AUUGN

Australian Unix systems User Group Newsletter

Volume 8

Number 1-2

Registered by Australia Post Publication No. NBG6524

The Australian UNIX* systems User Group Newsletter

Volume 8 Number 1-2

June 1987

CONTENTS

* UNIX is a registered trademark of AT&T in the USA and other countries.

AUUG General Information

Memberships and Subscriptions

Membership, Change of Address, and Subscription forms can be found at the end of this issue.

All correspondence concerning membership of the AUUG should be addressed to:-

The AUUG Membership Secretary,
P.O. Box 366,
Kensington, N.S.W. 2033.
AUSTRALIA

General Correspondence

All other correspondence for the AUUG should be addressed to:-

The AUUG Secretary,
Department of Computer Science,
Melbourne University,
Parkville, Victoria 3052.
AUSTRALIA

ACSnet: auug@munnari.oz

AUUG Executive

Ken McDonell, *President*

kenj@moncsbruce.oz — **(Temporary address is kjmcdonell@er.waterloo.cdn)**
Monash University, Victoria — (University of Waterloo, Canada)

Robert Elz, *Secretary*

kre@munnari.oz
University of Melbourne, Victoria

Chris Maltby, *Treasurer*

chris@gris.oz
Softway Pty. Ltd., N.S.W.

Chris Campbell, *Committee Member*

chris@olisyd.oz
Olivetti Australia, N.S.W.

John Lions, *Committee Member*

johnl@elecvax.oz
University of New South Wales, N.S.W.

Tim Roper, *Committee Member*

timr@labtam.oz
Labtam Limited, Victoria

Lionel Singer, *Committee Member*

lionel@pta.oz — **(This does not work)**
Lionel Singer Group, N.S.W.

Next AUUG Meeting

The next meeting will be held at NSWIT on the 27th and 28th of August.
Futher details are provided in this issue.

AUUG Newsletter

Editorial

Thank you again to those of you who support the Newsletter by sending in their articles and suggestions.

Some readers feel that the AUUGN is produced with the UNIX guru or systems programmer in mind and it is not relevent to the general UNIX user. Whoever that is !! As Editor, I can promise that this is not done on purpose. I publish the articles people want to write for the Newsletter, not what other people want to read. Perhaps you can help by writing a tutorial article on some aspect of UNIX you are familiar with or write some book reviews of good introductory texts. If you wish to see something in the AUUGN on a particular subject, you may be the best person to write the article, or encourage a person who you know is an expert in the field to produce it.

PLEASE SEND ME AN ARTICLE SOON.

Another comment is that the articles have a fanatical fervour when discussing our beloved Operating System. I think people should remember although we carry articles of general interest, that the aim of the AUUG, and its Newsletter, as stated in the constitution is "to promote the knowledge and understanding of the UNIX system". The people writing the articles are doing exactly that, PROMOTING UNIX, otherwise they would not bother writing for this particular publication with its audience in mind. If you feel that a particular author's position is too extreme, a letter to the Editor is probably appropriate. Chris Rusbridge has done extactly that, see his letter in this issue.

In the past two months I have been to the releases of two completely different machines which claim to be in the 20 to 30 MIP range. The one thing they both had in common was that they were multiprocessors running UNIX. I think we will see more of this particular type of the machine as the performance limits of a single CPU given any particular technology is realized. It is simplier and more cost effective for the designers to bolt on another CPU, without it being gobbled up by operating system overheads, which was a problem in the past. Other advantages of this approach is that gives expandablity, usually to a certain limit of CPU, and some measure of redundancy, as these systems allow crook CPUs to be removed. And remember folks it was UNIX that bought this cheap grunt to you because it was portable, flexible, and expandable.

Please note that the next AUUG Meeting is being held in Sydeny in August. Make an effort to attend, as I am sure that you will find it worthwhile.

REMEMBER, if the mailing label that comes with this issue is highlighted, it is time to renew your AUUG membership.

AUUGN Correspondence

All correspondence reguarding the AUUGN should be addressed to:-

John Carey
AUUGN Editor
Computer Centre
Monash University
Clayton, Victoria 3168
AUSTRALIA

ACSnet: auugn@monu1.oz

Phone: +61 3 565 4754

Contributions

The Newsletter is published approximately every two months. The deadline for contributions for the next issue is Friday the 14th of August 1987.

Contributions should be sent to the Editor at the above address.

I prefer documents sent to me by via electronic mail and formatted using *troff -mm* and my footer macros, troff using any of the standard macro and preprocessor packages (-ms, -me, -mm, pic, tbl, eqn) as well TeX, and LaTeX will be accepted.

Hardcopy submissions should be on A4 with 35 mm left at the top and bottom so that the AUUGN footers can be pasted on to the page. Small page numbers printed in the footer area would help.

Advertising

Advertisements for the AUUG are welcome. They must be submitted on an A4 page. No partial page advertisements will be accepted. The current rate is AUD$ 200 dollars per page.

Mailing Lists

For the purchase of the AUUGN mailing list, please contact Chris Maltby.

Disclaimer

Opinions expressed by authors and reviewers are not necessarily those of the Australian UNIX systems User Group, its Newsletter or its editorial committee.

Softway

a Techway company

for

- ✓ UNIX System V
- ✓ Documentor's Workbench 2.0
  - and various back-end drivers
  - PostScript support of plain text
  - support for graphs and images
- ✓ Ports & Device Drivers
- ✓ Intelligent Benchmarking
- ✓ SUN-III (ACSnet) + installation
- ✓ Biway - Bi-directional modem software for System V and 4bsd
- ✓ Courses:
  - Beginner's Workshop
  - Fast start to UNIX
  - System Administrators' workshop
- ✓ Technical Backup

- and all sorts of interesting software development.

Softway Pty Ltd. (Incorporated in NSW)
120 Chalmers St, Strawberry Hills, NSW.
PO Box 305, Strawberry Hills, NSW 2012.
☎ (02) 698 2322 Fax (02) 957 6914

AUUG

Winter Conference and Exhibition 1987

Sydney, August 27 and 28.

The Winter 1987 AUUG conference and exhibition will be held in Sydney, August 27 and 28 (Thursday/Friday) 1987.

The conference is being hosted by the New South Wales Institute of Technology. Local conference organisation is under the direction of Greg Webb of the NSWIT Computer Centre.

Registration forms, and additional information, will be mailed to members, and others, in the near future. More information on registrations can be obtained from Tony McGrath of Computing Science at NSWIT.

Correspondence should be addressed to the AUUG Conference, NSWIT Computer Centre, PO Box 123, Broadway NSW 2007. Phone (02) 218 9437, Fax to (02) 281 2498, or e-mail to **auug@nswitgould.oz**.

Michael Tilson, president of HCR, Toronto Canada, has been invited to give the keynote address at the conference, and Peter Weinberger, of AT&T Bell Laboratories has been invited as guest speaker. Other speakers are being considered, watch aus.auug and other announcements for more details.

We are now actively seeking papers for this conference. The programme committee chairman is Bob Kummerfeld from the University of Sydney.

Please send abstracts of papers to him at **bob@basser.oz**.

Paper abstracts can be addressed to Dr R.J. Kummerfeld, Basser Department of Computer Science, University of Sydney, NSW 2006.

The deadline for abstracts is Jul 10 1987. Authors will be notified of acceptance by July 30.

Authors of papers given at the conference will receive complimentary admission to the conference dinner. Authors who provide a written version of their paper by August 21 will have the conference registration fee waived.

In addition, AUUG has decided to hold a competition for the best paper by a full time student at an Australian educational institution. The prize for this competition will be an expenses paid trip to the AUUG meeting to present the winning paper. Students should indicate with their abstract that they wish to enter the competition, and then should provide the full written paper to the programme committee (which will be the sole judge) by August 14. AUUG reserves the right to not award the prize if no entries of a suitable standard are forthcoming.

AUUG has also set aside a sum of money to assist with travel expenses of students who have had papers accepted for the conference. Students submitting papers should make their interest in this known with their submission.

Adelaide UNIX Users Group

The Adelaide UNIX Users Group has been meeting on a formal basis for 12 months. Meetings are held on the third Wednesday of each month. To date, all meetings have been held at the University of Adelaide. However, it was recently decided to change the meeting time from noon to 6pm. This has necessitated a change of venue, and, as from April, meetings will be held at the offices of Olivetti Australia.

In addition to disseminating information about new products and network status, time is allocated at each meeting for the raising of specific UNIX related problems and for a brief (15-20 minute) presentation on an area of interest. Listed below is a sampling of recent talks.

D. Jarvis	"The UNIX Literature"
K. Maciunas	"Security"
R. Lamacraft	"UNIX on Micros"
W. Hosking	"Office Automation"
P. Cheney	"Commercial Applications of UNIX"
J. Jarvis	"troff/ditroff"

The mailing list currently numbers 34, with a healthy representation (40%) from commercial enterprises. For further information, contact Dennis Jarvis (dhj@aegir.dmt.oz) on (08) 268 0156.

Dennis Jarvis,
Secretary, AdUUG.

Dennis Jarvis, CSIRO, PO Box 4, Woodville, S.A. 5011, Australia.

PHONE: +61 8 268 0156
UUCP: {decvax,pesnta,vax135}!mulga!aegir.dmt.oz!dhj
ARPA: dhj%aegir.dmt.oz!dhj@seismo.arpa
CSNET: dhj@aegir.dmt.oz

A Supplemental Document For AWK

- or -

Things Al, Pete, And Brian Didn't Mention Much

John W. Pierce

Department of Chemistry
University of California, San Diego
La Jolla, California 92093
jwp%chem@sdcsvax.ucsd.edu

ABSTRACT

As **awk** and its documentation are distributed with *4.2 BSD UNIX** there are a number of bugs, undocumented features, and features that are touched on so briefly in the documentation that the casual user may not realize their full significance. While this document applies primarily to the *4.2 BSD* version of *UNIX*, it is known that the *4.3 BSD* version does not have all of the bugs fixed, and that it does not have updated documentation. The situation with respect to the versions of **awk** disitributed with other versions *UNIX* and similar systems is unknown to the author.

In this document references to "the user manual" mean *Awk - A Pattern Scanning and Processing Language (Second Edition)* by Aho, Kernighan, and Weinberger. References to "awk(1)" mean the entry for **awk** in the *UNIX Programmer's Manual, 4th Berkeley Distribution.* References to "the documentation" mean both of those.

In most examples, the outermost set of braces ('{ }') have been ommitted. They would, of course, be necessary in real scripts.

1. Known Bugs

There are three main bugs known to me. They involve:

Assignment to input fields.

Piping output to a program from within an **awk** script.

Using '*' in *printf* field width and precision specifications does not work, nor do '\f' and '\b' print formfeed and backspace respectively.

1.1. Assignment to Input Fields

[This problem is partially fixed in *4.3BSD*; see the last paragraph of this section regarding the unfixed portion.]

The user manual states that input fields may be objects of assignment statements. Given the input line

    field_one field_two field_three

the script

*UNIX is a trademark of AT&T

```
$2 = "new_field_2"
print $0
```

should print

```
field_one new_field_2 field_three
```

This does not work; it will print

```
field_one field_two field_three
```

That is, the script will behave as if the assignment to $2 had not been made. However, explicitly referencing an "assigned to" field *does* recognize that the assignment has been made. If the script

```
$2 = "new_field_2"
print $1, $2, $3
```

is given the same input it will [properly] print

```
field_one new_field_2 field_three
```

Therefore, you can get around this bug with, e.g.,

```
$2 = "new_field_2"
output = $1                     # Concatenate output fields
for(i = 2; i <= NF; ++i)        # into a single output line
      output = output OFS $i    # with OFS between fields
print output
```

In *4.3BSD*, this bug has been fixed to the extent that the failing example above works correctly. However, a script like

```
$2 = "new_field_2"
var = $0
print var
```

still gives incorrect output. This problem can be bypassed by using

```
var = sprintf("%s", $0)
```

instead of "*var* = $0"; *var* will have the correct value.

1.2. Piping Output to a Program

[This problem appears to have been fixed in *4.3BSD*, but that has not been exhaustively tested.]

The user manual states that *print* and *printf* statements may write to a program using, e.g.,

```
print | "command"
```

This would pipe the output into *command*, and it does work. However, you should be aware that this causes **awk** to spawn a child process (*command*), and that it *does not* wait for the child to exit before it exits itself. In the case of a "slow" command like **sort**, **awk** may exit before *command* has finished.

This can cause problems in, for example, a shell script that depends on everything done by **awk** being finished before the next shell command is executed. Consider the shell script

```
awk -f awk_script input_file
mv sorted_output somewhere_else
```

and the **awk** script

```
print output_line | "sort -o sorted_output"
```

If *input_file* is large **awk** will exit long before **sort** is finished. That means that the **mv** command will be executed before **sort** is finished, and the result is unlikely to be what you wanted. Other than fixing the source, there is no way to avoid this problem except to handle such pipes outside of the awk script, e.g.

```
awk -f awk_file input_file | sort -o sorted_output
mv sorted_output somewhere_else
```

which is not wholly satisfactory.

See *Sketchily Documented Features* below for other considerations in redirecting output from within an **awk** script.

1.3. Printf and '*', '\f', and '\b'

The document says that the *printf* function provided is identical to the *printf* provided by the *C* language **stdio** package. This is incorrect: '*' cannot be used to specify a field width or precision, and '\f' and '\b' cannot be used to print formfeeds and backspaces.

The command

```
printf("%*.s", len, string)
```

will cause a core dump. Given **awk**'s age, it is likely that its *printf* was written well before the use of '*' for specifying field width and precision appeared in the **stdio** library's *printf.* Another possibility is that it wasn't implemented because it isn't really needed to achieve the same effect.

To accomplish this effect, you can utilize the fact that **awk** concatenates variables before it does any other processing on them. For example, assume a script has two variables *wid* and *prec* which control the width and precision used for printing another variable *val:*

```
[code to set "wid", "prec", and "val"]

printf("%" wid "." prec "d\n", val)
```

If, for example, wid is 8 and *prec* is 3, then /fBawk will concatenate everything to the left of the comma in the *printf* statement, and the statement will really be

```
printf(%8.3d\n, val)
```

These could, of course, been assigned to some variable *fmt* before being used:

```
fmt = "%" wid "." prec "d"

printf(fmt "\n", val)
```

Note, however, that the newline ("\n") in the second form *cannot* be included in the assignment to *fmt.*

To allow use of '\f' and '\b', **awk**'s *lex* script must be changed. This is trivial to do (it is done at the point where '\n' and '\t' are processed), but requires having source code. [I have fixed this and have not seen any unwanted effects.]

2. Undocumented Features

There are several undocumented features:

Variable values may be established on the command line.

A **getline** function exists that reads the next input line and starts processing it immediately.

Regular expressions accept octal representations of characters.

A **-d** flag argument produces debugging output if **awk** was compiled with "DEBUG" defined.

Scripts may bè "compiled" and run later (providing the installer did what is necessary to make this work).

2.1. Defining Variables On The Command Line

To pass variable values into a script at run time, you may use

variable=value

(as many as you like) between any "**-f** *scriptname*" or *program* and the names of any files to be processed. For example,

```
awk -f awkscript today=\"‘date‘\" infile
```

would establish for *awkscript* a variable named **today** that had as its value the output of the **date** command.

There are a number of caveats:

Such assignments may appear only between **-f** *awkscript* (or *program* or [see below] **-R***awk.out*) and the name of any input file (or '-').

Each *variable=value* combination must be a single argument (i.e. there must not be spaces around the '=' sign); *value* may be either a numeric value or a string. If it is a string, it must be enclosed in double quotes at the time **awk** reads the argument. That means that the double quotes enclosing *value* on the command line must be protected from the shell as in the example above or it will remove them.

Variable is not available for use within the script until after the first record has been read and parsed, but it is available as soon as that has occurred so that it may be used before any other processing begins. It does not exist at the time the **BEGIN** block is executed, and if there was no input it will not exist in the **END** block (if any).

2.2. Getline Function

Getline immediately reads the next input line (which is parsed into *$1*, *$2*, etc) and starts processing it at the location of the call (as opposed to **next** which immediately reads the next input line but starts processing from the start of the script).

Getline facilitates performing some types of tasks such as processing files with multiline records and merging information from several files. To use the latter as an example, consider a case where two files, whose lines do not share a common format, must be processed together. Shell and **awk** scripts to do this might look something like

In the shell script

```
( echo DATA1; cat datafile1; echo ENDdata1 \
  echo DATA2; cat datafile2; echo ENDdata2 \
) | \
   awk -f awkscript - > awk_output_file
```

In the **awk** script

```
/^DATA1/ {        # Next input line starts datafile1
        while (getline && $1 !~ /^ENDdata1$/)
             {
             [processing for data1 lines]
             }
        }

/^DATA2/ {        # Next input line starts datafile2
        while (getline && $1 !~ /^ENDdata2$/)
             {
             [processing for data2 lines]
             }
        }
```

There are, of course, other ways of accomplishing this particular task (primarily using **sed** to preprocess the information), but they are generally more difficult to write and more subject to logic errors. Many cases arising in practice are significantly more difficult, if not impossible, to handle without **getline**.

2.3. Regular Expressions

The sequence "\ddd" (where 'd' is a digit) may be used to include explicit octal values in regular expressions. This is often useful if "nonprinting" characters have been used as "markers" in a file. It has not been tested for ASCII values outside the range 01 through 0127.

2.4. Debugging output

[This is unlikely to be of interest to the casual user.]

If **awk** was compiled with "DEBUG" defined, then giving it a **-d** flag argument will cause it to produce debugging output when it is run. This is sometimes useful in finding obscure problems in scripts, though it is primarily intended for tracking down problems with **awk** itself.

2.5. Script "Compilation"

[It is likely that this does not work at most sites. If it does not, the following will probably not be of interest to the casual user.]

The command

```
awk -S -f script.awk
```

produces a file named **awk.out.** This is a core image of **awk** after parsing the file *script.awk.* The command

```
awk -Rawk.out datafile
```

causes **awk.out** to be applied to *datafile* (or the standard input if no input file is given). This avoids having to reparse large scripts each time they are used. Unfortunately, the way this is implemented requires some special action on the part of the person installing **awk.**

As **awk** is delivered with *4.2 BSD* (and *4.3 BSD*), *awk.out* is created by the **awk -S ...** process by calling **sbrk()** with '0', writing out the returned value, then writing out the core image from location 0 to the returned address. The **awk -R...** process reads the first word of *awk.out* to get the length of the image, calls **brk()** with that length, and then reads the image into itself starting at location 0. For this to work, **awk** must have been loaded with its text segment writeable. Unfortunately, the *BSD* default for **ld** is to load with the text read-only and shareable. Thus, the installer must remember to take special action (e.g. "cc -N ..." [equivalently "ld -N ..."] for *4BSD*) if these flags are to work.

[Personally, I don't think it is a very good idea to give **awk** the opportunity to write on its text segment; I changed it so that only the data segment is overwritten.]

Also, due to what appears to be a lapse in logic, the first non-flag argument following **-R***awk.out* is discarded. [Disliking that behavior, the I changed it so that the **-R** flag is treated like the **-f** flag: no flag arguments may follow it.]

3. Sketchily Documented Features

3.1. Exit

The user manual says that using the **exit** function causes the script to behave as if end-of-input has been reached. Not menitoned explicitly is the fact that this will cause the **END** block to be executed if it exists. Also, two things are ommitted:

> **exit(***expr***)** causes the script's exit status to be set to the value of *expr*.
>
> If **exit** is called within the **END** block, the script exits immediately.

3.2. Mathematical Functions

The following builtin functions exist and are mentioned in *awk(1)* but not in the user manual.

int(*x***)**	*x* trunctated to an integer.
sqrt(*x***)**	the square root of *x* for *x* >= 0, otherwise zero.
exp(*x***)**	e-to-the-*x* for -88 <= *x* <= 88, zero for *x* < -88, and dumps core for *x* > 88.
log(*x***)**	the natural log of *x*.

3.3. OFMT Variable

The variable **OFMT** may be set to, e.g. "%.2f", and purely numerical output will be bound by that restriction in **print** statements. The default value is "%.6g". Again, this is mentioned in *awk(1)* but not in the user manual.

3.4. Array Elements

The user manual states that "Array elements ... spring into existence by being mentioned." This is literally true; *any* reference to an array element causes it to exist. ("I was thought about, therefore I am.") Take, for example,

```
if(array[$1] == "blah")
        {
        [process blah lines]
        }
```

If there is not an existing element of **array** whose subscript is the same as the contents of the current line's first field, *one is created* and its value (null, of course) is then compared with "blah". This can be a bit disconcerting, particularly when later processing is using

```
for (i in array)
        {
        [do something with result of processing
        "blah" lines]
        }
```

to walk the array and expects all the elements to be non-null. Succinct practical examples are difficult to construct, but when this happens in a 500 line script it can be difficult to determine what has gone wrong.

3.5. FS and Input Fields

By default any number of spaces or tabs can separate fields (i.e. there are no null input fields) and trailing spaces and tabs are ignored. However, if **FS** is explicitly set to any character other than a space (e.g., a tab: **FS** = "\t"), then a field is defined by each such character and trailing field separator characters are not ignored. For example, if '>' represents a tab then

```
one>>three>>five>
```

defines six fields, with fields two, four, and six being empty.

If **FS** is explicitly set to a space (**FS** = " "), then the default behavior obtains (this may be a bug); that is, both spaces and tabs are taken as field separators, there can be no null input fields, and trailing spaces and tabs are ignored.

3.6. RS and Input Records

If **RS** is explicitly set to the null string (**RS** = ""), then the input record separator becomes a blank line, and the newlines at the end of input lines is a field separator. This facilitates handling multiline records.

3.7. "Fall Through"

This is mentioned in the user manual, but it is important enough that it is worth pointing out here, also. In the script

```
/pattern_1/ {
            [do something]
            }

/pattern_2/ {
            [do something]
            }
```

all input lines will be compared with both *pattern_1* and *pattern_2* unless the **next** function is used before the closing '}' in the *pattern_1* portion.

3.8. Output Redirection

Once a file (or pipe) is opened by **awk** it is not closed until **awk** exits. This can occassionally cause problems. For example, it means that a script that sorts its input lines into output files named by the contents of their first fields (similar to an example in the user manual)

```
{ print $0 > $1 }
```

is going to fail if the number of different first fields exceeds about 10. This problem *cannot* be avoided by using something like

```
{
command = "cat >> " $1
print $0 | command
}
```

as the value of the variable **command** is different for each different value of *$1* and is therefore treated as a different output "file".

[I have not been able to create a truly satisfactory fix for this that doesn't involve having **awk** treat output redirection to pipes differently from output to files; I would greatly appreciate hearing of one.]

3.9. Field and Variable Types, Values, and Comparisons

The following is a synopsis of notes included with **awk**'s source code.

3.9.1. Types

Variables and fields can be strings or numbers or both.

3.9.1.1. Variable Types

When a variable is set by the assignment

```
var = expr
```

its type is set to the type of *expr* (this includes +=, ++, etc). An arithmetic expression is of type *number*, a concatenation is of type *string*, etc. If the assignment is a simple copy, e.g.

```
var1 = var2
```

then the type of *var1* becomes that of *var2*.

Type is determined by context; rarely, but always very inconveniently, this context-determined type is incorrect. As mentioned in *awk(1)* the type of an expression can be coerced to that desired. E.g.

```
{
expr1 + 0

expr2 ""     # Concatenate with a null string
}
```

coerces *expr1* to numeric type and *expr2* to string type.

3.9.1.2. Field Types

As with variables, the type of a field is determined by context when possible, e.g.

$1++   clearly implies that *$1* is to be numeric, and

$1 = $1 "," $2   implies that $1 and $2 are both to be strings.

Coercion is done as needed. In contexts where types cannot be reliably determined, e.g.,

```
if($1 == $2) ...
```

the type of each field is determined on input by inspection. All fields are strings; in addition, each field that contains only a number is also considered numeric. Thus, the test

```
if($1 == $2) ...
```

will succeed on the inputs

```
0      0.0
100    1e2
+100   100
1e-3   1e-3
```

and fail on the inputs

```
(null)    0
(null)    0.0
2E-518    6E-427
```

"only a number" in this case means matching the regular expression

```
^[+-]?[0-9]*\.?[0-9]+(e[+-]?[0-9]+)?$
```

3.9.2. Values

Uninitialized variables have the numeric value 0 and the string value "". Therefore, if *x* is uninitialized,

```
if(x) ...
if (x == "0") ...
```

are false, and

```
if(!x) ...
if(x == 0) ...
if(x == "") ...
```

are true.

Fields which are explicitly null have the string value "", and are not numeric. Non-existent fields (i.e., fields past NF) are also treated this way.

3.9.3. Types of Comparisons

If both operands are numeric, the comparison is made numerically. Otherwise, operands are coerced to type string if necessary, and the comparison is made on strings.

3.9.4. Array Elements

Array elements created by **split** are treated in the same way as fields.

An Overview of the UNIX World in Japan

Nobuo Saito[*]
Keio University
President of Japan UNIX Society

13 May 1987

1 UNIX History and Current Status in Japan

UNIX systems were introduced to Japan in 1977 for the first time. They were PDP-11 Version 6/7, and the users included Univ. of Tsukuba, Univ. of Tokyo , Keio University, etc. In 1980, VAX-11 4BSD's were introduced, and the users included Univ. of Tokyo, Keio University, SRA, JSD and so on. In 10 years it has become very popular both in business world and academic society. According to AT&T UNIX Pacific, the current number of source licensees in Japan is as follow:

Commercial license:	230
Academic license:	180

(January 1986)

There are also a lot of binary sublicenses given through the UNIX machines, and the total number of UNIX users nowadays may be more than 10 thousands.

