219

MVS update

December 2004

In this issue

© Xephon Inc 2004

A tcl PUBLICATION

MVS Update

Published by

Xephon Inc
PO Box 550547
Dallas, Texas 75355
USA

Phone: 214-340-5690
Fax: 214-341-7081

Editor

Trevor Eddolls
E-mail: trevore@xephon.com

Publisher

Bob Thomas
E-mail: info@xephon.com

Subscriptions and back-issues

A year's subscription to *MVS Update*, comprising twelve monthly issues, costs $505.00 in the USA and Canada; £340.00 in the UK; £346.00 in Europe; £352.00 in Australasia and Japan; and £350.00 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 2000 issue, are available separately to subscribers for £29.00 ($43.50) each including postage.

MVS Update on-line

Code from *MVS Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon .com/mvs; you will need to supply a word from the printed issue.

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.

Contributions

When Xephon is given copyright, articles published in *MVS Update* are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

© Xephon Inc 2004. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher.

Printed in England.

Using the PANEXIT feature of ISPF

The ISPF Dialog Manager (DM) provides several mechanisms, including DISPLAY, SELECT, and TBDISPL, that display panels and invite the user to respond. Within the panel definition, limited verification and modification of the field contents can be undertaken – which reduces the need to program exhaustive error-detection in the calling dialog since the returned result can be guaranteed to be valid.

The PANEXIT facility expands this ability and can dramatically reduce the complexity of dialogs. With a panel exit, it is easy (using a regular programming language, including REXX) to extend the verification of variables and to manipulate them. Since all of this is done 'on the panel' there are major benefits – the logic of the calling dialog can be significantly simplified, all calling dialogs automatically get the benefit of the PANEXIT, and maintenance of the PANEXIT code need be done in only one place.

For clarity, I will show a very simplified dialog. Though any programming language may be used for a PANEXIT, speed is not of the essence for this type of process and IBM has provided an advantage in using REXX – so I have used that here.

We need to define a small EXEC to call the panel, the panel itself, and the panel exit EXEC. This little EXEC repeatedly calls the panel until the user hits the 'end' key. When the *Enter* key has been pressed, it will display the value of the variables returned:

```
/*** rexx ***/
AC=Ø
DO WHILE (AC=Ø)
  "ADDPOP"
  "DISPLAY PANEL(PANEL1)"
  AC=RC
  "REMPOP"
  IF (AC=Ø) THEN DO
    SAY "Colour =" COLOUR||", Shade=" SHADE||,
                         ", Result=" STRIP(COLSHAD)||"."
  END
 END
EXIT Ø
```

Now we define a suitable panel, (PANEL1). Again, for clarity, this is a very simplified pop-up panel:

```
)ATTR DEFAULT($%[)
  [ TYPE(NEF) CAPS(ON)          /* Permanent in                 */
  % TYPE(NT)  SKIP(ON)          /* Permanent text               */
  # TYPE(FP)                    /* Hex BD  highlite text        */

)BODY WINDOW(72,9) EXPAND(\\)
%Command#=>[ZCMD
%
%   Colour#>[COLOUR%  (Red, Yellow, Green, Blue.
%                      Do not select Blue on a Monday, Green on a
%                      Wednesday or Dark-Red on any day but a Friday)
%
%    Shade#>[SHADE%   (Light, Dark)
%
\ \%Hit enter to process
)INIT
 .HELP  = PANEL1H
 &COLSHAD = 'aaaaaaaaaaaaaaaaaaaa'

)PROC
 VER (&COLOUR,NB,LIST,RED,YELLOW,GREEN,BLUE)
 VER (&SHADE,NB,LIST,LIGHT,DARK)
 PANEXIT((COLOUR,SHADE,COLSHAD),REXX,EXIT1)
 VER (&COLOUR,NB,LIST,RED,YELLOW,GREEN,BLUE)
 VPUT (COLOUR SHADE COLSHAD) PROFILE

)END
```

A typical situation is where the user options are too complex for simple panel analysis and a third variable, COLSHAD, has to be derived ‘on the panel’ from the value of two others. The dialog that calls this panel guarantees that the user selects a valid day-colour-shade combination before returning (or abandons the call instead).

Note: the PANEXIT statement passes three variable names (two from the panel and one created in the INIT section) to a REXX EXEC called EXIT1. As the final VER statement after the PANEXIT statement shows, if the exit makes the variable

COLOUR invalid, the panel will force the user to choose again.

Finally, we need the code for the panel exit EXEC (EXIT1) itself:

```
/* rexx */
CALL ISPREXPX 'i'                /* set up inbound variables from panel */

DAY=DATE(W)                                          /* get day of week */

SELECT               /* check and derive value for 'colshad' variable */
 WHEN (COLOUR=RED   )&(SHADE=DARK )&(DAY="Friday") THEN COLOUR=?
 WHEN (COLOUR=RED   )&(SHADE=LIGHT)     THEN COLSHAD=PINK
 WHEN (COLOUR=RED   )&(SHADE=DARK )     THEN COLSHAD=CRIMSON

 WHEN (COLOUR=YELLOW)&(SHADE=LIGHT)     THEN COLSHAD=LEMON
 WHEN (COLOUR=YELLOW)&(SHADE=DARK )     THEN COLSHAD=EGGYOLK

 WHEN (COLOUR=GREEN )&(DAY="Wednesday") THEN COLOUR ="?"
 WHEN (COLOUR=GREEN )&(SHADE=LIGHT)     THEN COLSHAD=LIME
 WHEN (COLOUR=GREEN )&(SHADE=DARK )     THEN COLSHAD=BRUNSWICK

 WHEN (COLOUR=BLUE  )&(DAY="Monday")    THEN COLOUR ="?"
 WHEN (COLOUR=BLUE  )&(SHADE=LIGHT)     THEN COLSHAD=SKY
 WHEN (COLOUR=BLUE  )&(SHADE=DARK )     THEN COLSHAD=MIDNIGHT
 OTHERWISE                                   COLOUR ="?"
 END

CALL ISPREXPX 't'                /* set up outbound variables for panel */
EXIT Ø                        /* send back return code. Ø=ok 8=not ok   */
```

Note the ISPREXPX statements – these are supplied by IBM and are vital in the REXX environment. The ‘i’ option prepares the ‘incoming’ variables (those specified on the panel) for processing in the exit code. The ‘t’ option makes the (possibly) modified variables available to the panel on completion. If the exit does not like the day-colour-shade combination, it resets COLOUR to ‘?’ and returns to the panel. The panel re-verifies the COLOUR value and discovers it is now invalid. The user must choose again.

Notes:

- ISPEXEC services cannot be called in a panel exit.
- PANEXIT receives specified variables from the panel, knows no others, and returns only those back to the panel.

It can create variables of its own but these cannot be passed back.

- All variables are passed to the exit in character format and must be returned in the same form.
- Variable length cannot be changed in the exit. The resulting value in the variables on output will be truncated or padded to exactly match the length of the variables on input. (For this reason, when the variable COLSHAD was created on the panel it was made quite long – lots of 'a's – in order to have the space to contain any possible value colour-shade from the exit.)
- The exit can 'send back' an error return code of 8. This signifies to the panel that the exit does not approve the selection. If a message is required to say this – much better than the generic default 'panel exit failed' message – then the message identifier must be specified on the PANEXIT statement of the panel (and it must be in an ISPMLIB concatenated library member as usual).

Deryck Swatman
System Programmer
HM Land Registry (UK)

© Xephon 2004

HSM management class policy tuning

It is not very easy to get your management policies correct first time. I have seen shops where the improper setting of management classes has caused many problems. Lots of jobs would wait for datasets to be recalled, which caused delays to the jobs. In a system running hundreds of jobs and running at 100% of the CPU, it would be impossible to notice which jobs are waiting for the recalls. HSM hangs because it is not able to handle all the requests, causing further delays to the jobs.

The key to any tuning is monitoring. I have developed a job a collect all the dataset movement operations of HSM from primary to ML1/ML2 or *vice versa*. This report will help you identify which jobs are getting delayed because they are waiting for the recalls. For completeness I have included all the data movement operations in the report.

Just to give an example we have identified three issues from the reports generated:

1 Datasets that are not supposed to get migrated are being migrated.

2 Since migration age was specified as 1, the datasets that are created before midnight are migrated during the space management running after midnight.

