32 TCP/SNA

December 1998

In this issue

update

TCP/SNA Update

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: t_eddolls@compuserve.com

North American office
Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Contributions
Articles published in *TCP/SNA Update* are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

TCP/SNA Update on-line
Code from *TCP/SNA Update* can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label.

Editor
Trevor Eddolls

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, EXECs, and other contents of this journal before making any use of it.

Subscriptions and back-issues
A year's subscription to *TCP/SNA Update*, comprising four quarterly issues, costs £125.00 in the UK; $185.00 in the USA and Canada; £129.50 in Europe; £33.00 in Australasia and Japan; and £132.00 elsewhere. In all cases the price includes postage. Individual issues, starting with the March 1991 issue, are available separately to subscribers for £32.00 ($47.00) each including postage.

Printed in England.

Centralized network console

Usually, once any system has been installed, it is equipped with at least one terminal that acts as a console.

The proliferation of consoles at a site (caused by different suppliers having consoles dedicated to their own equipment, which are often incompatible with the terminals from other suppliers) can produce logistical problems in terms of the availability of limited physical space and also difficulties in management.

Often it produces a complex topology with terminals from different producers (Sun, HP, DEC, IBM, Amdahl, etc) next to each other.

In such a complex environment, it is necessary to come up with a hardware and software solution.

One solution is a centralized system with a single point for the management of the different systems. The solution must:

- Centralize the control of the various systems and the networks.
- Reduce the number of operators and their workload.
- Automate the management of regular events.
- Provide remote control for all systems.
- Provide a single unified interface.
- Manage all system requests.

It must have the following characteristics:

- Configure different protocols (in order to converse with different equipment that does not necessarily support and/or use the same communication protocol – eg TCP/IP, DECnet, LAT, or SNA).
- Collect events in a log. The messages for each system must be managed by a ‘focal point’ (centralized console). This single system will be ‘virtually’ unattended.

- Define rules for the control of events. It is necessary for messages sent by each system to be captured, interpreted, and collected, and for predetermined actions to be generated.
- Manage access security to the console and the systems connected to the console. In order to access the centralized console and the systems connected to it, it is necessary to have a valid user-id and a password.

The centralized console must be suitably configured to manage events originating from various other systems.

These events can be grouped in accordance with their priority.

A working solution is represented by the following hardware/software combination:

- Workstation Digital Alpha 255/300.
- Terminal servers.
- 3762 control unit interlinks.
- Polycenter console.
- NetView.
- TCP/IP software on a mainframe.

The Alpha workstation 255/300 is equipped with graphic output – it is used as the centralized console. All the departmental systems are connected to it through the terminal server. Also, all the mainframe systems are connected to it. It is a focal point to which all system event messages are sent.

The terminal servers are hardware that interface between the LAN network and the various departmental systems, which are connected through serial ports.

The 3762 control unit interlinks are hardware devices that are connected between the LAN network and a mainframe system through parallel channels (bus and tag) or FDDI standard. They integrate the mainframe platforms with the LAN network using TCP/IP.

The Polycenter console is from DEC and provides management of the centralized console. It connects through a terminal server to the connected systems, using the LAT protocol.

NetView is the subsystem software used by IBM to manage the MVS console events to send to the centralized console.

It isn't possible to connect an MVS console to a terminal server, and so TCP/IP is used on the mainframe to connect to the centralized console using FTP and the remote shell functions.

With this centralized solution, it is possible to operate remotely and from a single location. The alarms for all the systems can also be managed from one screen. It is possible to connect to the centralized console, as an output device, a very large screen (for example a liquid crystal mega-screen with a high-definition image).

Obviously, this project is made up of various components.

The application described in this article is called the NetView Centralized Console. It represents the mainframe part of the centralized console system. In the MVS environment, it is the application that manages the selection of events to be sent to the Alpha workstation, where they are managed by the centralized console.

DESCRIPTION OF THE UTILITY

The utility is essentially composed of two parts. The first part captures the system events, interprets the events captured, and then prepares and executes the FTP and remote shell functions to the centralized console. The second part is an interface to show only the MVS events in a centralized way.

These functions integrate with the Polycenter console software so that the MVS system alarms are shown on the centralized console mega-screen.

The utility was developed and tested under MVS/ESA 4.3.0, TSO/E 2.5, ISPF/PDF 3.5, REXX, NetView 2.4.0, and TCP/Access 4.1.0.

In the message automation table, add one entry for every message/ event that it is to be managed by the centralized console. Every entry must trigger the execution of the appropriate CLIST that prepares the input necessary for procedure NCC, ie it must execute the CLIST NCCSC00 with the parameters ‘code message’ and ‘text of the message’.

Some examples of entries are shown below:

```
*
*/* Automation Message "IST663I" for NetView Centralized Console.
 IF MSGID='IST663I'
   THEN EXEC(CMD('IST664I ') ROUTE(ONE NCCOPR));
*/* Automation Message "IST1Ø5I" for NetView Centralized Console.
 IF MSGID='IST1Ø5I' & TEXT=MESSAGE
   THEN EXEC( CMD('IST1Ø5I ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "HASP19Ø" for NetView Centralized Console.
 IF MSGID='$HASP19Ø' & TEXT=MESSAGE
   THEN EXEC( CMD('$HASP19Ø ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "IEF45ØI" for NetView Centralized Console.
 IF MSGID='IEF45ØI' & TEXT(9)= CIAB .
  THEN EXEC( CMD('IEF45ØI ' CIAB) ) DISPLAY(Y);
*/* Automation Message "IEF238D" for NetView Centralized Console.
 IF MSGID='IEF238D' & TEXT(1)= RPLY & TEXT=MESSAGE
   THEN EXEC( CMD('IEF238D ' RPLY MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "IEF233A" for NetView Centralized Console.
 IF MSGID='IEF233A' & TEXT=MESSAGE
   THEN EXEC( CMD('IEF233A ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "IEF455D" for NetView Centralized Console.
 IF MSGID='IEF455D' & TEXT(1)= RPLY & TEXT=MESSAGE
   THEN EXEC( CMD('IEF455D ' RPLY MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "IEA911E" for NetView Centralized Console.
 IF MSGID='IEA911E' & TEXT=MESSAGE
   THEN EXEC( CMD('IEA911E ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "ESF499" for NetView Centralized Console.
 IF MSGID='§ESF499' & TEXT=MESSAGE
   THEN EXEC(CMD('§ESF499 ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "IOSØØØI" for NetView Centralized Console.
 IF MSGID='IOSØØØI' & TEXT=MESSAGE
   THEN EXEC( CMD('IOSØØØI ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "IOSØØ3A" for NetView Centralized Console.
 IF MSGID='IOSØØ3A' & TEXT=MESSAGE
   THEN EXEC( CMD('IOSØØ3A ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "IEC512I" for NetView Centralized Console.
 IF MSGID='IEC512I' & TEXT=MESSAGE
   THEN EXEC( CMD('IEC512I ' MESSAGE) ROUTE(ONE NCCOPR));
*/* Automation Message "SLSØ665E" for NetView Centralized Console.
```

```
 IF MSGID='.SLSØ665E' & TEXT=MESSAGE
   THEN EXEC( CMD('SLSØ665E ' MESSAGE) ROUTE(ONE NCCOPR));
*
```

It’s a good idea to define automated operations for all the activities of the centralized console.

In DSIOPF, add the following statement:

```
NCCOPR       OPERATOR    PASSWORD=NCCOPR
             PROFILEN    DSIPROFC
```

In the initialization process of NetView (CNME1034), add the following statement:

```
AUTOTASK OPID=NCCOPR,CONSOLE=xx
```

REXX EXECS IN THE NETVIEW ENVIRONMENT

CLIST NCCSC00

```
/* NCCSCØØ */
/*********************************************************************/
/* CLIST Main.                                                       */
/* Called by Console EVENTs.                                         */
/* It executes:                                                      */
/* - Delete/Allocation dataset to build JCL BATCH-UTILITY            */
/*    FTP and Remote Shell function.                                 */
/* - Delete/Allocation dataset to write report output-FTP.           */
/* - Delete/Allocation dataset for script remote shell.              */
/* - Call CLIST NCCSC1Ø for submit job utility.                      */
/*********************************************************************/
Trace ?e
Parse Arg codmsg message
rfound = Ø
/*********************************************************************/
/* nnrem         = Host-name workstation Digital (Centralized Console)*/
/* usrrem/pswrem = Login/password on nv3ØØ2 for Centralized Console  */
/* rshrem        = File name on nv3ØØ2 for Script Remote Shell       */
/*********************************************************************/
nnrem  = 'nv3ØØ2'
rshrem = 'NCCSVIL.COM'
usrrem = 'nccexec'
pswrem = 'nccexec'
apice    = "'"
microsec = substr(time(l),1Ø,6)
/*********************************************************************/
```

```
/*dsnjob = file MVS with JCL for EXEC of FTP and remote shell function*/
/*dsntxt = file MVS with text of the MVS system event                 */
/*dsnrsh = file MVS with script for EXEC on remote workstation(nv3002)*/
/*dsntab = file MVS with table of the MVS events for Centralized      */
/*                                                              Console*/
/* dsnmsg = file partitioned MVS for result (output) of FTP function  */
/*********************************************************************/
dsnjob = TEMP.NCCS.UTILITY.JOB.T||microsec
dsntxt = TEMP.NCCS.UTILITY.TXT.T||microsec
dsnrsh = TEMP.NCCS.UTILITY.RSH.T||microsec
rshid  = #||microsec
dsntab = NCC.SA01.UTL.TAB
dsnmsg = NCC.SA01.UTL.OUT||'('rshid')'
ADDRESS NETVIEW
"FREE F(NCCTXT)"
"FREE F(NCCRSH)"
"FREE F(NCCJOB)"
"FREE F(NCCTAB)"
/*********************************************************************/
/* Delete/Alloc dataset "TEMP.NCCS.UTILITY.JOB.Tmicrosec"    Type=work*/
/* Delete/Alloc dataset "TEMP.NCCS.UTILITY.TXT.Tmicrosec"    Type=work*/
/* Delete/Alloc dataset "TEMP.NCCS.UTILITY.RSH.Tmicrosec"    Type=work*/
/*********************************************************************/
ADDRESS NETVIEW
"ALLOC FI(NCCJOB) DATASET('"dsnjob"') MOD"
"FREE FI(NCCJOB) DATASET('"dsnjob"') DELETE"
"ALLOC FI(NCCTXT) DATASET('"dsntxt"') MOD"
"FREE FI(NCCTXT) DATASET('"dsntxt"') DELETE"
"ALLOC FI(NCCRSH) DATASET('"dsnrsh"') MOD"
"FREE FI(NCCRSH) DATASET('"dsnrsh"') DELETE"
"ALLOC DATASET('"dsnjob"') FILE(NCCJOB) SPACE(1,1) DSORG(PS)" ,
"RECFM(FB) LRECL(80) BLKSIZE(6160) TRACKS" ,
"UNIT(WORKA) NEW CATALOG"
if rc ¬= 0 then do
                nfile = dsnjob
                signal NCC_Alloc_Error
               End
"ALLOC DATASET('"dsntxt"') FILE(NCCTXT) SPACE(1,0) DSORG(PS)" ,
"RECFM(FB) LRECL(160) BLKSIZE(27840) BLOCK(2)" ,
"UNIT(WORKA) NEW CATALOG"
if rc ¬= 0 then do
                nfile = dsntxt
                signal NCC_Alloc_Error
               End
"ALLOC DATASET('"dsnrsh"') FILE(NCCRSH) SPACE(1,0) DSORG(PS)" ,
"RECFM(FB) LRECL(80) BLKSIZE(6160) BLOCK(2)" ,
"UNIT(WORKA) NEW CATALOG"
if rc ¬= 0 then do
                nfile = dsnrsh
                signal NCC_Alloc_Error
```

```
                End
"ALLOC DATASET('"dsntab"') F(NCCTAB) SHR FREE"
if rc ¬= Ø then do
                 nfile = dsntab
                 signal NCC_Alloc_Error
                End
say time() ' NCC - Read table 'dsntab '...'
ADDRESS MVS
  "NEWSTACK"
  "EXECIO * DISKR NCCTAB (STEM NCCR. FINIS"
  if rc ¬= Ø then do
                  nfile = dsntab
                  signal NCC_Read_Error
                 End
  "DELSTACK"
do i = 1 to nccr.Ø
    rtyp = substr(nccr.i,1,1)
    if rtyp ¬= '_' then iterate
    rmsg = word(nccr.i,2)
    if rmsg = codmsg then do
                           rtxt = word(nccr.i,3)
                           rvgb = word(nccr.i,4)
                           rfound = 1
                           leave
                          End
                     else iterate
End
if rfound ¬= 1 then signal NCC_NoEvent
               else do
/*********************************************************************/
/* Routine for Centralized Console interface for MVS events only     */
/* Refresh NetView global variables for the events                   */
/*********************************************************************/
                    ADDRESS NETVIEW
                    c1= '$'
                    interpret rvgb "='In Course'"
                    interpret c1||rvgb "='IH CR HR'"
                    interpret "'GLOBALV PUTT 'rvgb',$'rvgb"
                    interpret "'GLOBALV GETT 'rvgb',$'rvgb"
                   End
/*********************************************************************/
/* Create files of text and script for interface workstation NV3ØØ2  */
/* Network Centralized Console server                                */
/*********************************************************************/
say time() ' NCC - Interface creation for NCC-Server ...'
ADDRESS MVS
"NEWSTACK"
 QUEUE codmsg '-' message
 QUEUE
"EXECIO * DISKW NCCTXT(FINIS)"
```

```
if rc ¬= Ø then do
                 nfile = dsntxt
                 signal NCC_Write_Error
                End
"DELSTACK"
ADDRESS NETVIEW
"FREE FILE(NCCTXT)"
ADDRESS MVS
"NEWSTACK"
 QUEUE '$! Sviluppo=1'
 QUEUE '$copy 'rtxt 'lta1:'
 QUEUE '$exit'
 QUEUE
"EXECIO * DISKW NCCRSH(FINIS)"
if rc ¬= Ø then do
                 nfile = dsnrsh
                 signal NCC_Write_Error
                End
"DELSTACK"
ADDRESS NETVIEW
"FREE FILE(NCCRSH)"
/*********************************************************************/
/* Creation Job-Utility into "TEMP.NCCS.UTILITY.JOB.Tmicrosec".      */
/*********************************************************************/
nccusr=NCCOPER
jskj = nccusr||'#'
js.1='//'jskj' JOB (ZZNSØØØØ),'
js.2='//        NOTIFY='nccusr','
js.3='//        CLASS=S,'
js.4='//        MSGCLASS=Z,'
js.5='//        MSGLEVEL=(1,1)'
js.6='//*'
js.7="//FTP2     EXEC  PGM=FTP2,PARM='/FIOS'"
js.8='//STEPLIB  DD    DSN=SYS1.TCPICS.V4R1MØ.SAØ1.LINK,DISP=SHR'
js.9='//SYSPRINT DD SYSOUT=Z'
js.1Ø='//*SYSPUT    DD SYSOUT=*,DCB=BLKSIZE=133'
js.11='//SYSPUT  DD DSN='dsnmsg',DISP=SHR'
js.12='//SYSGET    DD *,DCB=BLKSIZE=8Ø'
js.13='OPEN 'nnrem
js.14=usrrem
js.15=pswrem
js.16='put '||apice||dsntxt||apice||' '||rtxt
js.17='put '||apice||dsnrsh||apice||' '||rshrem
js.18='close'
js.19='bye'
js.2Ø='/*'
js.21='//*'
js.22='//STP1 EXEC PGM=REMCMND,COND=(Ø,LT),'
js.23='// PARM=('apice'USER('usrrem') PASS('pswrem') 'nnrem'
@'rshrem||apice')'
```

```
js.24='//STEPLIB  DD DSN=SYS1.TCPICS.V4R1MØ.SAØ1.LINK,DISP=SHR'
js.25='//         DD DSN=TCPICS.V4R1MØ.SAØ1.LOAD,DISP=SHR'
js.26='//*NETRC    DD SYSOUT=X'
js.27='//SYSTERM  DD SYSOUT=X'
js.28='//FPRINTF  DD SYSOUT=X'
js.29='//SYSPRINT DD SYSOUT=Z'
js.3Ø='//SYSTSPRT DD SYSOUT=Z'
js.31='//*'
js.32='//DELWRK   EXEC PGM=IEBGENER,COND=(Ø,LT)'
js.33='//SYSPRINT DD SYSOUT=Z'
js.34='//SYSUT2   DD SYSOUT=(,INTRDR)'
js.35='//*'
js.36='//SYSUT1   DD *,DLM=EA'
/*********************************************************************/
/* Schedule the execution of CLIST NCCXDEL for the deletion of the   */
/* dataset used for the remote shell                                 */
/*********************************************************************/
dsnwrk = dsntxt
js.37='/*$VS,'||apice||'%EXCMD AUTO1,AFTER Ø1,NCCXDEL '||dsnwrk||apice
dsnwrk = dsnrsh
js.38='/*$VS,'||apice||'%EXCMD AUTO1,AFTER Ø2,NCCXDEL '||dsnwrk||apice
js.39='EA'
js.4Ø='//SYSIN    DD DUMMY'
js.41='//*'
js.Ø=41
do a=1 to js.Ø
  js.a=left(js.a,8Ø)
end
say time() ' NCC - Creation 'dsnjob '...'
ADDRESS MVS
  "NEWSTACK"
  "EXECIO * DISKW NCCJOB (STEM JS. FINIS"
  if rc ¬= Ø then do
                   nfile = dsnjob
                   signal NCC_Write_Error
                  End
  "DELSTACK"
ADDRESS NETVIEW
  "FREE FILE(NCCJOB)"
/*********************************************************************/
/* Call CLIST to submit job Batch-Utility NCC                        */
/*********************************************************************/
CALL NCCSC1Ø dsnjob jskj
esito = result
if esito = KO then do
                    ncctxt = 'Submit RSH in error. ',
                             'continuous only for NCC MVS.'
                    End
               else do
                    ncctxt = 'RSH in execution. ',
```

```
                        'Remote Shell id ' || rshid
                    'GLOBALV PUTT ncctxt'
                    say time() ' NCC - ' ncctxt
                    env = 'I'
/*********************************************************************/
/*Call CLIST NCCSC3Ø for Centralized Console interface to MVS events  */
/*only. Refresh NetView global variable of the events for the sessions*/
/*of the qualified operators to this procedure                       */
/*********************************************************************/
                    Call NCCSC3Ø rvgb env ncctxt
                   End
/*********************************************************************/
/* Call CLIST to update log of the events NCC                        */
/*********************************************************************/
'GLOBALV GETT ncctxt'
CALL NCCSC2Ø dsnrsh codmsg rshid esito ncctxt
esito = result
if esito ¬= Ø then do
        say time() '*****************************************************'
        say time() '***                                               ***'
        say time() '*** NCCSCØØ  -  NetView Centralized Console.    ***'
        say time() '***             Update Event Log in error.      ***'
        say time() '***             Call system support staff.      ***'
        say time() '***                                               ***'
        say time() '*****************************************************'
                   End
             else do
        say time() '*****************************************************'
        say time() '***                                               ***'
        say time() '*** NCCSCØØ  -  NetView Centralized Console.    ***'
        say time() '***             Update Event Log carried out.   ***'
        say time() '***                                               ***'
        say time() '*****************************************************'
                   End
Exit
NCC_Alloc_Error:
say time() '*********************************************************'
say time() '***                                                   ***'
say time() '*** NCCSCØØ  -  NetView Centralized Console.          ***'
say time() '***           Error in allocation phase of the file:  ***'
say time() '***                        'nfile'                    ***'
say time() '***           Call system support staff.              ***'
say time() '***                                                   ***'
say time() '*********************************************************'
Exit
NCC_Read_Error:
say time() '*********************************************************'
say time() '***                                                   ***'
say time() '*** NCCSCØØ  -  NetView Centralized Console.          ***'
say time() '***           Error in read phase of the file:        ***'
```

```
say time() '***          'nfile'                                    ***'
say time() '***           Call system support staff.                ***'
say time() '***                                                     ***'
say time() '*************************************************************'
Exit
NCC_Write_Error:
say time() '*************************************************************'
say time() '***                                                     ***'
say time() '*** NCCSC00  -  NetView Centralized Console.           ***'
say time() '***           Error in write phase of the file:         ***'
say time() '***          'nfile'                                    ***'
say time() '***           Call system support staff.                ***'
say time() '***                                                     ***'
say time() '*************************************************************'
Exit
NCC_NoEvent:
say time() '*************************************************************'
say time() '***                                                     ***'
say time() '*** NCCSC00  -  NetView Centralized Console.           ***'
say time() '***           Event 'codmsg' not managed.               ***'
say time() '***           Call system support staff.                ***'
say time() '***                                                     ***'
say time() '*************************************************************'
Exit
```

