group could not install some of the locally developed programs that were designed for our own user community. Also, based on the administrative boundaries and funding policies, it would be impossible for commercially licensed software, such as Lotus 123 or Interleaf, to be made available via the CSB. At this point, it was decided that only public domain software and locally developed software that could be distributed freely should be installed in the CSB. Finally, it was decided that the CSB would not be an infinite storage facility. This affected the implementation in two ways, the clients would mount the CSB read-only, and programs would use the local workstation disk for temporary files. This way, no programs would be writing to the CSB disk, possibly filling up the filesystem.

Implementation

The implementation of the CSB is highly site dependent. An explanation of the reasons behind our implementation is presented. The first thing noticeable is that we did not use **/usr/local** as the hierarchy. The basic reason for this is that in our client workstation model, **/usr/local** is on the local hard disk of the workstation and contains only the software required to be present when the network is not. In addition, we realized that other groups have been using **/usr/local** for their own purposes and we did not want to cause any conflicts if possible. This meant that we would have to create a new directory structure. We chose **/usr/std** for historical reasons. The choice of std actually worked out well, because users now reference the CSB as "standard" which can be construed as the meaning for std.

As illustrated in Figure 2, the layout of the CSB is based on architecture. Each architecture has its own hierarchy similar in layout to a normal **/usr/local** hierarchy. Each client workstation mounts the filesystem from the server at the architecture mount point to **/usr/std**. The development workstations have the ability to mount the CSB one level higher to have access to all architectures and the source directory in addition to mounting as a client workstation for normal access to the CSB. The dashed lines in Figure 2 show the NFS mounts that client and development workstations perform to gain access to the CSB. In a heterogeneous environment, the use of an automounter that can key on the workstation architecture facilitates the mounting process immensely.

The server utilized in our implementation is a Auspex NetServer NS5000. The Auspex server is the backbone of our dataless client administration model. As an added benefit, it has performed flawlessly in its role to serve large amounts of data to many clients. The important characteristics of this server that make it suitable for the CSB are:

- Industry leading NFS throughput
- Multiple Ethernet connectivity
- Large disk capacity
- Filesystems larger than 2GB

As demonstrated in the latest SPEC SFS Release 1.0 Benchmark Suite (097.LADDIS), Auspex is leading the pack in terms of NFS throughput. In particular, our Auspex is a hybrid between the NS5000 and NS5500. It is running version 1.5 of the Auspex operating system, has 23 SCSI drives ranging in size from 1GB to 2GB for a total usable disk space of 37GB, and has the maximum of 8 ethernets.

The Auspex server is capable of disk striping, concatenation, and mirroring. We have created a 5GB virtual partition that is striped across 5 disk drives for the CSB and other network available software. We limited the partition size to 5GB so it would fit conveniently on a Exabyte-8500 written tape.

With its 8 Ethernet interfaces, the Auspex server appears locally on 8 different subnets. This allows us to bypass a router, making the delivery of data even faster. Over 450 workstations and PCs have direct access to the server.

The current implementation of the CSB only supports hp300/hp400, hp700, and sun4 architectures. In addition, the sun4 architecture is limited to the least common denominator, SunOS 4.1.2. The supported architectures coincide with the architectures supported by our system administration group. We do not have access to the other platforms at DE such as Apollo Domain, Intergraph, and DEC Ultrix, and volunteers have not been recruited yet. Also included in the CSB are directories for PC-compatible and Apple Macintosh programs. The programs included for those architectures are network applications and UNIX-like utilities.

To save disk space, hard links are used between architectures where possible. This includes library files, info files, and manual pages. See Appendix A for the CSB client installation instructions and checklist.

It was a requirement that simple instructions were to be provided in order to make the CSB available to everybody. The client installation instructions are listed in Appendix B. It is a seven step process that needs to be done as root on the CSB client machine.

