Internal Use Only
非公開

TR－A－0158 30

HIP.ATR.CO.JP.

The Open Computing Environment

Ed Gamble

1993. 1. 8

（1992.11.30受付）

ATR視聴覚機構研究所

〒619-02 京都府相楽郡精華町乾谷 ☎07749-5-1411

ATR Auditory and Visual Perception Research Laboratories

Inuidani,Sanpeidani, Seika-cho, Soraku-gun, Kyoto 619-02 Japan

Telephone: +81-7749-5-1411
Facsimile: +81-7749-5-1408
Telex: 5452-516 ATR J

© ㈱ATR視聴覚機構研究所

HIP.ATR.CO.JP

The Open Computing Environment

Ed Gamble

ATR Auditory and Visual Perception Laboratories

2-2 Hikari-dai, Seika-cho, Soraku-gun

Kyoto 619-02 Japan

Contents

Abstract

This paper describes the implementation of hip.atr.co.jp as part of the open computing environment now in use within the ATR Auditory and Visual Perception and Human Information Processing research laboratories. Portions of this implementation, particularly Domain Name Service (DNS), have been adopted for use throughout ATR.

This is not a tutorial on network environments nor is it very difficult. The important programs, all simple shell scripts or data files, are included.

This paper has no technological significance whatsoever. It is an embarassment to the author to even describe this. However, documentation is warranted because of the unfortunate efforts expended to overcome the parochialism of all to many people.

There are three people here at ATR who through a combination of political and technical support, made hip.atr.co.jp a reality. They are: Yohichi Tohkura for his endorsement of the idea, Shigeru Katagiri for his authority needed to sway people's views and his recognition of the idea's validity, and finally, Satoru Fujita who very quickly picked up the ideas behind the uniform environment and helped tremendously. There were others who supported the idea, but these three actively supported the idea.

1 Introduction

This paper describes the open computer environment for the Human Information Processing laboratories within ATR. Open environments are based on the fundamental philosophies of uniformity, equality, and cooperation. These environments provide equal access to all and minimize wasted resources primarily of people attempting to learn about computers. A uniform, open environment means the people work together to improve the computational environment.

The environment is designed to increase scientific productivity by reducing the time spent on systems maintenance, software installation and general computer 'dirty' work. Here are the basic advantages of the "HIP Open Computing Environment:"

Uniformity If everyone shares the same environment, then everyone can help one another. For example, if you have a question, then most likely someone will know the answer because they use the same programs. Also, if someone installs some great, new software, then anyone else can also use it. And if someone writes a useful program, then he or she can easily make it available to all.

Reliability If your machine breaks down or crashes; no problem. You can use a different machine until your machine is fixed. Since all machines are nearly identical this 'machine independence' is easy to achieve.

Maintenance Regular tape backups need not be performed by you. Filesystems can be backed-up while online without you knowing. Our systems administrator does this while you are sleeping.

Upgradeability When a new machine arrives, a few simple files are updated and instantly every other machine in the laboratory knows about the new machine. Also, the machine is easily installed by the HIP systems administrator; you do not need to be familiar with installation procedures. Your machine is working within one hour of power on... guaranteed!

Similarly if a new person arrives. They get an account and instantly all computers throughout the laboratory know about the new user and all software becomes available to that person.

Science You do not have to be an expert at system maintenance any more. You do not have to learn about software installation. You do not have to learn about network addresses and gateways. You do not have to know what NFS is and why it is good.

HIP is currently implemented on over 60 machines within ATR. Currently all machines share the same NIS database and have identical filesystems. Therefore every machine is equal; a researcher can sit at any machine and have the same files, programs and environment.

In this paper we provide the details of the HIP environment. Such details include: the programs currently provided, the structure of filesystems, the purpose of file servers, the structure of network information service (NIS), sendmail and domain name service DNS. Most importantly, the environment has be designed to allow for individual freedom and variations among machines; this aspect of the HIP environment also is described.

We start with background information to illustrate that implementation of an open, uniform environment does not require new technology; in fact, uniform environments are simpler to implement than nonuniform, isolated machines.

1.1 Background

The idea of shared computer resources and an open computer network is a common, generally know idea. Institutions like M.I.T. and Stanford have campus wide computer networks with upwards of 500 computers of various manufacturers all seamlessly interfaced to provide a uniform environment for all students and staff.

At Stanford, for example, all students are given one account when they arrive at the university. This account enables the student to use hundreds of computers from which the student can access his own personal files and mail. The computers include Sun SparcStations, IBM RS/600s and Next WorkStations. One subnetwork at Stanford includes 100 computers; it is managed by one fulltime systems manager, one fulltime systems programmer and one part-time student.

At MIT, upon entry to the Institute, every student is given an account and therefore access to over 1000 campus wide computers. All computers have largely the same environment even though the computers are of different brands. The student can run X windows, editors, text formatters as well as sending mail. The student can easily connect to any other computer in the network.

The MIT AI contains nearly 300 computers. There are 100 lisp machines, over 100 SparcStations and various other Hewlett-Packard and Apple computers. Lisp Machines are drastically different from SparcStations; yet filesystems are shared, mail is accessible, and remote login sessions are possible. The AI lab employs one systems manager; only one person for all these machines.

Compare these massive computer networks with the previous system here at ATR. At ATR we had many small computer groups; in fact it seems that each research leader had his own small group. All the user accounts were distinct among each of these groups. Every group was managed by one or more people. These systems managers were responsible for setting up new machines, installing users and software, backing up and maintaining file systems and interfacing with the other groups. None of the groups were very big; my estimate is that the average size was 6 workstations or X-terminals.

It should be easy to see that large amounts of resources, both computer and more importantly human resources, were being wasted here at ATR.

This previous, unfortunate situation has been eliminated by the creation of the HIP environment.

1.2 The HIP Philosophy of Uniform Environments

The philosophy is summarized by the following statements.**When a researcher first arrives at ATR, the researcher should be given one and only one home directory. This home directory should be accessible from all the machines that the researcher would normally use during his research activities. Those possibly different machines should provide a nearly identical environment.** This statement applies for all machines running the Network File System (NFS) and therefore applies to all Suns, Alliants, HPs, DECs, etc.

Some of the direct advantages are: 1) the researcher need not maintain copies of his/her files on several machines which reduces confusion, wasted disk space and accessibility problems, 2) multiple disk space need not be reserved for the user which simplifies backup procedures and reduces cost, and 3) setup and configuration of the researcher's disk space is performed once, not many times when each new machine arrives.

There are some additional advantages that become easily implementable if home directories are uniformly accessible. First, mail access and delivery becomes distributed. That is, researchers can read and send mail from their local machine rather than being forced to log on to one machine. Second, utilities such as emacs, X-windows, and LaTeX text formatters and postscript previewers would be shared among all machines. Thus multiple copies of source, include, library and binary files need not be maintained on separate machines and most importantly *one person performs the installation for the entire network* thereby dramatically saving on computer and human resources. Furthermore, all users of the computers see a uniform environment. Programs are readily accessible across different machine architectures; users need not remember and learn multiple commands that depend on which machine the user is logged in to.

2 The Implementation of HIP

The implementation rests upon three fundamental concepts: NIS domains and File Server Clusters and Automount. These three concepts are described in the following sections:

2.1 The NIS domain: hip.atr.co.jp

The NIS concept is to efficiently share certain information needed to interface different machines. The implementation of NIS uses a master NIS server which maintains all the databases, slave NIS servers with keep copies of databases, and clients which make database requests and receive answers from the slave or master servers. The NIS implementation is robust to failure of any machine.