[*]Email address: ns%keio.junet@japan.cs.net

2 UNIX Systems and Machines

From the early stage, there were several domestic UNIX machines developed as commercial products in Japan. The developers include main framers, independent system houses and so on. They are CEC, Toshiba, NEC, Hitachi, DCL, Tateishi Elec. Co., Mitsubishi, OKI, Fujitsu, Sony etc. Recently, the Sigma project has stimulated the UNIX market, and the number of UNIX systems/machines is still increasing. They cover from PC UNIX to high quality personal workstations.

There are also Japanese branches or subsidiaries for major US UNIX systems machines; i.e. DEC Japan, AT&T and Ricoh, SUN Japan and CI(Chu Ito), Japan NCR, YHP(Yokokawa Hewlett Packard), IBM Japan, and so on.

It is important for Japanese UNIX system to provide users with Japanese language(Kanji character) processing facilities. They include Kanji input/output, Japanese document processing system, Japanese command shell and so on. Recently, most of the domestic or imported UNIX systems/machines are equipped with these facilities.

In any case, the UNIX system/machine builders have just started the tough competition in extending their market share, and this situation will be favorable for end users of UNIX.

3 JUS Activities

JUS(Japan Unix Society) was officially established in 1983, although its preparatory meeting was held in 1980. The current number of JUS members is over 1000. It is operated by the steering committee, which consists of about 30 committee members. They come from universities, UNIX system houses, software houses, main framers, national laboratories and so on. The regular activities of our society are as follows:

JUS Symposium:	2 time/ year (Summer, Winter)
	paper presentation, UNIX exhibition
UNIX Fair:	Tutorials and Exhibition 1 time/ 1-1.5 year
Ad Hoc Tutorials and Workshop:	any time necessary

The main purposes of our societies are to stimulate UNIX users and UNIX markets, to exchange technical and commercial informations, to watch the trends of UNIX systems and societies in the world. For these reasons, we are to publish our bulletin "/etc/wall" quarterly. We also plan to make a distribution tape for public domain software.

For the international coordination, we just decided to have an affiliation program with /usr/group. We would like to have the same relationship with oversea UNIX users groups. In 1985, the JUS representatives visited Korea to attend KUUG symposium. This time is our second visit to foreign UNIX user group.

4 Sigma Project

The Japanese Government MITI(Ministry of International Trade and Industry) started Sigma project in 1985. Its main purpose is to industrialize the software development process using UNIX systems. It consists of the following components:

- Sigma WS(High quality personal WS)
- Sigma OS(Extended UNIX)
- Sigma Network(LAN and WAN)
- Sigma Center(Network manager and DB service)

It is a 5 year project, and several prototype workstations appeared recently.

Sigma OS may become the standard of UNIX, but it is very important to watch the technology trends and strategy being developed on Sigma project. We should keep the technology standard as better and higher as possible.

JUNET
Japan {Unix, University} Network

Jun Murai
jun@japan.cs.net

University of Tokyo
2-11-16 Yayoi, Bunkyoku
Tokyo, 113 Japan

ABSTRACT

JUNET is a Japanese computer network connecting various types of organizations relating computer science. The operation of the network was started in October 1984 and currently connecting 73 organizations with more than 200 computers. It connects universities and major research laboratories in Japan, and the protocols currently used are TCP/IP over leased lines and UUCP over telephone lines. The services such as electronic mail and network news have been provided since the network was started, and special technologies for Japanese character handlings, name servers, and multimedia mail supports have been developed.

In this paper, the characteristics of JUNET are introduced.

1. Introduction

The history of JUNET[Murai85] started when University of Tokyo, Tokyo Institute of Technology and Keio University started exchanging mails using UUCP on telephone lines in October 1984. JUNET connects local area networks at research institutes in Japan and it also provides users with the means of the worldwide communications via various international links.

In Japan, there have been strong demands for computer networks where researchers can exchange various research information. Such a network, without a doubt, improves qualities of research and development environment at universities and research laboratories.

Since JUNET is the first and the only existing national wide computer network in Japan, most of the experimental studies about wide-area computer networks, from physical communication technology to application design, have been done using the network. The purpose in the very first stage of the network, thus, was providing actual network services to researchers and put Japanese communities into world academic networks. And then, we started to work to solve problems existing on the network such as Japanese character handling, naming systems, and multi-media message exchanges.

There are some historical reasons which have prevented Japanese research and development communities from establishing a network for them. One of such a reason is that there have been not so many good computers for communication in the communities. Another example of such a reason may be that there used to be more restrictions in using the telephone lines than there are now. Since 1983, UNIX† has become popular in Japanese research and development communities, and this has encouraged the communities to start a simple network such as the one using UUCP protocol over public telephone lines.

JUNET was founded when a gateway in Tokyo Institute of Technology started to

† UNIX is a trademark of Bell Laboratories.

poll Keio University's gateway, both running UNIX operating system, in October 1984. The origin of JUNET has thus been established including University of Tokyo's VAX-11 one week later. Applications used that time were electronic mails and electronic news.

At the time the three universities started the network, researches specificly required in Japanese environment, such as Japanese language interface for communication applications, Japanese character code standard, and interconnection to existing research and development networks, were started to take place. The domain addressing over UUCP network was introduced in May 1985 with a system to generate the address conversion software. At the same time, transmission rates of modems were switched to 1200bps at major sites in JUNET.

As for the internetworking between JUNET and other foreign networks, Kokusai Denshin Denwa Co., Ltd. (KDD), an international telephone and telegraph company, started investigating the USENET message qualities in July, 1983. After one year investigations, KDD decided to establish reliable links with USENET in September, 1984. In January, 1985, an active international link to USENET was set up, for the first time, with Centrum voor Wiskunde en Informatica, the Netherlands. Since then, several foreign gateways in USENET have been linked with this JUNET gateway. Some are located in Europe and the others in the United States. In December 1985, *utokyo-relay,* often called *japan* at University of Tokyo joined CSNET as a gateway between CSNET and JUNET. This link became a major route from world to Japanese academic communities and vise-versa.

JUNET currently connects more than 200 systems in 73 organizations and the growing number of organizations can be seen in Figure 1. The network covers from Hokkaido, the northern island, to Kyuushuu, the southern iland, however, concentrations in Tokyo and Osaka areas are obvious (Figure 2). Most of the links have been dial-up lines using 1200bps or 2400bps modems, although special mechanisms have developed in UNIX operating system in order to use fast dial-up modems with 9600bps or higher transmission rate. This mechanism is currently used for

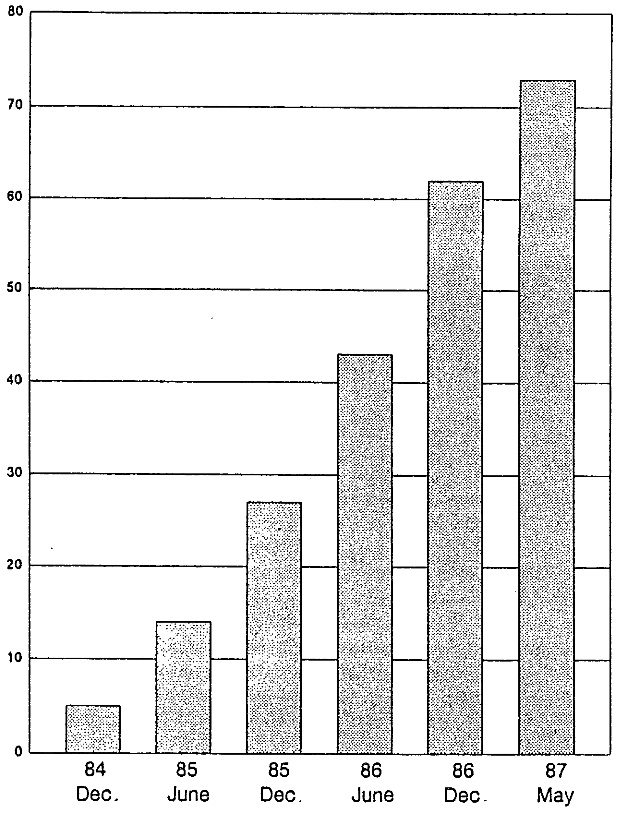
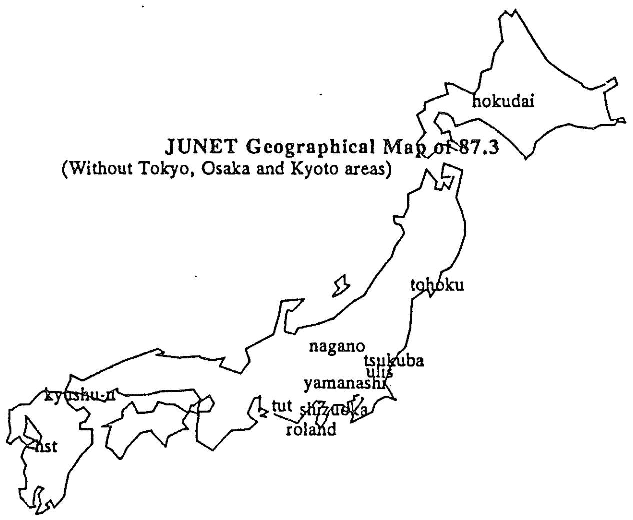


Figure 1. Number of domains



JUNET Geographical Map of 87.3
(Without Tokyo, Osaka and Kyoto areas)
hokudai
tohoku
nagano
tsukuba
ulis
yamanashi
tut
shizuoka
roland
kyushu-u
nst

Figure 2. JUNET Geographical Map

UUCP protocols and a mechanism for TCP/IP protocols over dial-up telephone lines is also developed for same kinds of modems.

Organizations connecting to the network are universities, research laboratories of computer software/hardware companies, and

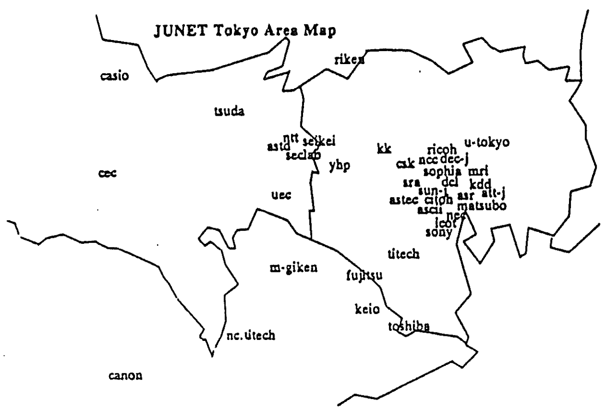


Figure 2a. Tokyo Map

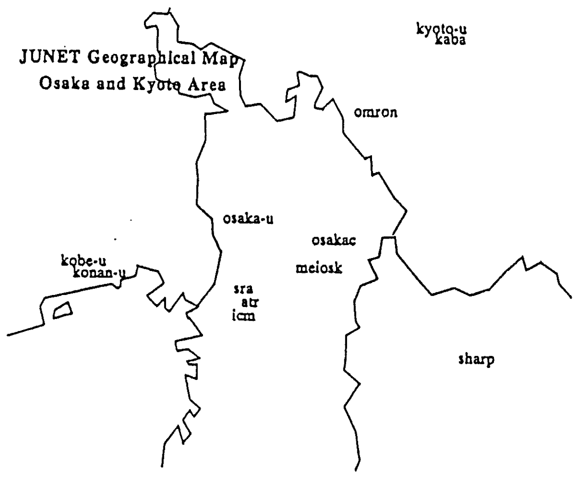


Figure 2b. Osaka and Kyoto Map

research laboratories of telephone companies. Users of the network are registered at gateways of each of the institutes in order to provide name server functions described later. All the functions are administrated by administrators at each of the institutes on totally volunteer basis.

There are several application services available in *JUNET.* Electronic mails and electronic news are the major applications among them. Since use of Japanese language in messages is the strong demand of users in the network, multi-languages are supported in text messages for both applications. Existence of Japanese character codes in messages involves some confusing problem at a system which have no functions to handle the Japanese codes. To avoid this problem, special considerations are made in a message header, yet keeping the RFC822[Crocker82].

Addressing and naming are handled by a JUNET naming system. This system provides:

- hierarchical domain-based addressing,
- name server functions,
- routing decision,
- and automatic naming database management.

2. JUNET Domain Addressing

In the hierarchy of JUNET domain structure, a domain called *junet* is the top domain, although we are now preparing to employ ISO's contry code for Japan *jp* as the top domain name. The second level domains are called *sub-domains,* and each of them represents a name of an institute or an organization. Lower level domains than the sub-domains are determined at each of the sub-domains. In any cases, the lowest level domains are names of hosts. The names of sub-domains usually are names well known to the society, but such names sometimes differ in intra/inter national environment. Therefore, one or more names can be registered as synonyms for a sub-domain name to help users to address with general knowledge on the name of organizations.

There are one of the distributed name server in each of the domains which handles definitions and deletions of names using a database dedicated to that domain. A name server of a domain thus has a database to define names of lower level domains adjacent to the domain, or names of resources, such as user names, if it is the lowest level domain. The knowledge of each name server is used in retrieving information of resource names and in delivering messages. The domain structure of JUNET is illustrated in Figure 3.

2.1. Design and Implementation

The message delivery system with the JUNET addressing functions is implemented in UNIX 4.2/3BSD. The message delivery is done by a modified version of sendmail[Allman83] whose rule is generated by a rule generating system of JUNET. The rules are described based on the following policies:

- Connections among sub-domains are UUCP links.

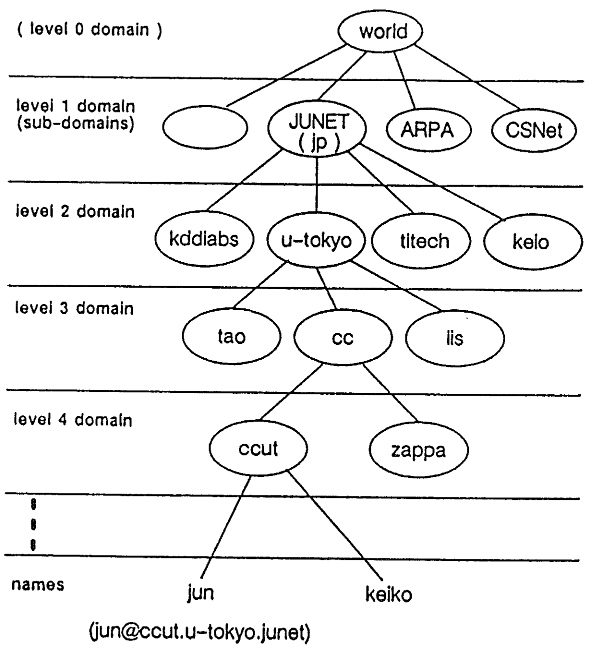


Figure 3. Domain Structure of JUNET

- Connections within a sub-domain can be either UUCP link, SMTP[Postel82] over Ethernet, or other kinds of links.
- Systems which are not running the addressing system of JUNET can be connected as one of the lowest level domain where functions of the higher level domain can not be working.
- Traditional UUCP notations (*a!b!c*) can be used anywhere.

Since the production rules of the JUNET sendmail system is differed site by site, the rules have to be generated at each site. To keep the consistency in the rules over JUNET sites, a generation system to generate the necessary rules is designed and implemented. The system reads a simple description file, generates the sendmail rules, and initializes the domain database. This database is accessed by the modified version of the sendmail system. The structure of the JUNET addressing system is shown in Figure 4.

In the description file, informations about the system is described under in the entries shown below:

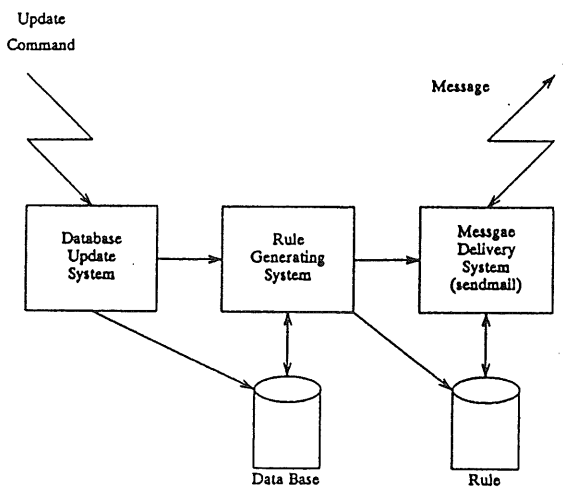


Figure 4. Structure of the System

entry tag	*information*
$make	whether this system is a gateway of a domain or not.
$name	a full name of a domain where this system belongs to, and its synonyms if any.
$site	a name of a system to which this system has a direct link, followed by a list of domain names which should be solved at the specified system.
$link	a system name which is directly connected to this system and is the entry point of the shortest path toward the system where the top domain name server exists.

The domain database contains relations between a physical link to a system and domain names which should be solved at that system. This is initialized in the generation operation, but not necessarily be the completed at the initialization time since the database is periodically updated automatically by a special message from the top domain. In other words, the initialized database is used as it is until the first update message arrives. Other than the sendmail, the *rmail* command which receives messages through UUCP links was modified to handle JUNET addresses efficiently.

3. Japanese Processing

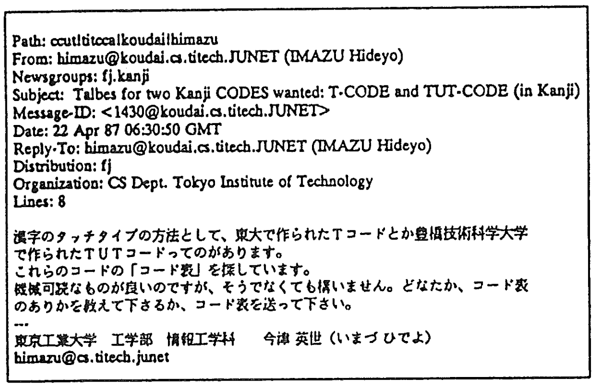
To satisfy the strong demands for Japanese character handling in the environment of JUNET, special considerations have to be paid to the communication software. The following two issues are considered according to the average environments of JUNET systems:

(1) There are several kinds of character codes which are actually used in operating systems as internal codes to represent Japanese characters including Kanji characters. Among them, there is the JIS Kanji code which is a 16-bit code set whose information is contains in two 7bit bytes. A sequence of the code is surrounded by a shift-in/shift-out pair when used in ASCII text. To avoid the complicated operations of switching modes when seeking such a byte stream according to the shift-in/out codes, some Kanji codes use two 8bit bytes to represent one Kanji character utilizing the the most significant bit for distinguishing the Kanji characters from ASCII characters.

(2) Two kinds of terminals are used by JUNET users. ASCII terminals which display only ascii characters, and ASCII/JIS terminals which can display both JIS Kanji codes and ASCII codes.

Since there are several Kanji codes used as described in (1), setting a standard of code set used in JUNET is necessary. We have selected the JIS Kanji code as the standard because it is the standard for peripherals and it consists of two 7bit codes. This allows the system to use the sign-bit for parity checking without compacting, and also allows systems to use some software which has functions of tricky usage of these bits: tty driver of UNIX operating system is masking these bits, and some text editor which is possibly used as a filter of text messages in the network uses the bits in special purposes.

When the JIS Kanji code set is used in ASCII text, the mode switching is done by the escape sequence defined at ISO. Most of the existing software based on 7bit ASCII code can be used for the sequence without change except that they have to accept the escape ASCII character which sometimes is removed in some communication software to avoid confusing problems of accepting object codes as text messages. When an operating system uses different Kanji character set other than JIS Kanji code, responsibility of conversion from the JIS Kanji codes to its internal code is in that system. Figure 5 shows example Kanji message.



```
Path: ccut!titcca!koudai!himazu
From: himazu@koudai.cs.titech.JUNET (IMAZU Hideyo)
Newsgroups: fj.kanji
Subject: Talbes for two Kanji CODES wanted: T-CODE and TUT-CODE (in Kanji)
Message-ID: <1430@koudai.cs.titech.JUNET>
Date: 22 Apr 87 06:30:50 GMT
Reply-To: himazu@koudai.cs.titech.JUNET (IMAZU Hideyo)
Distribution: fj
Organization: CS Dept. Tokyo Institute of Technology
Lines: 8

漢字のタッチタイプの方法として、東大で作られたＴコードとか豊橋技術科学大学
で作られたＴＵＴコードってのがあります。
これらのコードの「コード表」を探しています。
機械可読なものが良いのですが、そうでなくても構いません。どなたか、コード表
のありかを教えて下さるか、コード表を送って下さい。
---
東京工業大学　工学部　情報工学科　　今津 英世（いまづ ひでよ）
himazu@cs.titech.junet
```

Figure 5. Example Kanji message

To avoid breaking the screen image by transmitting the escape sequence to the pure ASCII terminals, we have set a convention in the header of text message format. Since we have two ways to represent Japanese language, Kanji representation and Romaji representation which is a phonetic representation of Japanese language using English alphabet, there are three possible formats in the 'Subject:' field of the header as shown in Figure 5. This helps a user at an ASCII terminal to refuse to show the contents of Kanji coded messages, then he/she might move to another terminal or print the message to a JIS Kanji printer in order to see the contents. User interfaces for mail and news such as *mh* and *rn* were modified to convert from/to network standard Kanji codes to/from local Kanji codes.

Figure 6 shows the increasing number of news articles for Japanese domestic newsgroups *fj*. Preference of Kanji characters in Japanese communities can be pointed out

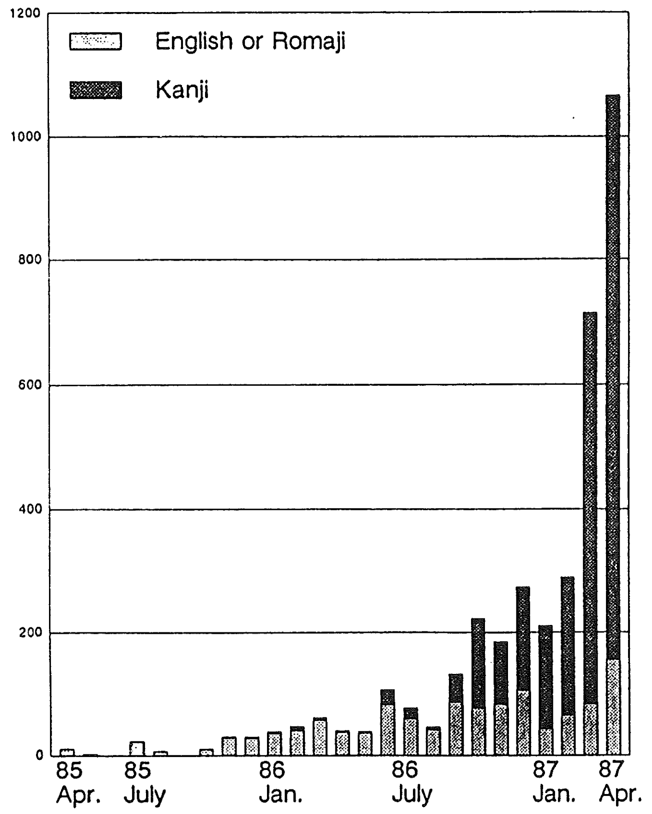


Figure 6. Number of articles

from the ratio of Kanji articles versus English articles shown in the figure.

4. Conclusion

The number of systems in JUNET has been increasing rapidly since it was started working in October 1984. Through studies of existing networks, the hierarchical domain addressing scheme was employed in JUNET.

The actual work for the addressing and routing of JUNET is achieved by the modifications of *sendmail* software and the rules, together with some other software. These programs provide an environment where the logical naming definitions and physical routing issues are clearly separated so that reliability, efficiency, extensibility, and flexibility of communication in the network are simultaneously achieved.

Internationalization of computer software is one of the most important issues in computer science, and some works have been achieved in JUNET communication software. In JUNET, electronic mail and news software are designed to enable 16bit Japanese character handlings, based on the standard JIS Kanji code.

Multi media message exchange mechanism is being developed. Rest of current studies on the JUNET project are as follows:

- Constructions of a distributed system over a widely interconnected distributed environment using JUNET. This provides general purpose system interfaces to programs based on hierarchical name spaces for general distributed resources such as files, users and machines.
- In the current system for message deliveries, the overheads caused by erroneous messages are large. This can be reduced by some mechanisms which handle name serving and message delivery functions dynamically based on distributed database technologies.

References

Allman83.
E. Allman, "Sendmail -- An Interconnecting Mail Rerouter, Version 4.2," in *UNIX Programmer's Mannual, 4.2 Berkeley Software Distribution*, vol. 2c, Virtual VAX-11 Version, Univ. of California, Berkeley, August 1983.

