33

AIX

July 1998

In this issue

update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 550955
From USA: 01144 1635 33823
E-mail: HarryLewis@compuserve.com

North American office

Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75067
USA
Telephone: 940 455 7050

Contributions

If you have anything original to say about AIX, or any interesting experience to recount, why not spend an hour or two putting it on paper? The article need not be very long – two or three paragraphs could be sufficient. Not only will you actively be helping the free exchange of information, which benefits all AIX users, but you will also gain professional recognition for your expertise and that of your colleagues, as well as being paid a publication fee – Xephon pays at the rate of £170 ($250) per 1000 words for original material published in AIX Update.

To find out more about contributing an article, see *Notes for contributors* on Xephon's Web site, where you can download *Notes for contributors* in either text form or as an Adobe Acrobat file.

Editor

Harold Lewis

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, scripts, and other contents of this journal before making any use of it.

Subscriptions and back-issues

A year's subscription to *AIX Update*, comprising twelve monthly issues, costs £175.00 in the UK; $265.00 in the USA and Canada; £181.00 in Europe; £187.00 in Australasia and Japan; and £185.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the November 1995 issue, are available separately to subscribers for £15.00 ($22.50) each including postage.

AIX Update on-line

Code from *AIX Update* is available from Xephon's Web page at www.xephon.com (you'll need the user-id shown on your address label to access it).

Printed in England.

Utility to purge old e-mail

PROBLEM

As an AIX administrator, you may occasionally have problems with the inboxes directory, which is normaly located at */var/spool/mail*. When this directory fills up, **sendmail** may hang and refuse connections and reject messages, giving the reason as 'insufficent space'. This not only results in the loss of potentially important messages, but also wastes bandwidth whenever the **mailer** daemon sends messages back to the sender to inform them of the problem. You can sometimes solve this problem temporarily by increasing the space available in */var*, by perhaps doubling or tripling its size. However, as you haven't addressed the root of the problem, it'll probably recur.

This article presents some solutions to this problem, including a utility to manage disk space. Of course, the utility doesn't represent a permanent solution, though it allows you to control the use of disk space for mail boxes.

CAUSES

There are several causes of the problem of runaway mailboxes, including:

- Users failing to check their mailboxes and read their e-mail.
- Users failing to delete e-mail that's been read and is not required.
- Users leaving an excessive number of read e-mails in their inbox for future reference.
- Inboxes belonging to user accounts that have expired and been deleted from the system – the inbox should have been deleted along with the user account.
- POP clients that use the option 'leave a copy on the server' – this option is available on Netscape Navigator, Internet Explorer, and other mail clients. I find this is the most common cause of running out of disk space for mail clients.

SOLUTIONS

There are many solutions to this problem, including:

- Increasing the disk space available to the */var* filesystem. This begs the question 'how much space is enough?' The answer to this question depends on the number of users you support and how active they are.
- Run an 'awareness campaign' to alert users of the importance of reading their inboxes, saving e-mail they want to keep, and deleting ones they don't want. Users should be taught the importance of conserving system resources. Moreover, they should be made aware that the disk space they use is a shared resource used by all other users on the system.
- Check the mail directory (*/var/spool/mail*) manually and move large inboxes to a temporary space. This can be done using the commands:

  ```
  cd /var/spool/mail
  ```

  and:

  ```
  ls -l > /tmp/mail.dir
  ```

  This will take some time to complete. Use the command below to sort inboxes by size

  ```
  sort -nr +4 /tmp/mail.dir > /tmp/mail.dir.sort
  ```

  and view the results in */tmp/mail.dir.sort*.
- Run the utility below to free some disk space.

THE UTILITY

The utility consists of two components – a C program, which determines the number of lines to be deleted from each inbox, and a shell script, which calls the C program and performs a number of tasks and commands. By specifying a date, the utility can be used to delete e-mail older than this date from inboxes.

A WORD OF CAUTION

Make sure that you have a recent back-up of the */var/spool/mail* directory before you run this utility. It would also be a good idea to inform users that you're planning to run the utility and purge their e-mail in future – this gives users a chance to save important messages in their inboxes. It would be an even better idea to schedule this utility to run at a certain time each month, say during preventive maintenance, and to ensure that users are aware of this.

IMPORTANT OPERATION NOTES

- I've called the C program *purge.c*, the executable *purge*, and the shell script *purge.csh*. You may rename them to whatever you wish, though you should make sure that the file names you use are reflected in the source code and you should also use a consistent naming convention.
- There are a few steps that have to be taken prior to running this utility:
  - Enter the target date by editing the shell script (some administrators may want to handle this by means of an argument in the script).
  - Make sure that no users are accessing or are able to access their inboxes during the purge process, as permissions to the directory that contains inboxes need to be changed. You can do this by a number of means, including making users log off, killing **inetd**, or disabling modems.
  - Stop **sendmail**. Doing this ensures that no e-mail is lost, as the sender's mail service defers in-coming e-mail, re-trying after a pre-defined interval.
- The time needed for the utility to finish executing depends on the size of your */var/spool/mail* directory. For example, it takes around 40 minutes to reduce a 2 GB mail directory with about 3,200 inboxes from 80% to 45% full.
- There is no need to reboot the system after running the utility. However, it is better to stop other processes on the system for

faster execution. This can be done using the commands:

```
kill -STOP process#
```

and

```
kill -CONT process#
```

- Make sure that the free space in */tmp* is at least twice the size of your largest inbox.
- The utility creates the following temporary files:
  - *uservar*. This is located in the same directory as the utility. It contains information about messages to be deleted from the inbox being processed.
  - */tmp/usermail*. This is located in */tmp* and contains a copy of the original state of the inbox being processed.
  - */tmp/usermail-tmp*. This is located in /tmp, and is the inbox after processing.
- The utility logs its activity in */tmp/logfile*. This file contains the date and time stamps of when the utility started and finished running, also containing the names of files processed and their size before and after e-mail is purged.
- You have to make sure that the correct path names are set in the C program and shell script if they are different in your environment.
- You need *root* authority to run this utility.
- Run the utility using the command below from the directory where the utility resides.

  ```
  nohup ./purge.csh &
  ```

- Once finished, make sure that the system is once more available to users by starting **inetd** and plugging modems back in.
- Start **sendmail**.

THE C PROGRAM, PURGE.C

To compile a list of lines to be deleted from each inbox, **purge**

analyses each line using a process that tries to match patterns used by **sendmail** to separate e-mail messages in the inbox file. The pattern is as follows:

```
From ..... day month xx hh:mm:ss year
```

Where:

- *day* is a member of: {Sat, Sun, Mon, Tue, Wed, Thu, Fri}.
- *month* is a member of: {Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec}.
- *xx* is the day of the month.
- *hh:mm:ss* is the time.
- *year* is the year in four-digit format.

In addition, each e-mail is also separated by a single empty line (except for the first e-mail, which starts at the beginning of the file).

PURGE.C

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
/* initialization of parameters */
int ch,i=0,j=0,k=0,indx=0,foundit=0;
int mth,dat,yr,mth1,dat1,yr1;
char str1[10000],str3[10000],fcap='F',ch1;

void date_capture();
void write_to_file(int value);

int main(int argc, char *argv[])
{
        int n=10000;
        FILE *fin;

        if(argc == 1)
                exit(0);

        mth=atoi(argv[1]);
        dat=atoi(argv[2]);
        yr=atoi(argv[3]);
```

```
        if ((mth < 1) || (mth > 12))
                exit(0);
        if ((dat < 1) || (dat > 31))
                exit(0);
        if (yr == 0)
        exit(0);
/* the file /tmp/usermail is a copy of the user's inbox
   copied by means of the shell script */
        fin=fopen("/tmp/usermail","r");
        if(fin == NULL)
        {
                printf("Error opening input file!\n");
                exit(0);
        } /* end of if */

        for (indx=0;indx < i;indx++)
        {
                str1[indx]=0;
                str3[indx]=0;
        } /* end of for */

        while (fgets(str1,n,fin) != NULL)
        {
                j++;
                i=strlen(str1);
                if ((str1[0] == fcap ) && (str1[1] == 'r' )
                  && (str1[2] == 'o' ) && (str1[3] == 'm' )
                  && (str1[4] == ' ' ) && (isdigit(str1[i-2]) != 0))
                {
                        date_capture();
                        if (yr < yr1)
                        {
                             if (j ==1)
                             {
                                write_to_file(0);
                                exit(0);
                             }
                             else
                             {
                                write_to_file(j-1);
                                exit(0);
                             }
                        }
                        else if (yr == yr1)
                        {
                                if(mth < k)
                                {
                                        if (j == 1)
                                        {
                                           write_to_file(0);
```

```
                                        exit(0);
                                }
                                else
                                {
                                        write_to_file(j-1);
                                        exit(0);
                                }
                        }
                        else if (mth == k)
                        {
                                if (dat < dat1)
                                {
                                        write_to_file(j-1);
                                        exit(0);
                                }
                                else if(dat >= dat1)
                                {
                                        ;
                                }
                        }
                        else
                        {
                                ;
                        }
                }
                else ;
        }
        else ;

        for (indx=0;indx < i;indx++)
        {
            str1[indx]=0;
            str3[indx]=0;
        } /* end of for */