CLIST NCCSC10

```
/* NCCSC10 */
/*********************************************************************/
/* Called by CLIST main NCCSC00.                                     */
/* It executes : - Submit job Batch-Utility NCC.                     */
/*********************************************************************/
Trace ?e
ARG dsnjob jskj
/*********************************************************************/
/* Routine to submit Batch-Utility NCC.                           */
/*********************************************************************/
n = 0
ADDRESS NETVIEW
NCC_Jescode:
'TRAP DISPLAY MESSAGES  CNM279I'
"SUBMIT '"dsnjob"'"
if rc ¬= 0 then
 do
  rtc = rc
  'TRAP NO MESSAGES'
  'FLUSHQ'
  SIGNAL NCC_ErrorA
 End
  'WAIT 40 SECONDS FOR MESSAGES'
```

```
  Select
    when (EVENT()='M') then
    do
      'MSGREAD'
      'TRAP NO MESSAGES'
      'FLUSHQ'
      Select
        When (MSGID()='CNM279I') then SIGNAL Submit_NCCutl
          Otherwise do
          say time() '************************************************'
          say time() '***                                          ***'
          say time() '*** NCCSC1Ø  -  NetView Centralized Console.  ***'
          say time() '***             Anomaly in Trap message.     ***'
          say time() '***          Function Remote Shell not active.***'
          say time() '***                                          ***'
          say time() '************************************************'
          ncctxt = 'Anomaly in Trap message. Function RSH not active.' ,
                         'Informs System Support staff.'
                'GLOBALV PUTT ncctxt'
                Return 'KO'
               End
      End
    End
    When EVENT() = 'G' then
     do
      'TRAP NO MESSAGES'
      'FLUSHQ'
      SIGNAL NCC_ErrorB
     End
    When (EVENT() = 'T') then
     do
      'TRAP NO MESSAGES'
      'FLUSHQ'
      SIGNAL NCC_Timeout
     End
    Otherwise do
       say time() '************************************************'
       say time() '***                                          ***'
       say time() '*** NCCSC1Ø  -  NetView Centralized Console.  ***'
       say time() '***         Trap unforeseen message condition.   ***'
       say time() '***         Informs System Support staff.        ***'
       say time() '***                                          ***'
       say time() '************************************************'
              ncctxt = 'Trap unforeseen message condition.' ,
                       'Informs System Support staff.'
              'GLOBALV PUTT ncctxt'
              Return 'KO'
             End
  END
Submit_NCCutl:
```

```
'GETMLINE MSG 1'
jobnum = substr(msg,18,8)
say time() ' NCC - Submit Job-Utility-NCC ...'
say time() ' NCC - Jobname ' jskj '...'
say time() ' NCC - Jobnumber ' jobnum '...'
/*************************************************************/
/* Delete dataset "TEMP.UTILITY.NCCS.Tmicrosec"        Type=work */
/*************************************************************/
ADDRESS NETVIEW
"ALLOC FI(SKE) DATASET('"dsnjob"') MOD"
"FREE FI(SKE) DATASET('"dsnjob"') DELETE"
return jobnum
/*************************************************************/
/* Routines for management errors.                           */
/*************************************************************/
NCC_ErrorA:
say time() '*********************************************************'
say time() '***                                                   ***'
say time() '*** NCCSC1Ø  -  NetView Centralized Console.         ***'
say time() '***             Submit remote shell in error.         ***'
say time() '***             Informs system support staff.         ***'
say time() '***                                                   ***'
say time() '*********************************************************'
ncctxt = 'Submit RSH in error.' ,
         'Informs System Support staff.'
'GLOBALV PUTT ncctxt'
Return 'KO'
NCC_ErrorB:
say time() '*********************************************************'
say time() '***                                                   ***'
say time() '*** NCCSC1Ø  -  NetView Centralized Console.         ***'
say time() '***             Trap message in error.                ***'
say time() '***          Informs system support staff. RC 'rtc'   ***'
say time() '***                                                   ***'
say time() '*********************************************************'
ncctxt = 'Trap message Submit in error.' ,
         'Informs System Support staff.'
'GLOBALV PUTT ncctxt'
Return 'KO'
NCC_Timeout:
n = n + 1
if n > 4 then do
    say time() '*********************************************************'
    say time() '***                                                   ***'
    say time() '*** NCCSC1Ø  -  NetView Centralized Console.         ***'
    say time() '***             Trap message in timeout.              ***'
    say time() '***             Informs System Support staff.         ***'
    say time() '***                                                   ***'
    say time() '*********************************************************'
    ncctxt = 'Trap message in Timeout.' ,
```

```
          'Informs System Support staff.'
'GLOBALV PUTT ncctxt'
Return 'KO'
End
     else Signal NCC_Jescode
```

Editor's note: this article will be continued in the next issue.

Espedito Morvillo
Systems Programmer (Italy)

Linking SNA and TCP/IP networking systems

Commonly, TCP/IP and SNA have grown independently within an organization, principally because the two networks work in different ways, the more so since the development of Advanced Peer-to-Peer Networking (APPN). Both systems scale well for large and small networks, and each supports a wide range of end stations. The result is that many organizations have two parallel networking systems developing side-by-side, yet unconnected. And when one considers the fundamental differences in design philosophies of the two systems – such as network infrastructure, addressing, and response to network connection – one can see the challenge in attempting seamless, user-transparent communication between TCP/IP and SNA networks.

THE PROBLEM

The greatest problem in linking SNA and TCP/IP networking systems is that the two differ considerably in almost every aspect of their operation.

SNA is a connection-oriented environment – a connection is established before any information is sent. This gives guaranteed, reliable bandwidth availability for key sessions, and hence SNA networks have traditionally supported mission-critical business applications where control and availability are vital.

TCP/IP is a connectionless environment. This 'open system' approach works by searching out the best route for the data packets while those

packets move between nodes. This approach is much more dynamic and can yield better performance, but it is also less vigilant and reliable.

SOLUTIONS

To counter the inefficiencies of running parallel networks, organizations have the broad choice of the following solutions for combining TCP/IP and SNA across one network transport:

- Do nothing.
- TCP/IP and SNA Logical Partitioning Across Frame Relay.
- SNA encapsulation in TCP/IP.
- SNA End-to-End through APPN or encapsulation of IP by SNA.

Do nothing

Doing nothing is the easy, but ultimately expensive, option. It may appear to cost nothing to leave the network alone, but two networks require twice the management and have twice the technical costs of one system. It requires people with different skills to run the two systems, they must be managed entirely separately, and they require separate lines and equipment.

TCP/IP and SNA Logical Partitioning

TCP/IP and SNA Logical Partitioning is a switch-based solution that creates logically separate interfaces and data paths across a single network. The requirements of both protocols are met, with the allocation of bandwidths, congestion management, and traffic flow priorities, without setting up parallel networks.

There are two systems used for TCP/IP and SNA Logical Partitioning: Frame Relay and ATM (Asynchronous Transfer Mode). The former is the most popular technology for Logical Partitioning of TCP/IP and SNA. ATM is still developing.

Frame Relay is a reliable and inexpensive Wide Area Network (WAN) architecture. It works by replacing a number of low-speed leased lines

with a single Frame Relay connection between the customer site and the Frame Relay provider's Point of Presence. Subscribers use a Frame Relay Access Device (FRAD), a router, controller, or switch to connect to the service. Frame Relay supports Logical Partitioning because it has the ability to assign traffic from different devices to individual Permanent Virtual Circuits (PVCs). These PVCs are administratively established and operate much like a dedicated connection between two remotely-connected devices, even though multiple devices may share the Frame Relay service.

Managing service delivery between TCP/IP and SNA is best achieved by establishing separate PVCs for both, each with its own CIR (Committed Information Rate), which is established when the PVC is set up, and specifies the minimum guaranteed throughput commitment from the Frame Relay network.

Properly configured, TCP/IP traffic will not interfere with the more delay-sensitive SNA traffic. This will reduce the likelihood of session timeouts and give predictable response times to SNA users. Also, because either protocol can take advantage of all the available bandwidth when the other is idle, Frame Relay bandwidth utilization is more effective than dedicated private lines.

For an organization wishing to integrate voice or video, or having high bandwidth requirements, ATM as a WAN architecture holds great promise after its initially cool reception.

Over the next few years ATM is likely to emerge as a dominant network architecture; not as a service interface, but as a transport infrastructure. It has been designed primarily for the international carrier community, with the most sophisticated of data prioritization capabilities, with more levels of prioritization, and is capable of real-time re-allocation of bandwidth in response to changing traffic requirements from attached devices. This will make ATM very effective at integrating TCP/IP and SNA, maximizing bandwidth utilization, and supporting extremely delay-sensitive traffic such as voice and video. However, high component costs and a lack of compliance between vendors' products has held back initial adoption.

SNA encapsulation in TCP/IP

SNA encapsulation in TCP/IP involves SNA frames being encapsulated in IP envelopes, routed on the TCP/IP network, and then de-encapsulated at their destination. With a router-based core WAN, SNA encapsulation seems the most obvious strategy. There are two types of SNA encapsulation to consider – IP Tunnelling and Data Link Switching (DLSw).