3 Since low threshold in Storage Group was specified as 0 as much as possible, all the datasets are moved to ML1/ ML2 during space management.

Because of the above problems, most of the jobs used to do recalls, which caused a lot of HSM activity. The migration age was changed to 2 and low threshold was changed to 40. We have seen a 10% to 20% improvement in batch throughput and 5% reduction is HSM CPU.

The function-specific records are written into SMF by HSM if the recording is enabled. The SMF record number for these records is 241 if defaults are not changed.

Otherwise look at the SETSYS SMF command in the ARCCMD00 member of your HSM parmlib dataset. The SMF record number foÞpfunction-specific records would be one more than the number set in the SETSYS SMF command – ie if you have SETSYS SMF(240), then the SMF record number of FSR would be 241. This needs to be updated in the job to extract SMF records if the number of your installation set is different from 240.

Follow the instruction on the job before the job is used.

© 2004. Reproduction prohibited. Please inform Xephon of any infringement.

The generated report looks like this:

```
HSM Functional Report On XXXX
Printed On Ø9/Ø9/Ø4 AT Ø9:22:Ø3 am
======== ======== ============================================ ========
Jobname  UserID   Data Set Name                                Date
======== ======== ============================================ ========
TEST1     USER1   USER1.TEST                                   2ØØ4Ø9Ø7
TEST2     USER2   USER2.TEST                                   2ØØ4Ø9Ø7
TEST3     USER1   USER1.TEST1                                  2ØØ4Ø9Ø7
TEST4     USER2   USER2.TEST1                                  2ØØ4Ø9Ø7
TEST5     USER1   USER1.TEST2                                  2ØØ4Ø9Ø7
TEST6     USER2   USER2.TEST2                                  2ØØ4Ø9Ø7
======== ======== ============================================ ========

=========== =========== =========== ======
Time Req    Time Req    Time Req    Func
Received    Started     Complete
=========== =========== =========== ======
21:Ø2:3Ø.17 21:Ø2:3Ø.27 21:Ø2:3Ø.86 M2->PR
21:Ø3:23.11 21:Ø3:23.11 21:Ø3:23.22 M1->PR
21:19:48.35 21:19:48.58 21:19:49.56 DELETE
21:31:55.37 21:31:55.39 21:31:55.56 DELETE
21:31:55.37 21:31:55.41 21:31:55.56 M2->PR
21:36:18.11 21:36:18.11 21:36:18.27 M1->PR
=========== =========== =========== ======
```

The *Function* column will have the following values:

- M1->PR – RECALL from ML1
- M2->PR – RECALL from ML2
- PR->M1 – migrate to ML1
- PR->M2 – migrate to ML2
- M1->M2 – migrate from M1 to M2 (usually done for HSM)
- DELETE – delete migrated dataset.

If you would like to see all the datasets that are deleted by HSM because they are expired change the included statement to:

```
INCLUDE COND=(43,1,BI,EQ,17)
```

HSMREP JCL

```
//HSMREP   JOB (G84747Ø1),'HSMREP',
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
JOB33223
//         CLASS=D,
//         NOTIFY=&SYSUID,
//         MSGCLASS=X
//*-----------------------------------------------------------------*
//*        Before Submitting:                                       *
//*                                                                 *
//*        Change XXXXXXX to your USERID                            *
//*        Change Input SMF dataset in SMFDUMP step to the dataset  *
//*        containing raw SMF data.
//*-----------------------------------------------------------------*
//* Delete the dataset if already present                           *
//*-----------------------------------------------------------------*
//DELETE   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
  DELETE 'XXXXXXX.SMF.DATA'
  SET LASTCC=Ø
  SET MAXCC=Ø
//***************************************************************
//*  Unload Required SMF records from SMFDATA
//***************************************************************
//SMFDUMP  EXEC PGM=IFASMFDP
//INDD1     DD DISP=SHR,DSN=SMF.DLYTAPE(-1)
//OUTDD1    DD DSN=XXXXXXX.SMF.DATA,DISP=(NEW,CATLG),
//             SPACE=(CYL,(5ØØ,3ØØ)),UNIT=SYSDA
//SYSPRINT  DD SYSOUT=*
//SYSOUT    DD SYSOUT=*
//SYSIN     DD *
   INDD(INDD1,OPTIONS(DUMP))
   OUTDD(OUTDD1,TYPE(241,11Ø))
/*
//*-----------------------------------------------------------------*
//* Generate HRECALL Report from SMF 241 records                    *
//*                                                                 *
//*        Record 241 Field 43 represents                           *
//*                                                                 *
//*         1 = Primary to level 1 migration                        *
//*         2 = Level 1 to level 2 migration                        *
//*             or level 1 to level 1 migration                     *
//*             or level 2 to level 2 migration                     *
//*         3 = Primary to level 2 migration                        *
//*         4 = Recall from level 1 to primary                      *
//*         5 = Recall from level 2 to primary                      *
//*         6 = Delete a migrated dataset                           *
//*         7 = Daily back-up                                       *
//*         8 = Spill back-up                                       *
//*         9 = Recovery                                            *
//*         1Ø = Recycle back-up volume                             *
```

© 2004. Reproduction prohibited. Please inform Xephon of any infringement.

```
//*         11 = Dataset deletion by age                             *
//*         12 = Recycle migration volume                            *
//*         13 = Full volume dump                                    *
//*         14 = Volume or dataset restore                           *
//*         15 = ABACKUP function (see WWFSRcontrol block)           *
//*         16 = ARECOVER function (see WWFSRcontrol block)          *
//*         17 = Expire primary or migrated datasets                 *
//*         18 = Partrel function                                    *
//*         19 = Expire or roll off incremental back-up version      *
//*         20 = (H)BDELETE an incremental back-up version           *
//*------------------------------------------------------------------*
//SORT     EXEC PGM=SORT
//SYSPRINT DD SYSOUT=*
//RECALREP DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SORTIN   DD DISP=SHR,DSN=XXXXXXX.SMF.DATA
//SORTOUT  DD DUMMY
//SORTWK1  DD UNIT=SYSDA,SPACE=(CYL,(20,5))
//SORTWK2  DD UNIT=SYSDA,SPACE=(CYL,(20,5))
//SORTWK3  DD UNIT=SYSDA,SPACE=(CYL,(20,5))
//SORTWK4  DD UNIT=SYSDA,SPACE=(CYL,(20,5))
//SORTWK5  DD UNIT=SYSDA,SPACE=(CYL,(20,5))
//SYSIN    DD *
  OPTION VLSCMP
  INCLUDE COND=(43,1,BI,GT,0,AND,43,1,BI,LT,7)
  SORT FIELDS=(43,1,BI,D,137,4,BI,A,141,4,BI,A)
  OUTFIL FNAMES=RECALREP,
  HEADER1=(1:'HSM Functional Report On XXXX',2/,
           1:'Printed On ',DATE,
             ' AT ',TIME=(12:),3/,
           1:'========',10:'========',
           19:'============================================',
           64:'========',73:'===========',
           85:'===========',97:'===========',109:'======',/,
           1:'Jobname',10:'UserID',19:'Data Set Name',
           64:'Date',73:'Time Req',85:'Time Req',97:'Time Req',
           109:'Func  ',/,
           1:' ',73:'Received',85:'Started ',97:'Complete',/,
           1:'========',10:'========',
           19:'============================================',
           64:'========',73:'===========',
           85:'===========',97:'===========',109:'======'),
  OUTREC=(1:1,4,5:19,8,14:35,8,23:45,44,
          68:137,4,DT1,
          77:141,1,HEX,C':',80:142,1,HEX,C':',83:143,1,HEX,C'.',
          86:144,1,HEX,
          89:145,1,HEX,C':',92:146,1,HEX,C':',95:147,1,HEX,C'.',
          98:148,1,HEX,
          101:149,1,HEX,C':',104:150,1,HEX,C':',107:151,1,HEX,C'.',
          110:152,1,HEX,113:43,1,
                     CHANGE=(6,
                           X'Ø1',C'PR->M1',
                           X'Ø2',C'M1->M2',
                           X'Ø3',C'PR->M2',
                           X'Ø4',C'M1->PR',
                           X'Ø5',C'M2->PR',
                           X'Ø6',C'DELETE'),
                           NOMATCH=(43,1)),
  TRAILER1=(1:'========',1Ø:'========',
            19:'============================================',
            64:'========',73:'===========',
            85:'===========',97:'===========',/,
            1:' End of the Report',/,
            1:'========',1Ø:'========',
            19:'============================================',
            64:'========',73:'===========',
            85:'===========',97:'===========',1Ø9:'======')
/*
```

SRG
Systems Programmer (UK) © Xephon 2004

Example batch job for the submission of selections to a performance monitoring system

INTRODUCTION

As published in the June 2004 issue of *MVS Update* (*High resource users – accumulated statistics suite based on SMF records*) we have been developing a tool-suite to help our users to select batch jobs for performance monitoring. The development of this tool-suite is an on-going task. At the completion of the second phase it was possible for us to make an interface to our in-house suite for the control of our performance monitoring.