    } /* end of while */
/* j contains number of lines to be deleted in the file and
   will be written in uservar */
    write_to_file(j);
    fclose(fin);
    return 0;
} /* end of main */

void date_capture()
{
    char *word1,*word2,*word3,*word4,*word5,*word6,*word7;
    word1=strtok(str1," ");
    word2=strtok(NULL," ");
    word3=strtok(NULL," ");
    word4=strtok(NULL," ");
```

```
        word5=strtok(NULL," ");
        word6=strtok(NULL," ");
        word7=strtok(NULL," ");
        dat1=atoi(word5);
        yr1=atoi(word7);
        if (strstr(word4,"Jan") != NULL)
                k = 1;
        else if (strstr(word4,"Feb") != NULL)
                k = 2;
        else if (strstr(word4,"Mar") != NULL)
                k = 3;
        else if (strstr(word4,"Apr") != NULL)
                k = 4;
        else if (strstr(word4,"May") != NULL)
                k = 5;
        else if (strstr(word4,"Jun") != NULL)
                k = 6;
        else if (strstr(word4,"Jul") != NULL)
                k = 7;
        else if (strstr(word4,"Aug") != NULL)
                k = 8;
        else if (strstr(word4,"Sep") != NULL)
                k = 9;
        else if (strstr(word4,"Oct") != NULL)
                k = 10;
        else if (strstr(word4,"Nov") != NULL)
                k = 11;
        else
                k = 12;
} /* end of void date_capture() */

void write_to_file(int value)
{
        FILE *fout;
/* uservar contains the number of lines to be deleted from the inbox */
        fout=fopen("uservar","w");
        if(fout == NULL)
        {
                printf("Error opening output file!\n");
                exit(0);
        } /* enf of if(fout == NULL) */
        fprintf(fout,"%d\n",value);
        fclose(fout);
} /* end of void write_to_file(int value) */
```

THE SCRIPT, PURGE.CSH

The script starts by setting certain parameters that should be checked carefully. It also changes permissions to */var/spool/mail* to ‘000’ to

prevent access to the directory during execution.

Next, it cleans up old files – if any – that remain from a previous execution of the command. During execution, the script writes a log of its activity to *logfile*, beginning with the current date and time, name of the file being processed, its size, how many lines are to be deleted, the new size of the file, and finally date and time when it finishes processing the file.

The script copies the inbox under processing to a temporary space, copying the processed version back to its original place and resetting its permissions and ownership to their original values. **sendmail** creates inboxes is using the following format:

```
-rw-rw---- 1 username mail size date&time username
```

The last step of processing is to restore the permissions of the directory containing mail boxes to their original state:

```
drwxrwsr-x   2 bin      mail       56320 date&time /var/spool/mail
```

PURGE.CSH

```
#!/bin/csh
set tmpdir="/tmp"
set dirs="/var/spool/mail"
set logfile="/tmp/logfile"

chmod 000 $dirs
/usr/bin/rm -f $logfile
/usr/bin/rm -f nohup.out
/usr/bin/rm -f uservar

echo Job started at `date` >> $logfile
echo "-------------------------------------------------" >> $logfile
echo Removing zero size inboxes >> $logfile
/usr/bin/find $dirs -size 0 -exec rm {} \;
echo Done removing zero size inboxes >> $logfile

foreach file (`ls ${dirs}`)
      echo "-------------------------------------------------" >>
$logfile
      set f_size=`ls -l $dirs/${file} | awk '{print $5}'`
      echo File name $file with size of $f_size >> $logfile
      if ( ${f_size} == 0 ) then
         echo ${file} has file size 0  Will be removed >> $logfile
         /usr/bin/rm -f $dirs/${file}
```

```
      else
        echo WORKING ON $file >> $logfile
# Copy the inbox to /tmp/usermail to work on it.
        /usr/bin/cp $dirs/$file $tmpdir/usermail
# ------------------------------------------------------
# Calling the "purge" program
# Change the date below to mm dd yyyy
# where mm is the month (1, 2, ..., 12)
#       dd is the day of the month (1, 2, ..., 31)
#       yyyy is the year (1998, 1999, ...)
# ------------------------------------------------------
        /"path_of_the_utility"/purge mm dd yyyy
# ------------------------------------------------------
# the file "uservar" contains the number of lines to be deleted
# from the inbox
        set lines=`cat uservar`
        if ($lines == 0) then
            echo There is no mail to purge in $file >> $logfile
        else
          echo There are $lines lines to purge in $file >> $logfile
          sed 1,${lines}d $tmpdir/usermail > $tmpdir/usermail-tmp
          /usr/bin/cp $tmpdir/usermail-tmp $dirs/${file}
          set f_sizenew=`ls -l $dirs/${file} | awk '{print $5}'`
          echo New size of $file is $f_sizenew >> $logfile
          if ( ${f_sizenew} == 0 ) then
             echo Removing file ${file} after mail purging >> $logfile
             /usr/bin/rm -f $dirs/${file}
          endif
        endif
# Clearing temporary files
        /usr/bin/rm -f $tmpdir/usermail $tmpdir/usermail-tmp
# Getting back the processed inbox into shape
        chmod 660 $dirs/$file
        chown $file $dirs/$file
        chgrp mail $dirs/$file
        echo JOB DONE WITH $file >> $logfile
      endif
end
chmod 775 $dirs
echo "------------------------------------------------" >> $logfile
echo Job ended at `date` >> $logfile
exit
```

Iyad Al-Bukhari
System Administrator
King Fahd University (Saudi Arabia)

Date and time manipulation

This article presents a number of utilities that we developed as part of the process of building systems to automate and monitor a number of crucial tasks on our RS/6000s. It also presents some examples to illustrate the way the tools are used. All the tools are for calculating and manipulating the date and time of these tasks. I have modified certain aspects of them in this article, as their original implementation was very much biased to our individual requirements. Therefore the ideas discussed are presented in a series of examples, so that you can decide the best way to implement them at your own installation.

I start by covering the two major scripts for calculating dates and times. These scripts are **calcdate_diff** and **calcdate_time**. I then present sample code that shows the way these scripts are used at our site. Therefore the article is split into three sections:

1 The **calcdate_diff** script

2 The **calcdate_time** script

3 Examples on implementing the scripts.

CALCDATE_DIFF SCRIPT

calcdate_diff is the main script that's used in all the examples provided later. Given its importance, I'll cover all its available options in this article. The script checks to ensure that the correct date syntax is used in the various options, but you have to develop your own error checking for any subsequent problems. The main criterion is that all dates must be in a *dd/mm/yyyy* or *d/m/yyyy* syntax. The script checks for a valid date (in the correct format) before it attempts any calculation on it. In the options list that follows the **-d**, **-m**, and **-y** flags can be used instead of any of the output formatting options below. Use the command **calcdate_diff ?** at a command prompt to produce a list of options and usage instructions. The date range supported is between the year 1900 and 2400. The criterion used to decide if the year is a leap year is that a leap year occurs on years divisible by four unless the year

is also divisible by 100. The second rule is overridden if the year is also divisible by 400 (see also *Leap year programming in AIX, AIX Update* Issue 24, October 1997).

OUTPUT FORMAT OPTIONS

-n	Day number in the range 0-6, where 0 is Sunday
-nn	Short day name (eg 'Mon')
-nnn	Long day name (eg 'Monday')
-d	Day of month with a leading zero (0)
-de	Day of month with an 'extension' (eg '21st')
-m	Month number with a leading zero (0)
-mm	Short month name (eg 'Jan')
-mmm	Long month name (eg 'January')
-y	Short year format (eg '97' – for those determined to write year 2000 bugs, even at this late stage)
-yy	Long year (eg '1997')
-j	Julian day
-yj	Year and julian day (eg '97261')
-sp	Output in the format *mm\/dd\/yy*
-s	Uses a space as the output separator
-/	Uses a forward slash (/) as the output separator.

HOW TO USE THE OPTIONS

The default option

```
calcdate_diff 1 -d -m -y
```

This returns the day before the current system date in *ddmmyy* format. For example, if the system date is June 3, 1998 the output is 020698. The first positional parameter can be positive or negative; a positive

value results in a past date being returned and a negative value results in a future date being returned.

The -c option

```
calcdate_diff -c dd/mm/yyyy dd/mm/yyyy
```

This results in the number of days between the two given dates being returned. For example, the command below returns a value of 1:

```
calcdate_diff -c 01/01/1997 02/01/1997
```

The first positional parameter must be **-c** to specify this option. Problems occur if the second date (the third positional parameter) is not greater or equal to the first date (the second positional parameter).

The -cd option

```
calcdate_diff -cd dd/mm/yyyy 1 -d -m -y
```

This option carries out a calculation based on the date input at the command line rather than using the current system date. The first positional parameter must be **-cd** to specify this option. This command can calculate future or past dates from the date given. A negative figure in the third positional parameter returns dates after the date supplied in the command and a positive value returns dates prior to the date supplied in the command.

The -v option

```
calcdate_diff -v dd/mm/yyyy
```

This option is used to validate the date provided in the command, returning a zero (0) if the date is valid. The first positional parameter must be **-v** to specify this option. The format of the given date has to be either *dd/mm/yyyy* or *d/m/yyyy*, otherwise an error occurs.

The -g option

```
calcdate_diff -g dd/mm/yyyy dd/mm/yyyy
```

This calculates the number of days between the first date (the second positional parameter) and the second date (the third positional parameter) provided. The first positional parameter must be **-g** to specify this option. If the second date is earlier than the first, a negative

value is returned. If the second date is after the first, a positive value is returned.

The -mc option

```
calcdate_diff -mc mmm
```

This converts the given month to the month number with a leading zero (0). Thus the command below returns the value ‘05’.