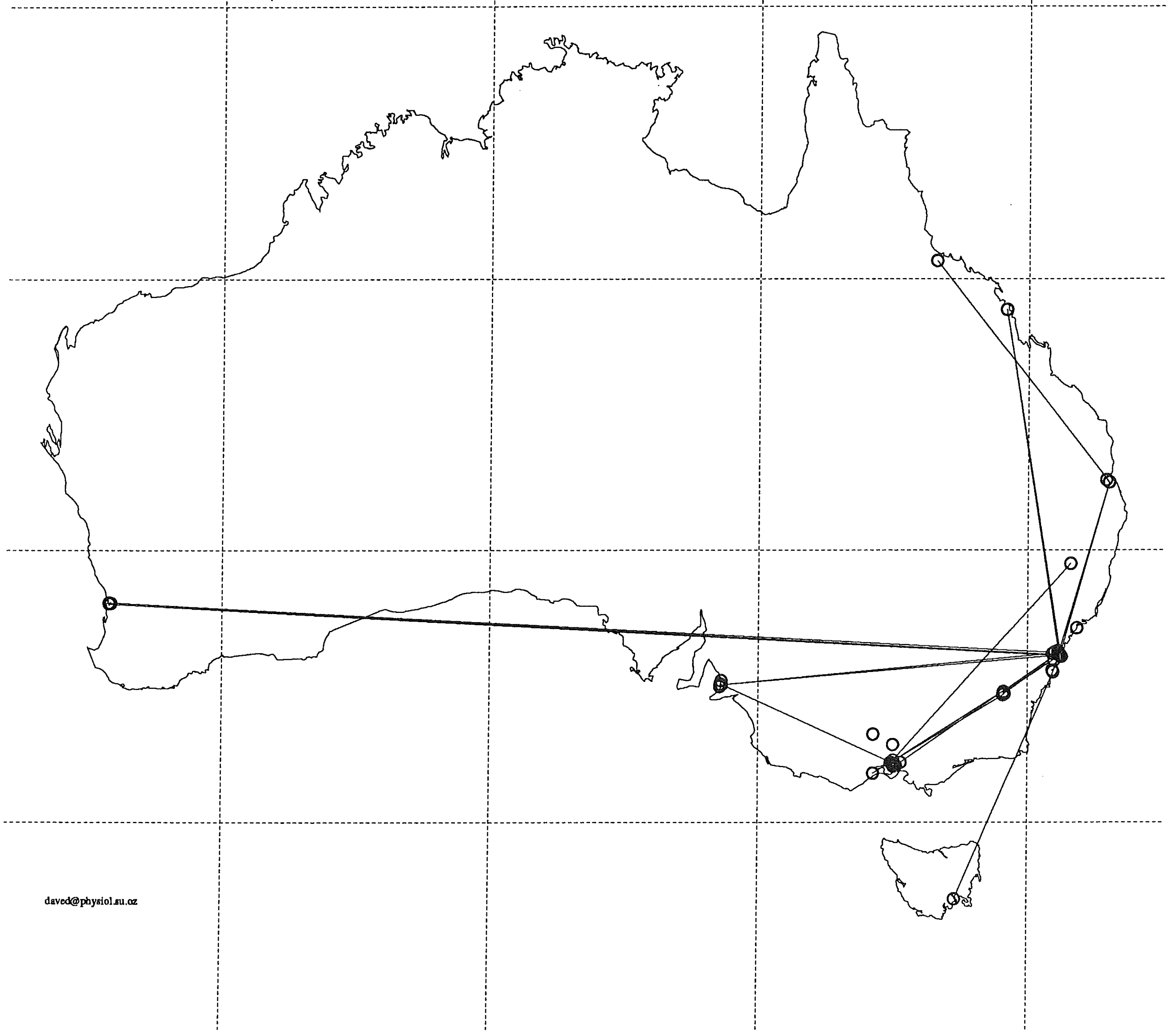
Crocker82.
D. H. Crocker, "Standard for the Format of ARPA Internet Text Messages," *RFC822*, University of Delware, August 1982.

Murai85.
Jun Murai and Tohru Asami, "A network for research and development communities in Japan - JUNET," *Proceedings of PCCS*, pp. 579-588, KAIST, Seoul, Korea, October 1985.

Postel82.
J.B. Postel, "Simple Mail Transfer Protocol," *RFC821*, SRI International, August 1982.

aus.map

circles are nodes, lines are news links



Review of Networking by CSIRO Division of Information Technology

Dr. Trevor Hales and Dr. Ian Richards

As those of you that were at the Canberra meeting of AUUG will know, we have carried out a review of networking within the academic and research community of Australia. The report has recently been finalised and as the Unix community and ACSnet are a significant part of this community, it seems appropriate to present parts of the report through the medium of this newsletter. What follows is edited highlights from the report, a full copy of which may be obtained from the CSIRO Division of Information Technology, 55 Barry Street, Carlton, 3053, Phone (03) 347 8644, ACSnet hales@ditmela.oz.

INTRODUCTION

Computer networking is a basic and essential tool of any research in that it facilitates the efficient exchange of ideas and information among colleagues working in related fields both nationally and internationally. Unfortunately this fact has not been well recognised within the Australian research environment, especially amongst funding bodies who have failed to support the establishment of academic and research networks as has occurred elsewhere in the world.

Instead, individual groups of users with common interests have had to establish their own methods of communication on a largely ad hoc basis. The only substantial networks to emerge have been CSIRONET and the Australian Computer Science Network (ACSnet). A number of other smaller "networks" are also in existence. This is in contrast with the British scene where the centrally funded Joint Network Team established and continues to fund the Joint Academic Network (Janet) amongst all academic and many research establishments. A similar situation exists in Europe with the establishing of RARE (Associated Networks for European Research). The United States has seen the establishment of a number of large and interconnected networks such as the Defence supported Arpanet. More recently the National Science Foundation has become an active participant in the establishment of research networking by funding the establishment of a super computer network within the USA.

The existing networks in Australia serve their users well in most cases and may be seen by many as being adequate for our relatively small academic and research environment. Indeed, the concept of a single unified network may be considered by some to be too restrictive and inflexible, and at best rather idealistic. However, the networking of Australia should be appraised in the context of the world academic and research community. Thus, it is important that a researcher can reach all those colleagues and other resources from his workstation in a straightforward and consistent way. Without doubt, the concept of a unified network has many difficult problems which must be addressed before it can be successfully implemented.

A unified network has some obvious advantages. Perhaps the most important is the potential interchange of ideas between different groups of users for whom communication had been virtually impossible due to lack of knowledge of each other's existence. Another advantage of a unified network is the minimization of effort in network planning and management, equipment and software evaluation and procurement, and in setting up peculiar inter computer gateways when some interworking is required.

While a number of fragmented networks have existed in Australia for a number of years, only recently has the possibility of a unified approach to networking across such a wide environment become a possibility. This is due to the currently emerging and stabilising International Standards for Open Systems Interconnection (OSI). The appearance of OSI products from many vendors within the next year or two now seems a certainty and they form the cornerstone of a unified approach to network planning and development.

CURRENT ENVIRONMENT

In reviewing current activities, we have sought to identify ''cultures'' or approaches to networking that may be using common methods or protocols. In some cases a particular culture may span a variety of types of organisation while in others they may be restricted to a particular category of institution or user interest group.

ACSnet

ACSnet is a loosely coupled network of machines throughout Australia that use a particular set of networking facilities known as SUNIII (Sydney University Network Version 3). There is no central administration or funding of the ACSnet, and for reasons that will be explained in a moment, none are really necessary.

It is somewhat difficult to know how many nodes are on ACSnet, but estimates in excess of 300 appear reasonable. All true ACSnet hosts run the UNIX operating system. Some non-Unix machines are able to attach as foreign nodes with limited functionality. Hosts exist in almost all of the areas with which this review was concerned, including Universities, CSIRO Divisions, Institutes of Technology, Telecom Research Laboratories, and a number of private research and development organisations as well.

The report then discussed the functionality of ACSnet. Some significant comments included the following.

ACSnet is a store and forward message handling network and should not in any way be confused with networks like Austpac or CSIRONET. The latter provide a network connection between nodes which can then use the connection for various purposes including remote login or file transfer. ACSnet relies on the provision of a network service between certain machines, although the store and forward capability means that a message can be sent between nodes which are not directly connected.

It does provide multiplexing facilities across network connections which may themselves be simple asynchronous lines, Local Area Networks, Austpac or CSIRONET. In this sense, ACSnet provides facilities equivalent to Layers 4 to 7 of the well known OSI Reference Model. However, all of its protocols are entirely non-standard. Most network connections are intermittent, eg. an Austpac call or a session on a dialup modem.

ACSnet Benefits and Problems

The consequences of this style of networking are that an extensive network can be easily and cheaply established without the need for expensive interfaces or software. Great advantage can be taken of the cheap bandwidth available within local areas using dialup modems and over CSIRONET in the wider area. Modem time can be effectively utilised by building a tree like topology of periodically interconnected sites. However, there are a number of significant disadvantages to this style of networking.

In particular, hosts are very dependent on other hosts that form the path to any destination to or from which they wish to send or receive messages. Intermediate hosts and links may introduce unpredictable transit delays due to such problems as congestion on a link or failure of a host or link for some period of time. An intermediate host may also cause the complete loss of messages in extreme cases. What is most disconcerting to the user is that under normal circumstances they have no way of knowing whether a message has reached it's destination or whether it has been delayed somewhere along the way. This is despite the best efforts of the ACSnet software to deliver a message or return a failure to deliver message, but if a link fails for a period, not even the failure message can get through.

The report then focussed on the News distribution aspects of ACSnet and examined the advantages and problems of news systems. The problems include the impact on network throughput of large quantities of news and the lack of control of news content and quality.

CSIRONET

The report discusses some of the history of CSIRONET and the development of the current network. Most readers of this Newsletter will be familiar with the communications aspects of CSIRONET, but perhaps not with its mail system.

In the early 1980s CSIRONET developed their own network mail system. This facility is provided by a particular host on the network. When a user logs in to any host on the network he is informed on any new mail that has arrived since the last notification. The reading and sending of mail messages requires that the user logs on to the mail machine. The centralised nature of this approach is a major weakness compared with a fully distributed mail service. Currently, there is no gateway from any of the other mail networks to this service. However, a number of ad-hoc systems have been produced where a pseudo terminal logs into the system to send and retrieve mail.

And with repect to CSIRONET's use of non standard protocols:

For many years CSIRONET had used their own unique protocol for the network layer. In the last three years the computer communications group of CSIRONET have embarked on the developed of X25 DTE/DCE interface for their micronodes. This software has been developed to conform to the 1984 standard and to date has been used for limited field trials. One of the first major applications of the revised network interface will be to provide access from the many CSIRO divisions to the new library machine. In addition CSIRONET is committed to migrating their existing network to this international standard interface as soon as possible. This migration will allow the many CSIRONET users to make use of the significant of software already available and using an interface conforming to the X.25 standard.

Spearnet

Spearnet (South Pacific Educational And Research NETwork) is an initiative by a number of University Computer Centres to establish a network amongst their facilities with the intention of providing facilities such as message transfer, file transfer, remote login, and job entry. The concept of Spearnet began in 1984 when a number of University Computer Centres made application to Digital Equipment for support of a pilot network, since Digital had supported similar proposals in the USA. During 1985, various discussions were continued and the idea was floated at a meeting of Directors of University Computer Centres. A meeting of the Spearnet management committee was held towards the middle of 1986. During this meeting it was decided to use the Coloured Book protocols, developed in the UK to support their academic and research network called JANET, as the initial protocol suite for Spearnet. The Coloured Book software was readily available to support such protocols on all of the relevant machines and thus a vendor independent network could be established quickly and with a high degree of confidence. It also has the advantage of using International Standards, namely X.25, for the lower layers, and thus the ultimate migration to the complete set of OSI standards would not imply a complete abandonment of the investment. The major interconnections were planned to use the public X.25 network, Austpac, rather than a private X.25 network as is used by JANET.

Soon after this, a number of sites had obtained the relevant Coloured Book software and were communicating with each other. Spearnet is slowly increasing in size as more sites join the network. To our knowledge, only four academic institutions in Australia (plus most of the academic sites in New Zealand) are using Coloured Book software. Despite this, the majority of Computer Centres appear to have a commitment to partake in a national academic and research network.

Coloured Book Protocols

The UK academic and research community has developed its own set of communication protocols for its own private wide area network. These standards are either called the Coloured Book or Rainbow Book series. When these standards were commenced there was an urgent need to provide file transfer and job submission between the academic sites. However, ISO had only just formulated the OSI reference model.

Thus the UK community developed its own set of interim standards for most of the applications specified in the OSI model, namely mail exchange, file transfer, terminal access, and job submission. These have been implemented on a wide variety of systems as a result of grants and contracts from the Joint Network Team in the UK.

Other Private Networks

A variety of other private networks have been established over wide areas either between institutions with geographically dispersed campuses or by groups of institutions with a desire to share certain resources.

A typical example of the former is the DECnet based network established by the South Australian Institute of Technology between its three campuses located at The Levels (a northern suburb of Adelaide), central Adelaide and Whyalla. The network is based on leased Telecom lines and Digital's proprietary DECnet protocols.

An example of the latter is VICNET, a terminal network among Victorian Colleges of Advanced Education and Institutes of Technology. It is based upon intelligent switches which provide port contention, queuing and data concentration between switches. Switches are interconnected by leased lines and via statistical multiplexors. As a terminal network, it is quite effective, but relatively little host-to-host communications can be supported.

Libraries have already become significant users of networking both for searching remote databases and catalogues, and for day-to-day activities such as inter-library loans. In Canberra, the Australia National Library, has a central facility for the storing and retrieval of the catalogue holdings of both its library material (books, periodicals etc) and those of the various associated institutions within Australia. The library has publicly declared its intention to use the OSI protocols for the host-to-host communications as and when they become available. This posture is very much in keeping with the activities being undertaken by libraries overseas.

University libraries are extensive users of the Australian Bibliographic Network operated by the National Library. They are also heavy users of overseas communications facilities in accessing various databases, primarily in North America and Europe. In addition, these libraries are increasingly using electronic mail for other library activities such as inter-library loan requests. A variety of mail systems are in use including OTC's Minerva, Telecom's Telememo and some commercial mail services.

Within CSIRO, there is a central library within the information resources unit of the Bureau of Information and Public Communication. This unit is currently developing CLINES - the CSIRO library network system. This system is planned to incorporate the various library functions and services, eg. cataloguing, inter-library loans, onto the one centralized host machine. In addition there are libraries associated with each of the forty odd CSIRO divisions. Each of these libraries will be able to have terminal access to the centralized host for examining the holdings of the other divisional libraries. These libraries also use the various mail systems for inter-library loans from their peers.

Public Facilities

OTC and Telecom both offer mail services, namely, Minerva and Telememo respectively. These two services, like CSIRONET Mail, are centralised services which require users to connect to the mail system. They do not permit interworking and thus exchange of mail between each others users is not possible. However, its should be noted that both organisations have declared their plans to interwork with each other.

CRITICAL ISSUES FOR THE FUTURE

In this chapter we attempt to identify and discuss several of the most critical issues and problems that emerge from the current environment and that require resolution in order that future networking can be developed in a successful and co-ordinated manner.

Functionality

The functionality of any future network will be dependent upon the compatability of products and the provision of gateways where compatibilty is not possible. Compatability will be ensured only by adopting certain standards and thus, most of this section is concerned with International Standards for computer communications, relevant vendor products compliant with those standards and the provision of gateways during the period of migration to those standards.

That we should adopt International Standards for communication protocols is not, we would contend, open to debate. We must, however, like so many other communities throughout the world, find a migration path to OSI (Open Systems Interconnection) that is as painless as possible. This requires some caution, a thorough understanding of the standards, and careful evaluation of vendors products as they emerge. The UK academic community are currently considering the problem and have circulated a publication "Transition to OSI Standards" for the migration from the colour book series of protocols to the emerging international standards.

Current Status of OSI

The current status of OSI standards is often misrepresented. There is no doubt that certain sub-sets of standards are well progressed and relatively stable. This fact is supported by the existence of a number of OSI functional products and announcements for the release of many more in the next twelve months. Despite this, some standards are clearly a long way off and it is vital when planning a network to understand what OSI can do now, what it will be able to do in a few years and what it may never be able to do at all.

Without doubt, the standards for the lower layers are now well established and further developments in these areas will primarily be driven by new technologies at the lowest layers, which will undoubtedly arrive in the form of ISDN. For our purposes, their is really no alternative to the use of X.25, at least in the wide area, and once more products appear, OSI Transport must be adopted.

The top three layers of OSI will often be bundled together into an application product and it it convenient to consider them as such, even though message handling, file transfer and virtual terminal may all use much the same parts of the Session Layer. The significant fact about developments in this regard is that message handling in conformance with CCITT Recommendation X.400 has gained rapid and widespread acceptance with a significant number of products appearing already. Ironically, these standards are not yet strictly part of OSI as ISO is only now working together with the CCITT on the latest version of X.400 due for release in 1988.

File Transfer, Access and Management (FTAM) is the OSI Application Layer standard nearest to completion, and should indeed achieve IS status during 1987. However, it is recognised as a large and complex standard, full implementations of which are not expected to appear quickly. What has given it some impetus is its inclusion in the MAP Functional Standard. This has resulted in a number of early partial implementations.

Virtual Terminal has continued to be a problem for OSI, and indeed for packet switched networks in general. The VT standard in its basic form is well progressed, however more sophisticated versions are some years away, and thus we can expect the triple X (X.3, X.28, X.29) CCITT Recommendations to be around well into the future.

Another significant area of OSI is Directory Services. The critical nature of these standards to the success of OSI was well recognised and they have been progressed at an extraordinary pace, although International Standard status is still about two years away. Similarly, management of OSI systems is progressing, albeit a little slower, and thus these standards cannot be considered in the immediate future.

It is worth emphasizing at this point that OSI has to do with the communications among "Open Systems", which are by definition those systems which are able to interwork using international standard protocols. These systems will almost always be heterogeneous. It is thus unreasonable to expect OSI to provide the level of system integration that users are coming to expect of systems on a LAN, eg. transparently sharable file systems, process migration, etc. That is not to say that OSI cannot work in harmony with such systems.

Functional Standards and Support for OSI

A well recognised problem with OSI standards is their complexity and myriad of options, undoubtedly resulting from a desire by the standards committees to please all parties. One potential consequence of this is that different OSI implementations might not be able to successfully interwork due to the different choices of options chosen. Conformance issues such as these are receiving considerable attention but resolution of the problems are some way off.

Pragmatic solutions to these problems have appeared from some perhaps unusual quarters, indicating the strong commitment by some organisations to OSI. In the United States, General Motors have embraced the standards as a way of retaining competitiveness with the Japanese and developed a so called "functional standard" defining subsets of OSI for manufacturing applications. MAP, the Manufacturing Automation Protocol, includes a cut down version of FTAM at the application layer. Another significant American corporation, Boeing, have developed a similar functional standard for the "office" known as TOP, the Technical Office Protocol.

In many cases the support has come from a group of companies. SPAG, the Standards Promotion and and Application Group, is just such a group. It consists of twelve information technology companies from the EEC who are engaged in both the monitoring of the emerging international standards and facilitating the interworking of implementations from various vendors as they become available. The European Economic Community has other bodies interested in the standardization process for communications. CEN/CENELEC, two standardization bodies that have development a functional profile for the harmonization of implementations.

COS, Corporation for Open Systems is a US initiative to achieve the reliable interoperability between the available communications products in a multi-vendor environment. Although COS has only been incorporated since January 1986, its participants already include almost every major computer vendor in the USA, including IBM, Digital, Burroughs, Wang and many others.

In Australia, the major equivalent initiative to these overseas organisations is the recently launched National Protocol Standards Centre (NPSC). It is supported by Telecom, OTC, the Australian Information Industry Association, the Department of Industry, Technology and Commerce, the Victorian State Government and ourselves. It will be involved in raising the level of awareness of OSI among Australian industry and providing support to companies involved in using or developing OSI products.

Vendor Offerings

Many of the vendors see OSI as a means of improving their marketing position and as an ideal solution to the problems inherent in a multi-vendor environment. In reality, provision of OSI products will be a necessity if vendors wish to market to many customers who are already demanding such products. These include most European governments, the United States Department of Defence, and of course many large corporations that are adopting such functional standards as MAP and TOP.

Vendors have responded to these demands for products with varying degrees of urgency. It is perhaps indicative of future trends to observe that almost all manufacturers offer an X.25 capability, with many now bringing their products up to 1984 specifications. X.400, MAP and TOP are being promised by many vendors for 1987 with Digital and ICL already having X.400 products in the marketplace. It is important to note that when an X.400, MAP or TOP product appear, it necessarily means that a complete protocol stack has been implemented to some extent.

Gateways

One of the important activities of networking is to provide a bridge between users who are either on physically separated networks, or using different communications protocols for their applications. Hence gateways appear in two different contexts in the context of networks. In the first case the gateway or relay is provided to link or connect together two separate networks. A typical case is the relay provided for linking the private UK X.25 network, JANET, to the IPSS X.25 network from British Telecom. This bridging function takes place at the network layer as specified in the OSI model.

When users using systems with equivalent functionality but provided by different protocols wish to interact, eg. exchange mail an from ACSnet site with its own mail protocol to a JANET site with a different mail protocol, then we have a functional gateway. The object of the gateway is to provide a mechanism for reliably and accurately transforming between the mail systems. A requirement of this system is that the originating mail message is stored in the gateway before being transformed in to the format of the receiving network and then shipped to its final destination.

The ACSnet user currently has access to a number of gateways. Most of these gateways are for overseas networks, eg. BITNET, EARN and NETNORTH, arpa csnet ean UUCP etc. ACSnet has developed a very simple addressing scheme consisting of user@hostname.network. The mail is automatically routed to the gateways at Melbourne University.

Naming and Addressing

The problem of how a network user or application working on behalf of a user can find another user or application on the network is a long standing problem with networks. It is clearly desirable that users should be able to use symbolic names for other network users or network resources that they wish to contact. Furthermore, these names should be relatively stable and should be capable of uniquely identifying the remote user or resource.

How does a network user find out the name of a remote user or resource? The problem is much the same as trying to find out someone's postal address or phone number - you look on the top of the letter they sent you, or look it up in some directory service. The standards for an Electronic Directory Service are stabilising at the moment and thus, implementations are perhaps a couple of years away. In the meantime, we must rely upon non standard electronic directories, and of course, looking at the return address on messages (this isn't so easy when looking for the name and address of a network resource).

In order to carry out the request by a user to make some form of contact with a named remote user or resource, the communications system must be able to translate that name into a route. This can occur at a number of levels, depending on the extent of store and forward messaging. In some cases the name may map directly to a network layer address, eg. an Austpac address, plus some additional information identifying the remote resource. In the case of mail, the remote resource will be a mail handler and the message will almost certainly contain additional information as to whom the mail should be delivered at the remote end. Where store and forward messaging is used, information carried in the message may allow the remote message handler to determine a subsequent remote message handler to which the message must be passed, and so on until it reaches its final destination. The overall route may be determined on the basis of the destination name before the message is sent or may be determined by each message handler along the route.

The details of how a route is determined from a network name should rarely be of any interest to the network user. One may argue that since the user may ultimately have to pay for the communications costs, that the route should be under their control. In fact, the user should be able to specify a quality and cost of service which, together with the destination name, will be used to determine the route.

Many naming schemes in the past have incorporated routing information into the actual name, the classic example being the Uucp network which used addresses like "host1!host2!host3!resource". The problem with such schemes, other than the obvious unfriendliness of such a name, is that the name is potentially volatile, eg. if host1 severs its link with host2, yet host3 can still be reached via host4, the name changes to "host1!host4!host3!resource".

A further administrative problem is that the uniqueness of names relies upon all host names being unique. Networks have now grown so large that such a requirement is virtually impossible to achieve. The usual way to achieve uniqueness is to have names issued by some naming authority. As this would be impossible on a global scale, it is appropriate for a central naming authority to delegate responsibility for some subset of addresses to subordinate authorities. This leads to a hierarchical control of names which is reflected in the Arpanet domain style addressing which is used by the Arpanet, Coloured Book (albeit with the syntax reversed) and ACSnet.

The naming proposed by X.400 is similarly hierarchical. Users are identified thus: Personal Name, Organisational Unit(s), Organisation, Private Domain, Administration Domain, Country. Each level of name corresponds to a higher level in the naming hierarchy which must ultimately reflect the way in which names are issued or verified. However, it is not possible to immediately infer anything about the route to a user from such a name, although the communication system must be able to do so based on the network configuration at that time.

A point worth emphasizing is that X.400, ACSnet and Coloured Book systems all use hierarchical names. This is indeed fortunate because it allows the translation of a name from one form to another, a necessary aspect of any message gateway between these subnetworks. The details of how this can be done are not trivial and we will not attempt to cover them here.

With such knowledge, it will be possible for network names to be assigned now such that they will not have to alter in the forseeable future (other than for the obvious reasons such as a person changing employers!). Unfortunately the syntactic appearance of a name will look rather different depending on the system being used, but the name will be semantically identical, eg.

```
testperson@psych.bonduni.oz.au        (ACSnet)

testperson@au.oz.bonduni.psych        (Coloured Book)

name="TESTPERSON",
unit="PSYCH",
organization="BONDUNI",
prdomain="OZ",
addomain="TELECOM",
country="AU"                        (X.400)
```

Note that the X.400 does not specify an actual human readable syntax and thus the above is merely an example of a possible representation. The X.400 Administration Domain reflects the requirement that only Administration Domain can be directly subordinate to a country specification. It is possible that the "TELECOM" part may be able to be inferred from the Country and thus omitted.

Providing the Network Service

In order to provide the functionality as perceived by its users, the network requires actual connections to exist (perhaps intermittently) between various hosts. Where an application can utilise a store-and-forward transfer mechanism, connections need not exist between each and every host provided some path exists, possibly via other hosts, between all hosts.

The network service is of particular interest because it largely dictates the performance of the network and it is an appropriate level at which to discuss the telecommunication costs involved with a network.

A network service between some or all hosts in a network could be provided in a wide variety of ways, including Dial-up lines between sites, Leased lines between sites, a Private Switched Network or a publicly or privately provided X.25 network. Each of these methods obviously have advantages and disadvantages.

Connectivity

The extent to which store and forward message transfer is necessary is largely dependent on the overall connectivity of hosts within the network. Clearly a network constructed by interconnecting hosts with leased lines will have minimal connectivity with the unavoidable result that many hosts will be switching messages from other hosts.

The use of dial-up modems can be quite attractive in this regard in that the switching capabilities of the Public Telephone Network can be used to allow a host to connect, at different times, with many other hosts. The shortcomings of dial-up access mostly stems from their relatively small bandwidth and consequent long connection times for any significant quantities of data. The other limiting factor is that a connection fully utilises a modem and an exchange line at each end. These factors combine to effectively limit the overall connectivity achievable using a dial-up modem. While ACSnet makes extensive use of dial-up facilities, careful planning is required to properly utilise "modem time", including the co-ordination of the times of day that different hosts attempt to connect to other hosts.