This suite is somewhat complicated and is directly based on the Strobe product from Compuware. For this reason it may not be of interest to all readers of *MVS Update*.

To enable others to constructively use the output generated from the on-line selection, I promised to write a simple REXX routine to generate a batch job that would provide an interface to a performance monitoring suite. This routine is described briefly below.

Although I have used the interface to Strobe as an example, I am sure that with a little bit of modification it could be used as an interface for other performance tools.

SELECTION LIST

The displayed selection list from the on-line routine (BTCHSDSP) is written to the dataset whose name is displayed after 'DSN =' on the panel. This dataset, which has a near identical record construction as displayed, is the input for the REXX routine.

```
BTCH04T                                                  Row 1 to 19 of 19
------------------- SELECTions to be processed List --------------------
Command==>                                                   Scroll=> CSR
AL13745                                                  15.07.2004 20:02

D (Delete)                 DSN =  'SYS4.STROBE.BTCHSTAT.SELECT'
-!--------!-------!--------!--------!-----!--------------------!
?!JOBNAME !STEPNUM!STEPNAME!PGMNAME !GOMIN!REQUESTOR & TIME    !
-!--------!-------!--------!--------!-----!--------------------!
  PALBLSTA 2       *OMVSEX  BPXPRECP 644   AL13745  150704 1809
  PALBL056 3       Z0MWBP00 DB2INITR 98    AL13745  150704 1809
  PALBL080 3       Z0MWBP00 DB2INITR 22    AL13745  150704 1809
  PALBL095 3       Z0MWBP00 DB2INITR 103   AL13745  150704 1809
  PALBL156 3       Z0MWBP00 DB2INITR 164   AL13745  150704 1809
  PALBL169 3       Z0MWBP00 DB2INITR 429   AL13745  150704 1809
  PALBL180 3       Z0MWBP00 DB2INITR 20    AL13745  150704 1809
  PALBL256 3       Z0MWBP00 DB2INITR 212   AL13745  150704 1809
  PALBL356 3       Z0MWBP00 DB2INITR 192   AL13745  150704 1809
  PALLFA11 3       Z0MWBP00 DB2INITR 447   AL13745  150704 1808
  PALLF005 3       DB2UNLD  IKJEFT1A 2     AL13745  150704 1809
```

PROCESS

A batch job, in this case BTCHSREQ, should be incorporated into the daily job schedule.

This job calls our REXX routine, which checks for the existence of the input selection dataset and that it is not empty. If the dataset exists and has one or more records, these records will be processed. The relevant information is extracted (jobname, stepnum, gomin, and requestor) and is pasted into the code of another batch job. This second batch job, stored on the stack, is automatically submitted at the end of the routine.

EXTRA FEATURE

It is normal when testing REXX routines to change the code to include the trace facility command, save it, and then rerun it. With the aid of REXX's powerful 'interpret' command, I have included a simple piece of code that will allow the trace function to be automatically switched on. The inclusion of an extra parameter allows 'trace' to be dynamically controlled with this parameter forming the trace option. The default value is 'O' and this switches trace to 'off'.

```
trace_ok = Ø
parse upper arg traceswitch
traceswitch = substr(traceswitch,1,1)
if traceswitch = 'A' | ,
   traceswitch = 'C' | ,
   traceswitch = 'E' | ,
   traceswitch = 'F' | ,
   traceswitch = 'I' | ,
   traceswitch = 'L' | ,
   traceswitch = 'N' | ,
   traceswitch = 'O' | ,
   traceswitch = 'R' | ,
   traceswitch = 'S' then trace_ok = 1
if trace_ok then
  nop
else
  traceswitch = 'o'
/* ****************************************************************** */
interpret 'trace' traceswitch
/* ****************************************************************** */
```

Example generated batch job for request to STROBE:

```
//AL13745S JOB (##DEF),
//             'LOAD STROBE REQS',REGION=ØM,
//             NOTIFY=&SYSUID,CLASS=Ø,MSGCLASS=T,MSGLEVEL=(1,1)
```

```
/*JOBPARM SYSAFF=ALFØ
//STROBESM EXEC PGM=STRBCSR
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
ADD PALBLSTA,NUMBER=2,GOMIN=716,SAMPLES=2ØØØØ,
DSNAME=ALEDS.STROBE,UNIT=SYSDA,NOTIFY=AL13745,DISPOSITION=CATLG,
NOLIMIT,ISPFFLAG=ØØØØ
ADD PALBLØ56,NUMBER=3,GOMIN=124,SAMPLES=2ØØØØ,
DSNAME=ALEDS.STROBE,UNIT=SYSDA,NOTIFY=AL13745,DISPOSITION=CATLG,
NOLIMIT,ISPFFLAG=ØØØØ
ADD PALBLØ8Ø,NUMBER=3,GOMIN=31,SAMPLES=2ØØØØ,
DSNAME=ALEDS.STROBE,UNIT=SYSDA,NOTIFY=AL13745,DISPOSITION=CATLG,
NOLIMIT,ISPFFLAG=ØØØØ
/*
```

SUMMARY

The whole suite in its current form provides:

- The collection of statistical information for batch jobs.
- Batch reports by estimated yearly SUs/average SUs resource usage.
- On-line display of the statistical information:
  - selection of *nnn* highest resource users by estimated yearly service units.
  - selection of *nnn* highest resource users by average service units.
- Selection of individual jobsteps for performance monitoring:
  - prevention of selections through the filter function.
  - summary of selections and selection deletion function.
- Automatic requests to performance monitoring tool (Strobe) based on selections.

The follow code is supplied:

- BTCHSPRP – batch job generation routine.
- BTCHSREQ – control JCL for the daily run.