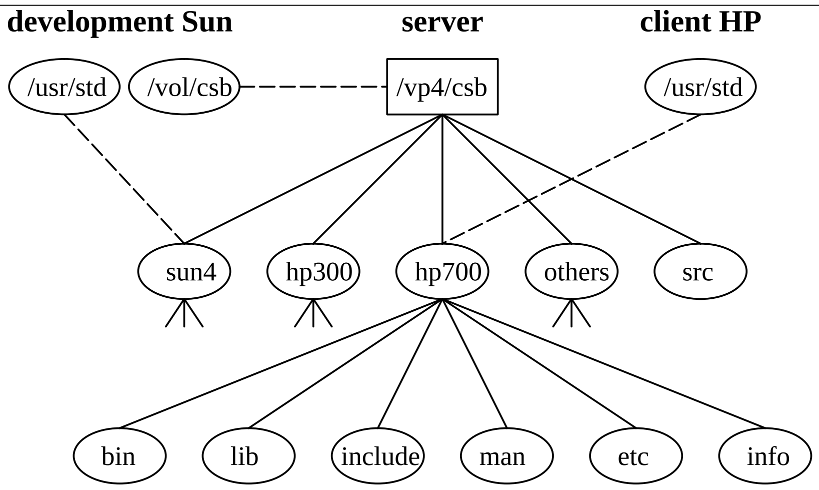

Figure 2: CSB layout

In the instructions, it is specified that the clients mount the CSB read-only. Since we did not want to control who had access, everyone gets access. We did not want to leave the CSB open for attacks of denial of service by filling up the disk drive, or becoming a privileged user and being mischievous. We also did not want to maintain a large **/etc/exports** file or NIS netgroup. The server has a simple exports line:

```
/vp4/csb -rw=kocrsv01:kocrsw15,
root=kocrsv01:kocrsw15:,anon=97
```

This gives read-write access only to the development machines and maps anonymous uids to the csb uid. The five development machines are sun4 workstations with SunOS 4.1.2 and SunOS 4.1.3, an hp300 workstation with HP-UX 8.07, and hp700 workstations with HP-UX 8.07 and HP-UX 9.01. We used [Stern91] and [Garfinkel91] as references.

Problems

During the implementation and the throughout its usage, there have been problems associated with the CSB, both technical and non-technical. The biggest non-technical problem has been the acceptance of the CSB. It has not be as widely accepted as possible because of the negative image some people (unfairly) associate with public domain software. This image usually arises due to fear of the unknown, the virus scare, and the licensing issues with shareware.

The other non-technical problem is the acceptance by other system administrators to use a resource not developed by themselves or not commercially supported. This attitude is difficult to change. In this situation, the merit of the software can do more persuading than words.

Another problem was spreading the word to all the users and system administrators that might benefit. Since there is no central system administration group and there are many different organizations using UNIX workstations, this was difficult. Notices were sent out to the members of the DE UNIX Users Group mailing list and to the system and network administrators in the DE Network Users Group. In addition, some GM users were introduced to it during a Software Tools User Group meeting. Finally, other GM users posting to NetNews asking about public domain software were sent the CSB information since they can access the CSB on the GM network.

Some software problems have been encountered. To reduce the security risks, we have not allowed any programs to be installed setuid/setgid. This means programs like the HP monitor program (which needs setgid to the kernel group to access kernel data structures) will not work when run as a normal user. It must be run as root on the client workstation to have access to the kernel.

Another problem was X11 programs in general. Most X11 programs like to be installed in the default X11 binary directory **/usr/bin/X11** and they like to have their application resource files installed in **/usr/lib/X11/app-defaults** along with their fonts in **/usr/lib/X11/fonts**. The standard X11 startup programs default to those locations. Since we wanted the CSB to be a standalone hierarchy, it was decided that installed X11 programs would have their application resources stored in **/usr/std/lib/app-defaults** and their fonts in **/usr/std/lib/fonts**. This paralleled the standard configuration of **/usr/lib/X11**. All our users have the XAPPLRESDIR environment variable pointing to a directory in their user area called **.app-defaults**. It took a while to figure out that the XAPPLRESDIR environment variable is like the PATH or MANPATH variable in that it can have multiple entries colon separated. Before that discovery we advised the users to copy the application resource files out of the **/usr/std/lib/app-defaults** directory into their **.app-defaults** directory. The X11 fonts in the CSB can be added with an **xset** command.