As data rates achievable through the Public Switched Telephone Network increase, these limitations may become less significant. This will happen to a small extent in the short term as 9600 bit/sec modems become available and then more significantly in the longer term as ISDN is introduced offering a 64Kbit/sec data channel. However, at that time the economics will almost certainly change also.

Most of the connectivity limitations mentioned above are overcome when a packet switched network is used to provide the network service. Because each site has a physical connection into the network and multiple virtual connections can be accommodated on that physical connection, the overall connectivity of the network is extremely high. Performance is a separate issue.

An X.25 packet switched network could be implemented privately or a network provided publicly could be used, ie. Austpac. The implementation and administration of a private X.25 network is without doubt a massive task and requires centralised funding of leased lines and switching nodes.

Cost Effectiveness

It is difficult to make meaningful comparisons of the cost effectiveness of various transmission techniques. Perhaps the most difficult aspect is the extent of capital and rental expenditure required and whether such expenditure is for a central facility or just for individual sites that wish to attach to the network.

The costs of using leased lines either for host-to-host connections or in the implementation of a private X.25 network would involve substantial central costs including installation and rental on lines and in the latter case, purchase of switches. As there are no volume or time related charges, the economy of leased lines in comparison with services which are charged by time or data quantity depends on the level of utilisation. We have not tried to do this analysis in detail at various levels of traffic as we believe a centrally funded network is inappropriate and inflexible in the current environment.

Of greater interest are the costs of X.25 traffic over a public network (ie. Austpac) or semi-public network (ie. CSIRONET) in comparison with the use of dial-up modems. The fact that local phone calls in Australia are not charged by time is an anomaly which gives such calls a distinct economic advantage. We are of course unable to predict for how long into the future this situation will exist! Thus, there is no point comparing Austpac or CSIRONET with local calls; the latter will always be more economical!

Over longer distances, and we will take the Sydney to Melbourne case as a typical example, the comparison becomes important. There are of course many factors that come into play due to the different nature of the mechanisms. For example, with a phone call you are paying for a full duplex channel with a capacity of say, 2400 bits/sec each way (new modems may provide up to 9600 bits/sec each way). With a packet switched network, the charges are essential based on the quantity of data transmitted, irrespective of the direction or the duration of the connection. There are connect time charges in Austpac but provided a connection is reasonably well utilised (ie. not an interactive terminal session!) they are insignificant in comparison with data volume charges.

An ACSnet connection over a 2400 bits/sec modem passes about 215 characters per second in each direction, allowing for some protocol overhead, retransmissions due to errors, etc. At Off Peak STD rates (10.00 pm - 8.00 am) this would give you about 172Kb for $1. In reality, there is rarely an equal quantity of data to send in each direction, so you could get as little as half of that quantity, ie. 86Kb for $1. Further, most current modems operate at 1200 bits/sec.

Over Austpac, again at Off Peak rates (6.00 pm - 8.00 am), $1 will buy you about 115Kb of data transmission, irrespective of direction. This appears to make Austpac look quite reasonable, although it must be remembered that Austpac charges are distance independent and thus Off Peak STD will match or better Austpac over shorter distances such as Sydney - Canberra.

The fixed costs must also be mentioned. A standard telephone costs $17.25 per month in rental while a 2400 bits/sec Austpac connection costs $275 per month. The significance of that difference largely depends on the level of traffic over which it can be apportioned. Further, a modem must be purchased or leased whereas the Austpac rental includes the modem. A significant development in the near future will be the offering of an X.32 dial-up X.25 service by Telecom which will hopefully allow many sites to achieve X.25 packet mode connectivity without such a heavy monthly rental.

The conclusions one can draw from this are that over a long distance, Austpac is acceptable in comparison with dial-up, but over shorter distances and especially in a local call zone, dial-up will be cheaper.

Choosing Quality of Service and Cost

Unfortunately the cost effectiveness of different methods of transmission are not the only factors affecting their choice. Looking at the extremes, Austpac can potentially offer a high level of service with almost immediate delivery and high reliability. Dial-up connections generally tend to be less reliable for one reason or another. Where store-and-forward is used over a series of dial-up connections, as with many ACSnet transfers, the transit times can become quite long. In some cases the time delay can be unacceptable, when one or more nodes or links are unavailable.

Further, charges for both dialup and Austpac vary with the time of day. Thus, one can save a great deal of money by using the half price charges late at night, but that may be unacceptable to a user who would like their mail delivered almost immediately.

It is clear that if a user is going to be charged for a service, there should be some opportunity to select the cost/performance level desired. For example, a user sending a mail item which will be transferred over Austpac should be able to indicate whether it should be sent immediately (during the day) or held until the evening when charges are half. The situation is similar where in a local area, a message could be sent quickly via Austpac or forwarded via a series of dial-up links with the commensurate longer delays. Further, a user may wish to avoid a store-and-forward transfer for security reasons so that there is no opportunity for system administrators on intermediate systems to view the message. It is unfortunate that current messaging systems do not provide users with such choices.

Administration

Opinions on the need for some central administration of a network vary considerably. In the ACSnet community, which is in some senses a counter-part of the UUCP community in the USA, there is a feeling that central administration is almost totally unnecessary. These kind of networks really emphasize the fact that a network exists because different sites agree to communicate with one another using agreed upon protocols at all levels. Absolutely no central administration is required to form such a network since becoming part of the network simply involves getting some software and finding some other friendly site already on the network who will let you connect to them, in whatever way is appropriate for your proximity, traffic levels and budget.

There are some dangers with this ad hoc network development. For instance, no-one is really considering the proper engineering of the main traffic trunks on the network. In the case of ACSnet, it means that no-one will seriously do anything about the lack of proper bandwidth between Melbourne and Sydney, simply because it is not their site's problem. The problem is really felt by the hosts on either end of the trunk.

The level of centralised network administration does, however, depend very much on the type of network service that is used. As was discussed in the previous section, if a private X.25 network were to be established, substantial central funding and administration would be required. On the other hand, the use of a separately provided network service from either Telecom or CSIRONET removes much of the need for central administration, since the network provider ensures that the network is properly engineered and maintained to an acceptable level of performance and reliability.

On the assumption that such an approach is viable, the remainder of this section examines some of the other problems that have implications for the overall network and thus potentially require some centralised administration.

Network-wide Services

Certain services will at some stage be required to service all users of the network. The most obvious example is a directory service. This is actually a special case of a database service of which there could conceivably be many in the future.

Another class of service is the gateway to some other network or mail system. An example might be as general as the current University of Melbourne gateway to various overseas mail systems. Gateways will also exist between various protocols within Australia which effectively form sub-networks. Thus, there may be one or more gateways that pass X.400 mail to Colourbook and vice versa. Similarly, gateways from X.400 to the existing ACSnet services are inevitable. Even when ACSnet changes to X.400, there is still likely to be a need for a store and forward facility, which is in effect a gateway, to pass mail to sites linked by dial-up modem to sites that have an X.25 connection.

While it is recognised that such services could be provided on a largely ad hoc basis, such an approach is likely to lead to various problems including lack of capacity, inequitable charging etc.

Accounting

The costs of developing and running a network include the capital cost of equipment and software, the transmission charges, and the costs of running a service such as a gateway or database which include all the usual operational costs associated with computers. In the absence of a fully centrally funded network, there will be a need to account for these costs and so far as is possible, pass them back to the end users of the network and its services. The end user in this context is a site such as a University; handling of accounting within a site is assumed to be a purely local issue, although accounting software may need to provide for local policies.

The major accounting problems arise in association with the exchange of electronic mail over network links which cost money, eg. an X.25 virtual circuit. One customer of the network service provider will normally be charged for that circuit. Normally it is the initiator of the call unless reverse charging is used and accepted. With most message protocols it is possible to exchange messages in both directions across that circuit with the obvious problem that whoever is paying may be paying for someone to send them mail. This has always been a problem with incoming overseas mail, the transmission charged for which have normally been paid at this end - a most undesirable situation!

The solution is obvious of course - only pass mail in one direction on a circuit, presumably from whoever is paying. So why have protocols such as ACSnet provided for bi-directional mail exchanges? In the case of ACSnet, the protocols have been designed to operate primarily over dial-up lines where it is vital to make as full a use as possible of the available bandwidth in both directions. However, the protocol can be used in a "one way" mode. This consideration does not apply over an X.25 network where you generally only pay for the data passed, irrespective of direction. X.400 does allow for bi-directional transfers but only one message can be moving in either direction at a time. Some implementations will allow you to prevent a circuit from being "reversed".

We believe that all other needs for accounting fall into the category of a network host providing some service for a network user. For instance, the operator of a database accessible from the network (and this includes a directory service) is providing a service which could be charged for on an individual use basis. Similarly, a host which is prepared to accept traffic from a dial-in host and to forward it across the network in a store-and-forward manner is providing a service to that host just as if it were a normal human user of that host. Thus, we believe that ACSnet style traffic can be accommodated into a suitable accounting regime without tremendous difficulty.

Network Planning and Development

Even where the network service is provided, the planning and development of the overall network structure and approach requires considerable effort. Indeed, this review is to a large extent, an example of such planning.

Many issues will require planning and consideration at a more detailed level in the future, in a manner which cannot reasonably be left to individual sites. The most obvious example is the choice and adoption of various standards. The evaluation of products which are appropriate to some proportion of sites is better handled centrally as is the identification and perhaps funding of special product development projects where vendor products are not available.

Other issues include the recognition for the need for certain network wide services and perhaps some funding to assist one or more sites to establish such a service, even if it's use is to be charged for. Again, the obvious example is a directory service which may require some capital funding for product purchases or development. If a site wishes to establish and offer such a service as an economically viable venture, well and good, but if not, some central planning is required to ensure that such a service is available.

A final area where central planning is desirable is the overall topology of the network and ways of minimising the telecommunications costs while maximizing reliability. Where direct connection across an X.25 network is used, topological considerations are less important. However, proper reliability will almost certainly rely upon the existence of some alternative links, perhaps via dial-up lines, satellite links etc. Proper planning of alternative links on major paths is an essential central planning function.

Centralised Funding

It would appear that some degree of centralised planning and development is at least desirable, perhaps even essential to the success of a national academic and research network. The problem of funding such centralised administration is difficult primarily due to the diversity of types of organisations involved. Central funding among even a particular sub group of the academic and research community, say the Universities, is difficult enough due to the high level of financial autonomy given to such individual institutions.

The situation is compounded by the existence of a quite significant number of semi government or entirely private organisations within the research community, some of whom already have interconnections via ACSnet or CSIRONET. There is generally strong support for interaction between industry and the traditional government academic and research organisations and so network interconnections must be considered to be desirable.

CONCLUSIONS

Our objective in undertaking the review was to develop a blueprint for the future development of Australian networks so as to improve the connectivity and communicability of Australian colleagues to peers both within Australia and overseas. This takes the form of a set of recommendations for essential policies, overall strategies, and both short and medium term (3 to 5 years) actions for the future development of networking within the Australian academic and research community.

Essential Policies

An initial observation concerning networking within Australia is its lack of acceptance and appreciation of the benefits for Australia of using a network. In the overseas environments there is substantial support for networking initiatives.

RECOMMENDATION 1. We recommend that all sectors of the Australian academic and research community provide clear recognition of the fundamental role played by computer communications in their activities and that policies give a high priority to the use and development of modern communication techniques, including participation in national and international networks.

As was recognised in Chapter 3, the adoption of International Standards for communications protocols is essential. The benefits of adhering to the OSI Standards have been well presented in many publications and forums as well as elsewhere in this report.

RECOMMENDATION 2. We recommend that all sectors of the Australian academic and research community show a clear commitment to the use of Open Systems Interconnection standards in the future and avail themselves of products at the earliest possible time.

In the first instance, such a commitment may take the form of policy statements, which may be seen by many as "words not actions". However, the usefulness of such a policy statement from major segments of the community should not be underestimated. Similar statements from comparable communities overseas have had a significant impact on the progress of communications work, especially on the development and release of communications products by computer vendors.

Certainly such a commitment must ultimately lead to actions in terms of developing decisions for the establishment of useful networking environments and the purchasing of the appropriate networking equipment and software. Following the UK model, we suggest that any future computer tenders should contain clauses referring in particular to the provision of OSI products by the sucessful vendor.

The benefits of the adoption of both Recommendations 1 and 2 will ultimately rely upon the widespread use of network facilities by ordinary members of the academic and research community.

RECOMMENDATION 3. We recommend that those responsible for the provision of network facilities within the Australian academic and research community give particular attention to the education in the effective use of network facilities of all members of their part of that community.

Goals and Strategies

During the early years of the next decade, we expect the technology of Integrated Services Digital Networks (ISDN) to have a significant impact on communications within the entire community, including the community under examination in this review. With the acceptance of ISDN we expect that exchange of electronic messages, access to databases, compute servers and many other services will become commonplace and economical for a wide variety of users. The vision of a global integrated services network comparable in penetration to the current telephone network will become a reality.

Our goal is to develop networking within the academic and research community in a manner which will allow us to take the maximum advantage of these new communication techniques and services as readily as possible. Many difficulties present themselves in achieving this goal. The incompleteness of many communication standards and the consequent lack of many products is one difficulty which we expect to diminish significantly during the next two to three years. High capacity, cost effective connectivity between all communicators as offered by ISDN, is still several more years away from being the norm. Hence in the meantime we must grapple with alternative methods which are riddled with compromises, charging anomalies and varying degrees of quality of service.

Thus, we must choose strategies which maximise the effective use of current and emerging communications techniques and products while attempting to steer a convergent path with the functionality that we expect during the next decade. To a very large extent, our policy on adoption of OSI will ensure a convergent path to ISDN.

Strategy - Connectivity

The connectivity of the sites within the Australian environment has been unplanned and followed a very ad hoc evolution. In many cases the selection of the "wet string" (the mechanism used for connections) has been decided on purely economic grounds often with a complete disregard for the desired quality of service.

STRATEGY 1. Communications between academic and research sites both within Australia and overseas should make use of a variety of connection technologies and services in order to take advantage of available economies while providing the desired quality of service.

We are in effect saying that before ISDN appears, we do not believe that a single method of interconnection exists that can satisfy the criteria of economy and quality of service in all situations.

Strategies - Functionality

The OSI model contains 7 layers from the physical to the application layer. In this section, we focus on the various possible applications that are provided by the "application layer". mail interchange. There is little point recommending applications that have no stable international standards available for developing their implementation.

STRATEGY 2. Emphasis should be placed upon the functionality provided by OSI standards that are the most mature and which are of significant use to the greatest number of users. Consequently, we expect the greatest emphasis to be placed upon electronic mail based on CCITT Recommendation X.400. Essential functionality unable to be met by OSI products should use products which have a migration path to OSI. Use of remote login to access services should be avoided where possible in favour of the use of effective host-to-host communications.

In stating these strategies, we are taking due regard of the less than ideal or complete state of communications standards and products at this time. Thus, we recognise that compromises will be necessary. Adoption of a single set of protocols, for instance, is simply too idealistic in the current climate. Similarly, while the ideal way of accessing a remote database is via an intelligent database enquiry protocol, such accesses will necessarily involve remote login for some time to come.

Administration

In the last chapter, we identified the need for a degree of centralised administration of the network even though the entire network need not be centrally funded. Any central administration will require some level of funding although we recognise that it should be as small as practicable. We also perceived some difficulties in bringing a wide variety of types of organisations together on an equitable basis. In order to take account of these conflicting factors, our strategy is to administer the network via a high level "umbrella" association while allowing individual organisations and groups of institutions as much autonomy as possible.

RECOMMENDATION 4. We recommend that an association be established, similar in concept to those found in overseas situations, e.g RARE or JNT, to coordinate central network administration functions. Some of the important functions are planning, evaluation of standards and vendor products, identification and possibly funding of special development projects, liaison with overseas organisations, liaison with the indigenous network providers such as Telecom, OTC and CSIRONET, bargaining power with the various providers of services, co-ordination of schemes for naming and addressing domains. Membership of the association should be by subscription which will allow for the funding of its activities and the establishment of an appropriate administrative structure. This association will harmonise the various communications initiatives begun by the computer centres, the computer science departments, and other like minded groups.

One of the early requirements is to establish the structure and composition of the association. Hence in the short term, it would be desirable for a small steering committee to be established to examine detailed issues such as membership categories, the extent of activities, an appropriate budget and the necessary level of subscriptions.

Funding was and continues to be a thorny issue for the Australian academic and research environments. Hence one quick, but inappropriate solution, would be for centralised funding to be provided for all costs associated with the individual sites. Certainly there is some inequity in the fact that the capital costs involved in establishing an X.25 connection together with the relevant software and then meeting the monthly rental and software maintenance charges are largely independent of the size of a site. However, as noted in the previous chapter, the funding arrangements of the various types of organisations that will be involved in such a network make central funding virtually impossible. Despite this, we believe there is some scope for achieving more equitable funding of the fixed costs.

RECOMMENDATION 5. We recommend that, where possible, groups of institutions with common funding sources, eg. Universities or CSIRO Divisions, consider the central funding of some or all of the capital and fixed recurrent costs involved in establishing an appropriate network connection for all their sites.

The question of transmission costs was also examined in the previous chapter and the problems involved in accounting were discussed. It is possible to make the following general recommendation at this stage, on the assumption that much of the interconnectivity of the network will be provided by a third party.

RECOMMENDATION 6. We recommend that, in general, usage costs be met by the users of the network and not by any central funding.

This strategy is in contrast with some overseas networks, eg. JANET. In these cases, those networks use private links for their interconnection facility, ie. there is no additional cost for traffic, except in terms of performance. Where traffic does cost real money, we firmly believe that a ''user pays'' approach has the benefit of tempering network usage.

The practicalities of the ''user pays'' approach and its implications on accounting procedures were covered in the previous chapter. This leads to a general recommendation for the handling of accounting problems.

RECOMMENDATION 7. We recommend that, wherever possible, the organisation making use of some network service should initiate and meet the costs of the network connection used to gain that service.

This recommendation has a variety of practical implications, some of which were mentioned in the previous chapter, eg. messages should be sent in one direction only over an X.25 connection. Clearly where the intent of the recommendation cannot be carried out, accounting procedures will be necessary to appropriately charge the actual user of the connection. In any case, accounting procedures will be required for other network services such as gateways and databases. Development of accounting procedures where necessary could be appropriately funded under the auspices of the central network administration association.

Connectivity

Given the commitment to OSI and the discussion in Chapter 3 regarding the economics of providing the network service, the starting point for our strategy must be a preference for the use of an X.25 network.

RECOMMENDATION 8. We recommend that the preferred method of interconnection of geographically separated sites be via a properly engineered X.25 network.

The choice of provider for that network must be made on the grounds of economics and performance.

RECOMMENDATION 9. We recommend that the association seeks tenders for the supply of X.25 network functionality and recommends a provider to member organisations based on both the functionality provided and economics of the service.

However, our strategy is to take account of the current economics of alternative methods of communication. The most obvious alternative is the use of high speed modems over the Public Switched Telephone Network (PSTN) while call charges are independent of the duration of the call. Standards are emerging at present in this area.

RECOMMENDATION 10. We recommend that, wherever possible, a low cost dial-up link or series of store and forward links be provided for passing traffic between sites within local or near local call zones. In the immediate future, existing ACSnet facilities will be the primary method of achieving such links. Future developments should make use of X.32 (dial up X.25) facilities both within ACSnet and based on emerging vendor solutions. Future developments should also allow users to make a choice between the more expensive X.25 path and the potentially higher delay, but low cost, path where such alternatives exist.

The remaining communications technology that must considered within our connectivity strategy is the use of the Australian satellite system, Aussat.

RECOMMENDATION 11. We recommend that the distribution via Aussat of broadcast type information, eg. Unix news and other information with widespread readership, be pursued in the immediate future.

The obvious benefits of this approach are the cost savings involved in using a broadcast medium. Obviously this must be weighed against the number of sites that are able, in the short term, to make use of this data and the capital costs to those sites of achieving that ability. However, we have no doubt that this strategy has long term significance.

The overall topology of the evolving academic and research network must be considered carefully and, as we mentioned earlier, this is one area in which an administrative association has a planning role. In order to maintain a high level of reliability and economy, our strategy is to ensure that diversity of major links is achieved.

RECOMMENDATION 12. We recommend that, as the network develops, careful consideration be given to the availability of alternative means of connectivity between major areas and sites. In particular, during the planning phase of the topology we recommend alternative X.25 network providers should be considered as should satellite links.

Functionality

The ultimate functionality of the network will be to offer a wide range of services, many of which have been discussed earlier. Unfortunately, many aspects of the desired functionality will be difficult to achieve for some time into the future. In particular, the sharing of resources between heterogeneous systems becomes more difficult as the type of resource becomes more system specific. We must also recognise that some services will be of interest to almost every network user while others will only be of immediate use to a subset of users. Thus, we must initially look towards functionality which can be provide the greatest benefits to the widest possible user base. Earlier, we stated that not all OSI applications have progressed at the same rate towards International Standardization. We believe those that have reached this maturity should be used at the earliest possible date.

Electronic Mail

Our strategy is to initially place emphasis on electronic mail with a desire to use X.400 products wherever possible. This strategy leads to a number of specific recommendations for the immediate future.

RECOMMENDATION 13. Due to the significant number of VAX/VMS sites within the Australian academic and research community, we recommend that, wherever possible, those sites procure and intercommunicate using the X.400 message handling products available on those systems.

The benefits of this approach are that a significant proportion of the academic and research community can immediately begin to communicate using electronic mail based on International Standards. It also allows users and system operators to gain experience with this long term solution. In the near future, many other implementations will interoperate with the Digital products. In particular, it is reasonable to expect that the mail systems provided by the PTTs (Australian Telecom and OTC) will have systems (Telememo and Minerva) which will conform to the CCITT X.400 standard and thus also become directly accessible from private systems.

Not all such sites may wish to adopt this approach immediately as it necessarily requires considerable expenditure, including the procurement of an X.25 connection. Some cost-saving alternatives exist, including the purchase of the Coloured Book software. It is still necessary to purchase the X.25 software and an Austpac connection. However, it must be realised that the Coloured Book component of this communications system, although relatively inexpensive, is a short term solution and ultimately the X.400 will be necessary. In the meantime, we recognise that this alternative will be attractive to many non-University Vax sites and that one or more appropriate gateways between Coloured Book and X.400 will be necessary.

Another option that some low traffic sites may wish to look towards is the use of an X.32 dial-up X.25 connection when such a service is offered, perhaps in a year's time. This would be justifiable primarily on the grounds of avoiding the substantial monthly rental on a dedicated X.25 connection.

RECOMMENDATION 14. We recommend that sites unable to procure X.400 software at this time, either for reasons of cost or unavailability, adopt the use of alternative non-OSI products for message handling provided the products have a stated commitment to move to X.400. This currently includes the Coloured Book software and ACSnet.

Functional Gateways

There is little doubt that two other major protocol cultures will continue to exist for some time into the future. These are the Coloured Book (CB) protocols imported from the United Kingdom and used extensively in New Zealand and the ACSnet protocols in widespread use amongst the Unix community within Australia.

RECOMMENDATION 15. We recommend that at least one Grey Book Mail to X.400 Mail gateway be established in the immediate future to permit exchange of mail between the different protocol standards.

The situation with gateways from ACSnet to X.400 is slightly more complicated because of the ability for ACSnet traffic to flow over a variety of network links, including dial-up lines. However, the general strategy does not change.

RECOMMENDATION 16. We recommend that a number of X.400 to ACSnet gateways be established in the immediate future. In particular, we expect such gateways to exist in most sites where both ACSnet and X.400 exist.

The requirement for apparently more such gateways reflects both the relative numbers of hosts supporting these protocols and the economy and reliability of forwarding ACSnet messages over ''terminal like'' links within local call zones to a remote X.400 gateway where X.25 could then be used. We do not endorse the use of X.25 for terminal-like connections where it can be avoided and indeed ACSnet over X.25 presents the problem of accounting for two-way traffic. It is thus preferable to gateway into X.400 for transfer across X.25.

RECOMMENDATION 17. We recommend that ACSnet be further developed to utilise OSI protocols with the aim of providing functionality not available from commercially available products. This is likely to include X.400 on some versions of Unix and X.32 dial-up access for carrying X.400 mail.

The benefits of being able to utilise dial-up links to carry X.400 have already been mentioned. In the shorter term, the handling of X.400 messages by ACSnet, even without the use of X.32 or X.25, will greatly simplify the gateway functionality required.

CSIRONET provides a central mail facility for its users. However, this system does not permit mail exchange with other mail systems.

RECOMMENDATION 18. We recommend that a gateway be developed for the CSIRONET mail system using the X.400 international standard.

Network Gateways

The gateway functionality currently provided by the University of Melbourne must be retained, as access to various networks overseas that do not yet conform to OSI standards is essential for our colleagues in Australia.

RECOMMENDATION 19. We recommend that the University of Melbourne continue to operate its overseas mail gateway to the UUCP, ARPAnet and other international networks and that X.25/X.400 access to that gateway be provided in the near future. In addition, we recommend that alternative gateways should be provided to ease the traffic flow to one machine and to introduce some redundancy into the network, eg. a BITNET gateway at the University of Queensland. We further recommend that work be carried out to improve the consistency and simplicity of addressing structures used by users wishing to access incompatible overseas mail networks.

The benefits of improving gateway functionality are the ease of use by inexperienced users and the consequent reduction of addressing failures which themselves can be quite expensive on overseas links.

As with any network service, an overseas mail gateway must be operated on an economic, cost recovery basis by charging users of the service. In the case of overseas mail, a difficulty that needs resolution is the fact that only limited bilateral arrangements exist with similar networks overseas to share transmission costs. Consequently, the majority of mail coming into Australia is paid for by the recipients rather than the senders. In some cases where researchers here have requested information, such a situation is reasonable, but in general, the sender should have to pay for the service.