Below are excerpts from the NIS implementation on the hip.atr.co.jp NIS domain. We do not show the standard NIS files "hosts" and "password" nor the detailed "services" and "protocols" files; only some unique files for hip.
From netgroup:

```
vision          (hubble,,) (helios,,) (hoshi,,) \
                (hesperus,,) (halley,,) (hole,,) (hyperion,,)

cognition       (hartree,,) (higgs,,) (hamilton,,) (hopfield,,) (hercules,,) \
                (holmes,,) (hollywood,,) (hebb,,) (hermite,,)

hearing         (hochspits,,) (hokuheki,,) (hobbes,,) (himal,,)\
                (hsun07,,) (hsun08,,) (hsun09,,) \
                (hsun10,,) (hsun11,,) (hsun20,,) (hsun21,,) (hsun22,,) \
                (hsun23,,) (hsun24,,) (hsun25,,) (hsun26,,) (hsun27,,)\
                (hana,,) (himawari,,) (herb,,) (hibiscus,,) (holly,,)

cm-net          (hcmfs,,) (hawking,,) (hsuncm,,) (hcm01,,) (hcm02,,) \
                (hazel,,) (hamburger,,)

hiphosts        hearing vision cognition cm-net
```

From **auto.master:**

```
# mount point   map-name        options
/-              auto.direct     -rw,hard,intr,noquota
/home           auto.home       -rw,hard,intr,noquota
/net            -hosts          -rw,soft,rsize=1024,wsize=1024,nosuid
/homes          -passwd         -rw,hard,intr,noquota
/DB             auto.db         -rw,hard,intr,noquota
/hip            auto.hip        -rw,hard,intr,noquota
```

From **auto.direct:**

```
# mount point           mount-options           location
/usr/vision             -rw,hard,intr,noquota   hubble:/export/vision
/usr/hearing            -rw,hard,intr,noquota   hsun10:/export/hearing
/usr/cognition          -rw,hard,intr,noquota   hermite:/export/cognition
/usr/local/share/src                            hubble:/export/source
```

From **auto.home:**

```
# mount point           mount-options           location
*                                               &:/export/&
```

Regarding the line “ &:/export/&” in auto.home. Within hip.atr.co.jp, a machine with disks mounts that disk under /export/hostname, for example /export/helios with home directories under /home/helios. When a machine has multiple disks the disks are mounted under hostname1, hostname2, etc and the user directories are listed as /home/hostname1, /home/hostname2 and so on. Thus every disk has a unique label (since every machine has a unique name) and that disk label is also in the hosts database. So, for example hsun08 has two disks mounted under /export/hsun08a and /export/hsun08b, users are given home directories of /home/hsun08a and /home/hsun08b and the hosts NIS database lists hsun08 as "133.186.40.58 hsun08 hsun08a hsun08b" and everything works easily.

This arrangement also is robust to machine failures. If a machine dies, the disks can be removed and temporarily placed in another machine and then only the NIS database needs to be updated. So, for example, say hsun08 dies. The disks are moved to hsun10 and the hosts database for hsun10 is changed to "133.186.40.60 hsun10 hsun08a hsun08b" and everything works. Users need not change anything. Nice...

2.2 File Server Clusters

File Server Clusters (FSCs) are an important concept. Our lab, under the hip.atr.co.jp environment, is built around several FSCs – roughly one for each department. Each FSC is composed of one fileserver machine and many workstations with nearly the same machine architecture i.e. Sun-4, DEC, HP, etc. All machines in an FSC share the same programs and databases. The fileserver machine has many large disks which are exported to all workstations in the FSC and which store the software shared by all machines. The fileserver also maintains database information about passwords, groups, mounting rules, and printer capabilities and NIS slave servers. When the fileserver gets updated with new software or a database entry, then the change appears immediately to all attached workstations. Figure 1 shows a rough view of FSCs.

Currently three FSCs exist: Vision, Hearing and Cognition. The Vision and Hearing FSC have been operating smoothly for many months now; first with helios as a server in Vision but more recently with a new SparcServer 630MP as server in Vision and a Sun-4/370 in Hearing. The more recent Cognition FSC is served by a SparcServer 690MP.

The workstations attached to each fileserver are configured as "dataless clients." A dataless client NFS mounts the /usr and other partitions from the fileserver and it uses its local disk for only the root partition, swap space and home partition. This configuration as dataless clients has the following advantages:

- Because all clients use the same /usr and other partitions, the clients are *guaranteed*

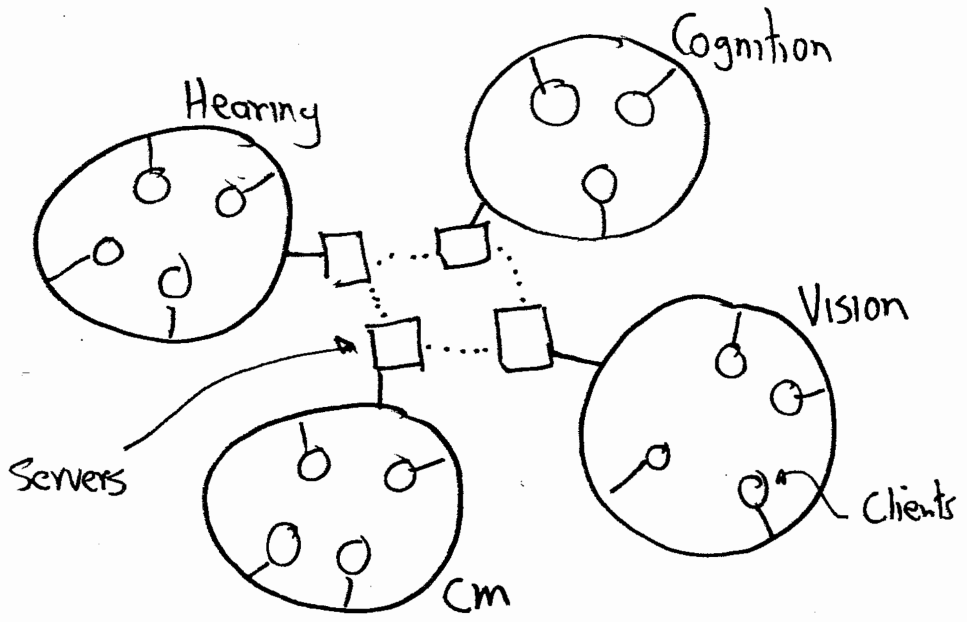

Figure 1: Overview of File Server Clusters.

to have an identical environment.

- Nearly all software is on one machine, the fileserver; therefore disk usage is minimized and filesystem backups are greatly simplified.
- Local usage of swap space and home directories ensures quick workstation response.
- Software grade-up occurs only on the fileserver but all clients can use the new version.
- File servers are designed for exactly this type of operation

Note that since each machine will have its own home partition, two issues arise: 1) every machine should be able to access another machines home directory and 2) the home directories must have unique names to be accessible by every other machine.

2.2.1 Server Disk Layout

The naming convention for the server disks is very important. A logically laid out and consistently named file system speeds a users understanding of her machine and improves her productivity.

Within the Unix filesystem, the naming conventions follows a repetitive pattern. Basically, at each directory hierarchy there is a similar structure; this consists of directories labeled: bin, lib, include, etc, src, man, share; as well as others. You can notice this structure by examining the "/" and "/usr" directories. Also, most software packages that can be purchased follow the same structure. Look at /usr/openwin or /usr/vision/khoros or /usr/vision/AnswerBook for examples. Note that the X11 software can also follow this structure. Because this structure is so common it is important to design the fileserver filesystem is the same way.

Such a repetitive structure has another important feature – all the software for one "system" is located in one location. This makes maintenance, backups and updates to software easier. If a new version of X11 arrives all the software gets installed in the "X11 directory;" if a new version of SunOS arrives, the "/" and "/usr" directories change *but the "X11" and "local" directory need not change.* This type of independence is an important concept for simplifying disk structure.

2.2.2 Example File Server Cluster

Here is the configuration of the Vision File Server Cluster. The fileserver for Vision is a Sparc 630MP called "hubble" which serves many clients including several Sparc 2 clients called helios, hoshi, halley and hesperus. All the clients mount their /usr, /usr/share, /usr/kvm and /usr/local partition from hubble. The partitions are mounted read-only on the clients. Hubble also maintains /export/vision, which is automounted as /usr/vision. This directory contains those programs that are frequently used by the vision group such as khoros, avs, video, and images. (Note, other groups can still use these programs from /usr/vision.). Here is part of the /etc/exports file from hubble.

```
/usr -access=hiphosts,root=hubble:hsun10:hermite
/usr/local -access=hiphosts,root=hubble:hsun10:hermite
/export -access=hiphosts,root=hubble:hsun10:hermite:hoshi
/export/source -access=hiphosts,root=hubble:hsun10:hermite:hoshi:helios
/export/mail -access=hiphosts,root=hubble:hsun10:hermite
/export/hubble -access=hiphosts,root=hubble:hsun10:hermite
```

The /usr, /usr/local and kmv partitions are really only needed only by vision machines (see NIS netgroup listing above) but since occasionally a non-vision machine uses hubble, the directories are exported to all hiphosts. Root access is granted to each of the file servers and to those machines of different architecture (hoshi Sparc-10, helios Sparc-2) needed to build kernels. One of the clients "helios" has home directories, here is it's /etc/exports

```
/export/helios -access=hiphosts
```

Note the everyone can mount the home directory and, if they have write permission on a directory, anyone can write to it. The directory cannot be mounted read-only because if I'm working on helios I must be able to write to my home directory on hoshi.