IP Tunnelling, although not generally sophisticated enough to support mission-critical networks, is the least complicated of all TCP/IP and SNA transport strategies, and is therefore a better solution for the smaller network. IP Tunnelling is the most basic form of SNA encapsulation, but its weakness lies in its inability to provide the predictable response times that SNA traffic demands, causing potential session timeouts for SNA devices, and frustration and inconvenience with network downtime.

Avoiding potential session timeouts involves allocating bandwidth on peak and not average demand, keeping the number of router hops to a minimum, or both.

Data Link Switching (DLSw) is a sophisticated hybrid of IP Tunnelling. SNA/NetBIOS frames are encapsulated, transported, then de-encapsulated as in IP Tunnelling, but DLSw compensates for unpredictable response times by sending 'keep alive' messages to attached SNA devices. The 'keep alive' message fools the SNA devices into believing an end-to-end connection was established, therefore eliminating session timeouts.

As with IP Tunnelling, unpredictable response times need to be addressed, despite DLSw advantages in solving session timeouts. Similarly again, minimizing the number of router hops and allocating more bandwidth (based on peak demand) may ease response time predictability problems.

SNA encapsulation of IP Tunnelling and DLSw is carried out by routers, but the encapsulation and de-encapsulation of SNA can be performed by software resident on the workstation. This will only offer a solution to the smallest of SNA implementations because of the

cost of software maintenance of every SNA user's desktop, and the cost of host processor cycles to perform the conversion of encapsulation and de-encapsulation.

SNA End-to-End

Although somewhat limited in their scope, there are a couple of SNA End-to-End solutions that may be applicable to an organization that is perhaps running an SDLC-based SNA network.

Most of these users will be using either a pure SNA environment, or are operating their SNA network parallel to, but separately from, an IP network. The solution to integrating IP and SNA could include APPN or the encapsulation of IP by SNA.

APPN and APPN High-Performance Routing (HPR) are similar to TCP/IP in that both are fully functioning network infrastructure transport protocols. They are designed to support peer-to-peer data communications, also addressing and directory (name-to-address) services, entity naming, and transport support for both TCP/IP and SNA (in the form of APPN). They can be supported over most routers and switches, and can be carried over any media type.

Although we cannot expect to see APPN emerging as a strong multi-protocol networking architecture, it does have advantages for the SNA environment, and APPN should greatly reduce the recurring operating and management costs of SNA networks. Despite this very obvious advantage, only a few organizations will adopt APPN as their primary protocol for wide area transport, and will rely instead on APPN-enabled switches and routers to encapsulate or logically partition APPN traffic.

Encapsulation of IP by SNA

If one can run SNA traffic down TCP/IP lines, then why not run TCP/IP over SNA via routers?

A router uses encapsulation schemes to route non-routable protocols such as SDLC and NetBIOS: the hardware transforms TCP/IP traffic into SNA for transport across SNA/APPN networks, then converts it back to TCP/IP at the other end. TCP/IP users gain the benefits of SNA

management and service, but may find that TCP/IP traffic is slowed down because there is no dynamic routing available.

Although this technology is not widely employed, it would be ideal for an organization where SNA-to-TCP/IP traffic ratio is very high. Routers can offer substantial operating flexibility – they provide single or multiple connections between different networks and are intelligent, and link termination eliminates traffic congestion caused by end-to-end polling and acknowledgements. However, it does pass general polling and acknowledgements end-to-end too, which increases traffic congestion and could cause delays during peak traffic periods with an inherent risk of SNA session timeouts.

CONCLUSION

A final decision as to which strategy will yield the most effective result is of course difficult. Every strategy has unique characteristics, of a positive or negative value. Issues of support (technical, hardware, and software), scalability, protocol, etc all need addressing. Will you have Web support, scripting, and programming tools; can it offer mobile and remote computing?

An ideal strategy for today's IT managers will need to be a robust solution offering enterprise-wide data access and connectivity, but which can easily be modified and managed to meet changing business requirements.

For those organizations that already recognize the attraction of mixing voice and data, the additional attraction of adding video too must be apparent. Whilst the consensus is that voice-over IP is the way to go at present, streaming video requires more bandwidth still and smarter networking solutions than most can afford. Unless your organization really does need streaming video-on-demand, the current cost of upgrading is unlikely to be worthwhile; wait a few years and broadband wireless systems should be up and running. These systems will provide 'virtual cabling' and should be capable of everything from cellular voice telephony to e-mail, video conferencing, Web browsing, video-on-demand, and mobile television. So, who needs cabling?

Nick Nourse
Independent Consultant (UK)

OS/2-TSO interconnection

PROBLEM ADDRESSED

There is an increasing acceptance that the two major IBM platforms – mainframes and PCs – are complementary. Although there are many standard packages (programs) that cater for data transmission between the two platforms (with varying degrees of efficiency), the use of a PC to directly invoke a mainframe procedure (command) is less well supported – IBM's Enhanced Conductivity Facilities (ECF) product is often thought of as 'overkill'. Even though the basic routines used by ECF (currently EHLLAPI and LU2, later APPC and LU6.2) are not difficult to use, their use has often been neglected, possibly because a number of different disciplines are needed – C, EHLLAPI, procedure language (REXX), etc, on the PC, and command procedures (REXX) on the mainframe. The TSO EXEC program described here is a generalized procedure language function that can be incorporated in an OS/2 procedure (cmd) to invoke a mainframe procedure (command) and return the results as a REXX variable.

PDS upload/download facility using CLISTs

A quick upload/download facility for PDSs is advantageous because typing lots of send/receive commands on a PC can be very laborious. The CLIST LSTMEMB is written in REXX and can be used to create a file containing a list of all the members of a PDS or a PDSE.

When executing this CLIST, a fully-qualified dataset name is required as the parameter, for example:

```
TSO %PDSUPLOA hlq.pdsname
```

PDSUPLOA

```
/* REXX                                                              */
TRACE All
/*                                                                   */
/*********************************************************************/
/* Procedure REXX .xmit all members of a PDS to a PC                 */
/*********************************************************************/
```

```
/* PROFILE NOPREFIX  */
/* ************************************************************** */
/* * xmit all PDS members on a PC                                 * */
/* ************************************************************** */
/*                                                                */
LIBRARY = ' '
   ADDRESS MVS;
   PARSE UPPER ARG LIBRARY;
/*                                                                */
/* +--------------------------------------------------------------+ */
/* | - CONTROL LIBRARY                                            | */
/* +--------------------------------------------------------------+ */
   ADDRESS MVS;
     MSGLONG  = 7Ø;
/* +--------------------------------------------------------------+ */
/* | - CONTROL DSORG OF THE FILES                                 | */
/* +--------------------------------------------------------------+ */
   ADDRESS TSO;
     DSORG = "NON OK";
     MEMBER = OUTTRAP("LINE.");
     "LISTDS '"LIBRARY"' MEMBER";
     SAY " ";
     DO INDEX = 1 TO 5;
        SAY LINE.INDEX;
        IF POS("PO",LINE.INDEX) > Ø THEN DO;
           DSORG = "OK";
        END;
     END;
     IF DSORG ¬= "OK" THEN DO;
        CFTANO = "ANOMALY : FILE NOT PARTITIONED";
        CALL CFT_MSG_ANOMALIE
     END;
     SAY " ";
     DO INDEX = 7 TO LINE.Ø;
        LINE.INDEX = STRIP(LINE.INDEX);
        DSNAME = LIBRARY"("LINE.INDEX")";
        SAY "===> XMIT '"DSNAME"'";
/* ADDRESS TSO " INDS$FILE GET "DSNAME" ASCII CRLF" */
   ADDRESS sas '++saslog'
    "rsubmit;"
    "filename pds '"library"' disp=shr;"
    "filename down 'c:"dsname"';"
    "proc download infile=pds outfile=down binary ;"
    "run;"
    "endrsubmit;"
      if rc = Ø then
           say 'sas download ok for "dsname"'
           else
           say 'sas download  for "dsname" abend with rc=' rc
     END;
```

```
     EXIT(Ø);
/* +-------------------------------------------------------------+  */
/* | - END PROCEDURE                                             |  */
/* +-------------------------------------------------------------+  */
   ADDRESS MVS;
     EXIT(Ø);
/***  SUBROUTINE NAMES                                           ***/
/* +-------------------------------------------------------------+  */
/* | - MESSAGE ANOMALIES                                         |  */
/* +-------------------------------------------------------------+  */
   CFT_MSG_ANOMALIE:
   ADDRESS MVS;
     QUANT  = RIGHT(DATE(N),4)"."RIGHT("ØØØ"DATE(D),3);
     LINEAA = "*";
     LINEBB = " ";
     LINEØ1 = DATE(W) DATE(N) "-" QUANT "-" TIME(N);
     LINEØ2 = "GROUPE RACF :" RACVAR('GROUPID');
     LINEØ3 = "USERID TSO  :" SYSVAR(SYSUID);
     LINEØ4 = CFTANO;
/*                                                                 */
   ADDRESS MVS;
     SAY "";
     SAY "****"LINEAA||COPIES("*",MSGLONG-LENGTH(LINEAA))"*";
     SAY "*  " LINEBB||COPIES(" ",MSGLONG-LENGTH(LINEBB))"*";
     SAY "* -" LINEØ1||COPIES(" ",MSGLONG-LENGTH(LINEØ1))"*";
     SAY "* -" LINEBB||COPIES(" ",MSGLONG-LENGTH(LINEBB))"*";
     SAY "* -" LINEØ2||COPIES(" ",MSGLONG-LENGTH(LINEØ2))"*";
     SAY "* -" LINEØ3||COPIES(" ",MSGLONG-LENGTH(LINEØ3))"*";
     SAY "* -" LINEBB||COPIES(" ",MSGLONG-LENGTH(LINEBB))"*";
     SAY "* -" LINEØ4||COPIES(" ",MSGLONG-LENGTH(LINEØ4))"*";
     SAY "*  " LINEBB||COPIES(" ",MSGLONG-LENGTH(LINEBB))"*";
     SAY "****"LINEAA||COPIES("*",MSGLONG-LENGTH(LINEAA))"*";
/*                                                                 */
   ADDRESS MVS;
     EXIT(9);
/*                                                                 */
```

LSTMEMB

```
/* REXX */
/*trace all */
/*********************************************************************/
/* Procedure REXX .lstmemb write to a file in output                 */
/*********************************************************************/
/* PROFILE NOPREFIX  */
/* ***************************************************************  */
/* *   list of members PDS PDSE library                          *  */
/* ***************************************************************  */
 ADDRESS TSO "FREE FILE(sortie)"
```

```
LIBRARY = ' '
   ADDRESS MVS;
   PARSE UPPER ARG LIBRARY;
FCode=SYSDSN("'expl69.lstmemb'")
If FCode = 'OK' Then
  Do
    Address TSO "DELETE 'expl69.lstmemb' NONVSAM"
  End
/*                                                                  */
Address TSO "ALLOCATE DATASET('expl69.lstmemb') FILE(Sortie)",
                     "DSORG(PS) SPACE(1,5) TRACKS RELEASE",
                     "LRECL(2Ø) BLKSIZE(629) RECFM(F,B)",
                     "MGMTCLAS(INTERIM) STORCLAS(BASE)",
                     "NEW CATALOG"
/*                                                                  */
   ADDRESS MVS;
   parse upper arg V_PARM;
/* +----------------------------------------------------------------+  */
/* | - CONTROLE V_PARM                                              |  */
/* +----------------------------------------------------------------+  */
   ADDRESS MVS;
     MSGLONG  = 7Ø;
/* +----------------------------------------------------------------+  */
/* | - Control DSORG for the files                                  |  */
/* +----------------------------------------------------------------+  */
   ADDRESS TSO;
     DSORG = "NON OK";
     MEMBER = OUTTRAP("LINE.");
     "LISTDS '"LIBRARY"' MEMBER";
     say " ";
     do INDEX = 1 to 5;
        say LINE.INDEX;
        if pos("PO",LINE.INDEX) > Ø then do;
           DSORG = "OK";
        end;
     end;
     if DSORG ¬= "OK" then do;
        CFTANO = "ANOMALIE : Fichier non partionne";
        call CFT_MSG_ANOMALIE
     end;
     say " ";
     do INDEX = 7 to LINE.Ø;
        LINE.INDEX = strip(LINE.INDEX);
        DSNAME = LIBRARY"("LINE.INDEX")";
 ADDRESS TSO "EXECIO *   DISKW SORTIE (STEM line. FINIS"
        say "===> write '"DSNAME"'" done;
        ADDRESS TSO;
        TSOMSG = msg()
        trace off;
     end;
```

```
     exit(Ø);
/* +-------------------------------------------------------------+  */
/* | - End procedure                                             |  */
/* +-------------------------------------------------------------+  */
 ADDRESS TSO "FREE FILE(sortie)"
   ADDRESS MVS;
     exit(Ø);
/***  SUBROUTINE NAMES                                           ***/
/* +-------------------------------------------------------------+  */
/* | - Message anomalies                                         |  */
/* +-------------------------------------------------------------+  */
   CFT_MSG_ANOMALIE:
   ADDRESS MVS;
     QUANT  = right(date(n),4)"."right("ØØØ"date(d),3);
     LINEAA = "*";
     LINEBB = " ";
     LINEØ1 = date(w) date(n) "-" QUANT "-" time(n);
     LINEØ2 = "Groupe RACF :" racvar('groupid');
     LINEØ3 = "userid TSO  :" sysvar(sysuid);
     LINEØ4 = CFTANO;
/*                                                                 */
   ADDRESS MVS;
     say "";
     say "****"LINEAA||copies("*",MSGLONG-length(LINEAA))"*";
     say "*  " LINEBB||copies(" ",MSGLONG-length(LINEBB))"*";
     say "* -" LINEØ1||copies(" ",MSGLONG-length(LINEØ1))"*";
     say "* -" LINEBB||copies(" ",MSGLONG-length(LINEBB))"*";
     say "* -" LINEØ2||copies(" ",MSGLONG-length(LINEØ2))"*";
     say "* -" LINEØ3||copies(" ",MSGLONG-length(LINEØ3))"*";
     say "* -" LINEBB||copies(" ",MSGLONG-length(LINEBB))"*";
     say "* -" LINEØ4||copies(" ",MSGLONG-length(LINEØ4))"*";
     say "*  " LINEBB||copies(" ",MSGLONG-length(LINEBB))"*";
     say "****"LINEAA||copies("*",MSGLONG-length(LINEAA))"*";
/*                                                                 */
```

Claude Dunand (France)

Why not share your expertise and earn money at the same time? *TCP/SNA Update* is looking for REXX EXECs, macros, CLISTs, program code, etc, that experienced networkers have written to make their life, or the lives of their users, easier. We will publish it (after vetting by our expert panel) and send you a cheque when the article is published. Why not call now for a free copy of our *Notes for contributors*?

Re-creating USS and Logon Mode table macros

VTAM uses Unformatted System Services (USS) tables to obtain messages that it directs to a terminal and to interpret user commands that it receives from terminals. Logon Mode tables describe session protocols that can be used to conduct sessions with sundry telecommunications device types. The macro instructions that are used to create entries in a Logon Mode table are MODETAB, MODEENT, and MODEEND. For USS tables, the macro instructions used are USSTAB, USSCMD, USSPARM, USSMSG, and USSEND.