Most public domain software available today seems to have been developed on a Sun or some machine running a BSD-derivative operating system. It will be interesting to see if this changes now that Sun delivers Solaris 2.x, a SysV-based operating system. Anyway, compiling software on HP-UX is not as easy as generic BSD or generic SysV since it seems to have a SysV kernel with BSD extensions. Of great help was the "INFO: GNU products on HP9000s700" information posted regularly to comp.sys.hp by Pierre Mathieu which lists the GNU programs that compile without modification and gives the modifications for those that need it to compile on HP-UX. In addition, the "BSD to HP-UX porting tricks" article posted regularly to the newsgroup comp.sys.hp by Mike Peterson has helped out. Finally, for software guaranteed to run on HP workstations, the HP Interworks organization has an anonymous ftp site, iworks.ecn.uiowa.edu, and the Liverpool UK HP-UX Software And Porting Archive organization has an anonymous ftp site, ftp.csc.liv.ac.uk. These two generous groups post updates on their software libraries regularly on the newsgroup comp.sys.hp.

There are three connectivity models for accessing the CSB:

- high speed network access – NFS
- low speed network access – **rdist**
- no network access – tape

The CSB was designed to perform best using a high speed network infrastructure. In testing, some software would fail at remote sites connected at 56KB. Other software would take excruciatingly long. Based on this, it was decided that anything connected at 56KB and less should install the CSB locally. This is solved relatively easily using **rdist**.

The instructions for remote site installation and the **rdist** distfile are shown in Appendix C.

Not every site we work with is connected to our network or Infranet. In those cases, we have distributed the CSB via tape. We can accommodate a wide variety of tape media. This is the worst case scenario.

There have been a couple of other miscellaneous problems encountered. Due to having workstations not administered by us and unknown to us mounting the CSB, the exported mount point is relatively permanent. This is only a minor inconvenience. Recently, when we upgraded the filesystem from 2GB to 5GB, it required a little extra work.

Another problem is one of unauthorized local modifications. For example, a program is installed correctly in the CSB but does not work quite right on a workstation, possibly due to an administration or configuration deficiency. The program is copied locally and modified to work around the problem. When we send out an upgrade, the copied program mysteriously quits working. This is difficult to track down and fix.

A relatively minor problem with HP-UX is that it only allows for one whatis manual database. It will now follow the MANPATH environment variable, but it requires a single merged database in **/usr/lib/whatis**. This is described in the HP-UX *catman* (1M) man page.

Upgrading existing packages in the CSB is currently done randomly. There is no established procedure. Upgrades are done when users or administrators notice that programs have new versions. Typical notification comes from the announce newsgroups or the source newsgroups and from mailing lists. This is one area that needs to be improved.

Usage History

There are three statistics that are considered important, the number of packages installed, the amount of disk space utilized, and the number of machines mounting the filesystem. These statistics are presented for information only, not for numerical analysis.

The first statistic is the number of packages installed. This number should be increasing to keep the resource from becoming stale. In addition, another important factor is keeping the existing packages up to date. When the CSB was announced to the user community in April, it contained over 50 packages installed for the supported UNIX platforms. In August, it contained over 60 packages for the UNIX platforms. See Figure 3 for more information. Of the packages installed, 35% have been upgraded.

The second statistic that is considered important is the amount of disk space utilized. There are 2 categories of utilization, source and working form. Figure 4 shows the change in utilization for the three binary architectures supported [hp300, hp700, sun4] and for the sources [src]. Due to their insignificance, we have neglected to show the utilization for the PC and Mac directories. The reason for the reduction in the src directory is that the source directory is also the build location and at any given time, some sources may be uncompressed and untarred for maintenance or installation. The amount of disk space utilized is important for remote site distribution.