Certain programs on the clients cannot be shared by NIS; examples are /etc/printcap and /etc/shells. The clients are updated with a nightly job running on hubble, a server. This nightly job distributes files to the clients using the program rdist. Here are some files for the nightly job on the server hubble:

```
root@hubble(87)# crontab -l
    00 02 * * * /usr/local/etc/nightly.server >/var/adm/nightly.log 2>&1
    00 03 * * 6 /usr/lib/newsyslog >/dev/null 2>&1

root@hubble(88)# cat /usr/local/etc/nightly.server
    #!/bin/sh
    #
    # Based on trix.ai.mit.edu     Ed Gamble, Mon Apr 6 1992
    #
    # This is the nightly job running on Master Server HIP.
    #
    /usr/ucb/rdate atrwide

    # Clean file systems and update openwindows
    /usr/local/etc/nightly.client

    # reload the nameserver...
    echo "Reloading the Nameserver"
    /usr/local/etc/update-DNS-maps -ping

    # update hosts
    #(var/hosts; make)

    # update Yellow Pages
    echo "Updating Yellow Pages."
    (cd /var/yp; make)

    # update the AMD file system maps from the Sun maps
    #/usr/local/etc/make_amd

    # update manual database
    echo "Using catman on manual databases."
    /usr/etc/catman -M /usr/man
    /usr/etc/catman -M /usr/local/man
    /usr/etc/catman -M /usr/local/openwin/man

    #
    # Today's modified files
    #
    /usr/local/etc/find-files -days 1 | \
        /usr/ucb/mail -s "Daily File Report (Hubble)" file-report

    distfile="/usr/local/etc/distfile.vision"
    if [ -f "$distfile" ]; then
    # perform server rdist update
    echo "Running rdist on ${distfile}."
    /usr/ucb/rdist -f $distfile
    fi

    date

root@hubble(89)# cat /usr/local/etc/nightly.client
    #
    # This is the nightly job run on all servers by cron.
    #
    # Get time from the ATR HIP master server
    /usr/ucb/rdate hip

    echo "Nightly job running on `/bin/hostname`."
    date

    /usr/local/etc/update-binder

    /usr/local/etc/update-printcap

    # ...still not uniform...
    #/usr/local/etc/update-sys-files
    #

    # clean up some unneeded junk files
    echo "Cleaning filesystems."
    df
    /usr/local/etc/cleanfs
    df

    date
```

Notice how cleanly the clients can be maintained. These jobs runs nightly; in the morning all clients are identical to the server. The clients also run nightly jobs to clean their filesystems and update various files. Once setup like this, no systems maintenance is required; everything is automatic.

Here are some additional programs that help maintain uniformity. The first involves /etc/printcap; which is not easily shared with NIS and the second involves various system files which change occasionally.

```
root@hubble# cat /usr/local/etc/update-printcap
```

```
#!/usr/local/bin/bash

usage="usage: `basename $0` [ -install | -help | -debug ]";

install=no
debug=no

case $1 in
  -install) install=yes;;
  -debug) debug=yes;;
  -help) echo $usage; exit 0;;
  "") ;;
  *) echo $usage; exit 1;;
esac

ETC=/usr/local/etc
HipPrinters=${ETC}/printcap
MyPrinters=${ETC}/printers/printcap.`hostname`

#
# Check if we need installation.  Force installation with the -install
# flag, otherwise check file modification dates.
#
if [ $HipPrinters -nt /etc/printcap ]; then
   install="yes"
else
   if [ -f $MyPrinters -a $MyPrinters -nt /etc/printcap ]; then
      install="yes"
   fi
fi

#
# Exit if nothing need be done.
#
if [ "$debug" = "yes" ]; then
   install=no;
else
   if [ "$install" = "no" ]; then
      echo "No printers updated."
      exit 1;
   fi
fi
#
# Install flag can be ignored now.  Debug flag has the information.
#


#
# Store the database in /tmp/printcap
#
if [ -f $MyPrinters ]; then
        cat ${ETC}/printcap $MyPrinters > /tmp/printcap
else
```

```
        cp ${ETC}/printcap /tmp/printcap
fi


if [ "$debug" = "no" ]; then
        echo -n "Installing /etc/printcap on `hostname`...";
        cp /etc/printcap  /etc/printcap.`date '+%m%d%y'`.`date '+%H%M%S'`;
        mv /tmp/printcap /etc/printcap;
        echo " Done."
else
        cp /etc/printcap /tmp/printcap.`date '+%m%d%y'`.`date '+%H%M%S'`
        cat /tmp/printcap
fi

rm -f /tmp/printcap

root@hubble# cat /usr/local/etc/update-sys-files
    #! /usr/local/bin/bash
    #
    # Trying to automate update of various system-files.  Performed by the
    # client side.
    #
    host=`bin/hostname`
    server=`/usr/local/bin/servername`
    cluster=`/usr/local/bin/clustername`
    #
    # Don't do on a server...
    #
    if [ "$server" = "$host" ]; then
       echo "No system-file updates for server machines."
       exit 0;
    fi
    #
    # A function to copy a file only if it is newer.
    #
    update-if-newer () {
      source=$1;
      target=$2;
      if [ $source -nt $target ]; then
        echo "Updating from $source to $target."
        cp $source $target;
      fi

    }

    #
    # Get some files from our server.
    #
    etc=/usr/local/etc
    #
    # These can't be updated on a server
    update-if-newer ${etc}/sendmail.cf.forward /etc/sendmail.cf
    update-if-newer ${etc}/resolv.conf.${cluster} /etc/resolv.conf
    #
    # These can be updated on a server.
    update-if-newer ${etc}/shells /etc/shells
    update-if-newer ${etc}/magic /etc/magic
    update-if-newer ${etc}/syslog.conf /etc/syslog.conf

root@hubble# cat /usr/local/etc/update-DNS-files
    #! /bin/sh

    ping=no;
    if [ "$1" = "-ping" ]; then
            ping=yes;
    fi

    cd /usr/local/etc/named
    rdist -f named-distribute

    SERVERS="hubble hsun10 hermite"

    for host in $SERVERS; do
            echo -n "Updating $host: named.boot "
            rcp ${host}.boot ${host}:/etc/named.boot

            echo -n "${host}.zone "
            rcp ${host}.zone ${host}:/var/named

            echo -n "/etc/resolv.conf "
            rcp resolv.conf ${host}:/etc/resolv.conf

            if [ "$ping" = "yes" ]; then
                    echo " ... (SIGHUP)."
                    PID=`rsh $host cat /etc/named.pid`
                    rsh $host kill -HUP $PID
            fi

            echo "  Done."
    done
```

We have just described how to maintain a uniform environment *within* one FSC. However, to have a uniform environment throughout ATR, different FSC such as Vision, Hearing, and Cognition must also be uniform. In the next section we address this issue.

2.2.3 Multiple File Server Clusters

In order to ensure uniformity throughout ATR, the files on *every* FSC fileserver should be nearly identical to all other fileservers. Generally each fileserver keeps all the partitions used by the clients like /usr, /usr/share, /usr/local, /usr/local/share, and so on. If one fileserver has different programs in /usr/local/bin then uniformity is not maintained. Consequently, a solution is needed to "synchronize" the fileservers. There are two solutions to maintain uniformity among fileservers.