After our VTAM systems programmer retired, his apprentice was asked to add a couple of entries to our USS and Logon Mode tables. He quickly learned, to his chagrin and my astonishment, that he did not have source statements that he could use to recreate those tables. In order to help him out of his predicament, I wrote two Assembler language programs that he used to recreate his tables. Unfortunately, neither program recreates comment statements.

The utility program, GENUSTAB, issues a LOAD instruction for our USS table – its name is USSTSNA. It then generates all of the control statements that can be used to recreate USSTSNA. Concerning operator messages, it creates statements to rebuild USSMSG entries only for terminal operator messages. We do not use VTAM operator messages, so I decided to forgo processing entries for them. However, GENUSTAB can be easily modified by any systems programmer so that it will generate them. Anyone who uses this utility must modify the LOAD statement used in it to reflect the name of his installation's USS table.

GENMODTB reconstructs a Logon Mode table from a load module entry whose name must be specified on the LOAD macro instruction contained in GENMODTB. GENMODTB does not create specifications for APPNCOA, DCODE, or ENCR since they are not used in our shop. These specifications could be readily generated by a simple modification to GENMODTB. Both PSNAMTAB and USSTSNA reside in SYS1.VTAMLIB, which is in the LNKLST concatenation of our MVS operating system.

To ensure that I could accurately recreate both tables, I assembled

them, link-edited them into a temporary dataset, and then used IBM's IMASPZAP to dump them and the entries from which they were created, then used a utility from Computer Associates to compare their contents for a match – which did not happen, of course, since their CCHHRs were different. A really good, energetic, enterprising systems programmer would have written a simple-minded Assembler program that issues a LOAD instruction for each of them and then used a CLCL instruction to compare them.

In our shop the 'USSMSG MSG=' statements were not specified in ascending order and the text for each of them was not contiguous with its associated USSMSG statement, so I had to rearrange my test so that it matched my utility's generated output.

I have included the source for both of my programs, the jobs that were used to create test tables, and the job-stream that was used to recreate the control statements and test that recreation.

BUILDTST

This is used to create a dataset to contain a test USS table and a test Logon Mode table.

```
//*********************************************************************
//*  JOB-STREAM USED TO CREATE A TEST LOGON MODE TABLE                *
//*********************************************************************
//STEP1 EXEC ASMACL,PARM.L='LIST,LET,XREF,RENT,REUS'
//SYSLIB DD DSN=SYS1.SISTMAC1,DISP=SHR
//       DD DSN=SYS1.MACLIB,DISP=SHR
//C.SYSIN  DD  *
*********************************************************************
*   SAMPLE ENTRIES IN A LOGON MODE TABLE                            *
*********************************************************************
PSNAMTAB MODETAB
         MODEENT LOGMODE=DSILGMOD,                                   *
               FMPROF=X'Ø3',TSPROF=X'Ø3',PRIPROT=X'B1',              *
               SECPROT=X'9Ø',COMPROT=X'3Ø8Ø',RUSIZES=X'87F8',        *
               PSERVIC=X'Ø2ØØØØØØØØØØØ185Ø185Ø7FØØ'
SAMPLE   MODEENT LOGMODE=SAMPLE,   LOGON MODE TABLE ENTRY NAME      *
               FMPROF=X'Ø3',       FUNCTION MANAGER PROFILE         *
               TSPROF=X'Ø3',       TRANSMISSION SERVICES PROFILE    *
               PSNDPAC=X'Ø4',                                       X
               SSNDPAC=X'Ø4',                                       X
               SRCVPAC=X'Ø4',                                       X
               PRIPROT=X'B1',      PRIMARY   LOGICAL UNIT PROTOCOL  *
               SECPROT=X'AØ',      SECONDARY LOGICAL UNIT PROTOCOL  *
```

```
              COMPROT=X'3Ø4Ø',COS=BATCH
         MODEEND
         END
/*
//L.SYSLMOD DD DSN=T5.AGØ3Z.VTAMLIB(PSNAMTAB),
//      DISP=(,PASS),UNIT=DISK,
//     SPACE=(CYL,(1,1,1))
//L.SYSPRINT DD DUMMY
//*                                                                 *
//STEP2 EXEC ASMACL,PARM.L='LIST,LET,XREF,RENT,REUS'
//SYSLIB DD DSN=SYS1.SISTMAC1,DISP=SHR
//       DD DSN=SYS1.MACLIB,DISP=SHR
//C.SYSIN  DD  *
USSTSNA  USSTAB FORMAT=DYNAMIC
*********************************************************************
*   SAMPLE ENTRIES IN A USS TABLE                                   *
*********************************************************************
WHO      USSCMD CMD=WHO,FORMAT=BAL
         USSPARM PARM=P1,DEFAULT=1Ø
         USSPARM PARM=P2,DEFAULT=LUNAME
SIGNNDC  USSCMD CMD=SIGNNDC,FORMAT=BAL,REP=LOGON
         USSPARM PARM=APPLID,DEFAULT=JES2
         USSPARM PARM=LOGMODE,DEFAULT=S3776
         USSPARM PARM=RMTNDC,DEFAULT=RMT414,REP=DATA
ROSCOE   USSCMD CMD=ROSCOE,FORMAT=BAL,REP=LOGON
         USSPARM PARM=APPLID,DEFAULT=ROSCOE
         USSPARM PARM=P1,REP=APPLID
         USSPARM PARM=M,REP=LOGMODE
LOGOFF   USSCMD CMD=LOGOFF,FORMAT=BAL
         USSPARM PARM=APPLID
         USSPARM PARM=TYPE,DEFAULT=COND
         USSPARM PARM=HOLD,DEFAULT=YES
EOD      USSCMD CMD=EOD,FORMAT=BAL
         USSPARM PARM=APPLID
         USSPARM PARM=TYPE,DEFAULT=UNCOND
         USSPARM PARM=HOLD,DEFAULT=NO
         USSMSG MSG=Ø,BUFFER=MSGØ
         DS    ØF
MSGØ     DC    AL2(MSGØE-MSGØS)
MSGØS    DC    X'4Ø154Ø154Ø154Ø154Ø154Ø1540154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø1540154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø1540154Ø15'
         DC    C'COMMAND WAS ACCEPTED: PLEASE STAND-BY..'
MSGØE    EQU   *
MESSAGES USSMSG MSG=1,BUFFER=MSG1
         DS    ØF
MSG1     DC    AL2(MSG1E-MSG1S)
MSG1S    DC    X'4Ø154Ø154Ø154Ø154Ø154Ø1540154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø1540154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø1540154Ø15'
         DC    C'PLEASE ENTER AN APPL-ID ',X'15'
```

```
MSG1E    EQU   *
         USSMSG MSG=2,BUFFER=MSG2
         DS    ØF
MSG2     DC    AL2(MSG2E-MSG2S)
MSG2S    DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    C'OIRMSG2 - APPL-ID IS UNKNOWN ',X'15'
         DC    C' TRY AGAIN ',X'15'
MSG2E    EQU   *
         USSMSG MSG=4,BUFFER=MSG4
         DS    ØF
MSG4     DC    AL2(MSG4E-MSG4S)
MSG4S    DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    C'ISTMSG4 - APPL-ID IS UNAVAILABLE ',X'15'
         DC    C' TRY AGAIN LATER ',X'15'
MSG4E    EQU   *
         USSMSG MSG=7,BUFFER=MSG7
         DS    ØF
MSG7     DC    AL2(MSG7E-MSG7S)
MSG7S    DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    C'ISTMSG7 - BIND ABORTED ',X'15'
MSG7E    EQU   *
         USSMSG MSG=1Ø,BUFFER=(MSG1Ø,LUNAME)
         DS    ØF
MSG1Ø    DC    AL2(MSG1ØE-MSG1ØS)
MSG1ØS   DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    C'                    '
         DC    C'       THIS TERMINAL IS AVAILABLE FOR ',X'15'
         DC    C'                    '
         DC    C'ACCESSING ON-LINE SYSTEMS',X'15'
         DC    X'4Ø154Ø154Ø154Ø15'
         DC    X'4Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø154Ø15'
         DC    C'NODE @@LUNAME -> ENTER AN APPL-ID ',X'15'
MSG1ØE   EQU   *
END      USSEND
         END
/*
//L.SYSLMOD DD DSN=T5.AGØ3Z.VTAMLIB(USSTSNA),DISP=(OLD,CATLG)
//L.SYSPRINT DD DUMMY
```