```
calcdate_diff -mc May
```

The first positional parameter must be **-mc** to specify this option. Note that the command expects the month in either short name format (‘Aug’) or in full name format (‘August’).

GENERAL CONSIDERATIONS

The options specified must be used exactly as discussed above, otherwise unpredictable results may occur. The formatting options can be specified in any order, and their order in the command line determines the format of the output date. A total of eight positional parameters is allowed; extending this requires the code to be modified.

CALCDATE_DIFF SCRIPT

```
#!/bin/ksh
##############################################################################
#                                                                            #
#       Written by :    Robert Russell                                       #
#       Use :           Date calculations                                    #
#       Copyright :     Robert Russell (1998)                                #
#       Date range:     1900 to 2400                                         #
#                                                                            #
##############################################################################

integer gto
integer DATEJ
integer YEAR
integer TOTAL_DAYS
integer TOTAL_1
integer TOTAL_2
integer OUTPUT_3
integer LEAP_DAYS
integer NUM_WEEKS
integer CALC_WEEKS
```

```
integer DIFF_DAYS
integer DAY_INT
integer jul
integer year
integer lyear
integer diff
integer LEAR_TOT
integer YEAR1
integer DAY1
integer YEAR2
integer DAY2

MINUS=$1
filedat[1]=31
filedat[2]=28
filedat[3]=31
filedat[4]=30
filedat[5]=31
filedat[6]=30
filedat[7]=31
filedat[8]=31
filedat[9]=30
filedat[10]=31
filedat[11]=30
filedat[12]=31
PASS[1]=$1
PASS[2]=$2
PASS[3]=$3
PASS[4]=$4
PASS[5]=$5
PASS[6]=$6
PASS[7]=$7
PASS[8]=$8

SET="N"
MONTH_CONV=""
JUL_TEMP_YEAR=0
JUL_TEMP_DAYS=0

leap_year_valid()
{
integer r
r=$1
let FOUR=$r/4*4
let HUNDRED=$r/100*100
let FOUR_HUN=$r/400*400
if [[ "$FOUR" = "$r" && "$HUNDRED" != "$r" || "$FOUR_HUN" = "$r" ]]
then
LEAP_TOT=366
SET="Y"
```

```
return 0
else
LEAP_TOT=365
SET="N"
return 1
fi
}

name_of_day ()
{
#Calculate day name
DATEJ=$jul
YEAR=$year
        FOUR=YEAR/4
        HUNDRED=YEAR/100
        FOUR_HUN=YEAR/400
        LEAP_DAYS=FOUR-HUNDRED+FOUR_HUN
if [ "$SET" = "Y" ]
then
        LEAP_DAYS=LEAP_DAYS-1
fi

TOTAL_DAYS=$YEAR*365+LEAP_DAYS+DATEJ
NUM_WEEKS=$TOTAL_DAYS/7
CALC_DAYS=$NUM_WEEKS*7
DIFF_DAYS=$TOTAL_DAYS-$CALC_DAYS

case "$DIFF_DAYS" in
        1)
                DAYNAME="Sunday"
                DAY_INT=0
        ;;
        2)
                DAYNAME="Monday"
                DAY_INT=1
        ;;
        3)
                DAYNAME="Tuesday"
                DAY_INT=2
        ;;
        4)
                DAYNAME="Wednesday"
                DAY_INT=3
        ;;
        5)
                DAYNAME="Thursday"
                DAY_INT=4
        ;;
        6)
                DAYNAME="Friday"
```

```
                DAY_INT=5
        ;;
        0)
                DAYNAME="Saturday"
                DAY_INT=6
        ;;
        esac
}

extra_day()
{
case $1 in
        01|21|31)
                echo "$1st"
        ;;
        02|22)
                echo "$1nd"
        ;;
        03|23)
                echo "$1rd"
        ;;
        *)
                echo "$1th"
        ;;
esac
}

convert_month_name()
{
case $1 in
        [jJ]an|[jJ]anuary)
                MONTH_CONV="01"
        ;;
        [fF]eb|[fF]ebruary)
                MONTH_CONV="02"
        ;;
        [mM]ar|[mM]arch)
                MONTH_CONV="03"
        ;;
        [aA]pr|[aA]pril)
                MONTH_CONV="04"
        ;;
        [mM]ay)
                MONTH_CONV="05"
        ;;
        [jJ]un|[jJ]une)
                MONTH_CONV="06"
        ;;
        [jJ]ul|[jJ]uly)
                MONTH_CONV="07"
```

```
        ;;
        [aA]ug|[aA]ugust)
                MONTH_CONV="08"
        ;;
        [sS]ep|[sS]eptember)
                MONTH_CONV="09"
        ;;
        [oO]ct|[oO]ctober)
                MONTH_CONV="10"
        ;;
        [nN]ov|[nN]ovember)
                MONTH_CONV="11"
        ;;
        [dD]ec|[dD]ecember)
                MONTH_CONV="12"
        ;;
esac
}

generate_year_day()
{
integer ADDMN
integer orig_jul
orig_jul=jul
jul=jul-MINUS
leap_year_valid $year
if [ $jul -le $LEAP_TOT ]
then
        DAY_LOOP=$jul
        while [ $DAY_LOOP -le 0 ]
        do
                year=$year-1
                leap_year_valid $year
                let DAY_LOOP=DAY_LOOP+$LEAP_TOT
        done
else
        let TEMP_LOOP=$LEAP_TOT-$orig_jul
        ADDMN=-$MINUS
        let DAY_LOOP=$ADDMN-$TEMP_LOOP

        year=$year+1
        leap_year_valid $year
        while [ DAY_LOOP -gt LEAP_TOT ]
        do
                let DAY_LOOP=$DAY_LOOP-$LEAP_TOT
                year=year+1
                leap_year_valid $year
        done
fi
let jul=$DAY_LOOP
```

```
}

M_D_Y_calc()
{
leap_year_valid $year
if [[ "$?" = "0" ]]
then
        let filedat[2]=29
else
        let filedat[2]=28
fi
leapjul=$jul
let i=1
let total=0
while [[ i -lt 13 ]]
do
        if [[ $total -lt $jul ]]
        then
                temp=${filedat[i]}
                let total=$total+$temp
                let month=$i
        fi
        let i=i+1
done
let temp=${filedat[month]}
let str=$total-$temp
let day=$jul-$str
amonth="$month"
aday="$day"
if [[ month -lt 10 ]]
then
        amonth="0$month"
fi
if [[ day -lt 10 ]]
then
        aday="0$day"
fi
}

month_name()
{
case "$amonth" in
        01)
                LONGMON="January"
        ;;
        02)
                LONGMON="February"
        ;;
        03)
                LONGMON="March"
```

```
        ;;
        04)
                LONGMON="April"
        ;;
        05)
                LONGMON="May"
        ;;
        06)
                LONGMON="June"
        ;;
        07)
                LONGMON="July"
        ;;
        08)
                LONGMON="August"
        ;;
        09)
                LONGMON="September"
        ;;
        10)
                LONGMON="October"
        ;;
        11)
                LONGMON="November"
        ;;
        12)
                LONGMON="December"
        ;;
        esac
}

workout_julian()
{
integer J_YEAR
integer J_MONTH
integer J_DAY
integer TOT_DAYS
J_YEAR=`echo $1|cut -f3 -d/`
J_MONTH=`echo $1|cut -f2 -d/`
J_DAY=`echo $1|cut -f1 -d/`
leap_year_valid $J_YEAR
if [[ "$?" = "0" ]]
then
        filedat[2]=29
else
        filedat[2]=28
fi
if [ $J_YEAR -lt 1900 -o $J_YEAR -gt 2400 ]
then
#       print "$1 Date format problem, year is out of range allowed"
```