RECOMMENDATION 20. We recommend that negotiations be initiated with similar overseas research network operators with the aim of achieving a more rational sharing of transmission costs for overseas mail.

Additional Services

The need for Directory Services which allow users to locate the proper address of other users with whom they wish to communicate has been widely acknowledged. Despite the instability of current standardisation efforts in this area, some facilities are required.

RECOMMENDATION 21. We recommend that the feasibility of providing a Directory Service accessible via X.25 be investigated in the near future with a view to establishing such a service in approximately a year's time. Such a service should be consistent with current standardisation efforts.

As with a gateway, the directory service should be operated on a user-pays principle. However, it may be appropriate to fund the development of the service centrally.

Providing other functionality is more difficult due to the instability of standards. Most other functionality relates to specific network services such as a database or compute service. Specific OSI protocols are under development for access to such services, but products are in most cases a year or more away. However, it should be noted that message handling can support many functions other than inter-personal mail, eg. file transfer, document exchange, news transfer, remote printing and even remote job entry, although detailed standards for these kinds of messages have not been defined.

RECOMMENDATION 22. We recommend that in the short term, file transfer and remote job transfer should be achieved using non-OSI methods such as Blue Book file transfer and Red Book remote job transfer from the Coloured Book suite. File and Job transfers using messaging systems, both X.400 and ACSnet, should also be supported with a view to moving towards the OSI standards (FTAM and JTAM) when they stabilise and products become available.

It may appear that this recommendation is simply saying "we must use what is here now because OSI standards aren't complete", but there is a more significant aspect to it. We have consistently argued against the use of dumb terminal access to remote services over packet switched networks. In the absence of suitably intelligent workstations and sophisticated virtual terminal protocols, remote terminal access is neither economical nor functionally satisfactory.

RECOMMENDATION 23. We recommend that sites resist the use of terminal access across wide area X.25 networks using unsophisticated protocols such as the so called "triple-X" ones (X.3, X.28, X.29).

This is in line with one of our earlier strategies for network functionality. It should be noted that in the absence of suitable OSI standards, use of file transfer packages like Kermit are far preferable than remote terminal access and thus deserve support in the short term.

Concluding Remarks

In the conclusion we have tried to develop a blueprint for the evolution of networking within Australia. We are conscious that this document is very much a snapshot of the current state of the Australia networks, the availability of vendor products, and the provision of communications links between various sites. All of the above components will evolve rapidly over the next 3 year period. However, we have tried to take account of likely changes to this environment. In particular for the long term success of networking there must be a fundamental commitment to OSI.

;login:

The USENIX Association Newsletter

Volume 12, Number 3 May/June 1987

CONTENTS

The closing date for submissions for the next issue of *;login:* is June 26, 1987

USENIX THE PROFESSIONAL AND TECHNICAL UNIX® ASSOCIATION

The RIACS Mail System

Matt Bishop
Research Institute for Advanced Computer Science
NASA Ames Research Center
Moffett Field, CA 94035

ABSTRACT

This document describes the configuration of the RIACS mail system, and how the mail configuration files implement this design. The rationale behind the design, as well as the configuration files, is discussed. The paper concludes with a description of features that may be used to debug the configuration files, and some basic guidelines on how to trace problems with configuration files that have been installed.

1. Introduction

Users who send or receive letters over an electronic network deal with only a small part of the mail system. Typically, sending mail involves several levels of services and protocols, summarized in Figure 1. (This model is called *the MHS model* and is from ISO X400.)

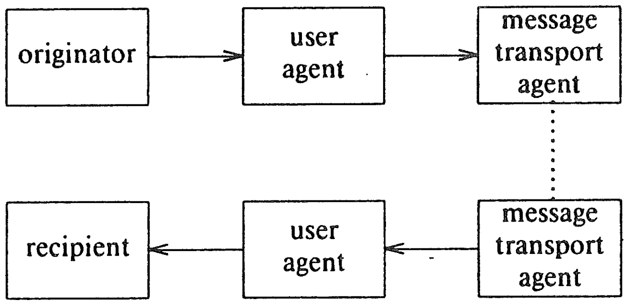


Figure 1. The MHS model.

In this figure, the person who wants to send the letter is called the *originator*. He uses a *user agent*, which is a program that enables him to type and format the letter. The user agent then hands the letter to a *message transport system*. This system is composed of one or more *message transport agents*; each agent accepts the message, determines if the recipient is on its machine, and if so passes the message to the appropriate user agent. If not, it determines which machine it should deliver the message to, and passes the letter on to the message transport agent on that machine. Finally, the letter reaches a user agent on the recipient's machine, and the user agent enables the recipient to read and possibly reply to the letter.

RIACS has many user agents available, ranging from the simple *binmail*(1) [UPM84] to the highly sophisticated *MH Mail Handling System* [ROSE86]. This manual leaves the description of the user agents to other documents; instead it concentrates on the message transport agents.

The next section describes the configuration of the machines at RIACS; the sections after it proceed line-by-line through the gateway and non-gateway *sendmail* configuration files. The final sections discuss how to use *sendmail* [ALLM84a, ALLM84b] to debug a configuration file, and how to install new configuration files. These last sections assume the reader is familiar enough with configuration files that the notation and general layout need not be reviewed; if not familiar with these, the reader should have a copy of [ALLM84b] readily available when reading these sections.

2. RIACS Mail Configuration

The RIACS environment is quite heterogeneous. It consists of a VAX-11/730[†] which serves as a gateway, a Sequent Balance 21000, an Intel Hypercube, a Ridge 32/V, a Silicon Graphics IRIS 3500, and several Sun 3s, all running some version of UNIX, some sharing file systems using NFS and others not. The following table shows the names of these machines:

Table 1. RIACS Computers and their Names

host name	*computer*
icarus	VAX-11/730
hydra	Sequent Balance 21000
cube	Intel Hypercube
daedalus	Ridge 32/V
pegasus	Silicon Graphics IRIS 3500
lavalite	Sun 3 (has file system)
miranda	Sun 3 (has file system; server)
phun	Sun 3 (client sharing *miranda*'s file system)
zeus	Sun 3 (client sharing *miranda*'s file system)
clavier	Sun 3 (client sharing *miranda*'s file system)
dora	Sun 3 (client sharing *miranda*'s file system)

There are also an Evans and Sutherland P300 graphics system, two Apple Macintoshes, and an IBM PC used for financial matters.

Of these machines, *icarus*, *hydra*, *daedalus*, and the Suns are the only ones which allow the full range of mailing facilities. *Cube* allows local mail to be sent, but does not have any facility for sending mail to another machine except via *uucp*. *Pegasus* does not allow any nonlocal mail to be sent; it, the Evans and Sutherland, the Macintoshes, and the IBM PC do not interact with the mail system.

There are a number of configurations that allow mail to be sent to nonlocal or remote machines. One is to require each host to have all the information needed, so when a letter is sent to *csl.sri.com* from *hydra*, *hydra*'s mail router will send the letter directly to *csl.sri.com*. On first blush, this seems to be the best option; however, it has drawbacks. Not every host can transmit all types of mail; for example, *icarus* is the only host that can be used to send *uucp* mail, because it is the only host with telephone lines connected. If some distant site changes its internet address, all machines must have their host tables updated (since some machines, such as *hydra*, do not use domain name resolving); worse, when a configuration file is changed to reflect a change in the way mail is handled, all configuration files must be changed. This leads to problems of mail being caught in transit when the change occurs.

Since all nonlocal mail must be sent through the gateway machine, why not send all nonlocal mail there for forwarding? In this configuration, whenever any letter being mailed has an address on some other machine, it is sent directly to *icarus*. *Icarus* then decides how to send it to its destination. For example, if a letter is sent to *csl.sri.com* from *hydra*, *hydra*'s mail router will send the letter to *icarus*; then *icarus*'s mail router will send the letter on to *csl.sri.com*. This scheme has advantages over the previous one, in that all hosts except the gateway need only know which host is the gateway; if there is any mail on the host for any other machine, it is sent to the gateway, which then decides how best to deliver it. If some distant site changes its internet address, only the host table on the gateway need be changed; the tables on the other hosts may be updated as convenient, but doing so (or failing to do so) will not affect the transfer of mail. One would suspect that this configuration is worse than the previous one, because if *icarus* goes down, no mail can be sent off site. However, since all our network traffic goes through *icarus*, if *icarus* is down, no off-site mail could be sent

[†]VAX is a Trademark of Digital Equipment Corporation.

anyway. The only problem is that intrasite mail will not go from one machine to another when *icarus* is down.

There is one other advantage to the latter configuration. RIACS uses the domain naming system of the ARPANET. As a result, having a central machine provides a very easy way to decide which machine is to be listed as "riacs.edu". All other machines are invisible to off site systems; any letter going out, regardless of the machine from which it originated, has the return address changed to read "*user*@riacs.edu". In essence, we can reconfigure our machines, change their internet addresses, and so forth without having to have people throughout the country update their host tables. Only the address of the machine answering to "riacs.edu" need be kept the same.

For these reasons, RIACS uses the latter configuration. Pictorially, this system is arranged as shown in Figure 2.

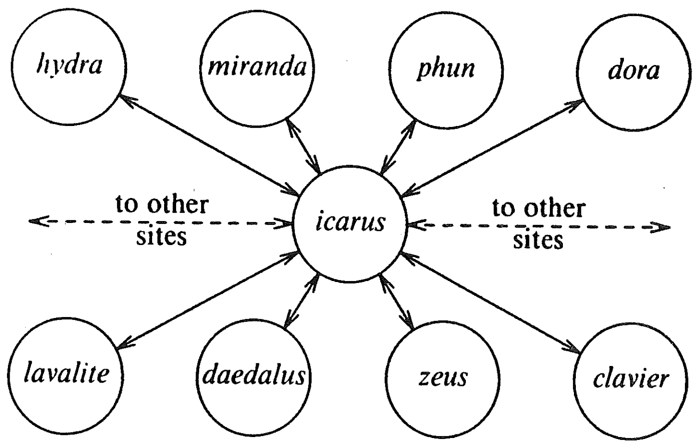


Figure 2. The RIACS Mail Configuration

There are two different configuration files: the first, the *gateway* version, resides on the RIACS gateway and handles routing over a variety of communications media; the second, the *non-gateway* version, resides on all other hosts.

3. *sendmail* Configuration File Preliminaries

This section describes options, macros, classes, mailers, and other characteristics of the configuration files. Rulesets are described in the next section.

3.1. Macros

Certain macros and options are set in both the gateway and non-gateway configuration files. Some are necessary for *sendmail* to work; others are peculiar to RIACS. In this section, those common settings are explained.

The first three macros in the gateway's file set the domain name N, the uucp host name U, and the version Z of this *sendmail* configuration file; in the non-gateway file, the uucp host name is not given but the gateway host G is. *sendmail* also requires six macros to be defined; otherwise, it will not function properly. Table 2 summarizes the definitions of these macros.

Table 2. RIACS Settings for Macros

macro	gateway setting	non-gateway setting
G	*not set*	icarus.riacs.edu
N	riacs.edu	
U	riacs.uucp	*not set*
Z	2.0G	2.0N
j	$w	
e	$j Sendmail $v/$Z ready at $b	
l	From $g $d	
n	MAILER-DAEMON	
o	.:%@!^=/[]	
q	$?x$x $.<$g>	

3.2. Options

The configuration files also set many options for *sendmail;* both the gateway and non-gateway configuration files set the same ones. Table 3 summarizes their settings.

Table 3. RIACS Configuration File Options

option	RIACS setting	option	RIACS setting
A	/usr/lib/aliases	a	10
B	_	d	b
F	0600	g	1
H	/usr/lib/sendmail.hf	o	*set*
L	9	r	1h
Q	/usr/spool/mqueue	s	*set*
S	/usr/lib/sendmail.st	t	PST,PDT
T	3d	u	1
W	*	x	8
X	12		

Some of these options require a bit of explanation.

A,a These refer to the alias file feature of *sendmail*. The alias file is located in */usr/lib/aliases*, and if *sendmail* detects the alias file is being rebuilt it will wait up to ten minutes before deciding the rebuild has failed and initiate one of its own.

B Older mailers allowed the use of blank characters in addresses; this causes all sorts of problems on many systems. This option directs *sendmail* to replace all blanks with underscores '_'. RIACS used to use periods '.', but this poses problems in a domain system; an underscore does not.

d This option instructs the daemon to run in background mode; the daemon will disconnect from the terminal.

F,Q The values of these options are the mode and location of the temporary and queue files used by *sendmail*. It is a RIACS policy that the mode shall be at least mode 0660, and preferably mode 0600; these settings prevent others from reading or altering mail while it is in the queue. On the gateway, this has never posed a problem; but it has on the Suns running NFS, since *root* on a client Sun cannot write to the server's queue directory. Worse, even if it could, the client's *sendmail* process could not read the files it put there! At this point, there are two options: either make the queue files mode 0660 and the queue directory 0777 (meaning anyone can delete queue files, although not necessarily read them), or make each Sun's queue directory a private one (by linking the queue directory to a directory in */private*). At RIACS, the latter is done; on the server, */usr/spool/mqueue* is a symbolic link to */private/usr/spool/mqueue*.

g,u	When *sendmail* spawns a subprocess to deliver mail, it resets the user and group identification numbers to 1 (for *daemon*). This provides a measure of security; if the mailer were running as *root*, a breach of security could result in compromise of the entire system rather than just daemons.
L	All mail coming into and going out of RIACS is logged; the sender and recipient are named in the log file, as is the queue number and other useful information. To reduce the amount of logging, lower this number. Logging is done via *syslog*(3), so to determine where information is being sent, look there.
o	Since many mailers use spaces rather than commas to delimit names, this option instructs *sendmail* to accept and handle such lists.
r	This option instructs *sendmail* to time out after 1 hour during a connection to send mail. Supposedly, this should "never happen," but we all know that the real world rarely lives up to our expectations. With the current state of the internet, a connection is often broken but one or both ends do not know this. Better to time out than wait forever (or for the next reboot).
S	This is where statistics on *sendmail* are kept. The file does not grow, so the */usr* file system will not overflow due to this.
s	This is a safety measure; it ensures the queue is always current, even when *sendmail* will deliver the message immediately. With mail, paranoia is a healthy state of mind!
T	After 3 days, the message will be returned as undeliverable. Another popular value is 15 days.
t	This is a relic of V6 UNIX and is ignored here.
W	This option applies to versions of *sendmail* compiled with the "wizard" option (4.3 BSD) or the "debug" option (4.2 BSD). Those versions of *sendmail* provide a special set of commands, ostensibly for debugging, that allow anyone with access to the SMTP server to break into the system. **DO NOT DELETE THIS OPTION!** With this option set, an attacker cannot break in this way even if the running version of *sendmail* contains the security hole.
X,x	These control how load averages affect SMTP connections. If the load average is 8 or more, incoming messages are queued and not delivered. If the load average is 12 or more, all requests for an SMTP connection are refused.

3.3. Precedences

This section is ignored unless one of the header fields is a "Precedence:" field, in which case the message is given the appropriate precedence with respect to all other undelivered messages. By default all messages have precedence 0 (the same as "first-class"). Table 7 summarizes the precedences RIACS recognizes.

Table 7. RIACS Mail Precedence Levels

class	priority
junk	-100
first-class	0
special-delivery	100

3.4. Trusted Users

When forwarding mail, *sendmail* must sometimes specify a different sender than in the message headers (this occurs in intra-network mailing). These users can use the appropriate command flags to do this; no-one else can. Putting a user in this field means he can change the sender. Currently, the trusted users are *root*, *daemon*, *uucp*, and *network*.

3.5. Mailers

This section describes the mailers for the gateway. There are five of them: *local*, which delivers mail with recipients on that host, *prog*, which delivers mail to programs or servers on that host, *uucp*, which delivers mail to the uucp network, *tcp*, which delivers mail over the internet, and *utcp*, which delivers uucp mail over the internet. Non-gateway hosts use three mailers (*local*, *prog*, and *tcp*) with identical descriptions to the mailers for the gateway.

The mailers each have flags indicating what actions *sendmail* should take when invoking them; these all follow the "F=" in each entry. All mailers require "Date: ", "From: ", and "Message-Id: " fields (flags **D**, **F**, and **M**) so if any of those fields are missing *sendmail* will add them before the mailer is invoked. The *local* and *prog* mailers both perform final delivery (the flag **l**); the letter will not be passed to another delivery agent when one of these mailers is invoked. Both the *local* and *tcp* mailers can deliver letters to multiple addresses simultaneously (the flag **m**) so *sendmail* will issue only one command to send the letter, rather than one such command per addressee. UNIX mailers do not conform to the standard, usually requiring headers of the form "From *user* ..." to be the first line in the letter; since the *local* mailer adds those lines, *sendmail* will not (the flag **n**); also, *sendmail* is to specify the recipient with the **-r** argument to the *local* mailer (the **r** flag). All mailers but the *tcp* mailer require quotes to be stripped from the address before being called, and *sendmail* does this (the **s** flag). The *prog*, *tcp*, and *utcp* mailers are expensive to connect to, so the gateway will only connect during a queue run; notice the **e** flag. (This flag only has an effect if the "c" flag is specified in the Options section. RIACS does not do this.) Also, the *uucp* mailer requires a special line at the top of the letter (this line is of the form "remote from *host* ..." and is also a violation of the standard) so the **U** flag has *sendmail* add it before sending the letter to that mailer. The *uucp*, *tcp*, and *utcp* mailers preserve the case of letters in user names (the **u** flag), and the *uucp* and *utcp* mailers also preserve the case of letters in host names (the **h** flag). Finally, the *tcp* and *utcp* mailers have a line length limit as specified in RFC 821 [RFC821]; *sendmail* will split lines in the letter if need be to keep lines short enough (the **L** flag).

Three mailers have other special considerations. Due to limits inherent in the *uucp* network, it is exceedingly unwise to send a letter with more than $2^{16}-1$ characters; the field "M=65535" in the *uucp* and *utcp* mailers prevents *sendmail* from sending files larger than that through this mailer. The *tcp* and *utcp* mailers use the two-character string "<CR><LF>" (that is, a carriage return followed by a line feed) to signal the end of a line.

The *uucp* mailer is peculiar to 4.3 UNIX. When called, */usr/bin/uux* is invoked as "uux - -z -a$f -gA $h!rmail ($u)"; this just means *uux* will read a letter from the standard input, send error messages to the sender (the **$f** is replaced with the sender's address), queue the message in the UUCP mail system with high priority (the "-gA" does this), and sends the input to the command *rmail* on the remote host **$h** with the recipient's name in parentheses on the command line.

The *tcp* mailer is really a "pseudo-mailer," because *sendmail* is the delivery agent. The string "P=[IPC]" tells *sendmail* to use an SMTP protocol to transmit the message. No other program is invoked (there is no program named "IPC", in fact). By default, communication is done over port 25. If some other port should be used, name the port after the "$h" in the "A=" string; for example, "A=IPC $h 100" initiates contact over port 100.

The *utcp* mailer is a compromise between the *uucp* mailer and the *tcp* mailer. Some sites on the internet are also uucp hosts, so rather than queue the message within the uucp system and send it that way, we use the SMTP protocol to transmit it directly. Thus, the mailer has the characteristics of both the *uucp* and *tcp* mailers in its description, but uses the *uucp* mailer's rewriting rules to rewrite addresses.

3.6. Headers

The header lines indicate what header fields should be added. The "Received:" field is inserted into every message, and the others are inserted depending on what flags the mailers in the *mailers* section above have. Table 5 shows which of the other headers are inserted into letters being handled by the mailers, and the flag present in the "F=" field of the mailer description that causes the header line to be inserted.

Table 5. Table of Header Lines

header line	*inserted for these mailers*	*flag*
Return-Path:	*none*	**P**
Resent-Date:	local, prog, uucp, tcp	**D**
Resent-Message-Id:	local, prog, uucp, tcp	**M**
Message-Id:	local, prog, uucp, tcp	**M**
Date:	local, prog, uucp, tcp	**D**
Resent-From:	local, prog, uucp, tcp	**F**
From:	local, prog, uucp, tcp	**F**
Full-Name:	*none*	**x**

If any of these lines are in the header, another new one will **not** be added. The test is done on the flag and not the header itself. For example, if a message has the header field "Date:", neither a new "Date:" nor a "Resent-Date" header field will be added, since a field tied to the **D** flag is present.

The next two sections describe the rulesets used to rewrite and deliver mail.

4. *sendmail* Gateway Configuration File

When *sendmail* obtains an address for processing, it runs that address through a series of rules grouped into rulesets. These rulesets can change the address, pass it on unchanged, or substitute a new address entirely for it. This section describes what RIACS' configuration file does.

4.1. Domain Names and Canonization

All the rulesets, except ruleset 3, assume the address is in a standard form; this reduces the number of rules that each ruleset needs. The form assumed is that the domain to which the letter is to be sent is surrounded by angle brackets '<', '>'. For example, the *canonical* form of "mab@riacs.edu" is "mab<@riacs.edu>"; the canonical form of "@purdue.edu:mab@riacs.edu" is "<@purdue.edu>:mab@riacs.edu" because the letter is to be sent to "purdue.edu", and from there to "riacs.edu". This form is internal; the angle brackets are removed before the address is written back into the header, or passed to the mailer. It is simply a convenience that allows simpler rewriting rules.

There are some cases where no legitimate domains exist to cover the addresses; for example, UUCP is not a domain in the sense that RFC 920 [RFC920] defines domain. But it is much easier to pretend that there is a domain ".uucp", and deal with addresses in that context; we can use the rewriting rules for legitimate domains to process addresses for these sites too. So the address "megatest!mab" is canonize to "mab<@megatest.uucp>". Also, many people use UUCP addressing to route letters through Internet sites; such addresses have the form "decwrl.dec.com!megatest!mab". In these cases, the canonization process does *not* add a ".uucp" to the domain; it uses the given domain. So, the above address would be canonized as "megatest!mab<@decwrl.dec.com>". This approach is actually a bit perilous, because many sites use domain names even though they are not on the Internet; such cases are handled by specific rules affecting only the address handed to a transport mechanism. This way, we can use the same canonical form of an address for all cases.

4.1.1. Ruleset 3

Ruleset 3 canonizes all addresses. We shall go through this ruleset in detail, showing how each rule moves the address towards canonization. When *sendmail* gets an address, it may be surrounded with '<', '>'; so we must strip these off. First, we handle the case where no address is provided, then we get the innermost address:

```
R<>                         $@@
R$*<$*<$*<$+>$*>$*>$* $4
R$*<$*<$+>$*>$*             $3
R$*<$+>$*                   $2
```

The first rule simply replaces the empty address with an "@" sign, and returns that as the new address. Other rulesets will deal with it later on. The next two rules deal with multiple nestings of addresses. This is not covered by the standard, so it is ambiguous. The convention most places seem to have adopted is to treat the innermost pair of angle brackets as delimiting the address. The final rule simply treats whatever is in angle brackets as the address.

The next rule,

```
R$+ at $+                   $1@$2
```

simply rewrites an archaic address specification to an acceptable one (RFC 733 [RFC733] allowed it, but RFC 822 [RFC822] does not). The old form of the address is now illegal, but since many sites will be slow to convert, it still should be recognized.

The next rule converts route specifications into an internal form:

```
R@$+,$+                     @$1:$2
```

This form replaces all ','s with ':'s, so (for example) "*@site1,@site2:user@site3*" would become "*@site1:@site2:user@site3*". This is not a legal form, but dealing with this form rather than the legal one makes rulesets much simpler. (The address is rewritten to the legal form before being output.) Note that we wish to forward the message to the first host in the specification.

Next, we deal with some of the more obvious errors.

```
R:$+                        $@$>3$1
R@:$*                       $@$>3$1
R$*@                        $@$>3$1
```

These three rules handle the case of a null domain name; in the first case, the separator ':' is deleted, and in the last two cases, the '@' introducing the null domain name is stripped and the address reprocessed.

Now we are ready to canonize:

```
R@$+:$+                     $@<@$1>:$2
R$+@$+                      $:$1<@$2>
R$+<$+@$+>                  $1$2<@$3>
R$*<@$*.>$*                 $@$>3$1@$2$3
R$+<@$+>                    $@$1<@$2>
```

The first rule handles route-specified addresses; note that it returns the result, since later rules shift the angle brackets right and mail is sent to the leftmost host in the route specified addresses rather than the rightmost. The second rule adds angle brackets to other addresses, and the third rule ensures that they are placed around the rightmost domain name. The fourth line deals with an error situation: when a domain name ending in a '.' is given, it is an error, so the offending '.' is deleted and the name recanonized. Otherwise, the result of the canonization is returned.

The last section of this ruleset handles syntaxes of addresses not falling within the scope of [RFC822]. These include UUCP and BITNET (among others).

```
R$+^$+                      $1!$2
R$-!$+                      $@$2<@$1.uucp>
```

```
R$+!$+              $@$2<@$1>
R$-:$+              $@$2<@$1>
R$-.$+              $@$2<@$1>
R$-=$+              $@$2<@$1.bitnet>
R$+%$+              $@$>9$1%$2
```

The first three rules deal with UUCP. The first rule converts an old UUCP address to the new form, and the second and third rules do the conversions described at the beginning of this section. The following two rules convert BERKNET addressing syntax to that of [RFC822]. The rule after than deals with BITNET syntax; like UUCP, BITNET is a pseudo-domain ".bitnet" and is dealt with similarly. The last rule translates "%" to "@" when appropriate; this is done by invoking ruleset 9, which will be discussed next.