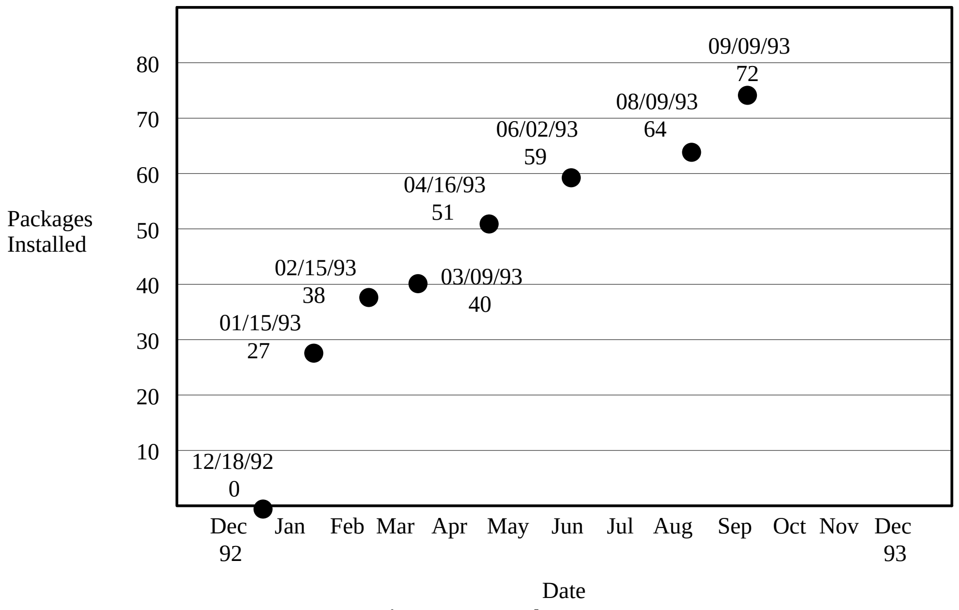

Figure 3: Packages

The number of machines utilizing the CSB is the third important statistic. This will show the number of machines actually using the CSB at a given time. Of the over 200 workstations that we administer, there may only be a fraction that have the CSB mounted due to the automounter. The ratio of our workstations to the total amount of workstations mounting the CSB is generally around 70%. See Figure 5 for a couple of data points, taken at random, of clients mounting the CSB across the network. This data was gathered by doing

```
grep csb /etc/rmtab | grep -v ^# | \
          uniq -u | wc -l
```

on the Auspex server. The Auspex at any given time is supporting over 6300 active NFS mounts from over 300 clients.

Future Enhancements

To most improve the CSB, the available list of software and if possible, the supported platforms must continue to grow. The available list of software needs to be increased by adding different binaries and making more sources available. In particular, including both the full X11R5 and GNU sources would be beneficial. As always, disk space is a consideration along with download time and