```
        exit 1
fi
if [ $J_MONTH -gt 12 -o $J_MONTH -lt 1 ]
then
        #print "$1 Date format problem: month is outside allowed range "
        exit 2
fi
if [ $J_DAY -gt ${filedat[$J_MONTH]} -o $J_DAY -lt 1 ]
then
        #print "$1 Date format problem: day is outside allowed range "
        exit 3
fi
let LOOP_UNTIL=J_MONTH-1
while [ LOOP_UNTIL -gt 0 ]
do
        TOT_DAYS=$TOT_DAYS+filedat[$LOOP_UNTIL]
        let LOOP_UNTIL=$LOOP_UNTIL-1
done
let TOT_DAYS=TOT_DAYS+J_DAY
JUL_TEMP_YEAR=$J_YEAR
JUL_TEMP_DAYS=$TOT_DAYS
}

calc_date_to_date()
{
        workout_julian $1
        year=$JUL_TEMP_YEAR
        jul=$JUL_TEMP_DAYS
        MINUS=0
        generate_year_day
        M_D_Y_calc
        name_of_day
        TOTAL_1=$TOTAL_DAYS
        workout_julian $2
        year=$JUL_TEMP_YEAR
        jul=$JUL_TEMP_DAYS
        MINUS=0
        generate_year_day
        M_D_Y_calc
        name_of_day
        TOTAL_2=$TOTAL_DAYS
        if [ TOTAL_2 -ge TOTAL_1 ]
        then
                OUTPUT_3=TOTAL_2-TOTAL_1
                P_OUTPUT="$OUTPUT_3"
        else
                OUTPUT_3=TOTAL_1-TOTAL_2
                P_OUTPUT="-$OUTPUT_3"
        fi
        echo "$P_OUTPUT"
```

```
}

calculate_the_differnce()
{
integer DIFF_TEMP
integer COUNT
COUNT=YEAR1
DIFF_TEMP=0
if [ $COUNT -lt $YEAR2 ]
then
        leap_year_valid $COUNT
        DIFF_TEMP=LEAP_TOT-DAY1
        COUNT=COUNT+1
        while [ $COUNT -lt $YEAR2 ]
        do
                leap_year_valid $COUNT
                DIFF_TEMP=DIFF_TEMP+LEAP_TOT
                COUNT=COUNT+1
        done
        DIFF_TEMP=DIFF_TEMP+DAY2
else
        DIFF_TEMP=DAY2-DAY1
fi
print $DIFF_TEMP
}

if [[ "$1" = "?" ]]
then
  print " Options available"
  print " -----------------"
  print " -n      Day number (0-6 0=Sunday)"
  print " -nn     Short day name (eg Mon)"
  print " -nnn    Long day name (eg Monday)"
  print " -d      Day of month with leading 0"
  print " -de     Day of month with extension (eg 21st)"
  print " -m      Month number with leading 0"
  print " -mm     Short month name (eg Jan)"
  print " -mmm    Long month name (eg January)"
  print " -y      Short year format (eg 97)"
  print " -yy     Long year format (eg 1997)"
  print " -j      Julian day"
  print " -yj     Year and julian day (eg 97261)"
  print " -sp     Output mm\/dd\/yy"
  print " -s      Prints a space between the output"
  print " -/      Print a / between the output"
  print
  print " Command use"
  print
  print " calcdate_diff 1 -d -m -yy "
  print "                 = Yesterdays date in ddmmyy format"
```

```
  print
  print " calcdate_diff -c dd/mm/yyyy dd/mm/yyyy"
  print "                 =The number of days between the two dates"
  print
  print " calcdate_diff -cd dd/mm/yyyy -1 -/ -d -m -yy"
  print "                 =The day after specified date in dd/mm/yyyy \
format"
  print
  print " calcdate_diff -v dd/mm/yyyy"
  print "                 =Validates the specified dd/mm/yyyy"
  print "         (returns 0 if the date is valid)"
  print
  print " calcdate_diff -g dd/mm/yyyy dd/mm/yyyy"
  print "                 =Checks which is the greater date"
  print "                 Displays the number of days \
from 1st date to 2nd"
  print
  print " calcdate_diff -mc May "
  print "                 =Converts give month to number of month"
  print "                 Includes leading 0 (eg 05)"
  print
  exit 0
fi

########Check Number of paramenters passed, exit if too many
if [ $# -gt 8 ]
then
        print "Too many parameters"
        exit
fi

if [[ "$1" = "-c" ]]
        then
                workout_julian $2
                        YEAR1=$JUL_TEMP_YEAR
                        DAY1=$JUL_TEMP_DAYS
                workout_julian $3
                        YEAR2=$JUL_TEMP_YEAR
                        DAY2=$JUL_TEMP_DAYS
                calculate_the_differnce $YEAR1 $YEAR2 $DAY1 $DAY2
                exit
        fi

if [[ "$1" = "-dc" || "$1" = "-cd" ]]
        then
                workout_julian $2
                year=$JUL_TEMP_YEAR
                jul=$JUL_TEMP_DAYS
                MINUS=$3
        else
```

```
                year=$(date +%Y)
                jul=$(date +%j)
        fi

if [[ "$1" = "-v" ]]
        then
                workout_julian $2
                exit 0
        fi

if [[ "$1" = "-g" ]]
        then
                calc_date_to_date $2 $3
                exit
        fi

if [[ "$1" = "-mc" ]]
        then
                convert_month_name $2
                print $MONTH_CONV
                exit
        fi

generate_year_day
M_D_Y_calc
name_of_day
month_name

SPACE=""
gto=1

while [[ gto -lt 8 ]]
do
gto=gto+1
if [ "${PASS[$gto]}" = "-n" ]
then
        OUT_PASS[$gto]=$DAY_INT
fi
if [ "${PASS[$gto]}" = "-nn" ]
then
        OUT_PASS[$gto]=`echo $DAYNAME|cut -c1-3`
fi
if [ "${PASS[$gto]}" = "-nnn" ]
then
        OUT_PASS[$gto]=$DAYNAME
fi
if [ "${PASS[$gto]}" = "-d" ]
then
        OUT_PASS[$gto]=$aday
fi
```

```
if [ "${PASS[$gto]}" = "-de" ]
then
        OUT_PASS[$gto]=`extra_day $aday`
fi
if [ "${PASS[$gto]}" = "-m" ]
then
        OUT_PASS[$gto]=$amonth
fi
if [ "${PASS[$gto]}" = "-mm" ]
then
        OUT_PASS[$gto]=`echo $LONGMON|cut -c1-3`
fi
if [ "${PASS[$gto]}" = "-mmm" ]
then
        OUT_PASS[$gto]=$LONGMON
fi
if [ "${PASS[$gto]}" = "-y" ]
then
        OUT_PASS[$gto]=`echo $year|cut -c3-4`
fi
if [ "${PASS[$gto]}" = "-yy" ]
then
        OUT_PASS[$gto]=$year
fi
if [ "${PASS[$gto]}" = "-j" ]
then
        OUT_PASS[$gto]=$jul
fi
if [ "${PASS[$gto]}" = "-yj" ]
then
        OUT_PASS[$gto]=`echo ${year}$jul|cut -c3-`
fi
if [ "${PASS[$gto]}" = "-sp" ]
then
        OUT_PASS[$gto]=$aday"\/"$amonth"\/"$year
fi
if [ "${PASS[$gto]}" = "-s" ]
then
        SPACE=" "
fi
if [ "${PASS[$gto]}" = "-/" ]
then
        SPACE="/"
fi
done

OUTPUT=""
gto=1
while [[ gto -lt 8 ]]
do
```

```
        gto=gto+1
        if [ "${OUT_PASS[$gto]}" != "" ]
        then
                addt=gto+1
                if [ "${OUT_PASS[$addt]}" != "" ]
                then
                        OUTPUT=$OUTPUT"${OUT_PASS[$gto]}"$SPACE
                else
                        OUTPUT=$OUTPUT"${OUT_PASS[$gto]}"
                fi
        fi
done
print $OUTPUT
#end
```

THE CALCDATE_TIME SCRIPT

The second script presented in this article is **calcdate_time**, which performs various time-related calculations. The way this script is used is demonstrated in examples later on. The script conforms to the same conventions as **calcdate_diff** in as much as all dates must be in *dd/mm/yyyy* format. In addition, all times must be in *HH:MM* 24-hour format. You may get unexpected results if these conventions are not followed. Both date and time syntax checking is done before the script carries out any calculations. Use the command below to get a list of available options for this script.