4.1.2. Ruleset 9

One very common situation arising on the Internet is that of network routing; for example, if someone at an ARPANET site wishes to send a letter to someone at a CSNET site, he must indicate that the message is to go to another network, the CSNET network. This requires the message to be sent to a site on both ARPANET and CSNET (such sites are called *relay sites* or *relays*). The obvious syntax is to use routing, as "@relay.cs.net:postmaster@vpi.csnet", but a far more common syntax is to write "postmaster%vpi.csnet@relay.cs.net". It is more desirable to have the mail routing mechanism worry about how to get the message from ARPANET to CSNET. With sites that allow this (such as RIACS), people tend to write "postmaster%vpi.csnet", that is, use a "%" rather than a "@". The proper way to deal with such an address is to send the message to the *last* site named. Ruleset 9 deals with addresses involving sequences of "%"; it changes the final such character to "@" and canonizes the result:

```
R$*%$*              $1@$2
R$*@$*@$*           $1%$2@$3
R$*@$*              $@$1<@$2>
```

The first rule changes all "%" characters in the address to "@" characters and the second rule changes all but the last "@" back. The third rule adds the '<' and '>' around the last site. This canonizes the name. Note that if no domain or site name is given after the "%", ruleset 9 is not called, so no error checking need be done. In fact, throughout the other syntaxes ruleset 3 recognizes, there is an implicit assumption that no attempt will be made to recover from a faulty address. Such addresses will cause the mail to fail and be returned to the sender at some point, either at RIACS or at a site farther along the mail path.

4.2. The Transport Mechanism

Ruleset 0 is the basis for transport. By the time this ruleset returns, the input address must be resolved to a mailer, a host, and an address. When ruleset 0 returns, *sendmail* instructs the named mailer to contact the named host and send the letter to the named address. In most cases, the mailer is determined by the domain, although there is a mechanism that allows the name of the site to determine the transport mechanism. By default, the pseudo-domain ".uucp" uses the *uucp* mailer; and other domains either use the *tcp* mailer or are rewritten to send to a relay site. The relay site may use either the *tcp* or *uucp* mailers (currently, they all use the *tcp* mailer). All addresses given to the *tcp* mailer are enclosed in angle brackets and contain the name of the host to which the letter is being given, because many hosts reject addresses not in angle brackets or without the name of the host.

4.2.1. Ruleset 0

The first few rules of ruleset 0 rewrite the address into a form that can be analyzed and turned into a mailer/host/address set. The first rule eliminates loops back to RIACS; this is an efficiency consideration, and could be dropped.

```
R$*<@$+>$*          $:$>6$1<@$2>$3
```

We shall discuss how this is done when we look at ruleset 6, below. The next rule handles messages with no addresses.

```
        R@                      $#local$:$n
```

Recall that ruleset 3 changed null addresses into "@"; this rule just sends the letter to the designated person. The next rule deals with domains to which RIACS does not have direct access:

```
        R$*<@$+>$*              $:$>8$1<@$2>$3
```

These must be reached through relays; all this resolution is done in ruleset 8, which is used to rewrite addresses here. Note that some subdomains of known domains are rewritten; this need not be done once RIACS uses a name resolver, but should be kept just in case the name resolver is replaced. The rewriting must be done if the top-level domain is one RIACS can only access through a relay (such as BITNET or CSNET).

Finally we must convert route specification addresses from the internal form to the legal form. This rule does so:

```
        R$*:@$*                 $1,@$2
```

Wherever the sequence ":@" appears, it is replaced by ",@". This converts the internal format of a route specified address to the legal form, which can then be output.

The address is now in a form that can be resolved to a mailer/host/address triple. Some RIACS hosts do not accept SMTP connections, and others are not accessible via *uucp*; table 6 summarizes the mailers used to reach each local host. They are divided into two classes based on the mailer used.

Table 6. Mailers for RIACS Local Hosts

host	mailer	class	host	mailer	class
clavier	tcp	T	icarus	tcp	T
cube	uucp	U	lavalite	tcp	T
daedalus	tcp	T	miranda	tcp	T
dora	tcp	T	phun	tcp	T
hydra	tcp	T	zeus	tcp	T

Rather than being rigid and rejecting all incorrectly-routed mail, mail is rerouted properly. The next four rules do this:

```
        R$*<@$=T.uucp>$*        $#tcp$@$2$:<$1@$2$3>
        R$*<@$=U>$*             $#uucp$@$2$:$1$3
        R$*<@$=U.$N>$*          $#uucp$@$2$:$1$3
        R$*<@$=U.arpa>$*        $#uucp$@$2$:$1$3
```

The first rule changes any mail sent to a RIACS SMTP host over *uucp* to use the SMTP mailer *tcp*. The next three rules change any mail sent to a RIACS host that does not accept mail from the *tcp* mailer to use *uucp*.

One last problem remains. Many sites in the "uucp" (and other) domains use the ARPA domains, and so mail must be routed to them on a per-site basis. Also, many "uucp" sites use the ARPANET as the communications medium for *uucp*; it makes more sense to send mail over the ARPANET rather than use *uucp*. This ruleset is used to do such routing. No mailers are invoked; the addresses are simply rewritten. Table 7 describes the sites the addresses of which are rewritten:

Table 7. Addresses that RIACS Rewrites

site names		mailers	
incoming	outgoing	original	replaced by
amdcad.amd.com	amdcad	*tcp*	*uucp*
ames	ames.arc.nasa.gov	*uucp*	*utcp*
decwrl	decwrl.dec.com	*uucp*	*utcp*
ll-xn	ll-xn.arpa	*uucp*	*utcp*
lll-crg	lll-crg.arpa	*uucp*	*utcp*
rutgers	rutgers.rutgers.edu	*uucp*	*utcp*
seismo	seismo.css.gov	*uucp*	*utcp*

The rules are:

```
R$*<@amdcad.amd.com>$*$#uucp$@amdcad$:$1$2
R$*<@ames.uucp>$*     $#utcp$@ames.arc.nasa.gov$:<$1@ames.arc.nasa.gov$2>
R$*<@decwrl.uucp>$*   $#utcp$@decwrl.dec.com$:<$1@decwrl.dec.com$2>
R$*<@ll-xn.uucp>$*    $#utcp$@$[ll-xn$]$:<$1@$[ll-xn$]$2>
R$*<@lll-crg.uucp>$*  $#utcp$@$[lll-crg$]$:<$1@$[lll-crg$]$2>
R$*<@rutgers.uucp>$*  $#utcp$@rutgers.rutgers.edu$:<$1@rutgers.rutgers.edu$2>
R$*<@seismo.uucp>$*   $#utcp$@seismo.css.gov$:<$1@seismo.css.gov$2>
```

More rules can be added as needed.

We next dispose of addresses within the "uucp" pseudo-domain:

```
R<@$+.uucp>:$+          $#uucp$@$1$:$2
R$+<@$+.uucp>           $#uucp$@$2$:$1
```

Note that these handle "uucp" addresses in both route specifications and other forms. Two rules are necessary because the name of the machine to which the letter is being sent is not included in the address; for example, a letter to "megatest!ametek!root" would call the *uucp* mailer with a host of "megatest" and an address of "ametek!root".

The next rule handles all addresses for the *tcp* mailer:

```
R$*<@$*>$*              $#tcp$@$2$:<$1@$2$3>
```

Note that the address handed to the mailer is enclosed in angle brackets. This handles both rout-specified addresses and other forms of addressing. Finally, any address not converted to a mailer/host/address triple is intended to be delivered on this machine, so the final rule in ruleset 0 does this:

```
R$+                     $#local$:$1
```

4.2.2. Ruleset 6

Earlier we mentioned some other rulesets; let us now look at ruleset 6. This ruleset eliminates the gateway as a target for a letter; otherwise, the gateway would just deliver mail to itself, a rather pointless exercise especially when it can be handled immediately. The rules in this ruleset are actually divided into two groups: the ones that do canonization, and the ones that recurse. The first six rules all recanonize after stripping the gateway's name; the next six do not change the address, but reinvoke this ruleset if the target host is the gateway.

The first two rules delete the top-level domain and uucp names from the address:

```
R$*<@$N>$*              $>3$1$2
R$*<@$U>$*              $>3$1$2
```

Note these simply delete the host name in angle brackets and recanonize. The next four deal with the name of the gateway:

```
R$*<@$=w>$*             $>3$1$3
```

```
R$*<@$=G>$*             $>3$1$3
R$*<@$=G.$N>$*          $>3$1$3
R$*<@$=G.arpa>$*        $>3$1$3
```

Because the gateway is known by so many different names, and the domain may or may not be appended, the first two rules deal with the gateway host name without the domain, the third with the fully qualified gateway host name, and the last with the old ".arpa" form of the gateway host name. The next six rules are similar, but just reinvoke this ruleset:

```
R$*<@$N>$*              $>3$1<@$N>$2
R$*<@$U>$*              $>3$1<@$U>$2
R$*<@$=w>$*             $>3$1<@$2>$3
R$*<@$=G>$*             $>3$1<@$2>$3
R$*<@$=G.$N>$*          $>3$1<@$2.$N>$3
R$*<@$=G.arpa>$*        $>3$1<@$2.arpa>$3
```

They are present in case the recanonization has focused on another version of the gateway's name.

4.2.3. Ruleset 8

Ruleset 8 deals with the domains to which RIACS does not have direct access. This requires the address to be rewritten in order to send the message through a relay host. There are two syntaxes used. The first, for unrouted addresses, is to replace the address with something the relay can use. For example, the address "root@munnari.oz" would become "root%munnari.oz@seismo.css.gov" since "seismo.css.gov" is the relay host for the domain "oz". The relevant rules are:

```
R$*<@decwrl.dec.com>$*$@$1<@decwrl.dec.com>$2
R$*<@decwrl.dec>$*      $@$1<@decwrl.dec.com>$2
R$*<@$+.au>             $@$1%$2.au<@seismo.css.gov>
R$*<@$+.bitnet>         $@$1%$2.bitnet<@wiscvm.wisc.edu>
R$*<@$+.csnet>          $@$1%$2.csnet<@relay.cs.net>
R$*<@$+.dec.com>        $@$1%$2.dec<@decwrl.dec.com>
R$*<@$+.dec>            $@$1%$2.dec<@decwrl.dec.com>
R$*<@$+.decnet>         $@$1%$2.decnet<@$[ames-io$]>
R$*<@$+.mailnet>        $@$1%$2.mailnet<@$[mit-multics$]>
R$*<@$+.oz>             $@$1%$2.oz<@seismo.css.gov>
R$*<@telemail>          $@$1%telemail<@orion.arc.nasa.gov>
```

The first two rules make sure that letters being sent to "*someone*@decwrl.dec.com", the DEC Easynet relay, are not addressed to "*someone*%decwrl.dec@decwrl.dec.com" (for some reason, "decwrl.dec.com" cannot handle this). The remaining nine rules simply rewrite the addresses as required. Be aware that the rules for "dec.com" will go away when the name resolver software becomes reliable; in fact, since RIACS has the table of DEC hosts, all rules involving "dec" (except "decnet", which is an Ames local network) may soon be commented out.

The second syntax is used for route-specified addresses; the relevant rules are:

```
R<@$+.au>$*             $@<@seismo.css.gov>:$1.au:$2
R<@$+.bitnet>$*         $@<@wiscvm.wisc.edu>:@$1.bitnet:$2
R<@$+.csnet>$*          $@<@relay.cs.net>:@$1.csnet:$2
R<@$+.dec.com>$*        $@<@decwrl.dec.com>:@$1.dec:$2
R<@$+.dec>$*            $@<@decwrl.dec.com>:@$1.dec:$2
R<@$+.decnet>$*         $@<@$[ames-io$]>:@$1.decnet:$2
R<@$+.mailnet>$*        $@<@$[mit-multics$]>:@$1.mailnet:$2
R<@$+.oz>$*             $@<@seismo.css.gov>:$1.oz:$2
R<@telemail>$*          $@<@orion.arc.nasa.gov>:@telemail:$2
```

The rules just put the relay before the host to which the letter is to be directed; for example, the address "<@munnari.oz>:*someone@otherhost*" will be rewritten as

"<@seismo.css.gov>:@munnari.oz:*someone@otherhost*". Notice that addresses to "decwrl.dec.com" are taken care of by the first two rules in the ruleset, and so need not be repeated here.

4.3. The Sender's Address

There are two philosophies for handling a sender's address. Either the address is rewritten to reflect the way the message is sent, or it is not rewritten unless failing to do so would produce known errors when that path is used as a reply path. RIACS takes the latter approach. We assume that other sites know best how to route their mail, and so (with the exception of *uucp*, for a reason explained below) we do not rewrite the sender's address unless the sender is at this site.

The rewriting rulesets applied to the sender's address depends in part on the mailers being used. In all cases, ruleset 1 is applied, then the mailer-dependent rules, and finally ruleset 4 operates on the result.

4.3.1. Ruleset 1

Ruleset 1 hides the name of the local host within RIACS by replacing it with the name of the gateway as a domain; so, "root@hydra" would be rewritten as "root@riacs.edu". This enables RIACS to hide its internal configuration from the rest of the world. It is most useful should the internal configuration change, since only the gateway would need to be updated. Table 8 summarizes this configuration; note the nicknames do not have any domain or pseudo-domain names appended.

Table 8. RIACS Local Hosts

official host name	nicknames
clavier.riacs.edu	clavier riacs-clavier
cube.riacs.edu	cube hypercube riacs-cube riacs-hypercube
daedalus.riacs.edu	daedalus riacs-daedalus ames-daedalus
dora.riacs.edu	dora riacs-dora
hydra.riacs.edu	hydra riacs-hydra
icarus.riacs.edu	icarus riacs riacs-icarus riacs-gw
lavalite.riacs.edu	lavalite riacs-lavalite
miranda.riacs.edu	miranda riacs-miranda
pegasus.riacs.edu	pegasus riacs-pegasus ames-pegasus
phun.riacs.edu	phun riacs-phun
zeus.riacs.edu	zeus riacs-zeus

This ruleset is quite simple, consisting of only four rules:

```
R$*<@$H>$*              $@$1<@$N>$3
R$*<@$=H.arpa>$*        $@$1<@$N>$3
R$*<@$=H.$N>$*          $@$1<@$N>$3
R$*<@$=H.uucp>$*        $@$1<@$U>$3
```

The first rule changes unqualified names, the second changes old-style ".arpa" names, and the third changes fully qualified hostnames. Notice the fourth, which translates local host names into the standard RIACS *uucp* designator.

4.3.2. Ruleset 10

The *local* and *prog* mailers next use ruleset 10, which consists of one rule:

```
R@                      $n
```

If there is a null address, this ensures the appropriate person gets the (rejected) letter; that person can then decide what action to take.

4.3.3. Ruleset 13

The *uucp* mailer uses ruleset 13. This ruleset must prepend the RIACS uucp name *riacs.uucp* to the address, so for example, "mab@<megatest.uucp>" would become "megatest!mab<@riacs.uucp>". If this is not done, the remote *uucp* mail handler will record the return address as "somewhere!mab" rather than "riacs!mab", making it impossible to reply to that letter. Five rules accomplish this:

```
R$*<@$U>$*          $@$1<@$U>$2
R$*<@$+.uucp>       $@$2!$1<@$U>
R<@$+.uucp>$*       $@<@$U>:@$1.uucp:$2
R$*<@$*>$*          $@$1<@$2>$3
R$*                 $@$1<@$U>
```

The first rule simply eliminates cases where *riacs.uucp* is already in the address. The next two rulesets add the name to addresses sent via *uucp* and via routing specifications. The fourth rule returns any nonlocal addresses; presumably, these are of the form "*someone*@amdcad.amd.com" where RIACS can only talk to "amdcad.amd.com" via *uucp*. In this case, the remote site's *uucp* mail agent must be able to handle the address; RIACS should not have to. The final rule adds the uucp domain to addresses from a local machine, so "mab" would be rewritten as "mab<@riacs.uucp>".

4.3.4. Ruleset 14

The *tcp* mailer uses ruleset 14. If the sender is local, this ruleset appends the RIACS domain to the address:

```
R$*<@$*>$*          $@$1<@$2>$3
R$+                 $:$1<@$N>
```

4.3.5. Ruleset 4

Finally, ruleset 4 does some miscellaneous cleanup:

```
R@                  $@
```

Just in case an empty address gets this far, it is ignored; the next mailer can deal with it. (This should never happen, but just in case this line is left here.) The next rule decanonizes the address by deleting the angle brackets '<', '>':

```
R$*<$+>$*           $1$2$3
```

Route-specified addresses must be completely surrounded by such brackets, and converted to a legal form. The next rules do this:

```
R$+:@$+             $1,@$2
R@$+:$+@$+          <@$1:$2@$3>
```

and the final rule rewrites uucp addresses to their usual form:

```
R$*@$+.uucp         $2!$1
```

Note that this will not affect uucp addresses which are part of a route specification.

4.4. The Recipient's Address

As with senders' addresses, there are two philosophies for handling a recipient's address. Either the recipient's address is not rewritten unless failing to do so would produce known errors when that path is used by the next host to which the message is sent, or the address is rewritten to reflect the routing used to get to the next host in the address. RIACS takes the former approach. We assume that other sites know best how to route their mail, and so we do not rewrite the recipient's address unless the recipient is at this site.

The rewriting rulesets applied to the recipient's address depends in part on the mailers being used. In all cases, ruleset 2 is applied, then the mailer-dependent rules, and finally ruleset 4 operates on the result.

4.4.1. Ruleset 2

sendmail runs the address given to the mailer through these rulesets, so ruleset 2 is used to recanonize these addresses. Note that this also works for recipient addresses handed directly to ruleset 2 (that is, those from the "To:" or "cc:" field) because defocusing and recanonizing is essentially a no-op in this case. Ruleset two has two rules:

```
R$*<$*@$*>$*            $:$1$2$3$3
R$*                     $@$>3$1
```

The first rule deletes all angle brackets, and the second rule just invokes ruleset 3.

4.4.2. Ruleset 20

The *local* and *prog* mailers next use ruleset 20, which needs to do nothing, so there are no rules in it.

4.4.3. Ruleset 23

The *uucp* mailer uses ruleset 23, which (like ruleset 20) needs to do nothing, so there are no rules in it.

4.4.4. Ruleset 24

The *tcp* mailer uses ruleset 24. If the recipient is local, this ruleset appends the RIACS domain to the address:

```
R$*<@$*>$*              $@$>8$1<@$2>$3
R$+                     $:$1<@$N>
```

Ruleset 4 has been discussed above.

4.5. Conclusion

This completes the discussion of why the gateway configuration file was set up as it is. Now let us look at the non-gateway file.

5. *sendmail* Non-Gateway Configuration File

This file is much simpler, because the strategy is simply to send any nonlocal mail to the gateway. Accordingly, sender and recipient addresses are not processed, and canonization is done only to aid in determining the mailer to be used.

5.1. The Transport Mechanism

Rulesets 3 and 9 are the same as for the gateway configuration file, so they will not be discussed again, and ruleset 0 is much simpler.

5.2. Ruleset 0

As with the gateway, the first rule in that ruleset ensures that all names of the host are deleted by invoking ruleset 6:

```
R$*                     $:$>6$1
```

At this point, the mailer can be determined. Either the letter is local, in which case the *local* or *prog* mailers need to be invoked, or it is not local, in which case the *tcp* mailer sends the letter to the gateway:

```
R@                      $#local$:$n
R$*<$*@$*>$*            $#tcp$@$G$:<$1$2@$3$4>
R$+                     $#local$:$1
```

Note that if the address is empty, the first rule will send the letter to the appropriate user, who can take whatever action is deemed appropriate. The next rule just forwards any mail with a host name to the gateway. Since names for this host were deleted by the first rule in this ruleset, such mail is destined for another host. The last rule just delivers the mail locally.

5.2.1. Ruleset 6

Ruleset 6 is somewhat different because it does not have to deal with domain names or "uucp" names, but only the names of this machine. As in the gateway configuration file, the first set of rules recanonize the address, and the next set determines whether or not to reinvoke ruleset 6:

```
R$*<@$=w>$*             $>3$1$3
R$*<@$=w.arpa>$*        $>3$1$3
R$*<@$=w.$N>$*          $>3$1$3
R$*<@$=w>$*             $>6$1<@$2>$3
R$*<@$=w.arpa>$*        $>6$1<@$2.arpa>$3
R$*<@$=w.$N>$*          $>6$1<@$2.$N>$3
```

5.3. Philosophy

The philosophy in writing the non-gateway configuration files has been to keep the rules as simple as possible. This is why local hosts will send mail to each other through the gateway rather than directly; since not all hosts can speak SMTP, only the gateway needs to keep track of how each host is to receive mail. This makes the gateway file the only one that needs to be changed when the configuration changes.

6. Debugging a *sendmail* Configuration File:

RIACS runs various versions of *sendmail* on its hosts. All are essentially the same program, so the techniques discussed below work for all versions of *sendmail*. In all cases, it is **strongly** recommended that the new configuration file **not** be installed until it is fully tested and debugged; instead, run *sendmail* with the -C option. So, to debug a configuration file *new.cf*, issue the command

```
sendmail -Cnew.cf other options ...
```

The first mode for testing is called *test mode*, appropriately enough, and is used to run addresses through rulesets to show what each ruleset is given, what it returns, and how the given address is rewritten. To use it, put the flag **-bt** on the command line. Here is a sample session, using the gateway configuration file; some of the longer lines have been split into two, in order to fit on the page.

```
% /usr/lib/sendmail -bt
ADDRESS TEST MODE
Enter <ruleset> <address>
> 0 riacs.edu!site.com!xyz
rewrite: ruleset 3   input: "riacs" "." "edu" "!" "site" "." "csnet" "!" "xyz"
rewrite: ruleset 3 returns: "site" "." "csnet" "!" "xyz" "<" "@" "riacs" "."
                    "edu" ">"
rewrite: ruleset 0   input: "site" "." "csnet" "!" "xyz" "<" "@" "riacs" "."
                    "edu" ">"
rewrite: ruleset 6   input: "site" "." "csnet" "!" "xyz" "<" "@" "riacs" "."
                    "edu" ">"
rewrite: ruleset 3   input: "site" "." "csnet" "!" "xyz"
rewrite: ruleset 3 returns: "xyz" "<" "@" "site" "." "csnet" ">"
rewrite: ruleset 6 returns: "xyz" "<" "@" "site" "." "csnet" ">"
rewrite: ruleset 8   input: "xyz" "<" "@" "site" "." "csnet" ">"
rewrite: ruleset 8 returns: "xyz" "%" "site" "." "csnet" "<" "@" "relay" "."
                    "cs" "." net" ">"
```

```
rewrite: ruleset  0 returns: "$#" "tcp" "$@" "relay" "." "cs" "." "net" "$:"
                             "<" "xyz" "%" "site" "." "csnet" "@" "relay" "."
                             "cs" "." "net" ">"
> 1,14,4 cde@phun.riacs.edu
rewrite: ruleset  3   input: "cde" "@" "phun" "." "riacs" "." "edu"
rewrite: ruleset  3 returns: "cde" "<" "@" "phun" "." "riacs" "." "edu" ">"
rewrite: ruleset  1   input: "cde" "<" "@" "phun" "." "riacs" "." "edu" ">"
rewrite: ruleset  1 returns: "cde" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset 14   input: "cde" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset  8   input: "cde" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset  8 returns: "cde" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset 14 returns: "cde" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset  4   input: "cde" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset  4 returns: "cde" "@" "riacs" "." "edu"
> 2,24,4 ghi@vpi.csnet
rewrite: ruleset  3   input: "ghi" "@" "site" "." "au"
rewrite: ruleset  3 returns: "ghi" "<" "@" "site" "." "au" ">"
rewrite: ruleset  2   input: "ghi" "<" "@" "site" "." "au" ">"
rewrite: ruleset  3   input: "ghi" "@" "site" "." "au"
rewrite: ruleset  3 returns: "ghi" "<" "@" "site" "." "au" ">"
rewrite: ruleset  2 returns: "ghi" "<" "@" "site" "." "au" ">"
rewrite: ruleset 24   input: "ghi" "<" "@" "site" "." "au" ">"
rewrite: ruleset  8   input: "ghi" "<" "@" "site" "." "au" ">"
rewrite: ruleset  8 returns: "ghi" "%" "site" "." "au" "<" "@" "seismo" "."
                             "css" "." "gov" ">"
rewrite: ruleset 24 returns: "ghi" "%" "site" "." "au" "<" "@" "seismo" "."
                             "css" "." "gov" ">"
rewrite: ruleset  4   input: "ghi" "%" "site" "." "au" "<" "@" "seismo" "."
                             "css" "." "gov" ">"
rewrite: ruleset  4 returns: "ghi" "%" "site" "." "au" "@" "seismo" "." "css"
                             "." "gov"
```

In this example, the first ruleset called is ruleset 3 (this is always the case, regardless of what rules you name). It is passed the line "riacs.edu!site.csnet!xyz", and returns the canonized form "site.csnet!xyz<@riacs.uucp>". Ruleset 0 is then invoked; its first rule calls ruleset 6, which strips off the "@riacs.edu" (since this example was run on *icarus*, which is also known as *riacs.edu*, there is no point whatsoever in remailing the message to *riacs.edu*) and calls ruleset 3 on the remainder of the string, "site.csnet!xyz". Ruleset 3 rewrites this as "xyz<@site.csnet>" and returns to ruleset 6, which returns control to ruleset 0. Ruleset 0 then calls ruleset 8, which rewrites the address to send it through the appropriate relay ("xyz%site.csnet<@relay.cs.net>") and then returns this new form to ruleset 0. Ruleset 0 rewrites the address as "$#tcp$@relay.cs.net$:<xyz%site.csnet@relay.cs.net>" and using this form calls the appropriate mail handling agent. The other two examples work in the same way, the second example using rulesets 3, 1, 14, and 4, and the third example using rulesets 3, 2, 24, and 4.