GENMODTB

```
//*********************************************************************
//* JOB-STREAM THAT WAS USED TO RECREATE AND AUTHENTICATE A LOGON     *
//* MODE TABLE                                                        *
```

```
//*********************************************************************
//STEP1 EXEC ASMACLG
//SYSLIB DD DSN=SYS1.SISTMAC1,DISP=SHR
//       DD DSN=SYS1.MACLIB,DISP=SHR
//C.SYSIN  DD  *
         TITLE 'REBUILD MODEENT ENTRIES IN PSNAMTAB'
*********************************************************************
*        THIS UTILITY PROGRAM ISSUES A LOAD FOR PSNAMTAB, THEN      *
*        RECREATES THE CONTROL STATEMENTS THAT CAN BE USED TO       *
*        REBUILD IT.  FOR RECREATING ANOTHER MODETAB, CHANGE        *
*        PSNAMTAB ON THE LOAD STATEMENT TO ITS NAME.                *
*        NOTE: APPNCOS, DCODE, AND ENCR ARE NOT REBUILT SINCE, AT   *
*              THIS TIME, THEY ARE NOT USED.                        *
*********************************************************************
         SPACE
GENMODTB CSECT
         PRINT NOGEN
         STM   14,12,12(13)
         LR    R1Ø,R15
         USING GENMODTB,R1Ø
         LA    R15,S               CHAIN
         ST    R13,S+4
         ST    R15,8(R13)
         LR    R13,R15
         SPACE
         OPEN  (PGDCB,OUTPUT)
         MVC   PGHOLD,PGCLEAR      CLEAR OUTPUT AREA
         MVC   PGHOLD(16),=CL16'PSNAMTAB MODETAB ' 1ST CONTROL STATEMNT
         BAL   R2,PGPUTOUT         WRITE IT
         SPACE
*********************************************************************
         SPACE
         LOAD  EP=PSNAMTAB         ===>  MEMBER IN VTAMLIB TO BE LOADED
         SPACE
*********************************************************************
         SPACE
         LR    R9,RØ               POINT TO ENTRY POINT
         CLI   Ø(R9),X'CØ'         TEST IF MODETAB CONTROL BLOCK
         BNE   *+2                 TERMINATE IF NOT
         SPACE
         LA    R8,8(R9)            ADDRESS OF 1ST MODEENT BLOCK
         USING MODEENT,R8          ESTABLISH MODEENT ADDRESSABILITY
         LH    R7,2(R9)            NUMBER OF MODEENT ENTRIES IN MODULE
         SR    R6,R6               CLEAR LENGTH REGISTER
         IC    R6,1(R9)            SIZE OF EACH ENTRY
         C     R6,MODESIZE         TEST IF LENGTH OF ENTRY MATCHES MINE
         BNE   *+2                 TERMINATE IF NOT
         SPACE
*********************************************************************
*   PROCESS A MODEENT BLOCK BEGINNING WITH THE LOGMODE SPECIFICATION  *
```

```
*********************************************************************
         SPACE
PGCMDLUP MVC   PGHOLD,PGCLEAR      CLEAR OUTPUT AREA
         MVC   PGHOLD(L'PGMODEC),PGMODEC SET CONSTANT IN OUTPUT AREA
         SPACE
         MVC   PGHOLD(8),LOGMODE   USE LOGMODE FOR A HANDLE ON CTL STMT
         MVC   PGHOLD+25(8),LOGMODE STUFF LOGMODE IN OUTPUT AREA
         EJECT
*********************************************************************
*        PROCESS THE COMPRES OPTION                                 *
*********************************************************************
         SPACE
         CLI   COMPRES,Ø           TEST FOR DEFAULT COMPRES OPTION
         BE    PGTCOMPR            BRANCH IF SO
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD,PGCLEAR      CLEAR OUTPUT AREA
         MVC   PGHOLD+15(L'PGCOMPRS),PGCOMPRS SET CONSTANT IN HOLD AREA
         CLI   COMPRES,1           TEST FOR A COMPRES OPTION OF REQD
         BNE   PGTCOMHI            BRANCH IF NOT
         MVC   PGHOLD+15+L'PGCOMPRS(L'PGCOMREQ),PGCOMREQ SET REQD OPT
         B     PGTCOMPR            CONTINUE...
         SPACE
PGTCOMHI CLI   COMPRES,2           TEST FOR A COMPRES OPTION OF PROHIB
         MVI   PGHOLD+15+L'PGCOMPRS,C'?' SHOW UNKNOWN OPTION
         BNE   PGTCOMPR            BRANCH IF NOT
         MVC   PGHOLD+15+L'PGCOMPRS(L'PGCOMHIB),PGCOMHIB SET PROHIB OPT
         SPACE 3
*********************************************************************
*        PROCESS THE COMPROT OPTION                                 *
*********************************************************************
         SPACE
PGTCOMPR CLC   COMPROT,PGØ         TEST FOR A DEFAULT COMPROT OPTION
         BE    PGTCOS              BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGCOMPRO),PGCOMPRO STOW 'COMPROT=' IN HOLD
         MVC   PGHOLD+15+L'PGCOMPRO(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGCOMPRO+L'PGX(5),COMPROT(3) CONVERT TO PD
         TR    PGHOLD+15+L'PGCOMPRO+L'PGX(4),PGTRANS-24Ø VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGCOMPRO+L'PGX+4,C'''' ENCLOSE VALUE IN 'S
         EJECT
*********************************************************************
*        PROCESS THE COS OPTION                                     *
*********************************************************************
         SPACE
PGTCOS   CLI   COSNAME,C' '        TEST FOR DEFAULT COS OPTION
```

```
         BE    PGTFMPRO            BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGCOS),PGCOS STOW 'COS=' IN HOLD AREA
         MVC   PGHOLD+15+L'PGCOS(8),COSNAME STOW NAME OF COS IN HOLD
         SPACE 3
*************************************************************************
*        PROCESS THE FMPROF OPTION                                      *
*************************************************************************
         SPACE
PGTFMPRO CLI   FMPROF,Ø             TEST FOR DEFAULT FMPROF OPTION
         BE    PGTPRIPR            BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGFMPRO),PGFMPRO STOW 'FMPROF' IN HOLD AREA
         MVC   PGHOLD+15+L'PGFMPRO(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGFMPRO+L'PGX(3),FMPROF(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGFMPRO+L'PGX(2),PGTRANS-24Ø VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGFMPRO+L'PGX+2,C'''' ENCLOSE VALUE IN 'S
         SPACE 3
*************************************************************************
*        PROCESS THE PRIPROT OPTION                                     *
*************************************************************************
         SPACE
PGTPRIPR CLI   PRIPROT,Ø            TEST FOR DEFAULT PRIPROT OPTION
         BE    PGTPSERV            BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGPRIPRO),PGPRIPRO STOW 'PRIPROT' IN HOLD
         MVC   PGHOLD+15+L'PGPRIPRO(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGPRIPRO+L'PGX(3),PRIPROT(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGPRIPRO+L'PGX(2),PGTRANS-24Ø VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGPRIPRO+L'PGX+2,C'''' ENCLOSE VALUE IN 'S
         EJECT
*************************************************************************
*        PROCESS THE PRIPROT OPTION                                     *
*************************************************************************
         SPACE
PGTPSERV CLC   PSERVIC,PGØ          TEST FOR DEFAULT PSERVIC OPTION
         BE    PGTPSND             BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGPSERV),PGPSERV STOW 'PSERVIC' IN HOLD AREA
```

```
         MVC   PGHOLD+15+L'PGPSERV(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGPSERV+L'PGX(9),PSERVIC(5) CONVERT TO PD
         UNPK  PGHOLD+15+L'PGPSERV+L'PGX+8(9),PSERVIC+4(5)
         UNPK  PGHOLD+15+L'PGPSERV+L'PGX+8+8(9),PSERVIC+4+4(5)
         TR    PGHOLD+15+L'PGPSERV+L'PGX(24),PGTRANS-240 VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGPSERV+L'PGX+8+8+8,C''''
         SPACE 3
*************************************************************************
*        PROCESS THE PSNDPAC OPTION                                     *
*************************************************************************
         SPACE
PGTPSND  CLI   PSNDPAC,0           TEST FOR DEFAULT PSNDPAC OPTION
         BE    PGTRUSIZ            BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGPSND),PGPSND STOW 'PSNDPAC=' IN HOLD AREA
         MVC   PGHOLD+15+L'PGPSND(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGPSND+L'PGX(3),PSNDPAC(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGPSND+L'PGX(2),PGTRANS-240 VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGPSND+L'PGX+2,C''''
         SPACE 3
*************************************************************************
*        PROCESS THE RUSIZES OPTION                                     *
*************************************************************************
         SPACE
PGTRUSIZ CLC   RUSIZES,PG0         TEST FOR DEFAULT RUSIZES OPTION
         BE    PGTSEC              BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGRUSIZE),PGRUSIZE STOW 'RUSIZES=' IN HOLD
         MVC   PGHOLD+15+L'PGRUSIZE(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGRUSIZE+L'PGX(5),RUSIZES(3) CONVERT TO PD
         TR    PGHOLD+15+L'PGRUSIZE+L'PGX(4),PGTRANS-240 VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGRUSIZE+L'PGX+4,C''''
         EJECT
*************************************************************************
*        PROCESS THE SECPROT OPTION                                     *
*************************************************************************
         SPACE
PGTSEC   CLI   SECPROT,0           TEST FOR DEFAULT SECPROT OPTION
         BE    PGTSRCV             BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGSECPRO),PGSECPRO STOW 'SECPROT=' IN HOLD
         MVC   PGHOLD+15+L'PGSECPRO(L'PGX),PGX STOW 'X'' IN HOLD AREA
```

```
         UNPK  PGHOLD+15+L'PGSECPRO+L'PGX(3),SECPROT(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGSECPRO+L'PGX(2),PGTRANS-240 VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGSECPRO+L'PGX+2,C''''
         SPACE 3
*********************************************************************
*        PROCESS THE SRCVPAC OPTION                                 *
*********************************************************************
         SPACE
PGTSRCV  CLI   SRCVPAC,0           TEST FOR DEFAULT SRCVPAC OPTION
         BE    PGTSSND             BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGSRCV),PGSRCV STOW 'SRCVPAC=' IN HOLD
         MVC   PGHOLD+15+L'PGSRCV(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGSRCV+L'PGX(3),SRCVPAC(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGSRCV+L'PGX(2),PGTRANS-240 VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGSRCV+L'PGX+2,C''''
         SPACE 3
*********************************************************************
*        PROCESS THE SSNDPAC OPTION                                 *
*********************************************************************
         SPACE
PGTSSND  CLI   SSNDPAC,0           TEST FOR DEFAULT SSNDPAC OPTION
         BE    PGTTSPRO            BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGSSND),PGSSND STOW 'SSNDPAC=' IN HOLD
         MVC   PGHOLD+15+L'PGSSND(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGSSND+L'PGX(3),SSNDPAC(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGSSND+L'PGX(2),PGTRANS-240 VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGSSND+L'PGX+2,C''''
         EJECT
*********************************************************************
*        PROCESS THE TSPROF OPTION                                  *
*********************************************************************
         SPACE
PGTTSPRO CLI   TSPROF,0            TEST FOR DEFAULT TSPROF OPTION
         BE    PGTTYPE             BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGTSPRO),PGTSPRO STOW 'TSPROF=' IN HOLDAREA
         MVC   PGHOLD+15+L'PGTSPRO(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGTSPRO+L'PGX(3),TSPROF(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGTSPRO+L'PGX(2),PGTRANS-240 VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGTSPRO+L'PGX+2,C''''
```

```
         SPACE 3
*************************************************************************
*        PROCESS THE TYPE OPTION                                        *
*************************************************************************
         SPACE
PGTTYPE  CLI   TYPE,1              TEST FOR DEFAULT TYPE OPTION
         BE    PGDONEXT            BRANCH IF IT WAS USED
         SPACE
         BAL   R2,PGEXTEND         SET-UP FOR CONTINUATION STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         SPACE
         MVC   PGHOLD+15(L'PGTYPE),PGTYPE STOW 'TSPROF=' IN HOLDAREA
         MVC   PGHOLD+15+L'PGTYPE(L'PGX),PGX STOW 'X'' IN HOLD AREA
         UNPK  PGHOLD+15+L'PGTYPE+L'PGX(3),TYPE(2) CONVERT TO PD
         TR    PGHOLD+15+L'PGTYPE+L'PGX(2),PGTRANS-24Ø VALU TO EBCDIC
         MVI   PGHOLD+15+L'PGTYPE+L'PGX+2,C''''
         EJECT
*************************************************************************
*        WRITE THE LAST MODEENT PARAMETER THEN PREPARE TO PROCESS       *
*        THE NEXT MODEENT TABLE ENTRY                                   *
*************************************************************************
         SPACE
PGDONEXT AR    R8,R6               POINT TO NEXT MODE TABLE ENTRY
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         BCT   R7,PGCMDLUP          THEN PROCESS IT
         SPACE
         MVC   PGHOLD+9(7),=CL7'MODEEND' CREATE ENDING CONTRL STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         MVC   PGHOLD+9(3),=CL3'END' CREATE END ASSEMBLER STATEMENT
         BAL   R2,PGPUTOUT         WRITE CONTROL STATEMENT
         CLOSE (PGDCB)             CLEAN-UP
         SVC   3                    THEN RETURN TO DUST
         SPACE 2
*************************************************************************
*        SET-UP FOR CONTINUATION OF A MODEENT CONTROL STATEMENT         *
*************************************************************************
         SPACE
PGEXTEND MVI   PGHOLD+71,C'C'      SET CONTINUATION CHARACTER
         LA    R1,PGHOLD+7Ø        POINT TO END OF CONTROL STATEMENT
         SPACE
PGDOLOC  CLI   Ø(R1),C' '          TEST IF BLANK
         BNE   PGDOCONT            BRANCH IF NOT
         BCT   R1,PGDOLOC           ELSE TRY AGAIN AD INFINITUM
         SPACE
PGDOCONT MVI   1(R1),C','          CONTINUE CONTROL STATEMENT
         BR    R2                  THEN CONTINUE WITH PROCESSING MODENT
         SPACE 2
PGPUTOUT PUT   PGDCB,PGHOLD        TRANSCRIBE A CONTROL STATEMENT
         MVC   PGHOLD,PGCLEAR      CLEAR OUTPUT AREA
         BR    R2                  THEN CONTINUE WITH PROCESSING MODENT
         EJECT
```

```
*************************************************************************
*        CONSTANTS AND OTHER SUCH JUNK                                  *
*************************************************************************
         SPACE
PGDCB    DCB   DDNAME=MODETAB,DSORG=PS,LRECL=8Ø,BLKSIZE=8Ø,RECFM=F,    L
               MACRF=PM
MODESIZE DC    A(MODSIZE)
S        DC    18F'Ø'
PGØ      DC    XL12'Ø'
         SPACE
PGMODEC  DC    CL25'         MODEENT LOGMODE='
         SPACE
PGCOMPRS DC    CL8'COMPRES='
PGCOMHIB DC    CL6'PROHIB'
PGCOMREQ DC    CL4'REQD'
         SPACE
PGFMPRO  DC    CL7'FMPROF='
PGTSPRO  DC    CL7'TSPROF='
         SPACE
PGCOMPRO DC    CL8'COMPROT='
PGPRIPRO DC    CL8'PRIPROT='
PGSECPRO DC    CL8'SECPROT='
         SPACE
PGRUSIZE DC    CL8'RUSIZES='
PGPSERV  DC    CL8'PSERVIC='
         SPACE
PGPSND   DC    CL8'PSNDPAC='
PGSSND   DC    CL8'SSNDPAC='
PGSRCV   DC    CL8'SRCVPAC='
         SPACE
PGCOS    DC    CL4'COS='
PGTYPE   DC    CL5'TYPE='
         SPACE 2
PGX      DC    CL2'X'''
         SPACE 2
PGTRANS  DC    C'Ø123456789ABCDEF'
         SPACE 2
PGCLEAR  DC    C' '
PGHOLD   DS    CL8Ø
         SPACE
         YREGS
         SPACE
         LTORG
         EJECT
MODEENT  DSECT
LOGMODE  DS    CL8  LOGMODE         - NAME OF ENTRY IN LOGON MODE TABLE
TYPE     DS    XL1  TYPE
FMPROF   DS    XL1  FM PROFILE      - FUNCTION MANAGEMENT PROFILE
TSPROF   DS    XL1  TS PROFILE      - TRANSMISSION SERVICES PROFILE
PRIPROT  DS    XL1  PRIPROT PROFILE - PROFILE OF PRIMARY LOGICAL UNIT
SECPROT  DS    XL1  SECPROT PROFILE - PROFILE OF SECONDARY LOGICAL UNIT
```

```
COMPROT  DS    XL2  COMPROT PROFILE - COMMON LOGICAL UNIT PROTOCOL
SSNDPAC  DS    XL1  SSNDPAC PROFILE - SECONDARY SEND PACING COUNT
SRCVPAC  DS    XL1  SRCVPAC PROFILE - SECONDARY RECEIVE PACING COUNT
RUSIZES  DS    XL2  RUSIZES PROFILE - RUSIZE SEC/PRI
PSNDPAC  DS    XL1  PSNDPAC PROFILE - PRIMARY SEND PACING COUNT
         DS    XL1  RESERVED
PSERVIC  DS    XL12 PRESENTATION SERVICES
         DS    AL1  ENCR VALUE
COSLEN   DS    XL1  LENGTH OF COSNAME
COSNAME  DS    CL8  DEFAULT FOR COSNAME
         DS    XL1  USER DATA LENGTH
         DS    XL1  LANGUAGE PROFILE
         DS    XL1  DEVICE CODE PROFILE
COMPRES  DS    XL1  COMPRES BYTENGTH
COSPARM  DS    CL8  PARM VALUE
         DS    XL1
         DS    XL1  RESERVED
MODSIZE  EQU   *-LOGMODE
         SPACE
         END
//L.SYSPRINT DD DUMMY
//G.STEPLIB DD DISP=SHR,DSN=T5.AGØ3Z.VTAMLIB
//G.MODETAB DD DISP=(,PASS),DSN=&&MODETAB,SPACE=(CYL,1),UNIT=DISK
//G.ABNLTERM DD SYSOUT=*
//G.SYSUDUMP DD SYSOUT=*
//*
//STEP2 EXEC ASMACL,PARM.L='LIST,LET,XREF,RENT,REUS'
//SYSLIB DD DSN=SYS1.SISTMAC1,DISP=SHR
//       DD DSN=SYS1.MACLIB,DISP=SHR
//C.SYSIN  DD  DISP=(OLD,PASS),DSN=&&MODETAB
//L.SYSLMOD DD DSN=&&VTAMLIB(MODETAB),DISP=(,PASS),SPACE=(CYL,(1,,1)),
// UNIT=DISK
//L.SYSPRINT DD DUMMY
//*
//X EXEC PGM=IMASPZAP,REGION=1ØØK
//SYSLIB DD DISP=(OLD,PASS),DSN=&&VTAMLIB
//SYSPRINT DD DSN=&&PSNAMTAB,DISP=(,PASS),SPACE=(CYL,1),UNIT=DISK
 DUMPT MODETAB  ALL
//*
//Y EXEC PGM=IMASPZAP,REGION=1ØØK
//SYSPRINT DD DSN=&&PSNAMGEN,DISP=(,PASS),SPACE=(CYL,1),UNIT=DISK
//SYSLIB DD DISP=SHR,DSN=T5.AGØ3Z.VTAMLIB
 DUMPT PSNAMTAB ALL
//*
//*********************
//*  IEBIBALL UTILITY *
//*********************
//LBRCOMP  EXEC LIBRCOMP
//SYSUT1   DD DSN=&&PSNAMTAB,DISP=(OLD,DELETE)
//SYSUT2   DD DSN=&&PSNAMGEN,DISP=(OLD,DELETE)
//REPORT   DD  SYSOUT=*
```

```
//SYSIN    DD *
 REP LST=CHANGES,DDNAME=REPORT,
 OLDLINE=((1,'OLD>'),(6,(1,90)),(100,'<OLD')),
 NEWLINE=((5,'NEW>'),(10,(1,90),NEW),(110,'<NEW')),
 TITLE=('-- LOCATE CHANGES IN CARD STREAMS ---')
 OLDFILE STRING=(1,72)
 NEWFILE STRING=(1,72)
//*                                                              *
//PRINT EXEC PGM=IEBGENER,REGION=118K
//SYSPRINT DD DUMMY
//SYSIN DD DUMMY
//SYSUT2 DD SYSOUT=*,DCB=(LRECL=80,BLKSIZE=80,RECFM=F),CHARS=GB12
//SYSUT1 DD  DISP=(OLD,DELETE),DSN=&&MODETAB
```

Editor's note: this article will be continued in the next issue.

Systems Programmer (USA)

An SNA quick reference guide

SNA was originally an IBM master plan for communications with and between IBM computers, terminals, and office systems. It was the company's vehicle for interconnection with other industry-standard networks such as X.25.

SNA contains specifications for the devices or nodes in a network and for the paths between those nodes. Both nodes and paths are organized in hierarchies of seven levels.

The SNA network components include:

- Host processors
- Distributed processors
- Communication controllers
- Cluster controllers
- Workstations
- SNA access methods

- Application subsystems
- Application programs
- Network management programs and network programs.

SNA NODES

Each element in an SNA network controls a specific part of the network. There are different types of node:

- Type 5 represents mainframes
- Type 4 represents communication controllers
- Type 2 represents cluster controllers
- Type 1 represents workstations, terminal controllers, and mini-computers.