```
calcdate_time ?
```

The -f option

```
calcdate_time -f dd/mm/yyyy HH:MM MMMM
```

This option calculates the date and time from the given date plus the given number of minutes. The first positional parameter must be **-f** to specify this option.

For example:

```
calcdate_time -f 01/06/1998 01:00 1440
```

Results in:

```
02/06/1998 01:00
```

The -b option

```
calcdate_time -b dd/mm/yyyy HH:MM MMMM
```

This option calculates the date and time from the given date minus the given number of minutes. The first positional parameter must be **-b** to specify this option.

For example:

```
calcdate_time -b 01/01/1998 01:00 1440
```

Results in:

```
31/12/1997 01:00
```

The -d option

```
calcdate_time -d dd/mm/yyyy HH:MM dd/mm/yyyy HH:MM
```

This option calculates the difference (in minutes) between the two given dates and times. The first positional parameter must be **-d** to specify this option.

For example:

```
calcdate_time -d 01/01/1998 01:00 02/01/1998 01:00
```

Results in:

```
1440
```

General considerations

The options must be exactly as shown. Any extra positional parameters specified are ignored.

THE CALCDATE_TIME SCRIPT

```
#!/bin/ksh
###############################################################################
#                                                                             #
#       Written by :     Robert Russell                                       #
#       Use :            Addition to calcdate_diff for times                  #
#       Copyright :      Robert Russell (1998)                                #
#       Calls programs : calcdate_diff                                        #
#                                                                             #
###############################################################################
integer i
```

```
integer a
integer TOTAL_MIN_DIFF
integer ONE_HOUR
integer ONE_MIN
integer TWO_HOUR
integer TWO_MIN
integer ONE_TOTAL
integer TWO_TOTAL
integer DAYS
integer TEMP
integer TEMP1
integer MINUTES

TOTAL_MIN_DIFF=0
COMMANDS=/usr/home/it032x/commands

date_check()
{
`$COMMANDS/calcdate_diff -v $1 >/dev/null`
CHECK=$?

case $CHECK in
        1)
                print "Problem with Year in Date"
                exit 1
        ;;
        2)
                print "Problem with Month in Date"
                exit 2
        ;;
        3)
                print "Problem with Day in Date"
                exit 3
        ;;
        0)
        ;;
esac
}

time_check()
{
integer H_CHECK
integer M_CHECK
S_CHECK=`echo $1 |cut -c3`
        if [[ "$S_CHECK" != ":" ]]
        then
                print "Time is not in correct format"
                exit 6
        fi
H_CHECK=`echo $1 |cut -f1 -d:`
```

```
M_CHECK=`echo $1 |cut -f2 -d:`
        if [ H_CHECK -lt 0 -o H_CHECK -gt 23 ]
        then
                print "Time is not in correct format"
                exit 4
        fi
        if [ M_CHECK -gt 59 -o M_CHECK -lt 0 ]
        then
                print "Time is not in correct format"
                exit 5
        fi
}

time_diff()
{
ONE_HOUR=`echo $ONE_TIME |cut -f1 -d:`
ONE_MIN=`echo $ONE_TIME |cut -f2 -d:`
TWO_HOUR=`echo $TWO_TIME |cut -f1 -d:`
TWO_MIN=`echo $TWO_TIME |cut -f2 -d:`
ONE_TOTAL=$ONE_HOUR*60+$ONE_MIN
TWO_TOTAL=$TWO_HOUR*60+$TWO_MIN
if [[ "$DATE_ONE" = "$DATE_TWO" ]]
then
        TOTAL_MIN_DIFF=$TWO_TOTAL-$ONE_TOTAL
else
        DAYS=`$COMMANDS/calcdate_diff -c $DATE_ONE $DATE_TWO`
                TEMP=1440-ONE_TOTAL
                TEMP1=$DAYS-1
                TOTAL_MIN_DIFF=1440*$TEMP1
                TOTAL_MIN_DIFF=$TOTAL_MIN_DIFF+$TWO_TOTAL+$TEMP
fi
        print $TOTAL_MIN_DIFF
}

time_add()
{
integer DAY_COUNT
DAY_COUNT=0
ONE_HOUR=`echo $ONE_TIME |cut -f1 -d:`
ONE_MIN=`echo $ONE_TIME |cut -f2 -d:`
ONE_TOTAL=$ONE_HOUR*60+$ONE_MIN
TEMP=$ONE_TOTAL+$MINUTES
DAY_ADD=$DATE_ONE
if [ TEMP -gt 1439 ]
then
        TEMP1=1440-$ONE_TOTAL
        DAY_COUNT=$DAY_COUNT+1
        MINUTES=$MINUTES-$TEMP1
        while [ $MINUTES -gt 1439 ]
        do
```

```
                DAY_COUNT=$DAY_COUNT+1
                MINUTES=$MINUTES-1440
        done
        TEMP=MINUTES
        DAY_ADD=`$COMMANDS/calcdate_diff -dc $DATE_ONE \
-$DAY_COUNT -d -m -yy -/`
fi
        TWO_HOUR=$TEMP/60
        TEMP1=$TWO_HOUR*60
        TWO_MIN=$TEMP-$TEMP1
        PRINT_HOUR=$TWO_HOUR
        PRINT_MIN=$TWO_MIN
        if [ TWO_HOUR -lt 10 ]
        then
                PRINT_HOUR="0$TWO_HOUR"
        fi
        if [ TWO_MIN -lt 10 ]
        then
                PRINT_MIN="0$TWO_MIN"
        fi
        print "$DAY_ADD $PRINT_HOUR:$PRINT_MIN"
}

time_back()
{
integer DAY_COUNT
DAY_COUNT=0
ONE_HOUR=`echo $ONE_TIME |cut -f1 -d:`
ONE_MIN=`echo $ONE_TIME |cut -f2 -d:`
ONE_TOTAL=$ONE_HOUR*60+$ONE_MIN
TEMP=$ONE_TOTAL-$MINUTES
DAY_ADD=$DATE_ONE
if [ TEMP -lt 0 ]
then
        TEMP1=$ONE_TOTAL
        MINUTES=$TEMP1-$MINUTES
        while [ $MINUTES -lt 0 ]
        do
                DAY_COUNT=$DAY_COUNT+1
                MINUTES=$MINUTES+1440
        done
        TEMP=MINUTES
        DAY_ADD=`$COMMANDS/calcdate_diff -dc \
$DATE_ONE $DAY_COUNT -d -m -yy -/`
fi
        TWO_HOUR=$TEMP/60
        TEMP1=$TWO_HOUR*60
        TWO_MIN=$TEMP-$TEMP1
        PRINT_HOUR=$TWO_HOUR
        PRINT_MIN=$TWO_MIN
```

```
        if [ TWO_HOUR -lt 10 ]
        then
                PRINT_HOUR="0$TWO_HOUR"
        fi
        if [ TWO_MIN -lt 10 ]
        then
                PRINT_MIN="0$TWO_MIN"
        fi
        print "$DAY_ADD $PRINT_HOUR:$PRINT_MIN"
}

if [[ "$1" = "-d" ]]
then
        DATE_ONE=$2
                date_check $DATE_ONE
        ONE_TIME=$3
                time_check $ONE_TIME
        DATE_TWO=$4
                date_check $DATE_TWO
        TWO_TIME=$5
                time_check $TWO_TIME
        time_diff $DATE_ONE $ONE_TIME $DATE_TWO $TWO_TIME
fi

if [[ "$1" = "-f" ]]
then
        DATE_ONE=$2
                date_check $DATE_ONE
        ONE_TIME=$3
                time_check $ONE_TIME
        MINUTES=$4
        time_add $DATE_ONE $ONE_TIME $MINUTES
fi

if [[ "$1" = "-b" ]]
then
        DATE_ONE=$2
                date_check $DATE_ONE
        ONE_TIME=$3
                time_check $ONE_TIME
        MINUTES=$4
        time_back $DATE_ONE $ONE_TIME $MINUTES
fi

if [[ "$1" = "?" ]]
then
        print " Command usage"
        print " -------------"
        print
        print " calcdate_time -f dd/mm/yyyy HH:MM MMMM"
```

```
        print "         prints the date and time of the added minutes"
        print
        print " e.g.    $ calcdate_time -f 01/01/1997 01:00 1440"
        print " 02/01/1997 01:00"
        print "  "
        print " calcdate_time -b dd/mm/yyyy HH:MM MMMM"
        print "         prints the date and \
time of the subtracted minutes"
        print
        print " e.g.    $ calcdate_time -b 01/01/1997 01:00 1440"
        print " 31/12/1996 01:00"
        print "  "
        print " calcdate_time -d dd/mm/yyyy HH:MM dd/mm/yyyy HH:MM"
        print "         prints the minutes \
between the given dates/times"
        print
        print " e.g.    $ calcdate_time -d \
01/01/1997 01:00 02/01/1997 01:00"
        print " 1440"
fi
#end
```

Changes needed to implement calcdate_time