The first solution is as follows. Designate one machine as the *FSC Master FileServer.* This FSC Master can be the NIS (and DNS) master also; but that is not required. If a new program or new software arrives that is useful to all people, then those programs should be installed into /usr/local. The installation if performed by installing the programs on the FSC Master and then using "rdist" to distribute the files to all the FSC Slave FileServers. So, for example, hubble is currently the FSC Master FileServer and hsun10 and hermite are FSC Slave FileServers. The following command ensures that all FileServers are identical.

```
root@hubble# rdist -f -
/usr/local -> (hsun10 hermite)
  install -R ;
^D
```

Here is an example of updating a directory on hsun10 from hubble.

```
root@hubble# rdist -f -
/usr/local/etc -> hsun10
  install ;
updating host hsun10
/usr/local/etc: Warning: remote mode 755 != local mode 775
updating: /usr/local/etc/auto.direct
updating: /usr/local/etc/group
updating: /usr/local/etc/aliases
updating: /usr/local/etc/named/hoshi/named.boot
updating: /usr/local/etc/named/hip.zone
updating: /usr/local/etc/named/hip.zone.rev.40
updating: /usr/local/etc/named/hip.zone.rev.44
installing: /usr/local/etc/named/distfile
installing: /usr/local/etc/named/atr.zone
updating: /usr/local/etc/named/hubble/named.boot
installing: /usr/local/etc/named/atr.zone.rev
updating: /usr/local/etc/hosts
updating: /usr/local/etc/netgroup
updating: /usr/local/etc/passwd
updating: /usr/local/etc/ypservers
installing: /usr/local/etc/cron.hubble
installing: /usr/local/etc/checkaliases
```

The second solution is as follows. Every cluster has its own directory, for example /usr/vision, /usr/hearing, and /usr/cognition. These directories are maintained on one fileserver; they are not duplicated on other fileservers but are available to all with automount.

Each directory contains subdirectories called bin, lib, include, src, and man. A FSC can put anything in its own subdirectory without introducing a non-uniform environment. Adding software to /usr/vision or /usr/cognition is an easy technique for maintaining uniformity while also allowing each group to have their own programs.

The design with multiple, identical fileservers is robust to failure of the fileserver. Since the environments are uniform, if a fileserver breaks, all the clients can be temporarily transferred to another FSC.

2.2.4 FSCs and Ethernet Subnetworks

The use of ethernet subnetworks is a useful structure to minimize network traffic. Within each FSC it is possible that network traffic could get heavy because many workstations may access programs or data from the fileserver at the same time. Therefore is is advantageous to structure an FSC on its own subnetwork. This installation was performed recently; each of the primary FSC's has it's own subnet. Currently network traffic is not an issue but it is possible, though unlikely, that in the future it might become an issue.

2.3 Automount

It seems that certain people are afraid of automount. Care must be taken when using the Sun version of automount. That version is "single threaded" which means that if one machine accesses another machine that is down, then the accessing machine will be frozen until the other machine comes up. This can be avoided in several ways:

1. If the machine is down for only a few minutes then the user should just wait until the down machine comes up again.

2. If a machine is down for a long time, temporarily transfer its files to another machine and add an alias to the NIS hosts database.

3. Avoid using machines that require regular maintenance, like atr-hr. Such machines will always cause a problem with automount.

4. Minimize dependence on other machines. Each machine depends primarily on the file-server only. If machine in a different cluster fails, there is no effect except in the cluster with the failed machine.

Of course, if you do not like Sun's automount you could always use AMD which is "multithreaded" and will not die if one machine dies.

Also, note that AMD exists for many machines besides Suns, therefore, the Alliant and HP's and IBM's can use AMD too.

3 Mail and DNS in HIP

This section describes the distribution of mail and the use of Domain Name Service (DNS). Both of these features are currently implemented across all of the Human Information Processing laboratory.

3.1 The Issue of Mail

Because of the uniformity of our environment, a user's mail file in (/usr/spool/mail) must be available from *all* machines. If they choose, a user will not have to log into one machine to get mail; a user's mail directory is automatically delivered to whatever machine the user is using. This is different from the current case where mail can only be read from one machine, atr-hr. As we all know, hr goes down regularly for maintenance which prevents mail delivery during that time.

As part of the new system, we now have a new e-mail address. It is hip.atr.co.jp. The name "HIP" represents all people within our lab; not just one department. One machine, hip.atr.co.jp. has been designated as the mailhost which will receive mail from atrwide. This new mailhost distributes mail to people in hip.atr.co.jp or it will forward mail to people who want to use atr-hr to read mail. After the A&VP labs disappears, mail from atrwide bound for atr-hr.atr.co.jp will go to hip.atr.co.jp. Still people can use atr-hr or hip.atr.co.jp; whichever they choose.

The mail machine happens to be the vision cluster's server 'hubble;' though it need not be. All hip machines mount mailhost:/export/mail, under /usr/spool/mail. This ensures uniformity. Also, hubble is a reliable machine that does not require regular maintainance. Hubble works all the time, every day. When a hip machines mounts its /usr/spool/mail directory, it's sendmail.cf file can be simplified; here it is

```
ebg@hoshi(598)$ head -20 /etc/sendmail.cf

    ############################################################
    #
    #      CLIENT SENDMAIL CONFIGURATION FILE
    #
    #   This sendmail does nothing other than forward all mail
    #   to the forwarding agent.  Everything is accepted and
    #   then forwarded.
    #
    ############################################################

    # official hostname
    Dj$w

    # forwarding agent
    DFmailhost

    ... other stuff to actually do the forwarding.
```

The mailhost uses the file /usr/lib/sendmail.main.cf as /etc/sendmail.cf and sends mail with DNS using sendmail.mx. This type of forwarding is what sendmail was designed for; it is a well tested technique.

3.2 Domain Name Service (DNS)

As a consequence of implementing DNS in hip.atr.co.jp, all of ATR has now adopted DNS. So for example, we now have a 'proper' atr.co.jp, atr-sw.atr.co.jp, atr-la.atr.co.jp and of course hip.atr.co.jp. Before the implementation of HIP, when a new machine arrived all other machines needed a modified copy of /etc/hosts. This included our gateway atrwide. Of course machines arrived so often that other machines could not always modify /etc/hosts. The HIP domain has solved this problem with the use of NIS. Now the NIS master is updated and all hiphosts will recognize the new machine.

NIS cannot be shared between the different labs at atr, like atr-la, atr-rd, atr-sw and atrwide. DNS provides a way to allow machines in other labs to easily learn about our new machines. But, more importantly, we can learn about other machines outside of ATR.

DNS is currently running on hip.atr.co.jp. Every fileserver is a DNS slave (and an NIS slave) and one machine is the DNS master (and NIS master also). Presently in the Vision FSC in.named is stared on a machine called "domainserver" with the details to follow:

```
root@hip# cat /etc/named.boot
;;
;; Boot File for name server -- hip domain
;;
;; Ed Gamble, Tue May 26 1992
;;
;; type domain source file or host
;;
directory /var/named
cache . ./root.zone
primary hip.atr.co.jp ./hip.zone
primary 40.186.133.in-addr.arpa ./hip.zone.rev.40
primary 41.186.133.in-addr.arpa ./hip.zone.rev.41
primary 42.186.133.in-addr.arpa ./hip.zone.rev.42
primary 43.186.133.in-addr.arpa ./hip.zone.rev.43
primary 44.186.133.in-addr.arpa ./hip.zone.rev.44
primary 45.186.133.in-addr.arpa ./hip.zone.rev.45
primary 0.0.127.in-addr.arpa ./hubble.zone

secondary atr.co.jp 133.186.1.10 ./atr.zone
secondary 186.133.in-addr.arpa 133.186.1.10 ./atr.zone.rev

secondary atr-sw.atr.co.jp 133.186.1.20 ./cs.zone
secondary 20.186.133.in-addr.arpa 133.186.1.20 ./cs.zone.rev
secondary 22.186.133.in-addr.arpa 133.186.20.117 ./cs.cmac.rev
secondary 29.186.133.in-addr.arpa 133.186.20.117 ./cs.take.rev
```

Here is /etc/resolv.conf which tells each machine how to find a another host's address. The stratagy is first look in /etc/hosts, then contact NIS, then finally ask the DNS servers.

```
root@hip# cat /etc/resolv.conf
domain hip.atr.co.jp
nameserver 127.0.0.1
nameserver 133.186.42.1
nameserver 133.186.41.1
nameserver 133.186.43.1
```