The version of *sendmail* on *icarus*, the RIACS gateway, has been modified in two ways. The first, relevant to debugging, prints control strings used in the configuration file as they are entered in that file [GILL86]. This modification has not been made to any other version of *sendmail*, so beware! Usually, the final line of the first case in the above example would be printed as:

```
rewrite: ruleset  0 returns: "^V" "tcp" "^W" "site" "." "com" "^X" "<" "xyz"
      "@" "site" "." "com" ">"
```

for 4.3 BSD's version of *sendmail*, or

```
rewrite: ruleset  0 returns: "^U" "tcp" "^V" "site" "." "com" "^W" "<" "xyz"
      "@" "site" "." "com" ">"
```

for other versions of *sendmail*. Table 9 shows the internal symbols corresponding to the special symbols used in rewriting rules; the version of *sendmail* on *icarus* uses those in the 4.3 BSD column, and all other machines use those in the 4.2 BSD column.

Table 9. Table of Configuration File Actions

Symbol	*Shown*		*Symbol*	*Shown*		*Symbol*	*Shown*	
	4.2BSD	4.3BSD		4.2BSD	4.3BSD		4.2BSD	4.3BSD
$*	^P	^P	$	^T	^U	$?	^Y(?)	^Z
$+	^Q	^Q	$#	^U	^V	$|	^Z(?)	^[
$-	^R	^R	$@	^V	^W	$.	^[(?)	^\
$=	^S	^S	$:	^W	^X	$[	*none*	^]
$~	^\	^T	$>	^X	^Y	$]	*none*	^^

There is also a set of debugging flags that are quite useful when one particular aspect of *sendmail* needs to be examined very closely. They differ from version to version; the ones listed in Table 10 are for version 5.51.

Table 10. Table of Debugging Flags

flag	*level*	*information about*
0	1	main: run as daemon in foreground
0	4	main: show canonical name and aliases of current host
0	15	main: print configuration table as loaded from configuration file
0	44	main: print command-line arguments to program
1	1	main: report sender of letter
1	9	main: allow user doing debugging to substitute new sender
2	1	exit: report exit status, flags
5	4	event: report alarms
5	5	event: report setting, clearing of event
5	6	event: show event processing
6	1	error: report how mail is dealt with on error
6	5	error: display state of *sendmail* on error
7	1	queueing: show names of queue file
7	2	queueing: show name of temporary queue file ("tf" file)
7	20	queueing: report processing of the queued file
10	1	deliver: show address given to mailer
11	1	deliver: show address to which letter is actually sent
12	1	parseaddr: show resolution of remote name in address
13	1	deliver: show to whom messages are to be sent
13	3	deliver: check for errors in sending messages
13	4	deliver: report to whom errors go
14	2	header: print list of names in header with commas
15	1	daemon (server): report requests
15	2	daemon (server): report forking to process a request
15	15	daemon (server): put network in debugging mode
16	1	daemon (client): report making remote connection
16	14	daemon (client): put network in debugging mode
18	1	smtp: report result of trying to make connection
18	100	smtp: pause after error in reading from remote connection
20	1	parseaddr: report address to be parsed
21	2	parseaddr: report input, output of each ruleset
21	3	parseaddr: report call to another ruleset (via "$>")
21	4	parseaddr: report result of call to another ruleset
21	10	parseaddr: report failure of ruleset to match

Table 10. Table of Debugging Flags

flag	*level*	*information about*
21	12	parseaddr: report ruleset to be matched and if match succeeds
21	15	parseaddr: report substitution due to matches substitution
21	35	parseaddr: show attempts to match
22	36	parseaddr: show address prescan
22	45	parseaddr: show what is being prescanned
22	101	parseaddr: show states during parsing of address
25	1	recipient: show recipient list (multiple addresses possible)
26	1	recipient: show an individual address
27	1	alias: report alias definition
30	1	smtp: report end of headers
30	2	smtp: report addition of "Apparently-To:" field
30	3	smtp: report processing of UNIX "From" line
31	6	header: show header line being read
32	1	header: show all headers to current letter
33	1	header: show header line as processed
35	9	macro: report definition
35	24	macro: show macro expansion
36	5	symbols: report addition and/or lookup
36	9	symbols: display symbol hash
37	1	configuration: show option settings in configuration file
40	1	queue: show queue run of file
40	4	queue: print queue file lines ("qf" file) as read
41	2	queue: report problem reading queue during ordering
45	1	envelope: report sender
50	1	envelope: report deallocation
51	4	queue: show deletion of transcript file
52	1	main: report disconnection from controlling terminal
52	5	main: do not disconnect from controlling terminal

Options are set with the -dflag, which has the format -d*flaglist.level.* To set flags 5, 6, 7, and 36 at level 10, for example, give the option "-d5-7,36.10". It is usually advisable to confine debugging flags to number 21 (which deals with the way addresses are parsed) or display all of them; also, specifying a level prints all messages from lower levels, too. To print any information that could be of importance, use "-d0-99.99"; the levels higher than 99 are used to debug *sendmail*'s internals.

As stated above, debugging flag 21 is useful enough to merit some special discussion. When more detail about the address parsing is needed, this flag prints each step in the ruleset and indicates how a match is being made. For example,

```
% /usr/lib/sendmail -bt -d21.99
Version 5.51
ADDRESS TEST MODE
Enter <ruleset> <address>
> 0 mab@riacs.edu
rewrite: ruleset  3   input: "mab" "@" "riacs" "." "edu"
-----trying rule: "<" ">"
ap="mab", rp="<"
----- rule fails
-----trying rule: "$*" "<" "$*" "<" "$*" "<" "$+" ">" "$*" ">" "$*" ">" "$*"
ap="mab", rp="$*"
ap="mab", rp="<"
ap="@", rp="<"
```

```
ap="riacs", rp="<"
ap=".", rp="<"
ap="edu", rp="<"
ap=<null>, rp="<"
----- rule fails
-----trying rule: "$+" "@" "$+"
ap="mab", rp="$+"
ap="@", rp="@"
ap="riacs", rp="$+"
ap=".", rp=<null>
ap="edu", rp=<null>
-----rule matches: "$:" "$1" "<" "@" "$2" ">"
$1: 7fffe2bc="mab"
$2: 7fffe2c2="riacs" 7fffe2c8="." 7fffe2ca="edu"
rewritten as: "mab" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset  3 returns: "mab" "<" "@" "riacs" "." "edu" ">"
rewrite: ruleset  0   input: "mab" "<" "@" "riacs" "." "edu" ">"
-----trying rule: "$*" "<" "@" "$+" ">"
ap="mab", rp="$*"
ap="mab", rp="<"
ap="<", rp="<"
ap="@", rp="@"
ap="riacs", rp="$+"
ap=".", rp=">"
ap="edu", rp=">"
ap=">", rp=">"
-----rule matches: "$:" "$>" "6" "$1" "<" "@" "$2" ">"
$1: 7fffe2bc="mab"
$2: 7fffe2c2="riacs" 7fffe2c8="." 7fffe2ca="edu"
-----callsubr 6
rewrite: ruleset  6   input: "mab" "<" "@" "riacs" "." "edu" ">"
-----trying rule: "$+" "<" "@" "riacs" "." "edu" ">"
ap="mab", rp="$+"
ap="<", rp="<"
ap="@", rp="@"
ap="riacs", rp="riacs"
ap=".", rp="."
ap="edu", rp="edu"
ap=">", rp=">"
-----rule matches: "$>" "3" "$1"
$1: 7fffe2bc="mab"
-----callsubr 3
rewrite: ruleset  3   input: "mab"
rewrite: ruleset  3 returns: "mab"
rewritten as: "mab"
rewrite: ruleset  6 returns: "mab"
rewritten as: "mab"
-----trying rule: "$+"
ap="mab", rp="$+"
-----rule matches: "$#" "local" "$:" "$1"
$1: 7fffe2bc="mab"
rewritten as: "$#" "local" "$:" "mab"
rewrite: ruleset  0 returns: "$#" "local" "$:" "mab"
>
```

(Note that parts of the above were edited to keep the output brief.) When ruleset 3 is first called, it tests to see if the address matches "<>". It does not, as the token "mab" is not the same as the token '<'. So it tries the next rule in the ruleset, and continues, until it finds a match in the rule "$+@$+". This rule replaces the previous form with "mab<@riacs.edu>", and continues the ruleset. Finally, it returns with "mab<@riacs.edu>". Ruleset 0 then calls ruleset 6, which strips off the "<@riacs.edu>" and calls ruleset 3 to recanonize the rest of the address "mab". Eventually, ruleset 6 returns "mab", and ruleset 0 turns this into the appropriate mailer command.

Occasionally a user will report a problem with mail requiring the maintainer of the mail system to determine at which hosts, if any, the problem occurred. To determine this, look at the headers in the letter that posed the problem. If there are error messages, read them first; if those do not provide enough information, scan the "Received: " lines. Lines added at RIACS indicate from whom the mail routers think the message was received; this can be used to track backwards. As an example, the message with the headers

```
Received: from icarus.riacs.edu (icarus.ARPA) by hydra.riacs.edu (4.12/1.6N)
          id AA14571; Fri, 16 Jan 87 16:41:02 pst
Received: from RELAY.CS.NET by icarus.riacs.edu (5.51/1.6G)
          id AA00679; Fri, 16 Jan 87 16:40:49 PST
Received: from icarus.riacs.edu by RELAY.CS.NET id aa02271; 16 Jan 87 19:30 EST
Received: by icarus.riacs.edu (5.51/1.6G)
          id AA00630; Fri, 16 Jan 87 16:29:16 PST
Message-Id: <8701170029.AA00630@icarus.riacs.edu>
Date: Fri, 16 Jan 87 16:29:16 PST
From: Matt Bishop <mab@riacs.edu>
To: mab%riacs.edu@RELAY.CS.NET
Subject: show how "Received: " header lines work
```

went from *icarus* (*icarus.riacs.edu*) to *RELAY.CS.NET* and back to *icarus* and from there to *hydra*.

Incidentally, the version of *sendmail* on *icarus* has been modified to report the name of the system from which *icarus* receives the letter. This modification defines the unsupported macro s so that the name of the sending host may be included on the "Received" header line, as well as allowing the receiving host to recognize nicknames as well as official names (these two fixes are from [LOVS86]). The *sendmail* files which have been altered are stored in their original form as *file*.orig in the *sendmail* source directory.

7. Installing New *sendmail* Configuration Files

In general, to install a new *sendmail* configuration file, do the following:

1. Locate and kill any currently-running invocations of *sendmail*. This is best done by typing "ps gaux | grep sendmail" as *root* and then killing all the processes this command lists. If some have died anyway, do not worry; they exited between the "ps" and the "kill".
2. Copy */usr/lib/sendmail.cf* somewhere. This way, if the new configuration file does not work right, the old one can be put back.
3. Move the new configuration file to */usr/lib/sendmail.cf*.
4. Freeze this file. This puts it into a form *sendmail* can load quickly. Type "/usr/lib/sendmail -bz"; the file that is created is */usr/lib/sendmail.fc*.
5. Restart the *sendmail* daemon. This command will be of the form "/usr/lib/sendmail -bd -q1h", but it varies from system to system. Look in */etc/rc.local* for the exact command; typing "grep 'sendmail.*-bd' /etc/rc.local" will print the exact command.
6. Type "mailq". On the server Sun, the command will print a message saying "Freeze file out of date", and delete the frozen configuration file; this is normal. (Why it is done is not known.) This ensures that the *sendmail* program is working correctly.

On the client Suns, the procedure is slightly different since they use the server's configuration file. On these systems, omit steps 2 through 4.

8. Changing the Configuration Files

This section describes how to make routine changes to the *sendmail* configuration files. It does not cover massive rewriting; for that, your best bet is to find a *sendmail* guru or just experiment.

8.1. Adding a New RIACS Host (Gateway Configuration File)

This is really quite easy. First, figure out how mail will be sent to the host; if it will go via *uucp* or *tcp*, you're in luck. (If not, you'll have to define a new mailer. See [ALLM84b] for a description of how to do this; use the already-created mailer descriptions as a guide.)

First, add all names of the new host in the class **H**. If the new host were named "hera.riacs.edu", with nicknames "riacs-hera.arpa", "hera", "ames-hera", and "riacs-hera", add a line of the form

```
CHhera riacs-hera ames-hera
```

to the appropriate section (where the rest of the hosts are defined). Then, if the mailer is *tcp*, add all the names to the class T; if the mailer is *uucp*, add all the names to the class U. If you are using some other mailer, define a new class for the mailer (remember, *sendmail* macro names are one character only!), and put this host into the class. For explanatory purposes, call the new class **W** and the mailer *newmail*. Then in ruleset 0, add the following rules to the end of the "resolve the mailers ... handle local hosts" section:

```
R$*<$=W>$*              $#newmail$@$2$:address
R$*<$=W.arpa>$*         $#newmail$@$2$:address
R$*<$=W.uucp>$*         $#newmail$@$2$:address
R$*<$=W.$N>$*           $#newmail$@$2$:address
```

(where address is the form of the address newmailer expects). Then reinstall the configuration file.

8.2. Sending UUCP Mail to A New Host Over the Internet (Gateway Configuration File)

These changes allow you to route mail to internet hosts over the internet, even when the sender has asked that mail be sent via *uucp*. It is strongly recommended that this be done only when *uucp* uses the internet as the communications medium; this ensures the site is really on the internet. Needless to say, this will not work unless the site accepts SMTP connections.

Suppose RIACS talks to the site "faraway.sub.dom" (with *uucp* name *faraway*) using *uucp* running over the internet. To route mail through SMTP but in such a way everyone else thinks the mail was sent through the uucp system, add the line

```
R$*<@faraway.uucp>$* $#utcp$@faraway.sub.dom$:<$1@faraway.sub.dom$2>
```

to ruleset 0, in the section "...handle special hosts". It can go anywhere in that section.

See the section entitled **Mailers** for a description of the *utcp* mailer.

8.3. Sending Internet Mail to A New Host Over UUCP (Gateway Configuration File)

These changes allow you to route mail to uucp hosts which use the internet domain naming scheme, even when the sender has asked that mail be sent via *uucp*. This simply must be done; it should be undone when (and if) the site joins the internet (in which case perhaps mail routed through *uucp* should be sent via the internet, as described in the section above).

Suppose RIACS talks to the site "faraway.sub.dom" (with *uucp* name *faraway*) using *uucp* running over a telephone line. It is not possible to pretend the message was sent over the internet, due to the limits of the uucp mailers, but adding the line

```
R$*<@faraway.sub.dom>$*$#uucp$@faraway$:$1$2
```

to ruleset 0, in the section "...handle special hosts" will route such mail over *uucp*. (This rule can go anywhere in that section.) Be aware the address will be run through the *uucp* mailer's sender and recipient address rewriting rules, so the receiving host should get a path to which it can reply. The emphasis is on the word "should."

8.4. Adding a New Domain Relay Site (Gateway Configuration File)

Suppose RIACS requires access to the domain "newdom", but that domain can only be reached through the host "relay.dom.net". The goal is to accept addresses like "*user@site*.newdom" and route the mail automatically.

Ruleset 8 is used to handle these cases. Two lines must be added, one for route addressing, and the other for unrouted addressing. In the section of this ruleset entitled "output fake domain stuff in user%host.fake@relay-host syntax", add a rule of the form

```
R$*<@$+.newdom>        $@$1%$2.newdom<@relay.dom.net>
```

(which maps "*user*<@*site*.newdom>" to "*user*%*site*.newdom<@relay.dom.net>") and to the section of this ruleset entitled "output fake domain stuff in route-specific syntax", add a rule of the form

```
R<@$+.newdom>:$*       $@<@relay.dom.net>:@$1.newdom:$2
```

(which maps "<@*site*.newdom>:*user@site2*" to "<@relay.dom.net>:@*site*.newdom:*user@site2*"). Then addresses without the relay site named explicitly will be sent to the relay site anyway.

8.5. Change the Gateway Host (Gateway and Non-Gateway Configuration Files)

To do this, both the gateway and non-gateway files must be changed. In the gateway file, change the definition of the class G to be all the names of the new gateway host. In the non-gateway file, change the definition of the macro G to be the name of the new gateway host (it is recommended this be the fully qualified internet name). Then install the gateway configuration file on the new gateway host, and the non-gateway configuration file on all other hosts.

This assumes the gateway is to handle all outgoing traffic. If not, a third configuration file must be written for the UUCP host. See the introductory comments to this section, above. (Translation: good luck!)

9. Conclusion

The goal of this work has been to make the RIACS mail system into a low maintenance system and, when maintenance is necessary, easy to maintain. This document is an integral part of that plan, because it is intended to allow a systems programmer to understand how the mail configuration files work, and why the mail system is set up as it is. Needless to say, only time will tell if these rather ambitious goals have been met.

Acknowledgement

My thanks to Barry Leiner for his comments on an earlier draft.

References

[ALLM84a] Allman, Eric, "Sendmail – An Internetwork Mail Router", in *UNIX System Manager's Manual, 4.2 Berkeley Software Distribution, Virtual VAX-11 Version*, Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA (Mar. 1984), as reprinted by the USENIX Association

[ALLM84b] Allman, Eric, "Sendmail – Installation and Operation Guide Version 4.2", in *UNIX System Manager's Manual, 4.2 Berkeley Software Distribution, Virtual VAX-11 Version*, Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA (Mar. 1984), as reprinted by the USENIX Association

[GILL86] Gilliam, Jeff, USENET message <2821@voder.uucp>, National Semiconductor, Santa Clara, CA (Dec. 11, 1986).

[LOVS86] Lovstrand, Lennart, USENET message <377@majestix.liuida.uucp>, Computer Science Department, University of Linkoping, Sweden (Dec. 28, 1986).

[RFC733] Crocker, D. H., Vittal, J. J., Pogran, K. T., and Henderson, D. A., *Standard for the Format of ARPA Network Text Message*, ARPANET Request for Comments No. 733, Network Information Center, SRI International, Menlo Park, CA (Nov. 1977)

[RFC821] Postel, Jonathan B., *Simple Mail Transfer Protocol*, ARPANET Request for Comments No. 821, Network Information Center SRI International, Menlo Park, CA (Aug. 1982)

[RFC822] Crocker, David H., *Standard for the Format of ARPA Network Text Message* (revised), ARPANET Request for Comments No. 822, Network Information Center SRI International, Menlo Park, CA (Aug. 1982)

[RFC920] Postel, J., and Reynolds, J., *Domain Requirements*, ARPANET Request for Comments No. 920, Network Information Center SRI International, Menlo Park, CA (October 1984)

[ROSE86] Rose, Marshall T., and Romine, John L., *The RAND MH Message Handling System: User's Manual*, UCI Version (Apr. 1986)

[UPM84] –, *UNIX User's Manual Reference Guide*, *4.2 Berkeley Software Distribution*, *Virtual VAX-11 Version*, Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA (Mar. 1984), as reprinted by the USENIX Association

The Fourth Annual USENIX Computer Go Tournament and Championship

```
    ABCDEFGHJKLMNOPQRST
19  +++++●+++++++++++++  19
18  ++OO●+●O++++O●+O+++  18
17  +O++O●+O++OO+O●+●++  17
16  ++OOO●●O+++OO●+●+++  16
15  +++O+O●O●O+O●++++++  15
14  +++●OO●●O●●●+++●+++  14
13  +++●O+O●O++++++++++  13
12  ++++●OO●+●+++++++++  12
11  +OO+++O●+++++●+++++  11
10  +●O+++O++++++++●●++  10
 9  +●+O++++++++++O++++  9
 8  ++●●OOO+++++O+++O++  8
 7  +●++●●++++++++O++++  7
 6  +●O●+++++++++++++++  6
 5  +●OO++++++++++●++++  5
 4  ++O+++●++●++++++●++  4
 3  +++O+O●+++++++●++++  3
 2  ++++O+O●+++++++++++  2
 1  +++++O+++++++++++++  1
    ABCDEFGHJKLMNOPQRST
```

Announcement

The fourth annual USENIX Computer Go Tournament will be held on Wednesday, June 10th, during the Summer 1987 USENIX conference in Phoenix, Arizona.

All interested parties are invited to submit programs. Programs may be written in any language as long as the binary will run under a UNIX operating system like 4.? BSD. The tournament rules will be essentially those established for the first USENIX Computer Go Tournament (see below).

This event will be a "championship." The winner will be the "USENIX Computer Go Champion" until the next championship is held (probably at the USENIX Conference the following summer).

Conference attendees may bring programs to submit with them as long as they get in touch with Peter Langston NO LATER THAN noon on the day before the tournament (preferably earlier). He can be reached through the USENIX Conference Office. People who are unable to attend the conference but would like to enter their programs can do so by sending a compilable source to one of the addresses below, (or by taking a chance and sending an "executable" file which may, or may not, function under last minute operating systems changes or machine changes, or ...).

The source code for the referee program to be used has been distributed through netnews "net.sources" and can be redistributed if interest warrants.

Comments or programs can be sent via electronic mail to *bellcore!psl* or *psl@BELLCORE.ARPA*. U.S. Mail should be sent to:

Peter Langston
Bell Communications Research
MRE 2D-396
435 South Street
Morristown, NJ 07960

USENIX Computer Go Tournament Rules

Revised April, 1986
P. Langston

- **A full size board will be used.** The board will be 19x19 with columns labeled "A" through "T" (excluding "I") left to right, and rows labeled "19" through "1" top to bottom.
- **Komi will be 5.5 points.** The second player gets a 5.5 point bonus.
- **There will be a time limit.** Each program will be limited to a total of 60 minutes of

accumulated "user" time. If a program goes over the time limit it will only be allowed 10 seconds of "user" time for each move (byo-romi). Subsequently, if a program uses more than 10 seconds of "user" time for a move it will immediately forfeit the game.

• **The programs must not be idle unnecessarily.** If 10 minutes of "real" time elapse with no increase in the current program's "user" time, it will be assumed that the program is stuck and the program will forfeit. (This rule is included to handle cases where a program loses synchronization or is doing something like: `for (;;) read(0, buf, sizeof buf);`.)

• **There will be no forking (around).** Each program must be a single process and must not fork other processes. Forking interferes with the timing mechanism and, like any attempt to evade or fool the timing, will result in a forfeit. The tournament will use a "referee" program to execute each competing pair of programs. There will be no command-line arguments, i.e. argc will be 1. All communication with the programs will be via the standard input and standard output, thus the programs must understand a specific set of commands and generate output of a specific form.

a) All input commands to the competing programs will be in the form of lines of text appearing on the standard input and terminated by a newline.

b) The first line of input to each program will be either "black" or "white" (lower case) to indicate which color the program will be playing (and thereby whether the program plays first or second).

c) The placement of a stone will be expressed as upper-case letter-number (e.g. "G7" note capitalization).

d) A pass will be expressed as "pass" (lower case).

e) The command "byo-romi" (lower case) means the time limit has been exceeded and all further moves must be generated within the 10 second time limit.

f) All output from the competing programs will be in the form of lines of characters sent to the "standard output" terminated by a newline, and had better either be flushed after every line or be unbuffered to start with (e.g. `setbuf(stdout, 0);`).

g) Any output lines not beginning with "A" through "T" (excluding "I") or "pass" will be considered garbage and ignored.

Any syntactically correct but semantically illegal move will be considered a forfeit. The three possibilities are: playing on a non-empty spot (occupied or off the board), ko violation, and suicide.

• **Play will end when both programs pass in sequence.** The programs may pass at any time, but once both pass concurrently, the game is over.

• **The decisions of the judge will be final.** A human judge will evaluate each game's results and may fill in missed dame or may judge a game incomplete if, in the judge's opinion, too much is unresolved. In general, Japanese rules will be used, (Nihon Kiin).

≡ ≡ ≡

We've Moved!

At the end of March, the USENIX Association moved to new office space. Our new postal address is:

P.O. Box 2299
Berkeley, CA 94710

The telephone number remains 415-528-8649.

Report on the Large Installation System Administrators' Workshop

Seventy people attended the first Large Installation System Administrators' Workshop in Philadelphia, PA, April 9-10, 1987. Each attendee was asked to present a one or two page paper detailing a certain aspect of system administration at their installation. The papers were then sorted into the following categories: network administration, backup and restore, software distribution and synchronization, mail and news, accounts and accounting, reliability enhancements, and I/O management. In addition to the papers, panel discussions and/or question and answer periods followed each of the sessions. Rob Kolstad, CONVEX Computer Corporation, and a member of the USENIX Association Board of Directors, and Alix Vasilatos, MIT Project Athena co-chaired the workshop. Their behind-the-scenes planning and on-site management of the sessions insured the success of the program. A questionnaire was distributed to the participants and their reflections on the two days indicated that the workshop was an outstanding success. It brought together a group of people who have a focused interest and allowed them an opportunity to share their problems/successes. This, and the ability to interact with other attendees in a one-on-one basis, were the most satisfying aspects of the workshop. Several attendees suggested that another workshop be considered a year from now in order to follow up on the progress generated by this workshop. A summary of the sessions follows.

Network Administration

Since all of the participants represented large installations, administering a network was a common concern. Security and convenience were the main topics in this session. Security seemed to be more of a concern to commercial companies than the university attendees. These companies often have third party developers using their systems and they need to protect areas such as research and development from outside access. Procedures ranged from simple encryption to elaborate, and expensive, call back features on dial-in ports. Even in-house departments, such as personnel, need to take precautions to safeguard files. It was generally felt that the read, write, execute permissions standard to most systems don't offer the needed protection. Convenience issues in a networked environment centered on transparency and reliability. A user may want to use a laser writer attached to one of the systems on the network, but doesn't want to go through some convoluted sequence of commands to access the device. Backup and recovery are also features that should be kept transparent to the user. Invaluable is the system administrator who restores the "lost" file/directory while the user continues along his productive path. Reliability can be achieved through redundancy (expensive) or maintaining a common environment across all machines on the network. When a machine is going to be out of commission for a day or two, the affected user(s) can be moved to another machine in a matter of an hour or so.