```
COMMANDS=/usr/home/it032x/commands
```

The COMMANDS variable is used to execute the **calcdate_diff** script from within **calcdate_time** script. This directory must be the same one that contains both the **calcdate_time** and **calcdate_diff** scripts.

This article concludes in next month's issue of AIX Update.

Robert Russell (UK)

Changing window titles

While it's handy to be able to use multiple windows from a Graphical User Interface (GUI), this may be confusing if they all have the same or similar titles. Unfortunately, this is often just what happens when you're working with multiple windows spawned by the same

application. When the windows or their content overlap, it's not unusual to find yourself wondering exactly which one you're using.

GRAPHICAL WINDOWS

Firstly, let's clear up a point that's often unclear. A graphical window's contents are character-oriented. This means that, even though you're using a GUI, the windows's contents are ordinary plain text. While the window may allow you to choose from a list of fonts and font sizes, this doesn't alter the fact that the contents are just plain text. The GUI's window manager puts a frame around this character-oriented window, usually adding a title to the frame in the title bar. What we want to be able to do is manipulate this region, including changing the title, if required.

This article discusses how to change the title of Common Desktop Environment (CDE) **dtterm** windows, AIX's **aixterm** windows, and generic Unix (X11) **xterm** windows. I also discuss how this can be done using a shell script that finds out which type of window you're using and then adapts itself to the window's peculiarities.

CDE DTTERM, AIX AIXTERM, AND UNIX X11 XTERM WINDOWS

CDE **dtterm**, AIX **aixterm**, and Unix X11 **xterm** window managers all use the same character sequence to put text in the title bar. It begins with the following four-character escape sequence:

```
<Esc> ] 2 ;
```

and ends with the following two-character sequence:

```
<Esc> <bell>
```

Text placed between the two escape strings is sent to the title bar.

It's worth knowing that the character sequence to change the title bar of Sun Solaris shell windows is different from that used by other versions of Unix – it starts with the three-character sequence (the last character is a one, not a lowercase 'L'):

```
<Esc> ] l
```

and ends with the two-character sequence:

```
    <Esc> \
```

Look out for this as it can cause problems in environments with Solaris workstations or servers.

MANAGING TITLES WITH A SHELL SCRIPT

We can use the fact that the environment variable TERM tells us which window type we're using to manage the windows's title and – should you wish to extend the script – other window attributes. The only parameter that the shell script requires is the text of the title itself.

SHELL SCRIPT WTITLE.SH

```
#! /bin/sh

usage()
{
    echo "usage:  $0 title"
    exit 1
}

if [ -n "$1" -a "$1" != "-help" ]
then
    case "$TERM" in
        "dtterm" | "aixterm" | "xterm" )
            echo "^[]2;" $1 "^[^G"
            ;;
        "sun-cmd" )
            echo "^[]l" $1 "^[\\"
            ;;
        *)
            echo "Sorry, TERM $TERM not supported"
            exit 1
            ;;
    esac
else
    usage
fi

exit 0
```

When entering the script, the two-character sequence ^*[* represents the <*Esc*> key and not the two characters ^ and *[*. If you're using **vi** to write the shell script, enter <*Ctrl*>-*V* before entering the <*Esc*> key to enter the <*Esc*> key's value. Similarly, ^*G* is the <*bell*> character

and not ^ and *G*. Again, if you're using **vi**, enter *<Ctrl>-V* before entering *G* to enter the *<bell>* character. You could also use *\033* and *\007* to enter these two values – these are the octal values of *<Esc>* and *<bell>*. As the shell interprets the backslash (\) as a special character, you need to enter it twice for it to be interpreted as one character. The first '\' tells the shell that the next character has no special meaning and the second '\' is the character itself.

To convert *wtitle.sh* from a text file to an executable shell script, use the following command:

```
chmod +x wtitle.sh
```

If there are spaces in the title, enclose the title in quotes. You can also use environment variables to make the title more informative. Here's an example that tells you which system you're working on using the $HOST environment variable.

```
wtitle.sh "$USER using $TTY"
```

CONCLUSION

While the ability to change a window's title may seem like a small improvement to the working environment, it makes it much easier to find the window that you need. You can now use the shell script **wtitle** to automate the process of managing titles on **dtterm**, **aixterm**, and **xterm** windows.

Werner Klauser
Klauser Informatik (Switzerland)

Running _CHECK_USER under AIX 4.2

This article provides some additional information to sites using the _CHECK_USER script published in *User administration* in *AIX Update* Issue 27.

Depending on the patch level, the **usrchk** command will not return the appropriate *xxxxx* codes in AIX 4.2. As the _USER_CHECK relies on this output, the script has to be updated to handle this situation.

REQUIRED OUTPUT

Below is the right format of the output of **usrck -n ALL**, which is necessary for _CHECK_USER to function correctly.

```
3001-644 The user wou_1 has no stanza in /etc/security/passwd.

3001-603 The UID 0 is duplicated for user root.

3001-664 The account for user daemon has expired.

3001-664 The account for user lpd has expired.

3001-662 User oracle is locked.

3001-603 The UID 0 is duplicated for user emersys.

3001-671 User wsha's minage is greater than maxage.

3001-661 There have been too many invalid login attempts by user xo6.
```

However, under AIX 4.2.*x*, the output below is sometimes produced when not all patch levels have been applied.

```
The user servdir has no stanza in /etc/security/passwd.

The user nuucp has no stanza in /etc/security/passwd.

The UID 0 is duplicated for user root.

The account for user daemon has expired.

The account for user bin has expired.

The account for user sys has expired.

The account for user adm has expired.

User guest is locked.

The account for user nobody has expired.

The account for user lpd has expired.

There have been too many invalid login attempts by user checkit.
```

WORKAROUND

To workaround this situation, locate the code below and replace it with the updated version shown.

CODE TO BE REPLACED

```
cat $check_file | egrep '(3001-664)'    | awk '{print "^^"$6}' | sort >
$exclude_file1
cat $check_file | egrep '(3001-662)'    | awk '{print "^^"$3}' | sort >
$exclude_file2
cat $check_file | egrep '(3001-661)'    | awk '{print
"^^"substr($12,1,length($12)-1)}' \
      | sort  > $exclude_file3
```

REPLACEMENT CODE

```
cat $check_file | egrep '(has expired)' | awk '{print "^^"$5}' | sort >
$exclude_file1
cat $check_file | egrep '(is locked)'  | awk '{print "^^"$2}' | sort >
$exclude_file2
cat $check_file | egrep '(too many invalid login)'   | awk '{print
"^^"substr($11,1,length($11)-1)}' \
      | sort  > $exclude_file3
```

OTHER CHANGES

In addition to replacing the segment of code above, you also need to make the following change:

Change the text heading:

```
echo All only-su  user accounts ---------------
```

to:

```
echo All only-su+ftp  user accounts ---------------
```

This last change is necessary as the **rlogin** and **login** flags are set to false. You can use **su** to switch to the user (if permitted) and use the account to **ftp** to the machine if the user is not listed in the */etc/ftpusers* file and the node offers an **ftp** service.

Michael Imhotep (Australia)

Printer header and trailer pages

For the average AIX administrator and end-user, printing on system printers is an everyday activity. This article explains a little-known but extremely useful feature of this type of print job – the use of custom header and trailer pages. This handy feature of AIX printing is useful in many situations, such as when you need to create cover pages for different kinds of print job or for different users sharing the same printer.

ENABLING AND DISABLING HEADERS AND TRAILERS

The printing of header and trailer pages for each print job is by default disabled when a new printer queue is created on an AIX system. This setting is recorded in the */etc/qconfig* file in the stanza about the printer queue device. You can always check the status of this feature using the following procedure:

1 To find the name of a queue device that corresponds to a particular queue, issue the command:

```
lsque -qps
```

Where *ps* is the name of a valid print queue.

The output of this command should look something like the lines below.

```
ps:
     device=lp0
```

Next, list the queue device properties using the command:

```
lsquedev -qps -dlp0
```

The output should be similar to the following:

```
lp0:
     file = /dev/lp0
     header = always
     trailer = never
     access = both
     backend = /usr/lib/lpd/piobe
```

The list below summarizes the possible values of the variables *header* and *trailer* in */etc/qconfig* and their effect on the printing of headers and trailers.

Value	*Effect on printing of header or trailer*
never	Don't print header or trailer.
always	Print header or trailer for each printed file.
group	Print header or trailer for each group of files printed using a single print command.

You can always change the default settings using **smit**. The procedure below works for all releases of AIX 4.

1 Issue the command **smit chpq** at a command prompt.

2 Fill in the PRINTER QUEUE name by selecting from a list.

3 Select the second option (*Default Print Job Attributes*).

4 Scroll down to section titled '*Header/Trailer page options*'.

5 Select one of the following options in the SEPARATOR PAGES field:

- 'none'
- 'header for job; no trailer'
- 'header for job; trailer for job'
- 'header for job; trailer for each file'
- 'header for each file; no trailer'
- 'header for each file; trailer for job'
- 'header for each file; trailer for each file'
- 'no header; trailer for job'
- 'no header; trailer for each file'.

4 Click OK.

It's also possible to override the default setting when you submit print

jobs by using flags supported by the various AIX print spooling commands. The table below summarizes the available options.

Command	*Flags*	*Action*
qprt or **enq**	**-Bnn**	Don't print header or trailer.
qprt or **enq**	**-Ban**	Print only a header for each file.
qprt or **enq**	**-Bna**	Don't print a header but print a trailer for each file.
qprt or **enq**	**-Baa**	Print both header and trailer.
qprt or **enq**	**-Bgg**	Print a header and trailer for each group of files.
qprt or **enq**	**-Bgn**	Print header for each group of files but no trailers.
qprt or **enq**	**-Bng**	Don't print any headers but print a trailer for each group of files.
qprt or **enq**	**-Bga**	Print a header for each group of files and print a trailer for each file.
qprt or **enq**	**-Bag**	Print a header for each file and a trailer for each group of files.
lpr	**-h**	Don't print any headers or trailers.
lpr	no flags	Print a header but no trailers.
lp	**-o** a **qprt** flag	Pass all flags to the **qprt** command.

If you wish to disable the printing of header pages permanently (for instance, on a printer connected to a personal workstation) you can set the **sh** attribute of the virtual printer to nothing. Do this by executing the command **lsvirprt**, choosing the appropriate virtual printer, then issuing the command **sh=** (without any trailing spaces), and pressing <ENTER>. Conclude by pressing <ENTER> again to exit **lsvirprt**.

DESIGNING CUSTOMIZED HEADERS AND TRAILERS

The header and template format files are located in the directory */usr/*

lib/lpd/pio/burst. Below is a list of the names and formats of all the files used.

File Name	*Format*	*Usage*
H.ascii	ASCII	Header page template for ASCII (plain text) printers.
H.gl	HP-GL	Header page template for HP-GL printers and plotters.
H.ps	PostScript	Header page template for PostScript printers.
bullH.ps	PostScript	Header page template for PostScript printers manufactured by Bull.
T.ascii	ASCII	Trailer page template for ASCII (plain text) printers.
T.gl	HP-GL	Trailer page template for HP-GL printers and plotters.
T.ps	PostScript	Trailer page template for PostScript printers.
bullT.ps	PostScript	Trailer page template for PostScript printers manufactured by Bull.

Below is the default ASCII text file:

```
*#########################################################################
*#########################################################################
*#########################################################################
*#########################################################################


**************************************************************************
**************************************************************************

%t  %T

%p  %P

%q  %Q

%h  %H

%s  %S
```

```
%d  =====> %D <=====

**************************************************************************
**************************************************************************

%a
%A

**************************************************************************
**************************************************************************
```

This file is interpreted as follows: characters prefixed with a percent sign (%) are predefined variables that are expanded by the AIX printing mechanism before the header or trailer file is printed. Variables whose names include only lower case letters are predefined labels, while variables whose names include only upper case letters are values extracted from the print process's environment. The following tables summarizes these variables.

Template variables

Name	*Type*	*Value*
%a	Label	'FLAG VALUES:'
%d	Label	'DELIVER TO:'
%e	Label	'END OF OUTPUT FOR:' (useful for trailer pages)
%h	Label	'PRINTED AT:'
%p	Label	'TIME PRINTED:'
%q	Label	'TIME QUEUED:'
%s	Label	'SUBMITTED BY:'
%t	Label	'TITLE:'
%%	Label	Special sequence to specify a percent sign
%A	Variable	Formatting flags from print command line and virtual printer
%D	Variable	The name of the user that should receive the printout

%H	Variable	The name of the host and queue devices that performed the printing
%P	Variable	The time when the print job was carried out
%Q	Variable	The time when the print job was carried out
%S	Variable	The name of user that issued the print command
%T	Variable	The title of the print job or the name of printed file.

The following is an example of a modified ASCII header file. I've used the command **/usr/bin/banner** to add a heading in large letters to enhance the appearance of the layout.

```
**************************************************************************

#######  #######  ######
#        #      # #      #
#        #      # #      #
#####    #      # ######
#        #      # #    #
#        #      # #     #
#        #######  #      #

   #     #        #######  #      #
  # #    #        #          #    #
 #   #   #        #            # #
#     #  #        #####        #
#######  #        #            # #
#     #  #        #           #   #
#     #  #######  #######  #      #

**************************************************************************
%T was printed by %S at %P to %H
**************************************************************************

%q  %Q

%d  =====> %D <=====

**************************************************************************
%a
%A
**************************************************************************
```

You can use the following technique to design even more eye-catching header pages in PostScript format with the aid of a Web browser. First create an HTML page that is to become your new template. You can use graphics, mixed font sizes, and other effects you want, and you may also include any template variables required. Next print the design to a file. Edit the file to ensure that all percent signs generated by PostScript are replaced by '%%' (except, of course, percent signs that precede template variables). Next save a copy of a default header or trailer file using the command below:

```
cp /usr/lib/lpd/pio/burst/H.ps  /usr/lib/lpd/pio/burst/H.ps.save.
```

Then copy your new template to the location of the original template:

```
cp /tmp/H.ps  /usr/lib/lpd/pio/burst/H.ps..
```

Below is an example of a header page in HTML format (Figure 1 shows the page as it appears when loaded into Netscape Navigator).

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<CENTER>
<UL>
<B>
<FONT SIZE=+2>Testing Department Custom Header Page</FONT>
</B>
<BR>
<HR WIDTH="100%">
<BR><IMG SRC="f50.gif" HEIGHT=200 WIDTH=200>
</CENTER>
<BR>
<HR WIDTH="100%">
<BR><FONT SIZE=+2>%a</FONT>
<BR><FONT SIZE=-2>%A</FONT>
<BR><FONT SIZE=+2>%d %D</FONT>
<BR><FONT SIZE=+2>%e %D</FONT>
<BR><FONT SIZE=+2>%h %H</FONT>
<BR><FONT SIZE=+2>%p %P</FONT>
<BR><FONT SIZE=+2>%s %S</FONT>
<BR><FONT SIZE=+2>%t %T</FONT>
<BR>
<HR WIDTH="100%">
<BR>
</UL>
```

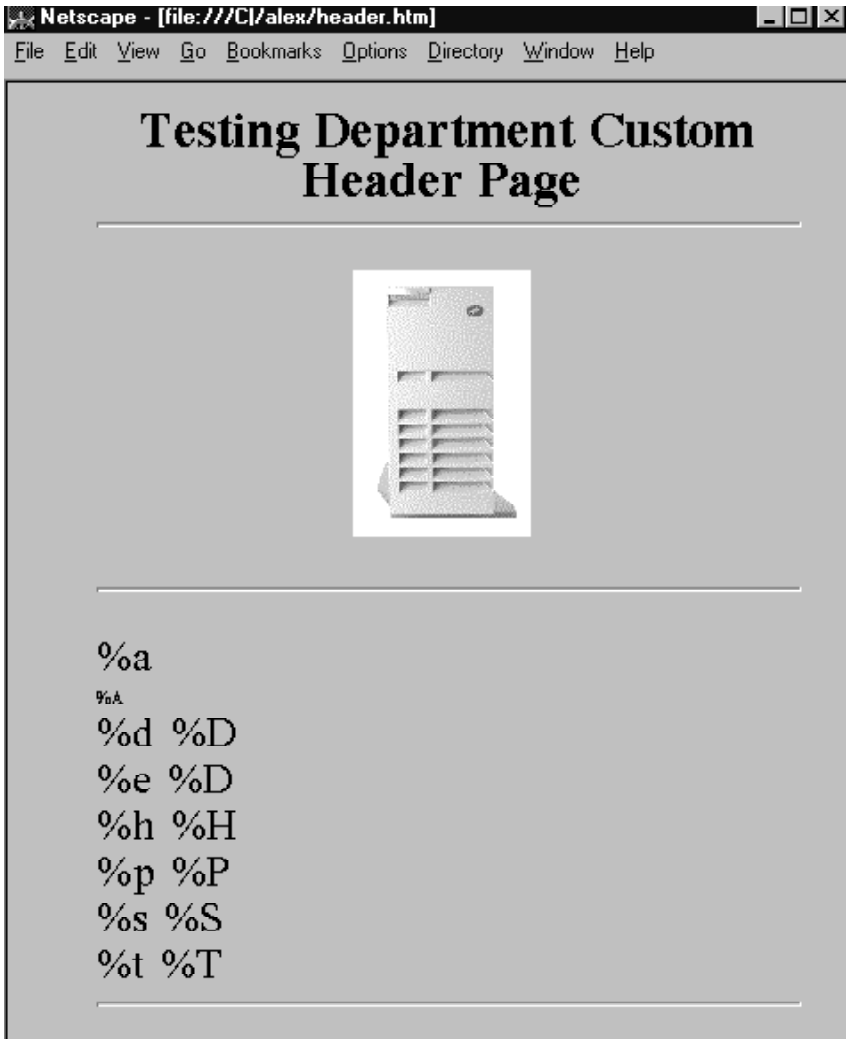

Figure 1: HTML header page viewed with Navigator

```
</BODY>
</HTML>

<PLEASE INSERT HERE PICTURE HEADER.GIF>
```

PAPER TRAYS FOR HEADER AND TRAILER PAGES

If your printer has multiple paper trays, you can specify different

paper sources for print jobs and header pages. This usually means selecting the smaller tray for header pages, possibly using distinctive stationery (say in a different colour). Use the following procedure to select the paper tray for the main part of print job:

1 Issue the command **smit chpq**.

2 Fill in the PRINTER QUEUE name by selecting from the list of available queues.

3 Choose the second option (Default Print Job Attributes).

4 Scroll to the section titled ‘Input Paper Source’.

5 Select the paper tray for the main print job (the list of trays depends on the printer).

6 Select ‘DO’ to commit changes.

To select the paper tray for headers use following procedure:

1 Issue the command **smit chpq**.

2 Fill in the PRINTER QUEUE name by selecting from the list of available printers.

3 Choose the first option (Printer Setup).

4 Scroll to field titled ‘Input PAPER TRAY for header page’ and select the paper tray for header pages (the list of trays depends on the printer).