Also here is abbreviated version /var/named/hip.zone which defines the machines and their properties for DNS:

```
root@hip# cat /etc/named.boot
;;
;; hip.zone                 Ed Gamble
;;
@               IN      SOA     hubble.hip.atr.co.jp.
```

```
                                  bug-hip-domain.hubble.hip.atr.co.jp. (
                               1        ; Serial
                               3600     ; Refresh 1 hour
                               300      ; Retry   5 mins
                               3600000  ; Expire  1000 hours
                               3600     ; Minimum 1 hour
                              )
                IN      NS      hubble.hip.atr.co.jp.
                IN      NS      hsun10.hip.atr.co.jp.
                IN      NS      hermite.hip.atr.co.jp.
                IN      NS      atrwide.atr.co.jp.

;
; Database Masters
;  Temporarily pointed to hubble.  (see above note)
;
loghost         IN      CNAME   hubble.hip.atr.co.jp.
mailer          IN      CNAME   hubble.hip.atr.co.jp.
ypmaster        IN      CNAME   hubble.hip.atr.co.jp.
domainserver    IN      CNAME   hubble.hip.atr.co.jp.
hip             IN      CNAME   hubble.hip.atr.co.jp.

;;;
;;; HOST MACHINES (44.atr.co.jp)
;;;
hip.atr.co.jp.  IN      A       133.186.44.30
                IN      MX      10      hubble.hip.atr.co.jp.
hip.atr.co.jp.hip.atr.co.jp.    IN      CNAME   hubble.hip.atr.co.jp.

;
; Vision
;
hubble          IN      HINFO   "Sun-4/630MP"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.30
                IN      MX      10 hubble.hip.atr.co.jp.
                IN      WKS     133.186.44.30   UDP tftp
                IN      WKS     133.186.44.30   TCP ( ftp telnet smtp finger
                                                supdup hostnames domain )
helios          IN      HINFO   "Sun-4/75 GX"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.31
                IN      MX      10 hubble.hip.atr.co.jp.
hoshi           IN      HINFO   "Sun-4/65"      "SunOS 4.1.2 JLE"
                IN      A       133.186.44.32
                IN      MX      10 hubble.hip.atr.co.jp.
hesperus        IN      HINFO   "Sun-4/75 GX"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.33
                IN      MX      10 hubble.hip.atr.co.jp.
halley          IN      HINFO   "Sun-4/75 GX"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.34
                IN      MX      10 hubble.hip.atr.co.jp.
hole            IN      HINFO   "CSPI RTS-860"  "VxWorks 5.0.2"
                IN      A       133.186.44.80
                IN      MX      10 hubble.hip.atr.co.jp.
```

```
black-hole      IN      CNAME   hole.hip.atr.co.jp.

hip.hip.atr.co.jp               IN      CNAME   hubble.hip.atr.co.jp.
hubble.hip.atr.co.jp            IN      CNAME   hubble.hip.atr.co.jp.
helios.hip.atr.co.jp            IN      CNAME   helios.hip.atr.co.jp.
hoshi.hip.atr.co.jp             IN      CNAME   hoshi.hip.atr.co.jp.
hesperus.hip.atr.co.jp          IN      CNAME   hesperus.hip.atr.co.jp.
halley.hip.atr.co.jp            IN      CNAME   halley.hip.atr.co.jp.
hole.hip.atr.co.jp              IN      CNAME   hole.hip.atr.co.jp.


;
; Cognition
;
hermite         IN      HINFO   "Sun-4/690MP"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.61
                IN      MX      10 hubble.hip.atr.co.jp.
                IN      WKS     133.186.44.61   UDP tftp
                IN      WKS     133.186.44.61   TCP ( ftp telnet smtp finger
                                                supdup hostnames domain )
hamilton        IN      HINFO   "Sun-4/75 GX"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.43
                IN      MX      10 hubble.hip.atr.co.jp.
higgs           IN      HINFO   "Sun-4/75 GX"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.44
                IN      MX      10 hubble.hip.atr.co.jp.
hartree         IN      HINFO   "Sun-4/75 GX"   "SunOS 4.1.2 JLE"
                IN      A       133.186.44.45
                IN      MX      10 hubble.hip.atr.co.jp.
hermite.hip.atr.co.jp           IN      CNAME   hermite.hip.atr.co.jp.
hamilton.hip.atr.co.jp          IN      CNAME   hamilton.hip.atr.co.jp.
higgs.hip.atr.co.jp             IN      CNAME   higgs.hip.atr.co.jp.
hartree.hip.atr.co.jp           IN      CNAME   hartree.hip.atr.co.jp.
;
; CM
... not shown ...

;;;
;;; HOST MACHINES (40.ATR.CO.JP)
;;;
;
; Hearing
... not shown ...
```