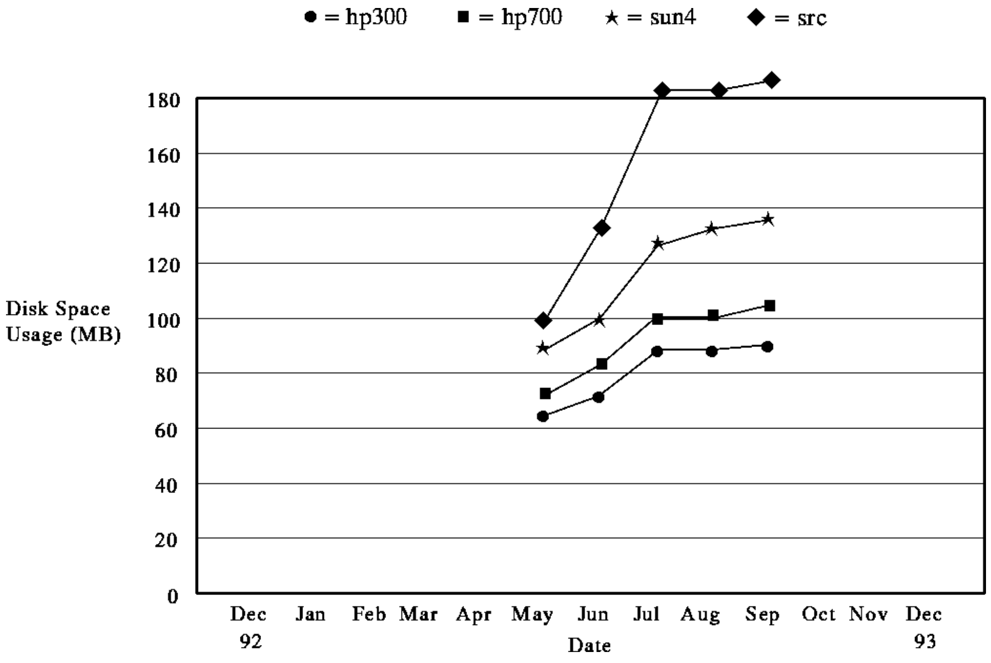

Figure 4: Disk Utilization

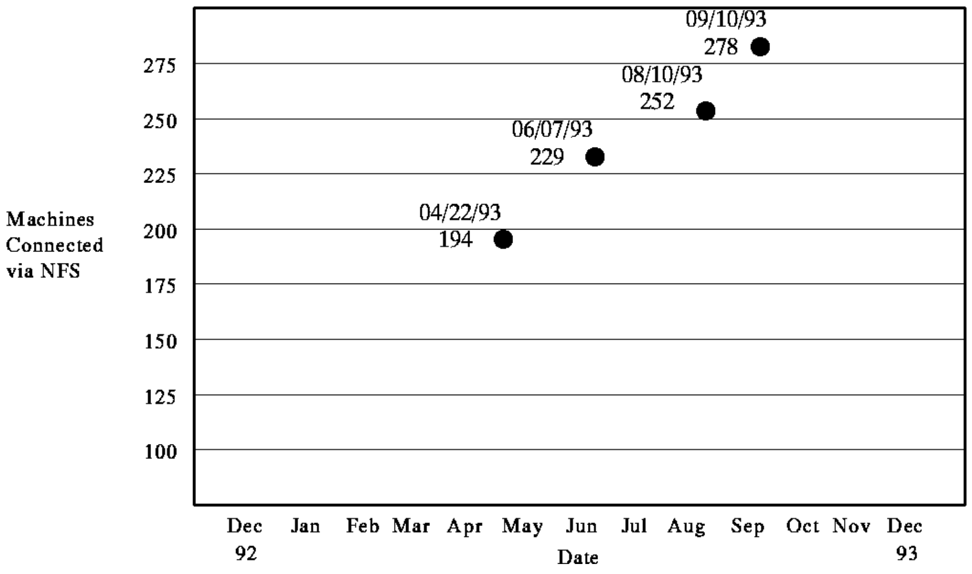

Figure 5: Machines using CSB

network bandwidth. In addition, more PC and Mac sources related to networking, email, and NetNews would be beneficial. The problem with those is finding someone knowledgeable and experienced to support them.

One possible solution to the diskspace dilemma is to utilize a CD jukebox facility to make X11R5, GNU, and other public domain software CDs available. Other possible CDs include the NetNews CD and Sun's Catalyst CDWare CD. This solution has not been explored further.

Another possible enhancement for the CSB would be to create and maintain a vendor patch directory. This would make vendor patches available to all the different system administrators. Since our primary vendors, Sun and HP, have patches available on the Internet, it would be relatively simple to start the patch directory. However, the difficult part would be maintaining it. This would include keeping separate directories for different operating system releases. Another maintenance task would be to remove patches that are superseded by newer patches. Since manpower is limited, this enhancement probably will not be implemented in its full capacity.