BTCHSPRP

```
/* REXX                                                              */
/* ***************************************************************** */
/* ### BTCHSPRP ###                                                  */
/* ----------------------------------------------------------------- */
/* Description:                                                      */
/* Routine to extract information from the selection list produced   */
/* by BTCHSDSP and to use this information to create a batch job.    */
/* The batch job, when run, will automatically create requests to    */
/* be measured by a performance measurement system, in this case     */
/* STROBE from COMPUWARE.                                            */
/*                                                                   */
/* Input dataset format:                                             */
/* JOBNAME STEPNUMBER STEPNAME PROGRAMNAME ESTMIN USER DATE TIME      */
/*                                                                   */
/* ESTMIN = estimated job runtime in minutes                         */
/*                                                                   */
/* Information used:                                                 */
/* JOBNAME STEPNUMBER ESTMIN USER                                    */
/*                                                                   */
/* ----------------------------------------------------------------- */
/* Created ....: 06.08.2004      Rolf Parker                         */
/* ----------------------------------------------------------------- */
/* ***************************************************************** */
/* ***************************************************************** */
/* trace switch                                                      */
/* to enable the control of the trace mode by use of a parameter     */
/* All, Commands, Error, Failure, Intermediates, Labels, Normal      */
/* Off, Results, and Scan                                            */
/* ***************************************************************** */
trace_ok = 0
parse upper arg traceswitch
traceswitch = substr(traceswitch,1,1)
if traceswitch = 'A' | ,
   traceswitch = 'C' | ,
   traceswitch = 'E' | ,
   traceswitch = 'F' | ,
   traceswitch = 'I' | ,
   traceswitch = 'L' | ,
   traceswitch = 'N' | ,
   traceswitch = 'O' | ,
   traceswitch = 'R' | ,
   traceswitch = 'S' then trace_ok = 1
if trace_ok then
  nop
else
  traceswitch = 'o'
/* ***************************************************************** */
interpret 'trace' traceswitch
```

```
/* ************************************************************** */

/* ************************************************************** */
/* global STROBE variables:                                       */
/* samples = number of samples attempted per request              */
/* samples/gomin = sample rate per minute                         */
/* hlq = high-level qualifier for the datasets to contain the samples */
/* unit = unit used for sample dataset allocation                 */
/* ************************************************************** */
  samples = 20000
  hlq = "ALEDS.STROBE"
  unit = "SYSDA"
/* ************************************************************** */
/* when available shows the origin of this called REXX routine    */
/* if unavailable the various variables will be shown as '?'      */
/* src3 = member name                                             */
/* src5 = dataset name                                            */
/* src4 = DD name                                                 */
/* ************************************************************** */
parse SOURCE src1 src2 src3 src4 src5 src6 src7 src8 src9
say 'running: 'src3' from: 'src5' DD: 'src4
/* ************************************************************** */
  retcode = 0
  seldd = 'BTCHSSEl'
/* ************************************************************** */
/* seldsn = dataset created by ISPF routine BTCHSDSP              */
/* ************************************************************** */
  seldsn = "'SYS4.STROBE.BTCHSTAT.SELECT'"
/* ************************************************************** */
/* check and proceed if input dataset is OK                       */
/* ************************************************************** */
  ADDRESS 'TSO'

  if SYSDSN(seldsn) = 'OK' then
    do
      "FREE DSNAME("seldsn")"
      "ALLOC DD("seldd") DSN("seldsn") SHR MOD "
      call prepmain
    end
exit retcode

/* ************************************************************** */
/* prepmain:                                                      */
/* read complete dataset, if non-empty then proceed               */
/* ************************************************************** */
prepmain:
  'EXECIO * DISKR  'seldd' (STEM infile. FINIS'
  if rc = 0 then
    do
      if infile.0 > 0 then
```

```
        do
          say 'processing ' infile.Ø 'records'
          call buildjcl
        end
      else
        do
          say 'dataset ' seldsn ' is EMPTY'
        end
    end
  else
    do
       retcode = rc
       say 'error reading dataset ' seldsn
    end
  "FREE DSNAME("seldsn")"
/* ************************************************************** */
/* delete used input dataset -                                    */
/* will be newly allocated by next call of BTCHSDSP               */
/* ************************************************************** */
/*  "DELETE "seldsn */
return

/* ************************************************************** */
/* buildjcl:                                                      */
/* build together JCL statements in the queue and then submit the */
/* contents of the queue                                          */
/*                                                                */
/* ************************************************************** */
buildjcl:
  ADDRESS 'TSO'
  "NewStack"
/* ************************************************************** */
/* Standard Jobcards                                              */
/* ************************************************************** */
  queue "//"USERID()"S JOB (##DEF),"
  queue "//             'LOAD STROBE REQS',REGION=ØM,"
  queue "//             NOTIFY=&SYSUID,CLASS=Ø,MSGCLASS=T,MSGLEVEL=(1,1)"
  queue "/*JOBPARM SYSAFF=ALFØ"
/* ************************************************************** */
/* STROBE Jobcards                                                */
/* ************************************************************** */
  queue "//STROBESM EXEC PGM=STRBCSR"
  queue "//SYSPRINT DD SYSOUT=*"
  queue "//SYSIN   DD *"
/* ************************************************************** */
/* variable STROBE request input                                  */
/* ************************************************************** */
  do i = 1 to infile.Ø
    call insert_request
  end
```

```
/* ****************************************************************** */
/* JOB END                                                            */
/* ****************************************************************** */
  queue "/*"
/* ****************************************************************** */
/* Queue END marker/label                                             */
/* ****************************************************************** */
  queue "$$"

  X = MSG('ON')
/* ****************************************************************** */
/* SUBMIT all lines up until the END marked by $$                     */
/* ****************************************************************** */
  "SUBMIT * END($$)"
  X = MSG('OFF')
return

/* ****************************************************************** */
/* insert_request:                                                    */
/* variable STROBE request input obtained from input file             */
/* ****************************************************************** */
insert_request:
  parse var infile.i jobnm stepnr . . gomin notif . .
  queue "ADD "jobnm",NUMBER="stepnr",GOMIN="gomin",SAMPLES="samples","
  queue "DSNAME="hlq",UNIT="unit",NOTIFY="notif",DISPOSITION=CATLG,"
  queue "NOLIMIT,ISPFFLAG=0000"
return
```

BTCHSREQ

```
//AL13745R JOB (SR00,SR016882,SR016882),'BTCHSREQ - STROBE',
//              CLASS=0,MSGCLASS=H,REGION=0M,
//              NOTIFY=AL13745,MSGLEVEL=(1,1)
/*JOBPARM L=0100,G=99999
//* -----------------------------------------------------------------
//*
//*  Daily job to send requests for performance monitoring to
//*  STROBE.
//*  The second parameter is a switch for the trace function
//*  within REXX.
//*  Default is 'o' for OFF.
//*
//* -----------------------------------------------------------------
//REXX01   EXEC PGM=IKJEFT1A,PARM='BTCHSPRP R'
//SYSEXEC   DD  DISP=SHR,DSN=AL13745.ISPF.EXEC
//SYSTSIN   DD  DUMMY
//SYSTSPRT DD  SYSOUT=*
/*
```

Rolf Parker
Systems Programmer (Germany)

© Xephon 2004

Object-oriented COBOL

OVERVIEW

In OO (object-oriented) COBOL, there are three kinds of program:

1 Class definitions

2 Method definitions

3 Client programs.

A class definition is similar to an ordinary program. It has the usual four divisions, but with various special features. In particular, the PROCEDURE DIVISION doesn't contain procedural code in the usual way. Rather, it contains all the code for all the methods of the class. Each method definition has four divisions of its own, and its PROCEDURE DIVISION contains the procedural code. Because of this arrangement, it is not possible to define some methods in one source file and others in another. All method definitions for a class must reside in the same source file. A class with many complex methods may require an unusually large source file.

A client program may be an ordinary program or a method definition. It uses the INVOKE verb, rather than CALL, to execute a method.

Defining a subclass is no different from defining a base class. In fact every class is a subclass, except for the built-in class SOMObject. A class may itself be an instance of a metaclass – a class of classes. You can define your own metaclasses, derived from SOMClass. Since a metaclass is just another kind of subclass, the syntax is the same as for any other subclass.

CLASS DEFINITIONS

IDENTIFICATION DIVISION

Instead of a PROGRAM-ID, a class definition has a CLASS-ID, followed by the name of the class: CLASS-ID. OBJECT1 INHERITS SOMObject. The INHERITS clause specifies the base class. All classes inherit directly or indirectly from SOMObject, a built-in generic class. The INHERITS clause may specify multiple inheritance by listing multiple base classes. The order in which the base classes are listed is significant. When two base classes have methods with the same name, the derived class inherits the method from whichever base class is listed earlier.

ENVIRONMENT DIVISION

In the CONFIGURATION section, a special REPOSITORY paragraph must declare each of the base classes and any other classes used by the methods. Optionally, it may also declare the class being defined:

```
REPOSITORY.  CLASS SOMObject IS 'SOMObject'
             CLASS OBJECT1   IS 'OBJECT1'
             CLASS OBJ2      IS 'OBJ2'.
```

Each CLASS clause maps the name of a class to the name by which it is known in the Interface Repository (IR). Thus a CLASS clause does for objects what SELECT...ASSIGN does for files. In my examples, the internal names match the external names. A class definition cannot have an INPUT-OUTPUT SECTION, because it cannot allocate any files. Methods, however, may access files.

DATA DIVISION

The DATA DIVISION, if present, can contain only a WORKING-STORAGE section, which defines per-instance data – each instance has its own set of WORKING-STORAGE variables. The syntax is similar to that of an ordinary WORKING-STORAGE section, except:

- The GLOBAL attribute is allowed but has no effect, since you can’t have nested programs in a class definition.
- The EXTERNAL attribute is not allowed.
- You can’t use a VALUE clause to initialize an item in WORKING-STORAGE. If you need to initialize something, you must do so by overriding the somInit method (corresponding to a default constructor in C++) to initialize the variable explicitly with a MOVE or a SET.