4 Installation on HIP

There are two primary installations that need to be performed on HIP; users and machines. The process of installation has been automated with three programs. "install-machine" is used following suninstall to provide the initial configuration of hip. This programs modifies the 'dot-files' for root, primarily .rhosts, sets the /etc/netgroup, /etc/defaultserver, /etc/sendmail.cf and /etc/resolv.conf files, changes /etc/exports to properly export all the machine's disks, updates /etc/fstab, and finally installs /etc/xdm for the X11 login. The program is shown below

```
root@hip# cat /usr/local/etc/install-machine
    #! /bin/sh
    #
    # Written by Ed Gamble, Thu Jul 23 1992
    #
    #  This is part 1: post suninstall but before rebooting for HIP.
    #  And part 2: after rebooting.
    #
    # Check for root user
    #
    if [ "`whoami`" != root ]; then
            echo ""
            echo "You must be root to run install-machine"
            echo ""
            exit 1
    fi

    progname=$0
    usage () {
            echo "usage: $progname [ -phase2 ] -name hostname -cluster name \
                     -server server"
            echo "        [ -home diskname ] "
            echo "            [ -spare1 diskname ]"
            echo "            [ -spare2 diskname ]"
            echo "            [ -spare3 diskname ]"
            echo "        [ -help ]"
            echo ""
            echo "Run this immediately after suninstall and before rebooting."
            echo " -spare? refers to extra disks used for databases."
            echo ""
            echo "After rebooting run:"
            echo "/usr/local/etc/install-machine -phase2 -name hostname \
                    -cluster name"
    }

    # command line arguments to fill
    hostname=""
    cluster=""
    server=""
    home=""
    spare1=""
    spare2=""
    spare3=""
    phase2=no

    # parse command line arguments
```

```
foundarg="yes"
while [ "$foundarg" = "yes" ]
do
    case "$1" in
        "-name") shift; hostname=$1; shift;;
        "-cluster") shift; cluster=$1; shift;;
        "-server") shift; server=$1; shift;;
        "-home") shift; home=$1; shift;;
        "-spare1") shift; spare1=$1; shift;;
        "-spare2") shift; spare2=$1; shift;;
        "-spare3") shift; spare3=$1; shift;;
        -phase2) phase2=yes; shift;;
        -help)
            usage; exit 0;;
        "") foundarg="no";;
        *) echo "Parse error: Unknown option \"$1\".  Check argument list."
           usage; exit 1;
            ;;
    esac
done

if [ "$hostname" = "" -o "$cluster" = "" ]; then
        usage;
        echo Hostname or Cluster missing.
        exit 1;
fi

case $cluster in
        vision | hearing | cognition )      ;;
        * ) echo "Unknown cluster named \"${cluster}\"... failing."
            exit 1
            ;;
esac

case $server in
        hubble | hsun10 | hermite ) ;;
        *)  echo "Unknown server named \"${server}\"... failing."
            exit 1
            ;;
esac

while :
do
        echo "Ready to install:"
        echo "  Hostname:    $hostname"
        echo "  Cluster:     $cluster"
        echo "  Server:      $server"
        echo "  Home:        $home"
        echo "  Spare1:      $spare1"
        echo "  Spare2:      $spare2"
        echo "  Spare3:      $spare3"
        echo -n 'Is this correct? [y|n] '
        read response
        case "$response" in
        y*) break;;
        n*) echo ""
                echo 'Please rerun this script with the proper arguments.'
                exit 1;;
        *)  echo ""
                echo 'Please answer "y" or "n"';;
        esac
done

if [ "$phase2" = "yes" ]; then
        echo "Installing Openwindows."
        /usr/local/openwin/bin/install_openwin
        echo "Modifying crontab"
        crontab /usr/local/etc/cron.client
        crontab -l
        echo "Starting the nightly.client now."
        /usr/local/etc/nightly.client
        echo "Phase 2 done."
        exit 1;
fi

if [ ! -x "/mnt/etc/install-machine" ]; then
        echo "The filesystem /usr/local is mounted in an unexpected place. \
                    Please unmount /usr/local"
        echo " and remount with 'mount ${server}:/usr/local /mnt'.  \
                    After that try"
        echo " /mnt/etc/install-machine [ args ] again."
        echo ""
        usage
        exit 1
fi


if [ ! -d /a ]; then
        echo "No /a directory.  Usually, when suninstall finishes, \
                    /dev/sd0a has been mounted"
        echo "under /a.  If you booted this machine after suninstall \
                    and before"
        echo "install-machine, no /a directory exists."
        echo ""
        echo "Do you wish to link \"/\" to \"/a\"?"
        while :
        do
                read response
                case "$response" in
                y*) echo "Linking / to /a"
                        cd /; ln -s . /a
                        linked=yes
                        break
                        ;;
                n*) echo ""
                        echo 'Rerun this script with the proper arguments.'
                        exit 1;;
                *)  echo ""
                        echo 'Please answer "y" or "n"';;
                esac
        done
fi

cd /a

echo "Copying .../user files into /"
cp /mnt/lib/user/.* .
cp -r /mnt/lib/user/Sj3 .
cp -r /mnt/lib/user/Mail .

echo "Defaulting .xinitrc, .cshrc, .bashrc, and .mh_profile for $cluster."

if [ -f .xinitrc.${cluster}.csh ]; then
        mv .xinitrc.${cluster}.csh .xinitrc
        chmod +x .xinitrc
        rm .xinitrc.*
fi

if [ -f .bashrc.${cluster} ]; then
        mv .bashrc.${cluster} .bashrc
        rm .bashrc.*
fi

if [ -f .cshrc.${cluster} ]; then
        mv .cshrc.${cluster} .cshrc
        rm .cshrc.*
fi

if [ -f .mh_profile.hip ]; then
        echo "Installing .mh_profile with user's name."
        sed -e "s/ebg/root/" .mh_profile.hip > .mh_profile
        rm .mh_profile.*
fi

echo "Adjusting .rhosts, etc/netgroup, and etc/defaultserver"
cat >.rhosts <<ADD_THIS
+@servers
ADD_THIS

cat >etc/netgroup <<ADD_THIS
+
ADD_THIS

cat >etc/defaultserver <<ADD_THIS
${server}
ADD_THIS

echo "Adjusting etc/sendmail.cf, and etc/resolv.conf"
cp /mnt/etc/sendmail.cf.forward etc/sendmail.cf
cp /mnt/etc/resolv.conf.${cluster} etc/resolv.conf

echo "Making /etc/xdm directory."
mkdir etc/xdm


echo "Editing etc/exports."
if [ "$home" != "" ]; then
   cat >>etc/exports <<ADD_THIS
/export/$home                  -access=hiphosts,root=hubble:hsun10:hermite
ADD_THIS
fi

if [ "$spare1" != "" ]; then
   cat >>etc/exports <<ADD_THIS
/export/$spare1                -access=hiphosts,root=hubble:hsun10:hermite
ADD_THIS
fi

if [ "$spare2" != "" ]; then
   cat >>etc/exports <<ADD_THIS
/export/$spare2                -access=hiphosts,root=hubble:hsun10:hermite
ADD_THIS
fi

if [ "$spare3" != "" ]; then
   cat >>etc/exports <<ADD_THIS
/export/$spare3                -access=hiphosts,root=hubble:hsun10:hermite
ADD_THIS
fi

echo "Editing etc/fstab.  Backup copy in etc/fstab.suninstall"
cp etc/fstab etc/fstab.suninstall
sed -e "/home/d" etc/fstab > /tmp/fstab
cat >>/tmp/fstab <<ADD_THIS
${server}:/usr/local                 /usr/local      nfs ro 0 0
mailhost:/export/mail                /var/spool/mail nfs rw 0 0
ADD_THIS
cp /tmp/fstab etc/fstab

echo "Editing etc/rc."
sed -e "/^exit 0/d" etc/rc >/tmp/rc
cat >>/tmp/rc <<ADD_THIS
if [ -f /usr/local/X11/bin/xdm ]; then
        /usr/local/X11/bin/xdm;
fi

exit 0
ADD_THIS
mv /tmp/rc etc/rc

if [ "$linked" = "yes" ]; then
        rm /a;
```

```
    fi

    echo ""
    echo "Installation nearly complete.  Please perform the following:"
    echo "  1) Comment out 'cesd' in /usr/rc.local."
    echo "  2) Reboot"
    echo "  3) As superuser type \"/usr/local/etc/install-machine -phase-2\""
    echo "  4) You might want to add the machine to \
                  /usr/local/lib/Tvtwmrc and to your"
    echo "     server's /usr/local/etc/distfile"
    echo ""
    echo "Done."
```