Although this does not seem feasible currently for a number of reasons, it would make sense to load any site licensed software and make it accessible to everyone via the CSB.

One of the more time consuming future activities will be creating a Solaris 2.x architecture. Since we do not anticipate supporting that architecture until the third quarter of 1994, we have plenty of time to prepare for it. Some software already has been ported; others will compile fine using existing SysV parameters. We will keep watching the net for hints.

Conclusion

The Corporate Software Bank is a valuable resource at Delco Electronics Corporation. It has saved both time and money. It is available on the DE Kokomo campus via NFS, on the DE wide area network or the GM wide area network via weekly rdists or to our unconnected customers via magnetic media. There is a README file describing the benefits, installation, structure, and installed packages available from the author via email or via anonymous ftp on the GM wide area network at kocrsv01.delcoelect.com in the doc directory.

The Corporate Software Bank is an evolving resource. It is continually being improved. All suggestions for improvement will be accepted. Please send them to the author.

Acknowledgments

Many thanks to the other Software/Computer Tool Development system administrators, Chris Cleary (DE), Ryland Rusch (DE), and Karen Kerr (DE). Thanks also to the CSB supporters, Dianne Glickfield (EDS), Mike Thoeny (DE) and Mike Scheidler (DE). Finally, many, many thanks to Rob Kolstad (BSDI), who at a critical period in my life, kept me on the system administration career path.

Author Information

Steven W. Lodin earned a B.S. in Electrical and Electronics Engineering with Computer Option from North Dakota State University in 1988. He has worked for General Motors since 1985. Currently he is working in the Software/Computer Tool Department of Delco Electronics Corp., which is a subsidiary of GM Hughes Electronics, which is a wholly-owned subsidiary of General Motors Corporation. As part of the System Administration Group, he is responsible for the administration of a heterogeneous world-wide network utilized for Systems Engineering and Software Development for most of Delco Electronics vehicle electronic systems. He is currently spending most of his time working with NetNews, email and of course, public domain software. Reach him via US mail at Delco Electronics Corp., MS CT-LL-M, One Corporate Center, Kokomo, IN 46904-9005 or send mail electronically to swlodin@kocrsv01.delcoelect.com.

References

[Garfinkel91] Simpson Garfinkel and Gene Spafford, "Practical UNIX Security", O'Reilly & Associates, June 1991.

[Harrison92] Helen Harrison, "So Many Workstations, So Little Time", LISA VI Proceedings, pp. 79-87, Long Beach, October 1992

[Nelson92] Bruce Nelson, Rapheal Frommer, & Auspex Engineering, "An Overview of Functional Multiprocessing for NFS Network Servers", Auspex Systems, August 1992.

[Schafer92] Peg Schafer, "bbn-public -- Contributions from the User Community", LISA VI Proceedings, pp. 211-213, Long Beach, October 1992

[Stern91] Hal Stern, "Managing NFS and NIS", O‘Reilly & Associates, June 1991.

Appendix A: Instructions for Package Installation (v3.0 09/01/93)

1) Put the source package in /vol/csb/src.
2) Unpack the package.
    - If the package ends in .tar.Z then uncompress and untar.
    - If the package ends in .tar.gz or .tar.z then gunzip and untar.

- If the package ends in .sh or .shar, the unshar it using sh.

3) Read the README and INSTALL files.
4) Change package configuration options:
   - Modify and config.h or conf.h files per instructions.
   - Modify Makefile per instructions.
5) Install the package:
   - If GNU-type package then
     - modify Makefile.in to change the prefix to be /usr/std from /usr/local
     - configure
     - make
   - If X11 package then modify the Imakefile to include:
     BINDIR = /usr/std/bin
     LIBDIR = /usr/std/lib
     XAPPLOADDIR = /usr/std/lib/app-defaults
     EXTRA_LIBRARIES = -lm
     Then

```
xmkmf
make
```