THE X.25/QLLC DATA LINK LAYER

The SNA network architecture for packet-switched data networks relies on Logical Link Control (LLC) protocols to provide services and to enhance the quality of those services.

These protocols are known as QLLC because they use 'Qualified' data packets to transfer information between nodes.

In this type of environment, established virtual circuits (generally referred as Switched VCs or SVCs) are viewed by the higher layer of SNA as switched lines while permanent virtual circuits appear as dedicated (leased) lines.

The QLLC protocol is dedicated to SNA nodes connected through X.25-based PSDNs. It uses the Qualifier bit set equal to 1 (Q-packets) to transfer HDLC-like commands and responses.

Q-PACKET FORMAT

The QLLC protocol uses the Q-packet as the basic element of transmission. It contains:

- F – flag (0x7E hexadecimal)
- A – address field
- C – control field
- FCS – Frame Check Sequence.

The Q-packet is composed as follows:

- The address field is 1-byte long and contains the hexadecimal value FF for commands and any value other than FF for responses.
- The control field is 1-byte long and contains the command/ response transmitted by the peer.
- The information field consists of a variable number of bytes and is used to carry QXID, QTEST, or QFRMR data.

QLLC FRAME TYPES

QLLC uses HDLC unnumbered frames, and supervisory commands and answers identical to their SDLC counterparts, which are carried as user data in Q-packets. QLLC commands and responses are initiated by the same higher-level events that initiate their SDLC counterparts. As in SDLC, all QLLC commands should have associated timeout processing.

Claude Dunand (France)

As a free service to subscribers and to remove the need to rekey the scripts, code from individual articles of *TCP/SNA Update* can be accessed on our Web site.

Subscribers need the user-id printed on the envelope containing their *Update* issue. Once they have registered, any code requested will be e-mailed to them.

A mailbox system for SMTP under MVS TCP/IP – 4

This issue we continue the code for the implementation of a mailbox system for SMTP, based on ISPF functions.

The CLIST MAILRECV used to receive mail:

```
 /*                                                                  */
 /*   MAILRECV
 /*   Receive SMTP Mail
 /*    Called from Panel INST81 and Command GETMAIL
 /*
 /*   Parameters:
 /*    MAILUSER: USERID on behalf of which mail is to be received;
 /*              defaults to current USERID in TSO on-line,
 /*              defaults to current USERID in TSO batch;
 /*              To receive on behalf of others OPER auth must be on.
 /*    TOUSER  : USERID to which mail is to be received;
 /*              defaults to current USERID in TSO on-line
 /*              defaults to current USERID in TSO batch
 /*    INVDS   : Indicates validity of dataset to be received;
 /*              when called recursively for invalid dataset this is
 /*              set to YES; default NO.
 /*    RECURSNO: No of recursive calls; Default Ø.
 /*
 /*   SUBROUTINES/EDIT MACROES:
 /*    %EDITRECV
 /*    %ISPFCPY
 /*    %MAILRECV recursively
 /*    %ZSMTPR
 /*
 /*   Utilities used:
 /*                                                                  */
PROC Ø DEBUG(nEBUG) LOGDS(SMTP) MAILUSER() TOUSER() INVDS(NO) +
RECURSNO(Ø)
CONTROL NOMSG NOFLUSH NOLIST NOCONLIST NOSYMLIST PROMPT NOCAPS
ATTN DO
 SET &FLUSH = FLUSH         /* NEXT STATEMENT MUST BE NULL LINE     */

END
ERROR DO
 SET &RET = &LASTCC
 RETURN
END
IF &SYSCAPS(&STR(&DEBUG)) NE DEBUG THEN DO
 IF &SYSISPF = ACTIVE THEN DO
  ISPEXEC VGET (DEBUG)
 END
END
```

```
IF &STR(&DEBUG) = &STR() THEN DO
 SET &DEBUG = NEBUG
END
IF &SYSCAPS(&STR(&DEBUG)) = DEBUG THEN DO
 CONTROL MSG NOFLUSH LIST CONLIST SYMLIST
 WRITE ======> Entering &SYSICMD <======
END
IF &SYSISPF = &STR(NOT ACTIVE) THEN DO
 WRITE ====> Sorry only executable under ISPF (&SYSICMD).
 EXIT CODE(16)
END
IF &FLUSH = FLUSH THEN DO
 IF &SYSISPF = ACTIVE THEN DO
  SET &ZEDSMSG = &str(Function interrupted)
  ISPEXEC SETMSG MSG(ISRZØØ1)
 END
 ISPEXEC LMFREE DATAID(&DID)
 DEL '&DSPREF..&LOGDS..&RCVDS'
 FREE DA('&TOPREF..&LOGDS')
 SET &ZISPFRC = Ø
 IF &SYSISPF = ACTIVE THEN DO
  ISPEXEC VPUT (ZISPFRC) SHARED
 END
 EXIT CODE(&ZISPFRC)
END
ISPEXEC CONTROL ERRORS RETURN
ISPEXEC VGET (ZSCREEN)
IF &RECURSNO > 2 THEN DO
 WRITE ====> Error, too many recursive calls (&SYSICMD).
 SET &ZISPFRC = 8
 ISPEXEC VPUT (ZISPFRC) SHARED
 EXIT CODE(&ZISPFRC)
END
SET &MAXSMTPLR = 251
SET &BLKSMTPLR = 27861
IF &SYSENV = FORE THEN DO
 SET &DSPREF = &SYSUID
END
ELSE DO
 SET &DSPREF = &STR(INSTPREF)      /* HARD-CODED GENERAL PREFIX   */
 SET &GENPREF = &DSPREF
 IF &SYSPREF NE &STR() THEN DO
  IF &SYSPREF NE &DSPREF THEN DO
   IF &SYSUID NE &STR() THEN DO
    SET &DSPREF = &SYSUID
    SET &SYSOUTTRAP = 99999
    SET &RET = Ø
    LISTC ENT('&DSPREF')
    SET &LISTCRET = &RET
    SET &SYSOUTTRAP = Ø
    IF &LISTCRET NE Ø THEN DO
```

```
     SET &DSPREF = &GENPREF
    END
   END
  END
 END
 PROFILE PREFIX(&DSPREF)
END
SET &RCVDS = RCV&ZSCREEN
SET &RCUSER = &STR()
IF &STR(&SYSNSUB(1,&MAILUSER)) NE &STR() THEN DO
 IF &STR(&SYSNSUB(1,&MAILUSER)) NE &SYSUID THEN DO
  SET &RCUSER = USERID(&STR(&SYSNSUB(1,&MAILUSER)))
 END
END
ELSE DO
 SET &MAILUSER = &SYSUID
END
SET &TOPREF = &DSPREF
IF &STR(&SYSNSUB(1,&TOUSER)) NE &STR() THEN DO
 IF &STR(&SYSNSUB(1,&TOUSER)) NE &SYSUID THEN DO
  SET &TOPREF = &STR(&SYSNSUB(1,&TOUSER))
 END
END
ISPEXEC VPUT (TOPREF)
SET &SYSOUTTRAP = 99999
SET &RET = Ø
LISTC ENT('&TOPREF')
SET &LISTCRET = &RET
SET &SYSOUTTRAP = Ø
IF &LISTCRET NE Ø THEN DO
 WRITE ====> Error &TOPREF does not have a catalog alias (&SYSICMD).
 SET &ZISPFRC = 8
 ISPEXEC VPUT (ZISPFRC) SHARED
 EXIT CODE(&ZISPFRC)
END
   /* PREALLOC RECEIVE LOG DATASET TO GET ENOUGH SPACE */
DEL '&DSPREF..&LOGDS..&RCVDS'
ALLOC FI(ZXCVZXCV) NEW CATALOG SPACE(1 15) CYLINDERS +
RECFM(V B) LRECL(255) +
MGMTCLAS(TEMP) STORCLAS(TEMP) +
DA('&DSPREF..&LOGDS..&RCVDS') REUSE
  /* to receive on behalf of other users and into other users         */
  /* the performing user must have OPER-command authority and         */
  /* RACF OPERATIONS; this would normally happen in batch.            */
SET &ERRDATASET = &STR(&DSPREF..ERROR.RECEIVE)
IF &STR(&SYSNSUB(1,&INVDS)) = YES THEN DO  /* if recursive call */
 SET &RECDATASET = &STR(DSNAME('&ERRDATASET'))
 PROFILE PROMPT WTPMSG MSGID
 SET &RECVRET = Ø
 SET &RET = Ø
 SET &SYSOUTTRAP = 999999
```

```
 CONTROL MSG
 RECEIVE LOGDATASET('&DSPREF..&LOGDS..&RCVDS') NODISPLAY &STR(&RCUSER)
 DATA PROMPT
&STR(&RECDATASET)
   /*                              /*  hit enter                   */
 ENDDATA
 SET &RECVRET = &RET
END
ELSE DO
   /* Prepare for datasets and messages pending receive; otherwise    */
   /* they will interfere with mail receipt and be lost.              */
   /* Allow receipt of up to 8 datasets with possible replace.        */
 PROFILE PROMPT WTPMSG MSGID
 SET &RECVRET = Ø
 SET &RET = Ø
 SET &SYSOUTTRAP = 999999
 CONTROL MSG
 RECEIVE LOGDATASET('&DSPREF..&LOGDS..&RCVDS') NODISPLAY &STR(&RCUSER)
 DATA PROMPT
   /*                              /*  hit enter and reply replace */
R
   /*                              /*  hit enter and reply replace */
R
   /*                              /*  hit enter and reply replace */
R
   /*                              /*  hit enter and reply replace */
R
   /*                              /*  hit enter and reply replace */
R
   /*                              /*  hit enter and reply replace */
R
   /*                              /*  hit enter and reply replace */
R
   /*                              /*  hit enter and reply replace */
R
 ENDDATA
 SET &RECVRET = &RET
END
IF &RECVRET = 968 THEN DO      /* IF SOME PROMPT NOT ISSUED */
 SET &RECVRET = Ø              /* IGNORE */
END
SET &APIRET = &RECVRET
SET &SYSOUTTRAP = Ø
SET &MAXLNE = &SYSOUTLINE
IF &SYSCAPS(&STR(&DEBUG)) NE DEBUG THEN DO
 CONTROL NOMSG
END
SET &XMITUSE = &STR()
SET &N = Ø
SET &RET = Ø
DO WHILE &N < &MAXLNE AND &MAXLNE > Ø
```

```
SET &N = &N + 1
SET &SYSDVAL = &STR(&SYSNSUB(2,&&SYSOUTLINE&N)) /*sysnsub(2 contents */
 /* if sysnsub(1 is used above, then later references must use       */
 /*  sysnsub(3 instead of sysnsub(2                                  */
SET &SYSDVAL = &STR(&SYSNSUB(1,&SYSDVAL))
READDVAL &C1 &C2 &C3 &C4 &C5 &C6 &C7 &C8
IF &LENGTH(&STR(&C1)) >= 8 THEN DO
 SET &MSGPFX = &SUBSTR(1:8,&STR(&C1))
 /* NO MORE OR NOTHING TO RECEIVE                                   */
 IF &STR(&MSGPFX) = INMR000I OR &STR(&MSGPFX) = INMR003I OR +
 &STR(&MSGPFX) = INMR900I OR &STR(&MSGPFX) = INMR901I +
 THEN DO
  IF &STR(&MSGPFX) = INMR000I THEN DO   /* NO MORE TO RECEIVE        */
   SET &N = &MAXLNE
   SET &INVDS = NO      /* terminate current recursive call         */
  END
  IF &STR(&MSGPFX) = INMR003I THEN DO   /* NOTHING TO RECEIVE        */
   DEL '&DSPREF..&LOGDS..&RCVDS'
   SET &N = &MAXLNE
   SET &INVDS = NO      /* terminate current recursive call         */
  END
 END
 ELSE DO
  WRITE &STR(&C1) &STR(&C2) &STR(&C3) &STR(&C4) +
  &STR(&C5) &STR(&C6) &STR(&C7) &STR(&C8) (&SYSICMD).
  SET &C = C
  SET &CX = CX
  SET &MAXC = 8              /* max no of variables after readdval */
  SET &CT = 0
  DO WHILE &CT < &MAXC
   SET &CT = &CT + 1
   SET &B = &STR(&SYSNSUB(2,&&C&CT))
   IF &STR(&SYSNSUB(1,&B)) = &STR() THEN DO
    SET &CT = &MAXC
   END
   ELSE DO
    SET &STARTSPC = 1
    SET &HA = 1
    SET &SRCHSPC = &STR(') /* Look for apost ' and remove for send */