The items in WORKING-STORAGE are accessible only to the methods of the class being defined.

PROCEDURE DIVISION

The PROCEDURE DIVISION consists entirely of method definitions, one after another. Each method definition may have four divisions of its own, including a PROCEDURE DIVISION, which performs the actions of the method. As a result, the entire class definition may contain multiple WORKING-STORAGE sections, multiple PROCEDURE divisions, and so forth. It takes getting used to, especially when you’re trying to find your way around in the editor.

Termination

The last statement in a class definition is a terminator:

```
END CLASS OBJECT1.
```

METHOD DEFINITIONS

Method definitions reside within the PROCEDURE DIVISION of a class definition. Each one has the usual four divisions, but with special features or restrictions.

IDENTIFICATION DIVISION

Instead of a PROGRAM-ID, code a METHOD-ID:

```
METHOD-ID. displayOnSysout.
```

© 2004. Reproduction prohibited. Please inform Xephon of any infringement.

If you are overriding a method defined in some parent class, add an OVERRIDE clause:

```
    METHOD-ID. somInit OVERRIDE.
```

ENVIRONMENT DIVISION

The only section allowed here is an INPUT–OUTPUT section for allocating files. You don't need a REPOSITORY paragraph – all the classes used should already be declared at the class level.

DATA DIVISION

In the FILE SECTION, any files described must be EXTERNAL.

The LOCAL-STORAGE SECTION is the same as in an ordinary program, except that the GLOBAL attribute has no effect, since you can't have nested programs in a method definition. This section has no counterpart in VS COBOL II. It declares variables, which exist only while the method is executing. They are created when you enter the method, and are destroyed when you exit. They are similar to 'auto' variables in C or C++.

The WORKING-STORAGE SECTION works in the usual way, except that GLOBAL has no effect. Items in WORKING-STORAGE occur once per class, not once per instance. Their values persist from one invocation to the next. They are accessible only to the methods that declare them, unless they are EXTERNAL. The LINKAGE SECTION also works in the usual way, except that GLOBAL has no effect.

PROCEDURE DIVISION

The procedural code looks normal, except that you cannot use any of the following constructs:

- Segmentation
- ENTRY
- GO TO

- ALTER
- USE FOR DEBUGGING
- EXIT PROGRAM (use EXIT METHOD or GOBACK instead).

Termination

Every method definition must end with a terminator:

```
    END METHOD displayOnSysout.
```

CLIENT PROGRAMS

Either an ordinary program or a method definition may use objects. The syntax is almost the same in either case.

IDENTIFICATION DIVISION

An ordinary program has a PROGRAM-ID, and a method definition has a METHOD-ID, as usual.

ENVIRONMENT DIVISION

Any program that uses objects must have a REPOSITORY paragraph in the CONFIGURATION SECTION declaring all the classes used:

```
    REPOSITORY. CLASS OBJECT1 IS 'OBJECT1'.
```

For a method, the REPOSITORY paragraph belongs in the class definition rather than in the method definition itself.

DATA DIVISION

You can't declare an object directly in the DATA DIVISION. Instead, you declare an object reference to it:

```
 Ø1  GENERIC-OBJ           USAGE IS OBJECT REFERENCE VALUE NULL.
*
 Ø1  OBJECT1-OBJ           USAGE IS OBJECT REFERENCE OBJECT1 VALUE NULL.
```

In the example above, GENERIC-OBJ is a universal object reference. It may refer to any object. OBJECT1-OBJ, however, is restricted to objects of the class OBJECT1, or of classes derived from OBJECT1. In each case the VALUE clause initializes the object reference so that it does not point to anything. Object references may occur in the WORKING-STORAGE SECTION, the LOCAL-STORAGE SECTION, or the LINKAGE SECTION. Presumably they may appear in the FILE SECTION as well, but not usefully. Like a pointer, an object reference is meaningful only during a particular execution of the program.

PROCEDURE DIVISION

Invoking methods

The only way to use an object is to invoke one of its methods:

```
     INVOKE OBJECT1-OBJ 'displayOnSysout' USING INDENTATION-DEPTH.
```

I won't try to list all the possible variations. Suffice it to say that an INVOKE works much like a CALL and offers pretty much the same options, except that you have to provide an object reference to identify the object whose method you are invoking.

Depending on compile-time options, the compiler may optionally consult the Interface Repository (IR) to verify that you are passing the right kinds of parameter to the method.

Attaching references to their objects

By using the SET verb, you can make one object reference refer to the same object as another object reference:

```
     SET GENERIC-OBJ TO OBJECT1-OBJ.
```

You can also detach a reference from its object:

```
     SET OBJECT1-OBJECT TO NULL.
```

Within a method definition, you can attach a reference to the object whose method you are defining:

```
     SET OBJECT1-OBJ TO SELF.
```

Creating and destroying objects

Declaring an object reference does not create a corresponding object. You must create every object explicitly by invoking the somNew method:

```
     INVOKE OBJECT1 'somNew' RETURNING OBJECT1-OBJ.
```

At first glance, this syntax appears different from the kinds of INVOKEs described above. You can't specify an OBJECT1, which doesn't exist yet, so you specify the class instead.

Actually this syntax is less different than it appears. OBJECT1 is an object. It is an instance of the class SOMClass (which is an instance of itself). The method somNew is not a method of OBJECT1 – it is a method of SOMClass. The real difference is that we don't need an object reference to the OBJECT1 class. The REPOSITORY paragraph serves the same function instead. When you have finished with an object, you should destroy it:

```
     INVOKE OBJECT1-OBJ 'somFree'.
```

This time you specify the object, not the class.

Special methods: somInit and somUninit

Whenever somNew creates an object, it invokes the somInit method, which is intended to initialize the object. It first invokes somInit for all the base classes, from the top down, and eventually invokes the somInit for the last-derived class, if one is defined.

Unfortunately somInit accepts no parameters. Left to its own devices, it will initialize each instance of a given class in exactly the same way. In practice you generally want to initialize different instances differently. It's up to you to come up with a way to do so.

When somFree destroys an object, it invokes the somUninit method for the last-derived class, if one is defined. Then it invokes the somUninit methods for all the base classes, from the bottom up – in the reverse order in which somNew invoked the corresponding somInit methods.

You are expected to override somInit and somUninit as needed. In particular, somUninit provides a way to automatically release any resources that may be associated with an object – such as files, dynamically allocated memory, or other objects.