Prior to installing a machine, the NIS and DNS databases must be updated with the new machine's address and characteristics. This allows all other machines to learn about the new machine and for the new machine to mount its filesystems from the cluster server. The following program performs: 1) check the the new machine has a unique name, aliases, ip address and netgroup, 2) update NIS hosts and netgroup, 3) update mail-to-hip to inform mailhost to accept mail for the new machine, 4) update fingerdir/clients, 5) update dserver for the online Japanese dictionaries, 6) update the DNS files named/hip.zone, named/hip.zone.rev.??, 7) install the changed NIS and DNS data, 8) warn about updating hubble's sendmail, tvtwmrc and openwin-menu.m4. Here is this program.

```
root@hip# cat /usr/local/etc/add-host-to-hip
    #! /bin/sh
    #
    # Written by Ed Gamble, Fri Jun 12 1992
    #  Should check for internal consistency of databases before updating.
    #  Particularly among mail-to-hip, dserver_access, netgroup, hosts,
    #  and hip.zone.
    #
    #
    # Check for root user
    #
    YPMASTER=hubble
    if [ "`whoami`" != root ]
    then
            echo ""
            echo "You must be root on $YPMASTER to run add-host-to-hip"
            echo ""
            exit 1
    fi
    if [ "`/bin/hostname`" != $YPMASTER ]
    then
            echo ""
            echo "You must be root on $YPMASTER to run add-host-to-hip"
```

```
        echo ""
        exit 1
fi


progname=`basename $0`
usage () {
        echo "usage: $progname -name hostname -address number"
        echo "              [-subnet 40]"
        echo "              [-cluster hearing]"
        echo "              [-type \"Sun 4/75 GX\"]"
        echo "              [-os \"SunOS 4.1.2 JLE\"]"
        echo "              [-owner \"(hip/fujita)\"]"
        echo "              [-aliases \"other-names...\"]"
        echo "              [-debug /tmp/new-host ]"
        echo "              [-help]"
}

# command line arguments to fill
hostname=""
address=""
subnet="40"
cluster="hearing"
type="Sun 4/75 GX"
os="SunOS 4.1.2 JLE"
owner="(hip/fujita)"
aliases=""
debug=""

# parse command line arguments
foundarg="yes"
while [ "$foundarg" = "yes" ]
do
    case "$1" in
        "-name") shift; hostname=$1; shift;;
        "-address") shift; address=$1; shift;;
        "-type") shift; type=$1; shift;;
        "-cluster") shift; cluster=$1; shift;;
        "-os") shift; os=$1; shift;;
        "-owner") shift; owner=$1; shift;;
        "-subnet") shift; subnet=$1; shift;;
        "-aliases") shift; aliases=$1; shift;;
        "-debug") shift; debug=$1; shift;;
        -help)
            usage; exit 0;;
        "") foundarg="no";;
        *) echo "Parse error: Unknown option \"$1\".  Check argument list."
           usage; exit 1;
            ;;
    esac
done

if [ "$hostname" = "" -o "$address" = "" ]; then
```

```
        usage;
        echo Hostname or Address missing.
        exit 1;
fi

#case $subnet in )
#   40 | 41 | 42 | 43 | 44 | 45 ) ;;
#   *) echo "Unknown subnet number ${subnet}.  Use only [40, 45]."
#      exit 1;
#      ;;
#esac

ip_address="133.186.${subnet}.${address}"

while :
do
        echo "Ready to configure:"
        echo "  Hostname:     $hostname"
        echo "  IP Address:   $ip_address"
        echo "  Cluster:      $cluster"
        echo "  Type:         $type"
        echo "  OS:           $os"
        echo "  Owner:        $owner"
        echo "  Aliases:      $aliases"
        echo "  Debug:        $debug"
        echo -n 'Is this correct? [y|n] '
        read response
        case "$response" in
        y*) break;;
        n*) echo ""
                echo 'Please rerun this script with the proper arguments.'
                exit 1;;
        *)  echo ""
                echo 'Please answer "y" or "n"';;
        esac
done

ETC=/usr/local/etc

#
# Check for duplicates.
#
# Should filter comment lines
echo Searching ${ETC}/hosts for ip address: ${ip_address}.
existp=`sed -e "/^#/d" -e s/#.*$$// ${ETC}/hosts | \
                    grep "${ip_address}[^0-9]"`
if [ "$existp" != "" ]; then
  echo "IP Address, ${ip_address}, already exists in the hip domain."
  echo "Not added."
  exit 1;
fi

echo Searching ${ETC}/hosts for hosts named: ${hostname} ${aliases}.
```

```
for name in ${hostname} ${aliases}; do
  existp=`grep "${name}[^a-z0-9A-Z]" ${ETC}/hosts`
  if [ "$existp" != "" ]; then
    echo "A host named \"${name}\" already exists in the hip domain."
    echo "Not added."
    exit 1;
  fi
done

echo Searching for subnet number ${subnet}...
if [ ! -f "${ETC}/named/hip.zone.rev.${subnet}" ]; then
    echo "Subnet number ${subnet} does not exist."
    echo "Not added."
    exit 1;
fi

#
# hosts
#
echo "Updating ${ETC}/hosts..."
if [ "$debug" = "" ]; then
        cp ${ETC}/hosts \
           ${ETC}/hosts.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/hosts
else
        cp ${ETC}/hosts ${debug}.hosts
        target=${debug}.hosts
fi
echo "$ip_address   $hostname $aliases                           # $type $owner" \
        >>$target

#
# netgroup
#
echo "Updating ${ETC}/netgroup..."
if [ "$debug" = "" ]; then
        cp ${ETC}/netgroup \
           ${ETC}/netgroup.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/netgroup
else
        cp ${ETC}/netgroup ${debug}.netgroup
        target=${debug}.netgroup
fi

group_spec=""
for name in ${hostname} ${aliases}; do
  group_spec="(${name},,) ${group_spec}"
done
cp ${ETC}/netgroup /tmp/netgroup

#Don't add to hiphosts. By Fujita 12/08/92.
#  -e "s/hiphosts[ \t]*/hiphosts    ${group_spec}/" \
sed -e "s/${cluster}[ \t]*/${cluster}       ${group_spec}/" /tmp/netgroup \
            >$target
rm /tmp/netgroup

#
# mail-to-hip
#
echo "Updating ${ETC}/mail-to-hip..."
if [ "$debug" = "" ]; then
        cp ${ETC}/mail-to-hip \
           ${ETC}/mail-to-hip.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/mail-to-hip
else
        cp ${ETC}/mail-to-hip ${debug}.mail-to-hip
        target=${debug}.mail-to-hip
fi
for name in ${hostname} ${aliases}; do
  echo "$name" >> $target
done

#
# fingerdir/clients
#
echo "Updating ${ETC}/fingerdir/clients..."
if [ "$debug" = "" ]; then
        cp ${ETC}/fingerdir/clients \
           ${ETC}/fingerdir/clients.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/fingerdir/clients
else
        cp ${ETC}/fingerdir/clients ${debug}.fingerdir.clients
        target=${debug}.fingerdir.clients
fi
for name in ${hostname} ${aliases}; do
  echo "$name" >> $target
done

#
# dserver_access
#
echo "Updating ${ETC}/dserver_access..."
if [ "$debug" = "" ]; then
        cp ${ETC}/dserver_access \
           ${ETC}/dserver_access.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/dserver_access
else
        cp ${ETC}/dserver_access ${debug}.dserver_access
        target=${debug}.dserver_access
fi
for name in ${hostname} ${aliases}; do
  echo "$name" >> $target
done

#
# hip.zone
```

```
#
echo "Updating ${ETC}/named/hip.zone..."
if [ "$debug" = "" ]; then
        cp ${ETC}/named/hip.zone \
           ${ETC}/named/hip.zone.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/named/hip.zone
else
        cp ${ETC}/named/hip.zone ${debug}.zone
        target=${debug}.zone
fi
cat >> $target <<ADD_THIS
$hostname                     IN      HINFO   "$type" "$os"
                        IN  A         $ip_address
                        IN  MX        10 mailhost.hip.atr.co.jp.
${hostname}.hip.atr.co.jp     IN      CNAME   ${hostname}.hip.atr.co.jp.
ADD_THIS

if [ "$aliases" != "" ]; then
  for alias in $aliases; do
        cat >> $target <<ADD_THIS
$alias              IN      CNAME   ${hostname}.hip.atr.co.jp.
ADD_THIS
  done
fi

#
# hip.zone.rev
#
echo "Updating ${ETC}/named/hip.zone.rev.${subnet}"
if [ "$debug" = "" ]; then
   cp ${ETC}/named/hip.zone.rev.${subnet} \
      ${ETC}/named/hip.zone.rev.${subnet}.`date '+%m%d%y'`.`date '+%H%M%S'`
   target=${ETC}/named/hip.zone.rev.${subnet}
else
        cp ${ETC}/named/hip.zone.rev.${subnet} \
                ${debug}.zone.rev.${subnet}
        target=${debug}.zone.rev.${subnet}
fi
cat >> $target <<ADD_THIS
${address}                    PTR     ${hostname}.hip.atr.co.jp.
ADD_THIS

#
# Install Databases
#
if [ "$debug" != "" ]; then
        echo ""
        echo "Debugging output in files:"
        echo "  ${debug}.hosts"
        echo "  ${debug}.zone"
        echo "  ${debug}.zone.rev.${subnet}"
        echo "  ${debug}.netgroup"
        echo "  ${debug}.mail-to-hip"
        echo "  ${debug}.fingerdir/clients"
        echo "  ${debug}.dserver_access"
        echo "No updating performed."
else
        echo "Updating NIS, DNS and Sendmail"
        echo " First NIS.  This will take a while."
        (cd /var/yp; make hosts netgroup)
        echo " Next DNS"
        (cd ${ETC}; ./update-DNS-maps -ping)
        echo " Next Sendmail"
        echo "  Not done. Try kill -HUP sendmail-pid on hubble"
        echo "Need to update /usr/local/lib/Tvtwmrc and openwin-menu.m4"
        echo "Finished!"
fi
```