5 Scroll to section titled ‘Input PAPER TRAY for trailer page’ and select paper tray for trailer pages (the list of trays depends on the type of printer).

6 Select ‘DO’ to commit the changes.

CUSTOMIZED HEADER AND TRAILER PAGES

The virtual printer’s **sh** attribute contains definitions of commands used to enable the printing of header and trailer files. As the spooler process inherits its environment variables from the environment of the user that submitted the print job, we can modify the **sh** attribute to

select header and/or trailer files for print jobs based on the value of the *$LOGIN* environment variable. Let's assume that some users have personalized header files in */usr/lpd/pio/burst/* and that these files are named after the user. We can change **sh** attribute (using the **lsvirprt** command) to test whether a personalized header file exists and use it instead of the standard template.

The following is the procedure to change the attribute:

1 Execute the **lsvirprt** command.

2 Select the print queue that you want to modify.

3 Type **sh~v** to edit the contents of the **sh** attribute using your favourite editor (this should be defined in the variable *$EDITOR* – use **vi** if *$EDITOR* is undefined).

4 The editor's buffer should contain two versions of **sh** – one that's displayed as a very long line and another that's displayed in table form, with the contents of **sh** in the left-hand column and comments in the right-hand column. You should modify **sh** by editing the left-hand column in the second version of **sh**.

5 Be careful to include any quote marks (‘’ or ‘"’) required. Save the edit buffer.

6 When you finish editing you'll be informed of any errors that were detected and given a chance to correct them.

Even if no errors where reported you are advised to re-check the contents of **sh** by issuing the **sh** command in a **lsvirprt** session.

The following is an example of the contents of the **sh** attribute before and after the incorporation of a test for the existence of a modified header template file. Don't panic if you've ruined the original attribute contents – you can always restore them by deleting and re-defining the print queue!

PIPELINE FOR HEADER PAGE BEFORE EDITING

```
sh = %Ide/pioburst %F[H] %Idb/H.ascii|%Ide/pioformat -@%Idd/%Imm-!%Idf/
pio5202 -L!-J! %IsH -u%IuH
```

```
%Ide INCLUDE:(Directory Containing Miscellaneous Modules)
‘/pioburst’
%F[H] if “-H] Argument” on Command Line,”-#Argument”->OUTPUT
‘ ‘
%Idb INCLUDE:(Directory Containing Header and Trailer Text Files)
‘/H.ascii|’
%Ide/ INCLUDE:(Directory Containing Miscellaneous Modules)
‘pioformat -@’
%Idd INCLUDE:(Directory Containing Digested Data Base Files)
‘/’
%Imm INCLUDE:(File Name Of (digested) Data Base; Init, By “piodigest”
(mt.md.mn.mq:mv))
‘-!’
%Idf INCLUDE:(Directory Containing Loadable Formatter Routines)
‘/pio5202 -L!-J!’
 %IsH INCLUDE:(FORMATTING FLAGS for header page)
‘-u’
%IuH INCLUDE:(Input PAPER TRAY for header page)
```

PIPELINE FOR HEADER PAGE AFTER EDITING

```
sh = { if -f /usr/lpd/pio/burst/$LOGIN; then %Ide/pioburst %F[H] %Idb/
$LOGIN|%Ide/pioformat -@%Idd/%Imm-!%Idf/pio5202 -L!-J! %IsH -u%IuH ;
else %Ide/pioburst %F[H] %Idb/H.ascii|%Ide/pioformat -@%Idd/%Imm-!%Idf/
pio5202 -L!-J! %IsH -u%IuH ; fi; }
```

```
`{if -f “/usr/lpd/pio/burst/$LOGIN”; then’
%Ide INCLUDE:(Directory Containing Miscellaneous Modules)
‘/pioburst’
%F[H] if “-H] Argument” on Command Line,”-#Argument”->OUTPUT
‘ ‘
%Idb INCLUDE:(Directory Containing Header and Trailer Text Files)
‘/$LOGIN|’
%Ide/ INCLUDE:(Directory Containing Miscellaneous Modules)
‘pioformat -@’
%Idd INCLUDE:(Directory Containing Digested Data Base Files)
‘/’
%Imm INCLUDE:(File Name Of (digested) Data Base; Init, By “piodigest”
(mt.md.mn.mq:mv))
‘-!’
%Idf INCLUDE:(Directory Containing Loadable Formatter Routines)
‘/pio5202 -L!-J!’
 %IsH INCLUDE:(FORMATTING FLAGS for header page)
‘-u’
%IuH INCLUDE:(Input PAPER TRAY for header page)
‘;else’
%Ide INCLUDE:(Directory Containing Miscellaneous Modules)
‘/pioburst’
```

```
%F[H] if "-H] Argument" on Command Line,"-#Argument"->OUTPUT
' '
%Idb INCLUDE:(Directory Containing Header and Trailer Text Files)
'/H.ascii|'
%Ide/ INCLUDE:(Directory Containing Miscellaneous Modules)
'pioformat -@'
%Idd INCLUDE:(Directory Containing Digested Data Base Files)
'/'
%Imm INCLUDE:(File Name Of (digested) Data Base; Init, By "piodigest"
(mt.md.mn.mq:mv))
'-!'
%Idf INCLUDE:(Directory Containing Loadable Formatter Routines)
'/pio5202 -L!-J!'
 %IsH INCLUDE:(FORMATTING FLAGS for header page)
'-u'
%IuH INCLUDE:(Input PAPER TRAY for header page)
';fi;}'
```

REFERENCES

- *AIX Version 4.1 Guide to Printers and Printing* (SC23-2783).
- *Printing for Fun and Profit Under AIX V4* (GG24-3570-01).

A Polak
APS (Israel)

Contributing to *AIX Update*

AIX Update is primarily written by practising AIX specialists in user organizations – not journalists, consultants, marketing people. In our view, such information is far more valuable to their AIX professionals than that from other sources.

If you're interested in contributing to *AIX Update*, please download a copy of *Notes for contributors* from Xephon's Web site at *www.xephon.com*. Articles to be considered for publication can be sent the editor at *HarryLewis@compuserve.com*.

AIX news

Adding to its range of PowerPC-based AIX systems, Bull has announced a new entry-level Escala S workgroup and departmental server. The new system is positioned below the Escala E and T SMP servers and is geared towards deploying workgroup and departmental applications. It comes in either desktop or mini-tower configurations with 333 MHz PowerPC604e (a 200 MHz model is also available). Prices start from US$6,000.

Bull has also begun shipping its first 64-bit models, the EPC1200 and EPC/S1200. The new EPC1200s start at US$130,000.

For further information contact:
Bull Information Systems, 2 Wall Street, Technology Park, Billerica, MA 01821, USA
Tel: +1 978 294 6000
Fax: +1 978 294 6440
Web: http://www.bull.co.uk

Bull Information Systems, Windsor House, 3-7 Albert Street, Slough SL1 2BH, UK
Tel: +44 1753 551554
Fax +44 1753 705678

* * *

SCO has announced Tarantella Version 1.1, an enterprise-level application broker that's now available on AIX and HP-UX (there's an optional mainframe connectivity package). The new package allows users to access any application from any client, without installing any new software on the client. Other features include central management of facilities, load-balancing, and intelligent caching of Java classes.

Out now, prices start at US$395 per user.

For further details contact:
SCO, 400 Encinal Street, PO Box 1900, Santa Cruz, CA 95061, USA
Tel: +1 408 425 7222
Fax: +1 408 458 4277
Web: http://www.sco.com

SCO, Croxley Centre, Hatters Lane, Watford, UK
Tel: +44 1923 816344
Fax: +44 1923 813817

* * *

Compuware has announced new versions of File-AID/CS and QARun products with support for year 2000 testing and conversion on client/server or mid-range systems. Version 1.5 of File-AID/CS enables faster data aging and conversion for year 2000 testing by executing directly on AIX.

For further details contact:
Compuware Corp, 31440 Northwestern Highway, PO Box 9080, Farmington Hills, MI 48334, USA
Tel: +1 248 737 7300
Fax: +1 248 737 7199
Web: http://www.compuware.com

Compuware Limited, 163 Bath Road, Slough SL1 4AA, UK
Tel: +44 1753 774000
Fax: +44 1753 774200