    SET &LENC1 = &LENGTH(&STR(&SYSNSUB(2,&&C&CT)))
    DO WHILE &HA > 0
     SET &HA = +
     &SYSINDEX(&STR(&SRCHSPC),&STR(&SYSNSUB(2,&&C&CT)),&STARTSPC)
     IF &HA > 0 THEN DO
      SET &&CX&CT = &SUBSTR(1:&HA-1,&STR(&SYSNSUB(2,&&C&CT)))
      SET &&CX&CT = &STR(&SYSNSUB(2,&&CX&CT))
      SET &&CX&CT = &STR(&SYSNSUB(2,&&CX&CT))+
      &SUBSTR(&HA+1:&LENC1,&STR(&SYSNSUB(2,&&C&CT))) /* any rest */
      SET &B = &STR(&SYSNSUB(2,&&CX&CT))
      SET &&C&CT = &STR(&SYSNSUB(1,&B))
     END
```

```
  END
 END
END
IF &STR(&MSGPFX) = INMRØØ1I THEN DO   /* restore of dataset success*/
 IF &SYSENV NE FORE THEN DO
  SE '&STR(&C1) &STR(&C2) &STR(&C3) &STR(&C4) +
  &STR(&C5) &STR(&C6) &STR(&C7) &STR(&C8)' +
  USER(&STR(&SYSNSUB(1,&MAILUSER))) LOGON
 END
END
IF &STR(&MSGPFX) = INMRØ56I THEN DO   /* not authorized to user-id */
 DEL '&DSPREF..&LOGDS..&RCVDS'
 SET &N = &MAXLNE
END
IF &STR(&MSGPFX) = INMRØ6ØI THEN DO   /* output dataset unusable   */
 IF &STR(&SYSNSUB(1,&INVDS)) = YES THEN DO    /* recursive call    */
  WRITE ====> Recursive call to invalid dataset terminated.
  WRITE ====> Recursive call to invalid dataset terminated.
  SET &INVDS = NO      /* terminate current recursive call         */
  SET &RECVRET = 4     /* set warning cc only */
 END
 ELSE DO
  SET &INVDS = YES     /* indicate recursive call to be activated  */
  IF &SYSENV NE FORE THEN DO
   SE 'Restore of dataset will be retried to &ERRDATASET' +
   USER(&STR(&SYSNSUB(1,&MAILUSER))) LOGON
  END
 END
END
IF &STR(&MSGPFX) = INMRØ63I THEN DO   /* alloc failure of ds-restor*/
 IF &SYSENV NE FORE THEN DO
  SE '&STR(&C1) &STR(&C2) &STR(&C3) &STR(&C4) +
  &STR(&C5) &STR(&C6) &STR(&C7) &STR(&C8)' +
  USER(&STR(&SYSNSUB(1,&MAILUSER))) LOGON
 END
END
IF &STR(&MSGPFX) = INMRØ9ØI THEN DO   /* logging failed            */
 DEL '&DSPREF..&LOGDS..&RCVDS'
 SET &N = &MAXLNE
END
IF &STR(&MSGPFX) = INMRØ91I THEN DO   /* log allocation error      */
 DEL '&DSPREF..&LOGDS..&RCVDS'
 SET &N = &MAXLNE
END
IF &STR(&MSGPFX) = INMRØ92I THEN DO   /* log open failed           */
 DEL '&DSPREF..&LOGDS..&RCVDS'
 SET &N = &MAXLNE
END
IF &STR(&MSGPFX) = INMRØ94I THEN DO   /* log  not sequential       */
 DEL '&DSPREF..&LOGDS..&RCVDS'
 SET &N = &MAXLNE
```

```
   END
   IF &STR(&MSGPFX) = INMR127I THEN DO   /* JES allocation failed     */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
   IF &STR(&MSGPFX) = INMR133I THEN DO   /* open failed for JES ds    */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
   IF &STR(&MSGPFX) = INMR149I THEN DO   /* exit suppressed logging   */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
   IF &STR(&MSGPFX) = INMR15ØI THEN DO   /* terminated by exit        */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
   IF &STR(&MSGPFX) = INMR151I THEN DO   /* terminated by exit        */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
   IF &STR(&MSGPFX) = INMR152I THEN DO   /* failed, node not known    */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
   IF &STR(&MSGPFX) = INMR153I THEN DO   /* User-id not available     */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
   IF &STR(&MSGPFX) = INMR8ØØI THEN DO   /* putget failed */
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &N = &MAXLNE
   END
  END
 END
END
IF &SYSDSN('&TOPREF..&LOGDS') = OK THEN DO
 LISTDSI '&TOPREF..&LOGDS'
  /* SMTP DOES NOT SUPPORT VARABLE RECORDS */
 IF &STR(&SYSDSORG) NE PO AND &STR(&SYSLRECL) NE &MAXSMTPLR AND +
 &STR(&SYSRECFM) NE FB THEN DO
  DEL '&TOPREF..&LOGDS'
 END
END
IF &SYSDSN('&TOPREF..&LOGDS') NE OK THEN DO
 ALLOC FI(ZXCVZXCV) NEW CATALOG SPACE(1 3) CYLINDERS DIR(45) +
 RECFM(F B) LRECL(&MAXSMTPLR) BLKSIZE(&BLKSMTPLR) +
 MGMTCLAS(STANDARD) STORCLAS(STANDARD) +
 DA('&TOPREF..&LOGDS') REUSE
END
IF &SYSDSN('&DSPREF..&LOGDS..&RCVDS') = OK THEN DO
```

```
SET &PFX = A
SET &SUF = Ø
SET &CNT = Ø
SET &MAXCNT = &EVAL(29*1Ø)
SET &CPYRET = 4
DO WHILE &CPYRET > Ø AND &CNT < &MAXCNT
 SET &CNT = &CNT + 1
 SET &NAME = &STR(&PFX)+
 &SUBSTR(1:2,&SYSSDATE)&SUBSTR(4:5,&SYSSDATE)&SUBSTR(7:8,&SYSSDATE)&SUF
 SET &RET = Ø
 %ISPFCPY &NAME &DSPREF..&LOGDS..&RCVDS &TOPREF..&LOGDS TYPE(DSNAME) +
 REPLACE(NOREPLACE) DEBUG(&STR(&DEBUG))
 SET &CPYRET = &RET
 IF &SYSCAPS(&STR(&DEBUG)) = DEBUG THEN DO
  WRITE ======> Reentering &SYSICMD <======
 END
 IF &CPYRET > 12 THEN DO
  WRITE ====> Error2Ø in creating member Rc &CPYRET (&SYSICMD).
  ISPEXEC LMINIT DATAID(DID) +
  DATASET(&STR('&TOPREF..&LOGDS')) ENQ(SHRW)
  ISPEXEC LMMSTATS DATAID(&DID) MEMBER(&NRSTR(&NAME)) +
  USER(RECEIVE)
  ISPEXEC LMFREE DATAID(&DID)
  FREE DA('&TOPREF..&LOGDS')
  DEL '&DSPREF..&LOGDS..&RCVDS'
  SET &ZISPFRC = 8
  ISPEXEC VPUT (ZISPFRC) SHARED
  EXIT CODE(&ZISPFRC)
 END
 SET &SUF = &SUF + 1
 IF &SUF > 9 THEN DO
  SET &SUF = Ø
  SET &STARTSPC = 1
  SET &RC = &STR(ABCDEFGHIJKLMNOPQRSTUVWXYZ$#@)
  SET &LOCSPC = &SYSINDEX(&STR(&PFX),&STR(&RC),&STARTSPC)
  IF &LOCSPC = Ø THEN DO
   WRITE ====> Error21 in creating member rc &CPYRET (&SYSICMD).
   ISPEXEC LMINIT DATAID(DID) +
   DATASET(&STR('&TOPREF..&LOGDS')) ENQ(SHRW)
   ISPEXEC LMMSTATS DATAID(&DID) MEMBER(&NRSTR(&NAME)) +
   USER(RECEIVE)
   ISPEXEC LMFREE DATAID(&DID)
   FREE DA('&TOPREF..&LOGDS')
   DEL '&DSPREF..&LOGDS..&RCVDS'
   SET &ZISPFRC = 8
   ISPEXEC VPUT (ZISPFRC) SHARED
   EXIT CODE(&ZISPFRC)
  END
  SET &RET = Ø
  SET &PFX = &SUBSTR(&LOCSPC+1:&LOCSPC+1,&STR(&RC))
  IF &RET > Ø THEN DO
```

```
    WRITE ====> Error22 in creating member Rc &CPYRET; +
    no more member names available (&SYSICMD).
    ISPEXEC LMINIT DATAID(DID) +
    DATASET(&STR('&TOPREF..&LOGDS')) ENQ(SHRW)
    ISPEXEC LMMSTATS DATAID(&DID) MEMBER(&NRSTR(&NAME)) +
    USER(RECEIVE)
    ISPEXEC LMFREE DATAID(&DID)
    FREE DA('&TOPREF..&LOGDS')
    DEL '&DSPREF..&LOGDS..&RCVDS'
    SET &ZISPFRC = 8
    ISPEXEC VPUT (ZISPFRC) SHARED
    EXIT CODE(&ZISPFRC)
   END
  END
 END
 DEL '&DSPREF..&LOGDS..&RCVDS'
 SET &REROUT = &STR()
 SET &ISPPROF = &MAILUSER..ISPF.ISPPROF(MAILROUT)  /* ISPF profile ds */
 SET &RET = Ø
 IF &SYSDSN('&ISPPROF') = OK THEN DO
  ALLOC FI(MAILROUT) DA('&ISPPROF') SHR REUSE
  OPENFILE MAILROUT INPUT
  SET &CNT = Ø
  SET &MAXCNT = 16
  SET &RET = Ø
  DO WHILE &RET NE 4ØØ AND &CNT < &MAXCNT
   SET &CNT = &CNT + 1
   SET &RET = Ø
   GETFILE MAILROUT
   IF &RET = Ø THEN DO
    SET &SYSDVAL = &STR(&SYSNSUB(1,&MAILROUT))
    SET &SYSDVAL = &STR(&SYSNSUB(1,&SYSDVAL))
    READDVAL &A1 &A2
    IF &STR(&SYSNSUB(1,&A1)) NE &STR() THEN DO
     SET &DOMAIN = &STR()
     IF &STR(&SYSNSUB(1,&A2)) NE &STR() THEN DO
      SET &DOMAIN = &STR(DOMAIN('&SYSNSUB(1,&A2)'))
     END
     %MAILSENS RC(&STR('&SYSNSUB(1,&A1)')) +
     ID(&STR('Rerouted mail from &MAILUSER')) +
     DS(&STR('&TOPREF..&LOGDS(&NAME)')) REROUTE(YES) +
     DEBUG(&STR(&DEBUG)) RCDOMAIN(YES) &DOMAIN
     SET &REROUT = YES
     WRITE &STR(====> Your Mail rerouted to &STR(&SYSNSUB(1,&A1 &A2)))
    END
   END
  END
  CLOSFILE MAILROUT
  FREE FI(MAILROUT)
 END
 IF &SYSISPF = &STR(ACTIVE) THEN DO
```

```
IF &REROUT NE YES THEN DO
 ISPEXEC VPUT (LOGDS)
 SET &FUNCTION = RECEIVE
 ISPEXEC VPUT (FUNCTION)
 %EDITRECV                    /* take edit recovery */
 IF &SYSCAPS(&STR(&DEBUG)) = DEBUG THEN DO
  WRITE =======> Reentering &SYSICMD <=======
 END
 ISPEXEC EDIT DATASET('&TOPREF..&LOGDS(&NAME)') MACRO(%ZSMTPR)
 IF &SYSCAPS(&STR(&DEBUG)) = DEBUG THEN DO
  WRITE =======> Reentering &SYSICMD <=======
 END
 ISPEXEC VGET (ZSMTPR)
END
IF &STR(&ZSMTPR) NE ERROR AND &STR(&ZSNTPR) NE SEVERROR THEN DO
 ISPEXEC LMINIT DATAID(DID) DATASET('&TOPREF..&LOGDS') ENQ(SHRW)
 ISPEXEC LMOPEN DATAID(&DID) OPTION(OUTPUT)
 SET &RET = Ø
 ISPEXEC LMMDEL DATAID(&DID) MEMBER(&NAME) NOENQ
 SET &LMMDELRET = &RET
 IF &LMMDELRET > Ø THEN DO
  SET &MAXCC = Ø
  SET &RET = Ø
  ISPEXEC LMMDEL DATAID(&DID) MEMBER(&NAME)
  SET &LMMDELRET = &RET
 END
 ISPEXEC LMCLOSE DATAID(&DID)
 ISPEXEC LMFREE DATAID(&DID)
 IF &LMMDELRET > Ø THEN DO
  ALLOC FI(DELZXCVB) DA('&TOPREF..&LOGDS') SHR REUSE
  DEL '&TOPREF..&LOGDS(&NAME)' FILE(DELZXCVB)
  FREE FI(DELZXCVB)
 END
END
ELSE DO
 IF &STR(&ZSMTPR) NE SEVERROR THEN DO
  SET &ZEDLMSG = +
  &STR(====> Mail saved to &TOPREF..&LOGDS(&NAME) <====)
  ISPEXEC SETMSG MSG(ISRZØØ1)
  ISPEXEC LMINIT DATAID(DID) +
  DATASET(&STR('&TOPREF..&LOGDS')) ENQ(SHRW)
  ISPEXEC LMMSTATS DATAID(&DID) MEMBER(&NRSTR(&NAME)) +
  USER(RECEIVE)
  ISPEXEC LMFREE DATAID(&DID)
 END
 ELSE DO
  SET &CPYRET = 16    /* SIMULATE COPY ERROR                          */
 END
END
SET &ZSMTPR = &STR()
ISPEXEC VPUT (ZSMTPR)
```

```
  SET &FUNCTION = &STR()
  ISPEXEC VPUT (FUNCTION)
 END
END
ELSE DO
 IF &SYSISPF = ACTIVE THEN DO
  ISPEXEC SETMSG MSG(INST349)
 END
 ELSE DO
  WRITE &STR(No mail to receive)
 END
END
FREE DA('&TOPREF..&LOGDS')
FREE DA('&DSPREF..&LOGDS..&RCVDS')
IF &STR(&SYSNSUB(1,&INVDS)) = YES THEN DO
 SET &TO = &STR()
 SET &ML = &STR()
 IF &STR(&SYSNSUB(1,&TOUSER)) NE &STR() THEN DO
  SET &TO = &STR(TOUSER(&SYSNSUB(1,&TOUSER)))
 END
 IF &STR(&SYSNSUB(1,&MAILUSER)) NE &STR() THEN DO
  SET &MA = &STR(MAILUSER(&SYSNSUB(1,&MAILUSER)))
 END
 %&SYSICMD DEBUG(&STR(&DEBUG)) LOGDS(&STR(&SYSNSUB(1,&LOGDS))) +
 INVDS(&INVDS) &STR(&SYSNSUB(1,&TO)) &STR(&SYSNSUB(1,&MA)) +
 RECURSNO(&EVAL(&RECURSNO+1))
 IF &SYSCAPS(&STR(&DEBUG)) = DEBUG THEN DO
  WRITE ======> Re-entering &SYSICMD <======
 END
 ISPEXEC VGET (ZISPFRC)
 SET &RECVRET = &ZISPFRC
END
SET &ZISPFRC = &RECVRET
ISPEXEC VPUT (ZISPFRC) SHARED
EXIT CODE(&ZISPFRC)
  /*                                                              */
```