Finally to install users a program exists. The steps are as follows: 1) update NIS passwd and group after checking that the new user has unidue ids, 2) check the user's cluster (vision, hearing, cognition), 3) create a home directory in the desired machine, 4) set up mail delivery, 5) copy .cshrc, .bashrc, .xinitrc, .emacs etc into the users home directory. Here is the program

```
root@hip# cat /usr/local/etc/add-user-to-hip
    #!/bin/sh
    #
    # Ed Gamble, Fri Jul 24 1992
    #
    #
    YPMASTER=hubble
    if [ "`whoami`" != root ]; then
      echo ""
      echo "You must be root on $YPMASTER to run add-user-to-hip"
      echo ""
      exit 1
    fi

    if [ "`/bin/hostname`" != $YPMASTER ]; then
      echo ""
      echo "You must be root on $YPMASTER to run add-user-to-hip"
      echo ""
      exit 1
    fi

    progname=`basename $0`
    usage () {
      echo "usage: $progname login-name user-id group-id \"full-name\" cluster"
      echo "                 home-partition mail shell [ -debug ] [ -help ] "
      echo ""
      echo "  Full-name it the user's actual name, use double quotes."
      echo "  Home-partition is the location of the user's home directory;"
      echo "  it should exist in the yp hosts table."
```

```
  echo ""
  echo "  Mail should be [ hip | hr ]; user gets mail on that machine"
  echo "  Shell should be [ bash | sh | csh | tcsh | newcsh ]"
  echo ""
}

if [ $# -lt 8 ]; then
  usage;
  exit 1;
fi

# command line arguments to fill
name=$1;    shift
id=$1;              shift
group=$1;   shift
full_name=$1;       shift
cluster=$1; shift
home=$1;    shift
mail=$1;    shift
shell=$1;   shift
debug=no;

foundarg="yes"
while [ "$foundarg" = "yes" ]
do
    case "$1" in
        "-debug") debug=yes; shift;;
        "-help") usage; exit 0;;
        "") foundarg="no";;
        *) echo "Parse error: Unknown option \"$1\".  Check argument list."
           usage; exit 1;;
    esac
done


while :
do
        echo "Ready to configure:"
        echo "  Name:         $name"
        echo "  User-ID:      $id"
        echo "  Group-ID:     $group"
        echo "  Full Name:    \"${full_name}\""
        echo "  Cluster:      $cluster"
        echo "  Home:         $home"
        echo "  Mail:         $mail"
        echo "  Shell:        $shell"
        echo ""
        echo "  Debug:        $debug"
        echo -n 'Is this correct? [y|n] '
        read response
        case "$response" in
        y*) break;;
        n*) echo ""
```

```
                echo 'Please rerun this script with the proper arguments.'
                exit 1;;
        *)  echo ""
                echo 'Please answer "y" or "n"';;
        esac
done

ETC=/usr/local/etc

# Check for a duplicate name
echo "Searching ${ETC}/passwd for user $name"
existp=`grep "^${name}:" ${ETC}/passwd`
if [ "$existp" != "" ]; then
  echo "User-name, ${name}, already exists in the hip domain."
  echo "Not added."
  exit 1;
fi

# Check for a duplicate user id
echo "Searching ${ETC}/passwd for user-id $id"
existp=`awk -F: '$3 == '$id' {print $1}' ${ETC}/passwd`
if [ "$existp" != "" ]; then
  echo "User-id, ${id}, already exists in the hip domain as user ${existp}."
  echo "Not added."
  exit 1;
fi

# Check that group-id exists
echo "Searching ${ETC}/group for group-id $group"
existp=`grep $group ${ETC}/group`
if [ "$existp" = "" ]; then
  echo "Group-id, ${group}, does not exist in the hip domain."
  echo "Not added."
  exit 1;
fi

# Check cluster
echo "Searching for cluster $cluster"
case $cluster in
  vision | hearing | cognition ) ;;
  *) echo "Cluster, ${cluster}, is not in [ vision | hearing | cognitionn ]"
     echo "Not added."
     exit 1;
     ;;
esac

# Check cluster
echo "Searching for shell $shell"
case $shell in
  sh)   shell=/bin/sh; sh_type=sh;;
  bash) shell=/usr/local/bin/bash; sh_type=sh;;
  csh)  shell=/bin/csh; sh_type=csh;;
  tcsh) shell=/usr/local/bin/tcsh; sh_type=csh;;
  newcsh) shell=/usr/local/bin/newcsh; sh_type=csh;;
  * ) echo "Shell, ${shell}, unknown; try \
            [ bash | sh | csh | tcsh | newcsh ]"
      echo "Not added."
      exit 1;
      ;;
esac

# Check the home location
echo "Searching for a home called /home/${home}"
existp=`ypmatch $home hosts`
if [ "$existp" = "" ]; then
  echo "Home directory /home/${home} is unknown.  A machine called"
  echo "\"${home}\" should be in NIS hosts table."
  echo "Not added."
  exit 1;
fi

if [ "$mail" != "hip" -a "$mail" != "hr" ]; then
  echo "Mail can only be delivered to hip or hr.  The mail address,"
  echo "${mail}, is unknown."
  echo "Not added."
  exit 1;
fi


#
#
# Create a new user ID and add to the password file
#
#
echo ""
echo "Adding user to  Password file...."
#
if [ "$debug" != "yes" ]; then
        cp ${ETC}/passwd \
           ${ETC}/passwd.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/passwd
else
        cp ${ETC}/passwd /tmp/passwd
        target=/tmp/passwd
        echo "  Debugging... check /tmp/passwd"
fi

echo "Adding the following entry to ${ETC}/passwd."
echo "  ${name}::${id}:${group}:${full_name}:/home/${home}/${name}:${shell}"
echo "${name}::${id}:${group}:${full_name}:/home/${home}/${name}:${shell}" \
            >> $target
if [ "$debug" != "yes" ]; then
        echo "Updating NIS passwd.  This may take a while..."
        (cd /var/yp; make passwd)
fi
echo ""
```

```
#
# Run yppassword to set the password
#
if [ "$debug" != "yes" ]; then
        echo "Setting Password. For old password, just hit return."
        yppasswd ${name}
fi

echo "Adding ${name} to ${ETC}/group"
if [ "$debug" != "yes" ]; then
        cp ${ETC}/group \
           ${ETC}/group.`date '+%m%d%y'`.`date '+%H%M%S'`
        target=${ETC}/group
else
        cp ${ETC}/group /tmp/group
        target=/tmp/group
        echo "  Debugging... check /tmp/group"
fi
cp ${target} ${target}.foobar
sed -e "s/${group}:/${group}:${name},/" ${target}.foobar > ${target}
rm ${target}.foobar
if [ "$debug" != "yes" ]; then
        echo "Updating NIS group."
        (cd /var/yp; make group)
fi
echo ""

#
# Add the user to the appropriate mailing list.
#
echo "Setting mail delivery"
if [ "$debug" != "yes" ]; then
  cp ${ETC}/aliases \
     ${ETC}/aliases.`date '+%m%d%y'`.`date '+%H%M%S'`
  target=${ETC}/aliases
else
  cp ${ETC}/aliases /tmp/aliases
  target=/tmp/aliases
  echo "  Debugging... check /tmp/aliases"
fi

if [ "$mail" != "hip" ]; then
  echo "${name}:            ${name}@atr-hr.atr.co.jp" >> $target
else
  cp $target ${target}.foobar
  sed -e "s/hipsters:[ \t]*/hipsters:       ${name},/" ${target}.foobar \
          >$target
  rm ${target}.foobar
fi

if [ "$debug" != "yes" ]; then
  echo "Updating NIS aliases."
  (cd /var/yp; make aliases;)
fi

#
# Make the Home directory on the the home server
#
dir=/home/${home}/${name}
echo ""
echo "Making home directory /home/${home}/${name} on machine $home."
if [ -d $dir ]; then
        echo "Directory, ${dir}, already exists; \
                    moving it to ${dir}-pre-hip."
        rsh ${home} "mv $dir ${dir}-pre-hip"
fi
rcp -r /usr/local/lib/user ${home}:${dir}
rsh $home "/bin/chmod 777 $dir"

echo "Linking ~/Sj3 with /usr/local/X11/lib/sj3/dict/user/${name} on hubble"
rsh hubble "ln -s ${dir}/Sj3/dict /usr/local/X11/lib/sj3/dict/user/${name}"

echo "Installing .xinitrc and shell files."

rsh $home "mv ${dir}/.xinitrc.${cluster}.${sh_type} ${dir}/.xinitrc"
rsh $home "rm ${dir}/.xinitrc.*"

rsh $home "mv ${dir}/.cshrc.${cluster} ${dir}/.cshrc"
rsh $home "mv ${dir}/.bashrc.${cluster} ${dir}/.bashrc"
rsh $home "rm ${dir}/.cshrc.* ${dir}/.bashrc.*"

echo "Installing .mh_profile with user's name."
rsh $home "rm ${dir}/.mh_profile.*"
sed -e "s/ebg/${name}/" /usr/local/lib/user/.mh_profile.${mail} \
            > /tmp/.mh_profile
rcp /tmp/.mh_profile ${home}:${dir}/.mh_profile
rm /tmp/.mh_profile

echo ""
echo "Changing modes and owners."
rsh $home "/bin/chmod 777 /export/${home}/${name}"
rsh $home "/etc/chown -R ${id}.${group} /export/${home}/${name}"

echo
echo "User ${name} added"
echo
```

Note that the above programs are not essential to the functioning of hip.atr.co.jp; they just centralize the information needed for the installation.

5 Conclusions

We now have an open, uniform environment within the ATR A&VP and H.I.P. research laboratories. Considerable time has been expended to develop an environment which is fair, understandable and customizable by all. I hope that the uniform environment has been a benefit.