Akila Balaji
Programmer/Analyst (India) © Xephon 2004

A peek at WLM's decision making – part 2

This month we continue the code for collecting SMF type 99 records.

```
S2AS_CS_FMCT=c2d(SUBSTR(x.i,ofo+16,4))
                                /* No. of CS frames the ASIDs owns   */
S2AS_ES_FMCT=c2d(SUBSTR(x.i,ofo+2Ø,4))
                                /* No. of ESP frames the ASIDs owns  */
S2AS_PPS_TAR=c2d(SUBSTR(x.i,ofo+24,4))  /* ASID protective process  */
                       /* storage target. This is the only target   */
                               /* non-monitor ASIDs can have.       */
Select
when zk = Ø then
           col1=left(date('n',SMF99DTE,'j'),12)||left(SMF99TME,12)
otherwise  col1=left(' ',24,' ')
End
exrec.b = col1 left(S2PCNM,9)    left(S2AS_ANAM,9),
          right(S2AS_ASID,4)     right(S2AS_CS_FMCT,5),
          right(S2AS_ES_FMCT,4)  right(S2AS_PPS_TAR,4)
b = b + 1
end
Return
RQS:
parse arg off len num
if (len <> Ø) Then do
do zw = Ø to num -1
 ofw  = (off + (zw*len))- 3
    /*-------------------------------------------------------------------*/
   /*  SMF99 subtype 2: Remote Queue Server Data.                        */
   /*  Contains information on the state of a batch work queue on a     */
   /*  specific system. There is one RQDATA section for each system a*/
   /*  batch work queue is registered on including the local system. */
    /*-------------------------------------------------------------------*/
```

```
S2RQ_SYS_NAME = SUBSTR(x.i,ofw,8))      /* Name of system this RQDATA */
                                          /* section represents        */
S2RQ_FLAGS    = ,                         /* System flags              */
          x2b(c2d(SUBSTR(x.i,ofw+8,4)))
 qz1 =  substr(S2RQ_FLAGS,1,1)
 qz2 =  substr(S2RQ_FLAGS,2,1)
 qz3 =  substr(S2RQ_FLAGS,3,1)
 qz4 =  substr(S2RQ_FLAGS,4,1)
 qz5 =  substr(S2RQ_FLAGS,5,1)
   Select
    when qz1 =1 then SServer ="Server just started"
                                  /* This system started at least     */
                                  /* one server for this work         */
                                  /* queue in the policy interval     */
                                  /* that this data represents        */
    when qz2 =1 then SServer ="Server cannot start"
                                  /* This system cannot start any     */
                                  /* servers for this work queue      */
                                  /* due to some constraint           */
    when qz3 =1 then SServer ="Deferred starting server"
                                 /* This system wanted to add servers*/
                                 /* for this work queue on the       */
                                 /* just-completed policy interval,  */
                                 /* but deferred since another       */
                                 /* system appears to be a           */
                                 /* better candidate.                */
    when qz4 =1 then SServer ="Work queue is managed"
                                 /* Work queue is managed on         */
                                          /* this system             */
    when qz5 =1 then SServer ="Assess data valid"
                                /* originator sent valid assess data*/
    otherwise        SServer =
   End
S2RQ_ACTIVE_SERVERS = ,                 /* 10-second average No. of   */
        c2d(SUBSTR(x.i,ofw+12,4))      /* active servers scaled by 16 */
S2RQ_TOTAL_SERVERS  = ,            /* 10-second average total servers.*/
        c2d(SUBSTR(x.i,ofw+16,4)) /*Includes active and idle servers.*/
S2RQ_AVG_TOTAL_REQ  = ,                /* Average total requests for  */
        c2d(SUBSTR(x.i,ofw+20,4))   /* the queue eligible to run on  */
                                       /* the system respresented by  */
                                  /* this RQDATA entry. Corresponds   */
                                  /* to the last point plotted on the */
                                  /* queue delay plot. Scaled by * 16.*/
S2RQ_DEFERRAL_INFORMATION = ,             /* This info valid only if  */
            SUBSTR(x.i,ofw+24,40)    /* S2rqdat-a_assess_data_valid  */
                                          /* is on                    */
S2RQ_#_SERVERS  = ,                  /* No. of servers required for  */
        c2d(SUBSTR(x.i,ofw+24,4))         /*      receiver value      */
S2RQ_PI_DELTA   = ,                  /* PI delta for donor period of */
        c2d(SUBSTR(x.i,ofw+32,4))   /* highest importance if servers */
```

```
                                                /* are started              */
S2RQ_HIGHEST_IMP = ,                       /* Highest importance of donor   */
        c2d(SUBSTR(x.i,ofw+36,2))     /* periods negatively affected */
                                                /* if servers are started   */
S2RQ_WAITING_FOR_SYSNAME  = ,          /* System name sender is deferring*/
            SUBSTR(x.i,ofw+40,8)   /* to. Blank if deferring only to */
                                 /* collect data from other systems  */
S2RQ_DONOR_CLASS  = ,              /* Class name for donor period most */
            SUBSTR(x.i,ofw+48,8)     /* impacted by starting servers */
S2RQ_PER#  = ,                     /* Period no. with in class of donor*/
        c2d(SUBSTR(x.i,ofw+56,4))
S2RQ_DONOR_RGROUP  = ,                      /* Resource group name for */
            SUBSTR(x.i,ofw+60,8)       /* donor period most impacted */
                                                /* by starting servers      */
S2RQ_PA_SKIP =c2d(SUBSTR(x.i,ofw+68,2))  /* Policy adj.skip clock     */
S2RQ_Q_SKIP = c2d(SUBSTR(x.i,ofw+70,1))
                                             /* Defer processing skip clock */
S2RQ_Q_SKIP_REASON  = ,              /* Reason defer processing skip  */
        c2d(SUBSTR(x.i,ofw+71,1))        /* clock was set            */
S2RQ_AVG_QUEUED_REQUESTS = ,       /* Average no. of queued requests */
        c2d(SUBSTR(x.i,ofw+72,4))  /*over a policy interval scaled by*/
                                                /* 16                       */
S2RQ_AVG_INELIGIBLE_REQUESTS = ,         /*Average no. of ineligible */
        c2d(SUBSTR(x.i,ofw+76,4))        /* queued requests over a    */
                                 /* policy interval scaled by * 16   */
end
Return
SMF: procedure