   Then
     strip the executables
     make install
     remove any library or man pages installed
     link libraries and man pages
     make clean
   - Otherwise, follow the instructions in the README or Makefile.
6) Repeat step 5 for all platforms applicable.
7) Save the package with the modified configuration:
   Change directory to parent above the package.
   Tar and compress the package:

```
tar cf package.tar package
compress package.tar
```

8) Remove the package build directory.

```
rm -rf package
```

9) Add entry to the README file. Increment the minor release number at the top. Copy the new README into the anonymous ftp directory on kocrsv01.delcoelect.com (~ftp/sys_info/CSB.README).
10) Post to NetNews in the delco.software.csb newsgroup. Post man pages sections if possible. If a minor upgrade, post the CHANGES files.

Appendix B: CSB Installation

Now that you know more about the CSB and you've decided that you want to start using it, there are some fairly simple steps (executed as root) that need to be taken. I am going to illustrate a single architecture mount example for the HP700 platform. This assumes the workstation is configured properly running DNS (Domain Naming System) or has a current hosts file. The most up-to-date hosts file can be retrieved from kocrsv01.delcoelect.com via anonymous ftp in the sys_info directory.

- Outside CTC Building
  1. Create directory /usr/std (mkdir /usr/std)
  2. Modify /etc/checklist to add the following line:
     kocrsv04:/vp4/csb/hp700 /usr/std nfs ro,bg,soft,intr 0 0
  3. Mount the directory (mount /usr/std)
  4. Modify the default or each user's PATH environment variable to include /usr/std/bin
  5. Modify the default or each user's MANPATH environment variable to include /usr/std/man
  6. Modify the default or each user's XAPPLRESDIR environment variable to include /usr/std/lib/app-defaults
- Inside CTC Building
  Modify the line in /etc/checklist or /etc/fstab to use one of the following instead of kocrsv04. This is because the CSB server has multiple ethernet ports which allow direct access to some floors/wings bypassing the CTC router.

If on floor	Use	If on floor	Use
CTC-LL	sv04_02	CTC-1E	sv04_04
CTC-2E	sv04_06	CTC-3E	sv04_08
CTC-3W	sv04_09	CTC-4E	sv04_10
CTC-4W	sv04_11		

Appendix C: Weekly Rdist Installation Instructions

This is the suggested method for access to the CSB from WAN connections (DE WAN or Infranet). The binaries, libraries, and man pages are all available locally, with weekly updates via the rdist command.

Required:

- The rdist command in /usr/ucb/rdist. We can provide this for HP workstations since its not included in HP-UX.
- Enough disk space to hold the CSB for each architecture requested. This is a constantly increasing number. Check with the CSB administrator for the latest guess.
- A local account, csb, on the local CSB server. It should not be in NIS. A suggested /etc/passwd entry is:

```
csb:*:97:9:Corporate Software Bank,,,:
     /vol/csb:/bin/csh
```

- It will also need a .rhosts file in its home directory with the following machine entries (these are all for just one machine):

```
kocrsv04
kocrsv04.delcoelect.com
sv04_03
sv04_03.delcoelect.com
```

Rdist file

```
REMOTE_CSB_ALL = ( mwbchp01 )
REMOTE_CSB_SUN4 = ( gmpxhp02 edssun ele1 lxeusv01 sun01 )
REMOTE_CSB_HP700 = ( flidh102 gmpxhp02 edssun sun01 hp22 )
REMOTE_CSB_HP300 = ( flidh102 )

inst:
        (/vol/csb/sun4 /vol/csb/hp700) ->
        (sun01) install;

csb.all:
        (/vol/csb/sun4 /vol/csb/hp700 /vol/csb/hp300) ->
                (${REMOTE_CSB_ALL}) install -R;

csb.sun4:
        (/vol/csb/sun4) ->
                (${REMOTE_CSB_SUN4}) install -R;

csb.hp700:
        (/vol/csb/hp700) ->
                (${REMOTE_CSB_HP700}) install -R;

csb.hp300:
        (/vol/csb/hp300) ->
                (${REMOTE_CSB_HP300}) install -R;
```