The following JCL will execute a general receive program in the background.

Such a job must run under a user who has OPER command authority to receive on behalf of others, and has RACF OPERATIONS to receive into any user's SMTP-mailbox dataset.

```
//SMTPRECV EXEC ISPFBAT,REGION=6M,
//             PARM.ISPFBAT='ISPSTART CMD(%SDSFBACK FUNC(O) PFX(SMTP)
//             RUNTYPE(UPDATE) CMD(S) TYPE(2) JNUM(3) JOBPOS(1)ADT(S))'
```

JCL to receive your own mail in batch:

```
//STEPM1   EXEC ISPFBAT,REGION=6M,
```

```
//    PARM.ISPFBAT='ISPSTART CMD(%MAILRECV)'
```

The CLIST ZSMTPR will parse the received mail log dataset and split it into separate mail, and save them into the mailbox. Also, constructed mail, whether sent or with send suppressed, as well as replies to received mail, will be saved into the mailbox by this routine with an ID in ISPF statistics classifying the mail as MAILED, RECEIVE, SAVED, or REPLY:

```
 /*                                                                   */
 /*  ZSMTPR
 /*  Edit macro to split up SMTP mail from receive or send command;
 /*  Mail is saved into mailbox with id of MAILED, RECEIVE, SAVED, or
 /*  REPLY dependent upon function.
 /*   Called from MAILRECV and MAILSENS
 /*
 /*   SUBROUTINES/EDIT MACROES:
 /*    %EDITRECV
 /*    %ZSMTPI
 /*
 /*   Utilities used:
 /*    SLEEP
 /*    TSOLINE1
 /*                                                                   */
PROC Ø DEBUG(nEBUG)
CONTROL NOMSG NOFLUSH NOLIST NOCONLIST NOSYMLIST NOCAPS
ERROR DO
 SET &RET = &LASTCC
 RETURN
END
IF &SYSCAPS(&STR(&DEBUG)) = DEBUG THEN DO
 CONTROL MSG NOFLUSH LIST CONLIST SYMLIST
 WRITE ======> Entering &SYSICMD <======
END
IF &SYSISPF = &STR(NOT ACTIVE) THEN DO
 WRITE ====> Sorry only executable under ISPF (&SYSICMD).
 EXIT CODE(16)
END
IF &SYSNEST = NO THEN DO
 ISREDIT MACRO PROCESS
END
ISPEXEC CONTROL ERRORS RETURN
ISPEXEC VGET (ZSCREEN)
ISPEXEC VGET (LOGDS,TOPREF)
IF &TOPREF = &STR() THEN DO
 SET &TOPREF = &SYSUID
END
IF &TOPREF = &STR() THEN DO
 SET &TOPREF = &STR(INSTPREF)       /* HARD-CODED GENERAL PREFIX */
END
```

```
ISPEXEC VGET (FUNCTION,SENDSUP,REPLY)
ISPEXEC CONTROL DISPLAY LINE START(14)
ISREDIT (LBOUND,RBOUND) = BOUNDS
SET &ZSMTPR = &STR(OK)
IF &FUNCTION NE MAIL THEN DO
  /* CONTROL MSG NOFLUSH LIST CONLIST SYMLIST */
   /* translate hex code to national characters or special characters */
 ISREDIT C =E5 } ALL            /* Swedish and Danish                */
 ISREDIT C =C5 $ ALL            /* Swedish and Danish                */
 ISREDIT C =E4 { ALL            /* Swedish                           */
 ISREDIT C =C4 # ALL            /* Swedish                           */
 ISREDIT C =F6 : ALL            /* Swedish                           */
 ISREDIT C =D6 @ ALL            /* Swedish                           */
 ISREDIT C =E6 { ALL            /* Danish                            */
 ISREDIT C =C6 # ALL            /* Danish                            */
 ISREDIT C =F8 : ALL            /* Danish                            */
 ISREDIT C =D8 @ ALL            /* Danish                            */
 ISREDIT C =3D '=' ALL          /* ascii equal sign to equal sign    */
 ISREDIT C '= '  ' ' ALL        /* cc:mail peculiarity to blank      */
 ISREDIT C '!'  ' ' ALL
 ISREDIT C P'.'  ' ' ALL        /* translate non-displayable to blank */
 ISREDIT EXCLUDE ALL
 ISREDIT F P'^' ALL
 ISREDIT DEL ALL X
 ISREDIT RESET
END
ISREDIT (MAXLINE) = LINENUM .ZLAST
ISREDIT (JOBEND) = LINENUM .ZLAST
SET &RET = Ø
ISREDIT LABEL &JOBEND = .E Ø
IF &RET > 8 THEN DO
 ISPEXEC CONTROL DISPLAY LINE START(14)
 SLEEP 1
 WRITE ====> Error15 setting first label for jobend (&SYSICMD).
 SET &ZSMTPR = &STR(SEVERROR)
 SET &RET = Ø
 SYSCALL FREERSC &FUNCTION &ZSMTPR DEBUG(&STR(&DEBUG))
 /* SYSCALL RETURN CODE DOES NOT TAKE ERROR ROUTINE */
 SET &RET = &LASTCC
 EXIT CODE(1)
END
ISREDIT CURSOR = 1 Ø
ISREDIT (CURLINE) = LINENUM .ZFIRST
ISREDIT (JOBBEGIN) = LINENUM .ZFIRST
IF &JOBBEGIN = &JOBEND THEN DO
 ISPEXEC CONTROL DISPLAY LINE START(14)
 SLEEP 1
 WRITE ====> Error16 Only one line in mail (&SYSICMD).
 SET &ZSMTPR = &STR(SEVERROR)
 SET &RET = Ø
 SYSCALL FREERSC &FUNCTION &ZSMTPR DEBUG(&STR(&DEBUG))
```

```
 /* SYSCALL RETURN CODE DOES NOT TAKE ERROR ROUTINE */
 SET &RET = &LASTCC
 EXIT CODE(1)
END
SET &RET = Ø
ISREDIT LABEL &JOBBEGIN = .B Ø
IF &RET > 8 THEN DO    /* rc8 indicates existing label replaced */
 ISPEXEC CONTROL DISPLAY LINE START(14)
 SLEEP 1
 WRITE ====> Error13 setting first label for jobbegin (&SYSICMD).
 SET &ZSMTPR = &STR(SEVERROR)
 SET &RET = Ø
 SYSCALL FREERSC &FUNCTION &ZSMTPR DEBUG(&STR(&DEBUG))
 /* SYSCALL RETURN CODE DOES NOT TAKE ERROR ROUTINE */
 SET &RET = &LASTCC
 EXIT CODE(1)
END
SET &NOMORE = NO
SET &BEGJOB = NO
SET &USER = MAILED
IF &STR(&REPLY) = YES THEN DO
 SET &USER = REPLY
END
IF &STR(&SENDSUP) = YES THEN DO
 SET &USER = SAVED
END
SET &SUBJTX = &STR(Subject:)
SET &LSUBJTX = &LENGTH(&STR(SUBJECT:))
SET &MAXCNT = 512   /* SET LOOP CONTROL TO MAX 512 SCANS */
SET &CNT = Ø
DO WHILE &NOMORE NE YES AND &BEGJOB NE YES AND &CNT < &MAXCNT
 SET &CNT = &CNT + 1
 IF &CNT = &MAXCNT THEN DO
  WRITE ====> Error, loop on CNT terminated (&SYSICMD).
 END
 SET &RET = Ø
 ISREDIT LABEL &JOBBEGIN = .B Ø
 IF &RET > 8 THEN DO
  ISPEXEC CONTROL DISPLAY LINE START(14)
  SLEEP 1
  WRITE ====> Error14 setting label for jobbegin (&SYSICMD).
  SET &ZSMTPR = &STR(SEVERROR)
  SET &RET = Ø
  SYSCALL FREERSC &FUNCTION &ZSMTPR DEBUG(&STR(&DEBUG))
  /* SYSCALL RETURN CODE DOES NOT TAKE ERROR ROUTINE */
  SET &RET = &LASTCC
  EXIT CODE(1)
 END
 IF &FUNCTION NE MAIL THEN DO
  SET &USER = RECEIVE
  SET &MRRET = 4
```

```
ISREDIT (MRMSGL,MRMSGR) = CURSOR
SET &MRMSGLM = 2**24-1
SET &RET = Ø
ISREDIT SEEK ' ** MESSAGE ** '
SET &MRRETM = &RET
IF &MRRETM = Ø THEN DO
 ISREDIT (MRMSGLM,MRMSGRM) = CURSOR
END
SET &MRMSGLA = 2**24-1
ISREDIT CURSOR = &MRMSGL Ø
SET &RET = Ø
ISREDIT SEEK ' ** ACKNOWLEDGMENT ** '
SET &MRRETA = &RET
IF &MRRETA = Ø THEN DO
 ISREDIT (MRMSGLA,MRMSGRA) = CURSOR
END
SET &MRMSGLR = 2**24-1
ISREDIT CURSOR = &MRMSGL Ø
IF &MRRETM > Ø AND &MRRETA > Ø THEN DO
 ISREDIT CURSOR = &MRMSGL Ø
END
SET &RET = Ø
ISREDIT SEEK 'RECEIVE         ' 1
SET &MRRETR = &RET
IF &MRRETR = Ø THEN DO
 ISREDIT (MRMSGLR,MRMSGRR) = CURSOR
END
IF &MRRETM = Ø OR &MRRETA = Ø OR &MRRETR = Ø THEN DO
 SET &MRRET = Ø
 SET &MRMSGL = &MRMSGLM
 SET &MRMSGR = &MRMSGRM
 IF &MRMSGL > &MRMSGLA THEN DO
  SET &MRMSGL = &MRMSGLA
  SET &MRMSGR = &MRMSGRA
  SET &NAME = ACKNLM
  SET &ORGNAME = ACKNLM
  SET &SUBNAME = &STR()
 END
 IF &MRMSGL > &MRMSGLR THEN DO
  SET &MRMSGL = &MRMSGLR
  SET &MRMSGR = 4
  SET &NAME = OTHER
  SET &ORGNAME = OTHER
  SET &SUBNAME = &STR()
 END
 ISREDIT CURSOR = &MRMSGL &MRMSGR
END
IF &MRRET > Ø AND &CURLINE = 1 THEN DO
 ISREDIT RESET
 ISREDIT RESET LABEL
 EXIT CODE(1)    /* NO JOB AT ALL */
```

```
 END
 ELSE DO
  SET &CURLINE = Ø
  ISREDIT (JOBMSGL,JOBMSGR) = CURSOR
  IF &MRRET = Ø THEN DO
   SET &RET = Ø
   ISREDIT LABEL &JOBMSGL = .AST 1
   SET &LABRET = &RET
   ISREDIT CURSOR = &JOBMSGL Ø
   SET &RET = Ø
   ISREDIT SEEK RECEIVE WORD .AST .AST
   IF &RET = Ø OR &LABRET > 8 THEN DO
    ISREDIT SHIFT ( &JOBMSGL 15          /* LETS SEE TIMESTAMP */
    ISREDIT SHIFT ( &EVAL(&JOBMSGL+1) 15 /* LETS SEE TIMESTAMP */
   END
  END
  ELSE DO
   SET &RET = Ø
   SYSCALL FREERSC &FUNCTION &ZSMTPR DEBUG(&STR(&DEBUG))
   /* SYSCALL RETURN CODE DOES NOT TAKE ERROR ROUTINE */
   SET &RET = &LASTCC
   EXIT CODE(1)
  END
 END
END
ISREDIT (JOBBEGIN,JOBFROW) = CURSOR
ISREDIT (JOBFREC) = LINE &STR(&SYSNSUB(1,&JOBBEGIN))
IF &JOBMSGR > 3 OR &FUNCTION = MAIL THEN DO
 SET &ENDJOB = YES
 SET &RET = Ø
 ISREDIT LABEL &JOBBEGIN = .B Ø
 IF &RET > 8 THEN DO
  ISPEXEC CONTROL DISPLAY LINE START(14)
  SLEEP 1
  WRITE ====> Error11 setting label for jobbegin (&SYSICMD).
  SET &ZSMTPR = &STR(SEVERROR)
  SET &RET = Ø
  SYSCALL FREERSC &FUNCTION &ZSMTPR DEBUG(&STR(&DEBUG))
  /* SYSCALL RETURN CODE DOES NOT TAKE ERROR ROUTINE */
  SET &RET = &LASTCC
  EXIT CODE(1)
 END
 IF (&STR(&SYSNSUB(1,&NAME)) NE ACKNLM AND +
 &STR(&SYSNSUB(1,&NAME)) NE OTHER AND &FUNCTION NE MAIL) OR +
 &FUNCTION = MAIL THEN DO
  SET &NAME = NONAME
  SET &ORGNAME = NONAME
  SET &SUBNAME = &STR()
  ISREDIT CURSOR = &EVAL(&JOBBEGIN + 1) 1
  SET &RET = Ø
```

```
 ISREDIT SEEK '&SUBJTX'
 IF &RET = Ø THEN DO
  ISREDIT (NAMBEGIN,NAMFROW) = CURSOR
  ISREDIT (NAMFREC) = LINE &STR(&SYSNSUB(1,&NAMBEGIN))
  SET &LENNAM = &LENGTH(&STR(&SYSNSUB(1,&NAMFREC)))
  SET &SYSDVAL = +
  &SUBSTR(&NAMFROW+&LSUBJTX:&LENNAM,&STR(&SYSNSUB(1,&NAMFREC)))
  SET &SYSDVAL = &STR(&SYSNSUB(1,&SYSDVAL))
  READDVAL &A1 &A2 &A3 &A4 &A5 &A6 &A7 &A8 &A9
  SET &SU = &STR(&SYSNSUB(1,&A1)&SYSNSUB(1,&A2)&SYSNSUB(1,&A3)+
  &SYSNSUB(1,&A4)&SYSNSUB(1,&A5)&SYSNSUB(1,&A6)&SYSNSUB(1,&A7)+
  &SYSNSUB(1,&A8)&SYSNSUB(1,&A9))
  IF &STR(&SYSNSUB(1,&SU)) NE &STR() THEN DO
   SET &NAME = &SYSCAPS(&STR(&SYSNSUB(1,&SU)))
   SET &ORGNAME = &STR(&SYSNSUB(1,&A1) &SYSNSUB(1,&A2) +
   &SYSNSUB(1,&A3) &SYSNSUB(1,&A4) &SYSNSUB(1,&A5) +
   &SYSNSUB(1,&A6) &SYSNSUB(1,&A7) &SYSNSUB(1,&A8) &SYSNSUB(1,&A9))
  END
 END
END
```

Editor's note: this article will be continued in the next issue.

Nils Plum
Systems Programmer (Denmark)

March 1995 – December 1998 index

Items below are references to articles that have appeared in *TCP/SNA Update* since March 1995. References show the issue number followed by the page number(s). All these back-issues of *TCP/SNA Update* can be ordered from Xephon. See page 2 for details.

Bay Networks has announced its plan to use the TCP/IP network infrastructure for the convergence of SNA traffic.

New features include RSVP for DLSw, which is for end-to-end QoS, priority queueing for bandwidth management, high-performance routing over IP for IP convergence, IP Multicast with DLSw for lower network traffic, and back-up peers for added redundancy.

RSVP for DLSw allows network administrators to reserve bandwidth based on time of day, by device, or at system start-up for SNA/DLSw traffic across an IP backbone.

Priority queueing features mean up to eight queues can be used for each WAN interface. Traffic may be prioritized among all queues, bandwidth may be apportioned by percentage, and service may be delivered in priority order.

HPR over IP is said to result in improved availability and scalability. IP Multicast with DLSw improves network performance by eliminating overhead.

For further information contact:
Bay Networks, 4401, Great America Parkway, PO Box 58185, Santa Clara, CA 95052-8185, USA.
Tel: (703) 264 8000.
URL: http://www.baynetworks.com.

* * *

Cisco has introduced the IGX8450 wide-area switch, which supports traditional WAN applications, including ATM, Frame Relay, circuit emulation, SNA, compressed voice, voice switching, serial circuit data, and video, with emerging IP applications.

It uses tag switching, Cisco's implementation of multi-protocol label switching, to deliver IP QoS in an ATM multiservice backbone. IP and ATM QoS comes via support for the extension of ABR traffic management to Cisco 7000 routers.

The 8450 works with all platforms in the IGX 8400 series and uses the same network management tools. The IP+ATM feature set is also available as an expansion shelf that can be added to any IGX 8400 series.

For further information contact:
Cisco, 5305 Gulf Drive, Suite 1, New Port Richey, FL 34652, USA.
Tel: (813) 817 0131.
URL: http://www.cisco-ps.com.

* * *

OpenConnect has announced OC:// WebConnect AutoVista, a rules-based engine for converting 3270 and 5250 character-based mainframe application screens into graphical Web-like interfaces. It's said to convert the screens in real time, instantly providing users with a graphical interface to existing mainframe applications. Without any programming or performance loss, the conversion is done through an installation working in concert with the company's Web-to-host software, OC:// WebConnect Pro.

For further information contact:
OpenConnect Systems, 2711 LBJ Frwy, Suite 800, Dallas, TX 75234-7324, USA.
Tel: (972) 484 5200.
URL: http://www.openconnect.com.