/* REXX - convert a SMF time to hh:mm:ss:hd format   */
arg time
    time1    = time % 100
    hh       = time1 % 3600
    hh       = RIGHT("0"||hh,2)
    mm       = (time1 % 60) - (hh * 60)
    mm       = RIGHT("0"||mm,2)
    ss       = time1 - (hh * 3600) - (mm * 60)
    ss       = RIGHT("0"||ss,2)
    fr       = time //  1000
    fr       = RIGHT("0"||fr,2)
    rtime  = hh||":"||mm||":"||ss||":"||fr
    return rtime
Tctext:
   Parse Arg TCOD
   Select
     when TCOD=1      Then text ="STA_RECOVERY_RETRY"
     when TCOD=2      Then text ="STA_RECOVERY_PERC"
     when TCOD=3      Then text ="STA_RECOVERY_REDRIVE_SE"
     when TCOD=10     Then text ="RA_AUXP_DEC_MPL"
     when TCOD=20     Then text ="RA_AUXP_NO_ACTION"
     when TCOD=30     Then text ="RA_MP_NO_ACTION"
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
when TCOD=40      Then text ="RA_OU_DEC_MPL"
when TCOD=50      Then text ="RA_OU_NO_ACTION"
when TCOD=60      Then text ="RA_SWAP_FOR_MPL"
when TCOD=70      Then text ="RA_UP_DECREASE_MPL"
when TCOD=80      Then text ="RA_UP_NEW_CAND"
when TCOD=90      Then text ="RA_UP_NO_ACTION"
when TCOD=100     Then text ="RA_UU_INC_MPL"
when TCOD=105     Then text ="RA_UU_ADD_SRV_GR"
when TCOD=106     Then text ="RA_UU_ADD_SRV_RR"
when TCOD=107     Then text ="ADD_SRV_ASSESS"
when TCOD=108     Then text ="ADD_SRV_ASSESS2"
when TCOD=110     Then text ="RA_UU_NO_ACTION"
when TCOD=120     Then text ="RA_UP_SWAP_OUT"
when TCOD=130     Then text ="SWAP_DETECTED_WAIT"
when TCOD=140     Then text ="SWAP_EXCHANGE"
when TCOD=150     Then text ="SWAP_LONG_WAIT"
when TCOD=160     Then text ="SWAP_UNILATERAL"
when TCOD=170     Then text ="RA_MON_PAG_COST_HI"
when TCOD=180     Then text ="RA_MON_POLICY_DIR"
when TCOD=190     Then text ="RA_UNMON_ALL_P_OK"
when TCOD=195     Then text ="RA_UNMON_NO_CAPT"
when TCOD=200     Then text ="TX_END_UNMON"
when TCOD=210     Then text ="NS_STOR_TAR_ACTION"
when TCOD=220     Then text ="PA_ADD_TRANS_DISP"
when TCOD=222     Then text ="PA_AS_BET_DISPS"
when TCOD=224     Then text ="PA_AS_FROM_DISP"
when TCOD=226     Then text ="PA_AS2_TRX_DISP"
when TCOD=227     Then text ="PA_AS2_NONTRX_DISP"
when TCOD=230     Then text ="PA_DELETE_DISP"
when TCOD=232     Then text ="PA_ADDDISP_MT_EN_Q"
when TCOD=233     Then text ="PA_ADD_DISP_MT_EN"
when TCOD=235     Then text ="PA_ADDDISP_ST_EN_Q"
when TCOD=236     Then text ="PA_ADD_DISP_ST_EN"
when TCOD=240     Then text ="PA_GREC_CAND"
when TCOD=245     Then text ="PA_NA_NO_MPL"
when TCOD=246     Then text ="PA_DRV_PRO_SKIPPED"
when TCOD=250     Then text ="PA_NA_NO_PROBLEM"
when TCOD=251     Then text ="PA_ADDDISP_SCSP"
when TCOD=252     Then text ="PA_ADDDISP_SCSP_Q"
when TCOD=253     Then text ="PA_ADDDISP_SCMP"
when TCOD=254     Then text ="PA_ADDDISP_SCMP_Q"
when TCOD=255     Then text ="PA_ADDDISP_MCMP"
when TCOD=256     Then text ="PA_ADDDISP_MCMP_Q"
when TCOD=260     Then text ="PA_NA_UNKNOW_DELAY"
when TCOD=265     Then text ="PA_NA_SYSPLEX_ONLY"
when TCOD=270     Then text ="PA_REC_CAND"
when TCOD=280     Then text ="PA_RREC_CAND"
when TCOD=290     Then text ="PA_USE_DISC_CENT"
when TCOD=300     Then text ="PA_USE_DISC_EXP"
when TCOD=305     Then text ="PA_STOR_DONOR"
when TCOD=306    Then text ="SH_STOR_DONOR"
when TCOD=307    Then text ="SV_STOR_DONOR"
when TCOD=308    Then text ="PA_DONOR_PERIOD"
when TCOD=310    Then text ="WLM_Q_REQ"
when TCOD=311    Then text ="WLM_Q_MISC"
when TCOD=315    Then text ="PA_CPC_MOVE_DOWN"
when TCOD=320    Then text ="PA_CAL_PI_NO_FOREIGN_FA"
when TCOD=500    Then text ="HSK_FROM_SPC_DP"
when TCOD=510    Then text ="HSK_TO_SPC_DP"
when TCOD=520    Then text ="HSK_XFROM_SPC_DP"
when TCOD=525    Then text ="HSK_UNBUNCH_PRTY"
when TCOD=526    Then text ="PA_PCC_NO_OCC_PRTY"
when TCOD=527    Then text ="PA_PCC_NO_UNO_PRTY"
when TCOD=528    Then text ="PA_PCC_BLKR_MOVED"
when TCOD=529    Then text ="PA_PCC_BLKR_VIOLTN"
when TCOD=530    Then text ="PA_PMDO_DON"
when TCOD=531    Then text ="PA_PCC_DON_VIOLTN"
when TCOD=532    Then text ="PA_PCC_BLKR_IS_DON"
when TCOD=533    Then text ="PA_PCC_BLKR_IS_REC"
when TCOD=534    Then text ="PA_PCC_BLKR_NETVAL"
when TCOD=540    Then text ="PA_PMDU_DON"
when TCOD=550    Then text ="PA_PMD_DON_NETVAL"
when TCOD=560    Then text ="PA_PMD_GDON_NETVAL"
when TCOD=565    Then text ="PA_PMD_GREC_NETVAL"
when TCOD=570    Then text ="PA_PMD_RDON_NETVAL"
when TCOD=573    Then text ="PA_PMD_REC_NETVAL"
when TCOD=576    Then text ="PA_PMD_RREC_NETVAL"
when TCOD=580    Then text ="PA_PMD_SEC_DON"
when TCOD=590    Then text ="PA_PMU_DON_NETVAL"
when TCOD=595    Then text ="PA_PMU_DON_SEC_REC"
when TCOD=600    Then text ="PA_PMU_GDON_NETVAL"
when TCOD=605    Then text ="PA_PMU_GREC_NETVAL"
when TCOD=610    Then text ="PA_PMU_RDON_NETVAL"
when TCOD=613    Then text ="PA_PMU_REC_NETVAL"
when TCOD=616    Then text ="PA_PMU_RREC_NETVAL"
when TCOD=620    Then text ="PA_PMUO_REC"
when TCOD=630    Then text ="PA_PMUUA_REC"
when TCOD=635    Then text ="PA_PMUUB_REC"
when TCOD=640    Then text ="PA_PMU_SEC_REC"
when TCOD=650    Then text ="PA_PMU_TO_SPC_DP"
when TCOD=651    Then text ="PA_PMU_SPC_NXT_DP"
when TCOD=655    Then text ="PA_PMU_SPC_UP_FAIL"
when TCOD=660    Then text ="PA_PRO_DECP_DON"
when TCOD=665    Then text ="PA_PRO_DECP_MPL"
when TCOD=670    Then text ="PA_PRO_DECP_SEC"
when TCOD=675    Then text ="PA_PRO_DECP_BLKR"
when TCOD=690    Then text ="PA_PRO_DON_DEPEN"
when TCOD=720    Then text ="PA_PRO_GREC_NETVAL"
when TCOD=730    Then text ="PA_PRO_GREC_RECVAL"
when TCOD=740    Then text ="PA_PRO_INCP_DON"
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
     when TCOD=750    Then text ="PA_PRO_INCP_REC"
     when TCOD=760    Then text ="PA_PRO_INCP_SEC"
     when TCOD=770    Then text ="PA_PRO_INCP_BLKR"
     when TCOD=780    Then text ="PA_PRO_INCP_SC"
     when TCOD=850    Then text ="PA_PRO_NA_NO_DONOR"
     when TCOD=870    Then text ="PA_PRO_NA_SPC_DP"
     when TCOD=880    Then text ="PA_PRO_RDON_CAND"
     when TCOD=890    Then text ="PA_PRO_REC_DEPEN"
     when TCOD=900    Then text ="PA_PRO_REC_NETVAL"
     when TCOD=910    Then text ="PA_PRO_REC_RECVAL"
     when TCOD=920    Then text ="PA_PRO_RREC_NETVAL"
     when TCOD=930    Then text ="PA_PRO_RREC_RECVAL"
     when TCOD=933    Then text ="PA_PRO_SERVED_GDON"
     when TCOD=936    Then text ="PA_PRO_SERVED_GREC"
     when TCOD=938    Then text ="PA_PRO_TO_SPC_DP"
     when TCOD=939    Then text ="PA_PRO_SPC_UP_FAIL"