Backup and Restore

The range of services and resources varied widely among the installations represented. Some sites only had enough disk space for a day or two of incremental dumps, while others kept as much as three weeks of information on-line. Once the information needed to be dumped to tape, 6250 drives with 3600 foot tapes were often used. As much as 225 megabytes of data can be stored on this medium. Most attendees indicated that after three years of use the tapes needed to be relegated to the circular file, while some installations are still using tapes that are ten years old. Interestingly enough, some installations did incremental dumps on active systems. Production environments, where down time to do incremental dumps is not possible, performed the backups while the systems were in use. The time slot assigned to this task was often in the middle of the night when use of the system was reduced, limiting the number of files which might be affected. Restoration of files/directories often required the system administrator's intervention because of the need for super user privileges.

Some sites had enough faith in their operations staff to allow them to have limited superuser privileges to carry out the recovery. One site charged a $25 fee, in hard cash, every time a restore operation was requested. The money was required up front, and a significant drop in requests was experienced when the policy was instituted.

Software Distribution and Synchronization

The ideal environment is one in which all hardware/software is identical across the network. Software distribution can be automated from a single server and installation insured to maintain commonality among all systems. Although such environments exist, we all know they are not the rule. The other extreme is to put out an update and let the various system administrators decide what they want to implement. Yes, a few of these installations were represented, as well. Most seemed to agree that a central source for software distribution is desirable and more efficient.

Mail and News

One company represented had over 5,000 machines worldwide they wanted to make sure could receive mail from the home office. Demanding that each machine have a common mail system enabled this to happen. Having a news facility to address questions/answers from the user community was implemented at another installation. As entries were aged off the system they were placed on a passive news system. Each time a submission was made, the entire data base was searched to determine if it had been addressed before. Reliability of mail systems was another issue. Can you be sure the mail reached its destination? Just because it didn't come bouncing back to you doesn't mean it didn't fall into a sinkhole. Responding to mail is one way to insure it was received, but do you respond to the response so that they respondee will know you received his response...? News has become the bane of some administrators' existences, while it is an inalienable right to others. The amount of resources expended on shipping various news groups around the world is staggering.

Accounts and Accounting

University environments constantly need to prune the password file of users who have become inactive. However, inactive users often surface at a later date and want their accounts re-activated. Automating a procedure to add, delete and re-activate users eases this task. Charge back accounting in a university often includes a variety of algorithms for students, staff, faculty, and even commercial accounts. Accounting files are often used to identify abuses in a system and correct them before they become too great of a problem. Students often find it a challenge to develop ways to "beat the system."

Reliability Enhancements

How does one insure the reliability of computer resources? One way is through redundancy of hardware, but this is expensive. Military applications and air traffic control operations are often forced to go this expensive route. An environment that consists of similar machines running the same software provides a fairly reliable situation. Moving users from system to system is simple and can be implemented when a certain machine is out of service for a lengthy period of time.

I/O Management

Remote device usage often requires that you know the path to the machine hosting that device. A presentation in this session described a network where each machine knew only the devices that were available locally, and passed unknown requests to a parent system in search of the device. The request was passed up the tree until it found the resource it needed, or arrived at the root node where the location of all devices in the network was kept. To expedite the request, a *.destination* file in the user's login directory could be used to identify the path to the remote device. Routing multiplexors were also discussed in this session. Users are able to connect to a variety of systems in the network through this method. The pros and cons of various multiplexor manufacturers' offerings were also discussed.

Summary

The true success of a meeting is gauged by feedback from the participants. Over 65% of the attendees took the time to fill out the questionnaire and the results were overwhelmingly positive. Administrators with common problems/solutions were brought together and the interaction afforded by a workshop of limited size was one of the most-often mentioned advantages they appreciated. Most of the solutions that were presented are in the public domain and available from the presenters. A proceedings consisting of the papers is available on request, as long as they last, from Rob Kolstad, CONVEX Computer Corporation, 701 Plano Road, Richardson, TX 75081.

John Donnelly

≡ ≡ ≡

Future Meetings

4th Computer Graphics Workshop Oct. 8 & 9, 1987, Cambridge, MA

Abstracts are due by May 1, 1987. For information, contact Tom Duff, the program chair, at *research!td* or (201) 582-6485.

USENIX 1988 Winter Conference and UniForum - Dallas

The USENIX 1988 Winter Conference will be held on February 10-12, 1988, at the Registry Hotel in Dallas, Texas. It will be concurrent with UniForum 1988, which will also be in Dallas. The Conference will feature tutorials and technical sessions.

USENIX 1988 Summer Conference and Exhibition - San Francisco

The USENIX 1988 Summer Conference and Exhibition will be held on June 21-24, 1988, at the Hilton Hotel in San Francisco, California. There will be a conference, tutorials, and vendor exhibits.

Long-term USENIX Conference Schedule

Jun	8-12	'87	Hyatt Regency, Phoenix, AZ
Feb	10-12	'88	Registry Hotel, Dallas, TX
Jun	21-24	'88	Hilton Hotel, San Francisco, CA
Feb	1- 3	'89	Town & Country Inn, San Diego, CA
Jun	13-16	'89	Hyatt Regency, Baltimore, MD
Feb		'90	Washington, DC
Jun	11-15	'90	Marriott Hotel, Anaheim, CA
Jun		'91	Nashville, TN

Letters to the Editor

Apparently-To: ukc!monul!auugn
From: phcomp!addw@munnari.oz
Date: Wed, 29 Apr 87 00:20:00 GMT

Hello John,

I've tried calling you a couple of times on the 'phone, no luck.

I am doing much of the work on the EUUG (European Unix Users' Group) newsletter, this is a recent thing -so I'm still learning.

This means that I am on the look out for interesting material to put into it. One of the things that I would like is to have a set of small regular columns (1 - 2 pages) on certain topics. These columns would cover a variety of things:

- Review of what is happening: Who, What, Where, When.
- Thoughts on what may happen. This may also include items that would lead to a discussion; ie "What do you guys think about ..."

I am looking for someone who would be a good columnist for what is happening in Australia. I want someone who would be likely to continue to do so for a reasonable period, ie establish a character to the column. As editor of the AUUGN you seemed a good place to start looking for someone, or could provide a few leaders.

The EUUG newsletter is a quarterly publication that is distributed to all members of the EUUG (European Unix Users Group). Copy dates (one month before publication):

1 February	for	1 March
1 May	for	1 June
1 August	for	1 September
1 November	for	1 December

Let me know what you think. Phone/email for a chat on it. I'll try & 'phone you again in the near future.

Thanx

Alain Williams
addw@phcomp   mcvax!ukc!phcomp!addw
+44-1-435-0200

Phil County -- phil@latcs1 has volunteered to do this task.

AUUGN Editor.

From: cccar@max.sait.oz (Chris Rusbridge)
To: john@moncskermit.oz
Date: Wed, 29 Apr 87 16:05:03 cst
Subject: AUUGN letter to editor

Re: The Claytons Unix Programmer

I was entertained by Greg Rose's presentation at the Unix conference, and again by his paper reprinted in AUUGN. However, one of the implications from his article clashes with my recent experience, and I believe it may be worth a comment.

Greg Rose makes the analogy between a computer runing Unix, and a car with a simple, N cylinder petrol engine. The implication is that Unix, is safe; it is standard; there are lots of people who know how to fix it if it goes wrong. In summary, business will buy Unix to solve their problems rather than take the risk of a steam turbine operating system.

I wish it were so. I think that commercial exploitation of Unix will be a good thing, especially if we can get it up into the large system bracket, running transaction processing tasks for large organisations. That *will* be the way to get a fair share of DP spending into the Australian market! However, I'm getting off my track a bit here.

We recently bought a couple of Unix systems. As usual, there were lots of factors in the choice; one was our lack of experience with Unix, while another was our desire to provide an environment similar to the commercial world. For these and other reasons, we bought a System V-based Unix system from one of the longest established vendors of Unix into the commercial world.

When we came to install these systems, I was quite horrified by the state in which they were supplied. It has gradually become apparent to us (as we painfully learn more about it), that security is totally absent from the system, as supplied. We have had to spend months of effort trying to tighten up to the extent that we could let our users loose. We have had to develop arcane skills, and can see that getting our very own Unix "guru" is going to be essential. I doubt our system is "safe" yet!

To go back to Greg Rose's analogy, we find we have bought a car with a 4 cylinder, petrol engine, but without brakes. The controls to this vehicle are so complex that we must hire a chauffeur to drive it.

If Unix is to be successful, I mean *really* successful in the sense that Greg meant it, the packaging needs to be improved out of sight. It must be possible to install the system in confidence that one's commercial secrets are secure. It should be possible to install software with little more expertise than is required for installing MS-DOS software.

Can this be achieved? From my present viewpoint, where the complexities and lack of standardisation in Unix loom very large, I rather doubt it. Is this a task that the Australian Unix community should take up? Since many of Greg's arguments in favour of Unix seem reasonable, and since Australia does seem to be at least in the vanguard of commercial exploitation of Unix, I would like to see this happen.

--

Chris Rusbridge, Academic Computing Service Manager
S. A. Institute of Technology.
ACSnet: cccar@max.sait.oz
Phone: +61 8 343 3098 Telex: AA82565
Post: PO Box 1, Ingle Farm, SA 5098 Australia

AT&T

AT&T Unix Pacific Co.,Ltd.
No. 1 Nan-oh Bldg., 5th Fl.
2-21-2, Nishi-Shinbashi
Minato-ku, Tokyo 105 Japan
Tel : 03-431-3305
Telefax : 03-431-3680
Telex : J34936 ATTUP

March 17, 1987

Dear User Group,

We would like to add a section to our $ echo describing the recent activities as well as the plans of user groups in our region. We hope that this will help give the user groups more exposure, and a chance to know what each other is doing.

We would appreciate if you could send us a few paragraphs on what your group has been doing, and any immediate plans. Last issue we ran a list of contact information for the user groups in this region. If we get a good response to writing about user groups, we would like to continue doing this. Let me know if you have any questions, or have any suggestions on what you would like to see in $ echo.

Sincerely,

[illegible]

Ryerson E. Schwark
Account Executive
Software Licensing

Tokyo-Japan-RS-hy

Enc.
$ echo

Computer Centre
Monash University
Clayton, Victoria 3168
AUSTRALIA

Monday 15th June, 1987

Ryerson E. Schwark
Account Executive - Software Licencing
AT&T Unix Pacific Co., Ltd.
No. 1 Nan-oh Bld., 5th Floor
2-21-2, Nishi-Shinbashi
Minato-ku, Tokyo 105 JAPAN

Dear Sir,

Thank you for you letter asking us for information about the AUUG to be published in *$echo*.

I have passed your request on to Phil County who is producing a similar column for the EUUGN.

I also read an the article in the February 1987 issue of *$echo* by James Arnold entitled *Shared Libraries on UNIX System V.* I feel that AUUGN readers whould be interested in such developments.

Please would AT&T give permission to reproduce this article and any other articles I find appropriate in the future.

Yours Faithfully,

John Carey

John Carey,
AUUGN Editor.

AT&T

AT&T Unix Pacific Co.,Ltd.
No. 1 Nan-oh Bldg., 5th Fl.
2-21-2, Nishi-Shinbashi
Minato-ku, Tokyo 105 Japan
Tel : 03-431-3305
Telefax : 03-431-3680
Telex : J34936 ATTUP

May 18, 1987

Dr. Ken J. McDonell - President, Australian UNIX Systems User Group
Mr. Patrick Ong - President, Singapore UNIX User Group
Mr. Sung Yang Bang - President, Korean UNIX User Group
Mr. Ian M. Howard - President, New Zealand UNIX Systems
Prof. Kanchana Kanchanasut - President, Thainix
Mr. Stan Shih - Taipei Computer Association
Dr. Teo Sen Chong - President, Malnix

Dear Mr. Messrs:

AT&T Unix Pacific Co., Ltd. is pleased to offer its first UNIX* System Software Technology Seminar in Hong Kong on July 9th and 10th of this year. We will be bringing some of the leading developers and researchers in UNIX System technology to discuss the current innovations and future directions of the UNIX Operating System. We believe that the information will prove informative and useful to you.

I have enclosed more infomation on the speakers as well as registration forms for you and anyone you think might be interested. Your cooperation in supporting this seminar is highly appreciated. If you need more information, please feel free to contact us.

We look forward to your participation.

Sincerely,

[illegible]

Ryerson E. Schwark
Account Executive
Software Licensing

Tokyo-Japan-RS-hy

* Registered Trademark of AT&T in the USA and other countries.

Computer Centre
Monash University
Clayton, Victoria 3168
AUSTRALIA

Monday 15th June, 1987

Ryerson E. Schwark
Account Executive - Software Licencing
AT&T Unix Pacific Co., Ltd.
No. 1 Nan-oh Bld., 5th Floor
2-21-2, Nishi-Shinbashi
Minato-ku, Tokyo 105 JAPAN

Dear Sir,

Thank you for sending the AUUG information and application forms for the 1987 UNIX* System Technology Seminar being held in July.

Unfortunately we are unable to pass these on to our members because of the following reasons:-

We need about 200 copies if we where to distribute copies to all our members with the Newsletter. We only received approximately 30 copies from you.

We need the information send to us with plenty of lead time as out Newsletter is published every two months. The Seminar deadline was before the next Newsletter will reach our members.

Also we normally ask people to pay AUD $200 for advertiseing in the AUUGN. In your case it is borderline whether it is advertising or a community announcement.

We would prefer that you bought our mailing list and distributed your material directly to our members.

Yours Faithfully,

John Carey

John Carey,
AUUGN Editor.

* UNIX is a trademark of AT&T in the USA and other countries.

/usr/group

The International Network of UNIX Users

February 27, 1987

Dear UNIX Vendor:

The 1987 UNIX Products Directory has been produced and your company has helped to make it the largest publication devoted entirely to UNIX and UNIX-based products and services.

The 1987 Directory is your resource tool to the UNIX systems market.

. 3,164	UNIX Products and Services
. 826	UNIX-Specific Vendors
. 1,587	Vertical and Horizontal Software Packages
. 732	System Software and Development Tools
. 104	Publications-Books, Magazines, Newsletters

Refer to the attached sheet for additional details. Call or write /usr/group for ordering information. Because usr/group members receive a 50% discount, you may choose to become a /usr/group member. Bulk purchase rates are also available.

Sincerely,

Joe W. Fagenstrom

Joe W. Fagenstrom
Members Services/Marketing Manager

enclosure

UNIX is a trademark of AT&T

4655 Old Ironsides Drive, Suite 200 Santa Clara, CA 95054 (408) 986-8840

NORTH-HOLLAND

POSTAL ADDRESS:
P.O. Box 1991
1000 BZ Amsterdam
The Netherlands

Telephone (020) 5862 911
Telex 10704 ESPOM NL
Cables Espom

OFFICE ADDRESS:
Sara Burgerhartstraat 25
1055 KV Amsterdam
The Netherlands

Secretariat
Australian Unix User Group
AUUG
P.O. Box 366
Kensington
2033 NEW SOUTH WALES
Australia

Direct line (020)5862473 Amsterdam, March 24, 1987

RE: X/OPEN PORTABILITY GUIDE, 2nd edition 1987

Dear Secretariat ,

As you are probably aware of, the 1987 edition of the X/OPEN PORTABILITY GUIDE was recently published. The interest in the new revised and expanded edition is overwhelming; this was especially true during CEBIT ´87 in Hannover.

We feel confident that the members of the Australian Unix User Group AUUG would be interested in obtaining a copy of the Guide and therefore we like to ask you to enclose a copy of the brochure as per enclosed specimen with a mailing to your members.

Please let me have your reaction as soon as possible, stating the number of brochures you need for this purpose. We will immediately supply you with the required quantity together with a complimentary copy of the 1987 edition of the X/OPEN PORTABILITY GUIDE.

I look forward to hearing from you.

With kind regards,

Yours sincerely,

[illegible]

Joop Dirkmaat
Marketing Manager

JD/td

INFORMATION & BUSINESS DIVISION
a branch of
ELSEVIER SCIENCE PUBLISHERS B.V.

THIS PAGE INTENTIONALLY LEFT BLANK

AUUG

Membership Categories

Once again a reminder for all "members" of AUUG to check that you are, in fact, a member, and that you still will be for the next two months. This is especially important just now, as membership rates in AUUG are set to rise, this will be your last chance to join, or rejoin at the old rate.

There are 4 membership types, plus a newsletter subscription, any of which might be just right for you.

The membership categories are:

Institutional Member
Ordinary Member
Student Member
Honorary Life Member

Institutional memberships are primarily intended for university departments, companies, etc. This is a voting membership (one vote), which receives two copies of the newsletter. Institutional members can also delegate 2 representatives to attend AUUG meetings at members rates. AUUG is also keeping track of the licence status of institutional members. If, at some future date, we are able to offer a software tape distribution service, this would be available only to institutional members, whose relevant licences can be verified.

If your institution is not an institutional member, isn't it about time it became one?

Ordinary memberships are for individuals. This is also a voting membership (one vote), which receives a single copy of the newsletter. A primary difference from Institutional Membership is that the benefits of Ordinary Membership apply to the named member only. That is, only the member can obtain discounts on attendance at AUUG meetings, etc, sending a representative isn't permitted.

Are you an AUUG member?

Student Memberships are for full time students at recognised academic institutions. This is a non voting membership which receives a single copy of the newsletter. Otherwise the benefits are as for Ordinary Members.

Honorary Life Memberships are a category that isn't relevant yet. This membership you can't apply for, you must be elected to it. What's more, you must have been a member for at least 5 years before being elected. Since AUUG is only just approaching 3 years old, there is no-one eligible for this membership category yet.

Its also possible to subscribe to the newsletter without being an AUUG member. This saves you nothing financially, that is, the subscription price is the same as the membership dues. However, it might be appropriate for libraries, etc, which simply want copies of AUUGN to help fill their shelves, and have no actual interest in the contents, or the association.

Subscriptions are also available to members who have a need for more copies of AUUGN than their membership provides.

To find out if you are currently really an AUUG member, examine the mailing label of this AUUGN. In the lower right corner you will find information about your current membership status. The first letter is your membership type code, N for regular members, S for students, and I for institutions. Then follows your membership expiration date, in the format exp=MM/YY. The remaining information is for internal use.

Check that your membership isn't about to expire (or worse, hasn't expired already). Ask your colleagues if they received this issue of AUUGN, tell them that if not, it probably means that their membership has lapsed, or perhaps, they were never a member at all! Feel free to copy the membership forms, give one to everyone that you know.

If you want to join AUUG, or renew your membership, you will find forms in this issue of AUUGN. Send the appropriate form (with remittance) to the address indicated on it, and your membership will (re-)commence.

As a service to members, AUUG has arranged to accept payments via credit card. You can use your Bankcard (within Australia only), or your Visa or Mastercard by simply completing the authorisation on the application form.

Robert Elz

AUUG Secretary.

AUUG

Application for Ordinary, or Student, Membership Australian UNIX* systems Users' Group.

*UNIX is a registered trademark of AT&T in the USA and other countries

To apply for membership of the AUUG, complete this form, and return it with payment in Australian Dollars, or credit card authorisation, to:

AUUG Membership Secretary
P O Box 366
Kensington NSW 2033
Australia

- Please don't send purchase orders — perhaps your purchasing department will consider this form to be an invoice.
- Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.

I, ... do hereby apply for

☐ Renewal/New*	Membership of the AUUG	$50.00	
☐ Renewal/New*	Student Membership	$30.00	(note certification on other side)
☐ International Surface Mail		$10.00	
☐ International Air Mail		$50.00	
Total remitted			AUD$______ (cheque, money order, credit card)

* Delete one.

I agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months commencing on the first day of the month following that during which this application is processed.

Date: __/__/__ Signed: ____________________

☐ Tick this box if you wish your name & address withheld from mailing lists made available to vendors.

For our mailing database - please type or print clearly:

Name: .... Phone: .... (bh)

Address: .... (ah)

....

Net Address: ....

....

.... *Write "Unchanged" if details have not altered and this is a renewal.*

....

Please charge $_____ to my ☐ Bankcard [redacted] ☐ Mastercard.

Account number: ______ ______ ______ ______. Expiry date: __/__.

Name on card: ______________________________ Signed: ____________________

Office use only:

Chq: bank ________ *bsb* ___-___ *a/c* ____________ *#* ______________

Date: __/__/__ *$* *CC type* ___ *V#* __________

Who: ____________ *Member#* ________

Student Member Certification *(to be completed by a member of the academic staff)*

I, ... certify that

.... *(name)*

is a full time student at .... *(institution)*

and is expected to graduate approximately ___/___/___.

Title: __________________________________ Signature: __________________

AUUG

Application for Institutional Membership Australian UNIX* systems Users' Group.

*UNIX is a registered trademark of AT&T in the USA and other countries.

To apply for institutional membership of the AUUG, complete this form, and return it with payment in Australian Dollars, or credit card authorisation, to:

AUUG Membership Secretary
P O Box 366
Kensington NSW 2033
Australia

● Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.

.... does hereby apply for

☐ New/Renewal* Institutional Membership of AUUG	$250.00
☐ International Surface Mail	$ 20.00
☐ International Air Mail	$100.00
Total remitted	AUD$______ (cheque, money order, credit card)

* Delete one.

I/We agree that this membership will be subject to the rules and by-laws of the AUUG as in force from time to time, and that this membership will run for 12 consecutive months commencing on the first day of the month following that during which this application is processed.

I/We understand that I/we will receive two copies of the AUUG newsletter, and may send two representatives to AUUG sponsored events at member rates, though I/we will have only one vote in AUUG elections, and other ballots as required.

Date: __/__/__ Signed: ____________________

Title: ____________________

☐ Tick this box if you wish your name & address withheld from mailing lists made available to vendors.

For our mailing database - please type or print clearly:

Administrative contact, and formal representative:

Name: .... Phone: .... (bh)

Address: .... (ah)

.... Net Address: ....

....

.... *Write "Unchanged" if details have not altered and this is a renewal.*

....

Please charge $_____ to my/our ☐ Bankcard [redacted] ☐ Mastercard.

Account number: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _. Expiry date: __/__.

Name on card: ______________________________ Signed: ____________________

Office use only: **Please complete the other side.**

Chq: bank ________ *bsb* ___-___ *a/c* ______________ *#* ______________

Date: __/__/__ *$* *CC type* ___ *V#* __________

Who: ____________ *Member#* ________

Please send newsletters to the following addresses:

Name:	....	Phone:	....	(bh)
Address:	....		....	(ah)
	....	Net Address:	....	
	....			
	....			
	....			

Name:	....	Phone:	....	(bh)
Address:	....		....	(ah)
	....	Net Address:	....	
	....			
	....			
	....			

Write "unchanged" if this is a renewal, and details are not to be altered.

Please indicate which Unix licences you hold, and include copies of the title and signature pages of each, if these have not been sent previously.

Note: Recent licences usally revoke earlier ones, please indicate only licences which are current, and indicate any which have been revoked since your last membership form was submitted.

Note: Most binary licensees will have a System III or System V (of one variant or another) binary licence, even if the system supplied by your vendor is based upon V7 or 4BSD. There is no such thing as a BSD binary licence, and V7 binary licences were very rare, and expensive.

☐ System V.3 source	☐ System V.3 binary
☐ System V.2 source	☐ System V.2 binary
☐ System V source	☐ System V binary
☐ System III source	☐ System III binary
☐ 4.2 or 4.3 BSD source	
☐ 4.1 BSD source	
☐ V7 source	

☐ Other *(Indicate which)* ....

AUUG

Application for Newsletter Subscription
Australian UNIX* systems Users' Group.

*UNIX is a registered trademark of AT&T in the USA and other countries

Non members who wish to apply for a subscription to the Australian UNIX systems User Group Newsletter, or members who desire additional subscriptions, should complete this form and return it to:

AUUG Membership Secretary
P O Box 366
Kensington NSW 2033
Australia

- Please don't send purchase orders — perhaps your purchasing department will consider this form to be an invoice.
- Foreign applicants please send a bank draft drawn on an Australian bank, or credit card authorisation, and remember to select either surface or air mail.
- Use multiple copies of this form if copies of AUUGN are to be dispatched to differing addresses.

Please *enter / renew* my subscription for the Australian UNIX systems User Group Newsletter, as follows:

Name: .... Phone: .... (bh)

Address: .... (ah)

.... Net Address: ....

....

.... *Write "Unchanged" if address has*

.... *not altered and this is a renewal.*

For each copy requested, I enclose:

☐ Subscription to AUUGN	$ 50.00
☐ International Surface Mail	$ 10.00
☐ International Air Mail	$ 50.00

Copies requested (to above address) ________

Total remitted AUD$______

(cheque, money order, credit card)

☐ Tick this box if you wish your name & address withheld from mailing lists made available to vendors.

Please charge $_____ to my ☐ Bankcard [redacted] ☐ Mastercard.

Account number: _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _. Expiry date: __/__.

Name on card: ______________________ Signed: ______________________

Office use only:

Chq: bank ________ *bsb* ____-____ *a/c* ______________ *#* ______________

Date: __/__/__ *$* *CC type* ___ *V#* __________

Who: ____________ *Subscr#* _________

AUUG

Notification of Change of Address
Australian UNIX* systems Users' Group.

*UNIX is a registered trademark of AT&T in the USA and other countries.

If you have changed your mailing address, please complete this form, and return it to:

AUUG Membership Secretary
P O Box 366
Kensington NSW 2033
Australia

Please allow at least 4 weeks for the change of address to take effect.

Old address (or attach a mailing label)

Name:	Phone: .... (bh)
Address:	.... (ah)
....	Net Address: ....
....	
....	
....	

New address (leave unaltered details blank)

Name:	Phone: .... (bh)
Address:	.... (ah)
....	Net Address: ....
....	
....	
....	

Office use only:

Date: ___/___/___

Who: ____________ *Memb#* _________