     when TCOD=940    Then text ="PA_PRO_UNC_DON"
     when TCOD=950    Then text ="PA_PRO_UNC_REC"
     when TCOD=960    Then text ="PA_PRO_UNC_SEC_DON"
     when TCOD=970    Then text ="PA_PRO_UNC_SEC_REC"
     when TCOD=975    Then text ="PA_SDO_DONFAIL_SPC"
     when TCOD=976    Then text ="PA_SDO_ADD_DGRP"
     when TCOD=978    Then text ="PA_SDO_CLR_FLGS"
     when TCOD=980    Then text ="PA_TA_EA_MOV_UBA"
     when TCOD=981    Then text ="PA_TA_EA_MOV_BDEV"
     when TCOD=982    Then text ="PA_TA_EA_NA_TIME"
     when TCOD=983    Then text ="PA_TA_EA_NA_DONPIO"
     when TCOD=984    Then text ="PA_TA_EA_NA_IOSQL"
     when TCOD=987    Then text ="PA_TA_EA_DON_L1MIN"
     when TCOD=988    Then text ="PA_TA_EA_REC_L1MIN"
     when TCOD=989    Then text ="PA_TA_EA_NA_CUQDT"
     when TCOD=990    Then text ="PA_TA_GA_MOV_UBA"
     when TCOD=991    Then text ="PA_TA_GA_MOV_BDEV"
     when TCOD=992    Then text ="PA_TA_GA_INV_RDEV"
     when TCOD=993    Then text ="PA_TA_GA_NA_DONPIO"
     when TCOD=994    Then text ="PA_TA_GA_NA_IOSQL"
     when TCOD=995    Then text ="PA_TA_GA_DON_L1MIN"
     when TCOD=996    Then text ="PA_TA_GA_REC_L1MIN"
     when TCOD=997    Then text ="PA_TA_RRPATOD"
     when TCOD=998    Then text ="PA_TA_GA_DONGTREC"
     when TCOD=999    Then text ="PA_TA_GA_NA_CUQDT"
     when TCOD=1000   Then text ="PA_TA_EA_PASS_NO"
     when TCOD=1900   Then text ="PA_0C9_suppressed"
     when TCOD=2010   Then text ="PA_DEC_PSI_TAR"
     when TCOD=2011   Then text ="PA_DEC_PSI_TAR_GP"
     when TCOD=2020   Then text ="PA_INC_PSI_TAR"
     when TCOD=2021   Then text ="PA_INC_PSI_TAR_GR"
     when TCOD=2030   Then text ="PA_PSI_NA_NET_VAL"
     when TCOD=2031   Then text ="PA_PSI_GREC_NETVAL"
     when TCOD=2040   Then text ="PA_PSI_NA_REC_VAL"
```

```
when TCOD=2041   Then text ="PA_PSI_RREC_RECVAL"
when TCOD=2050   Then text ="PA_PSI_TAR_UNAB"
when TCOD=2060   Then text ="PA_REM_PSI_TAR"
when TCOD=2061   Then text ="PA_REM_PSI_TAR_GP"
when TCOD=2070   Then text ="PLOT_X_REM_PSI_TAR"
when TCOD=2071   Then text ="PLOT_X_REM_PSI_GP"
when TCOD=2075   Then text ="PLOT_X_REM_RCS_TAR"
when TCOD=2080   Then text ="SH_DEC_PSI_TAR"
when TCOD=2081   Then text ="SH_DEC_PSI_TAR_GP"
when TCOD=2090   Then text ="SH_REM_PSI_TAR"
when TCOD=2091   Then text ="SH_REM_PSI_TAR_GP"
when TCOD=2100   Then text ="TDH_AS_DEC_PSI_TAR"
when TCOD=2101   Then text ="TDH_AS_DEC_PSI_GP"
when TCOD=2110   Then text ="TDH_AS_REM_PSI_TAR"
when TCOD=2111   Then text ="TDH_AS_REM_PSI_GP"
when TCOD=2120   Then text ="TDH_ME_DEC_PSI_TAR"
when TCOD=2121   Then text ="TDH_ME_DEC_PSI_GP"
when TCOD=2130   Then text ="TDH_ME_REM_PSI_TAR"
when TCOD=2131   Then text ="TDH_ME_REM_PSI_GP"
when TCOD=2140   Then text ="TDH_UA_DEC_PSI_TAR"
when TCOD=2141   Then text ="TDH_UA_DEC_PSI_GP"
when TCOD=2150   Then text ="TDH_UA_REM_PSI_TAR"
when TCOD=2151   Then text ="TDH_UA_REM_PSI_GP"
when TCOD=2160   Then text ="RV_HSK_INC_PSI_TAR"
when TCOD=2161   Then text ="RV_HSK_INC_PSI_GR"
when TCOD=2170   Then text ="WSM_DEC_PSI_TAR"
when TCOD=2171   Then text ="WSM_DEC_PSI_TAR_GP"
when TCOD=2180   Then text ="WSM_REM_PSI_TAR"
when TCOD=2181   Then text ="WSM_REM_PSI_TAR_GP"
when TCOD=2510   Then text ="PA_DEC_PRT"
when TCOD=2520   Then text ="PA_INC_PRT"
when TCOD=2530   Then text ="PA_PRT_NA_NET_VAL"
when TCOD=2540   Then text ="PA_PRT_NA_REC_VAL"
when TCOD=2550   Then text ="PA_PRT_NA_SRVR_UD"
when TCOD=2555   Then text ="PA_PRT_NA_ENCLAVE"
when TCOD=2560   Then text ="PA_PRT_NO_WSS"
when TCOD=2570   Then text ="PA_PRT_TAR_UNAB"
when TCOD=2580   Then text ="PA_REM_PRT"
when TCOD=2590   Then text ="RV_HSK_INC_PRT"
when TCOD=2600   Then text ="SH_DEC_PRT"
when TCOD=2610   Then text ="SH_REM_PRT"
when TCOD=2620   Then text ="TDH_DEC_PRT"
when TCOD=2630   Then text ="TDH_REM_PRT"
when TCOD=2640   Then text ="WSM_DEC_PRT"
when TCOD=2650   Then text ="WSM_REM_PRT"
when TCOD=3010   Then text ="PA_CSI_NA_NET_VAL"
when TCOD=3020   Then text ="PA_CSI_NA_REC_VAL"
when TCOD=3030   Then text ="PA_CSI_TAR_UNAB"
when TCOD=3040   Then text ="PA_INC_CSI_TAR"
when TCOD=3050   Then text ="TDH_DEC_CSI_TAR"
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
when TCOD=3060   Then text ="TDH_REM_CSI_TAR"
when TCOD=3070   Then text ="PA_INC_XMEM_TAR"
when TCOD=3080   Then text ="PA_XMEM_NA_NET_VAL"
when TCOD=3090   Then text ="PA_XMEM_NA_REC_VAL"
when TCOD=3095   Then text ="PA_XMEM_NA_SRT"
when TCOD=3100   Then text ="PA_XMEM_TAR_UNAB"
when TCOD=3110   Then text ="TDH_DEC_SSI_TAR"
when TCOD=3120   Then text ="PA_SHR_TAR_UNAB"
when TCOD=3130   Then text ="PA_SHR_NA_REC_VAL"
when TCOD=3140   Then text ="PA_SHR_NA_NET_VAL"
when TCOD=3150   Then text ="PA_INC_SHR_TAR"
when TCOD=3160   Then text ="PA_DEC_SHR_DEL"
when TCOD=3510   Then text ="B16M_SHORT_DEC_MPL"
when TCOD=3520   Then text ="PA_DEC_MPL"
when TCOD=3521   Then text ="PA_DEC_MPL_GP"
when TCOD=3530   Then text ="PA_INC_MPL"
when TCOD=3531   Then text ="PA_INC_MPL_TS"
when TCOD=3540   Then text ="PA_INC_MPL_GR"
when TCOD=3541   Then text ="PA_INC_MPL_RR"
when TCOD=3550   Then text ="PA_MPL_NA_NET_VAL"
when TCOD=3551   Then text ="PA_MPL_NETVAL_RR"
when TCOD=3552   Then text ="PA_MPL_NETVAL_GR"
when TCOD=3560   Then text ="PA_MPL_NA_REC_VAL"
when TCOD=3561   Then text ="PA_MPL_RECVAL_RR"
when TCOD=3562   Then text ="PA_MPL_RECVAL_GR"
when TCOD=3580   Then text ="PA_MPL_NA_SHORTAGE"
when TCOD=3590   Then text ="PA_SWAP_FOR_MPL"
when TCOD=3600   Then text ="TDH_DEC_MPL"
when TCOD=3601   Then text ="TDH_DEC_MPL_FOR_GR"
when TCOD=3602   Then text ="TDH_DEC_MPL_FOR_RR"
when TCOD=3603   Then text ="TDH_DEC_QMPL_GR"
when TCOD=3604   Then text ="TDH_DEC_QMPL_RR"
when TCOD=3605   Then text ="TDH_INC_QMPL_GR"
when TCOD=3606   Then text ="TDH_INC_QMPL_RR"
when TCOD=3607   Then text ="TDH_MOD_SERVINST"
when TCOD=3608   Then text ="TDH_STRT_MIN_SP"
when TCOD=3610   Then text ="RV_HSK_INC_MPL"
when TCOD=3613   Then text ="TDH_DEC_QMOV_GR"
when TCOD=3614   Then text ="TDH_DEC_QMOV_RR"
when TCOD=3615   Then text ="TDH_DEC_QSWP_GR"
when TCOD=3616   Then text ="TDH_DEC_QSWP_RR"
when TCOD=3617   Then text ="TDH_DEC_QSVT_GR"
when TCOD=3618   Then text ="TDH_DEC_QSVT_RR"
when TCOD=3620   Then text ="TDH_NA_INI_BAL"
when TCOD=3621   Then text ="TDH_MPL_VCAL_ERR"
when TCOD=3622   Then text ="TDH_MPL_SVLCAL_ERR"
when TCOD=4010   Then text ="ESPOL_NSW_LRU"
when TCOD=4020   Then text ="ESPOL_NSW_SP_AVAIL"
when TCOD=4050   Then text ="ESPOL_SWP_LRU"
when TCOD=4060   Then text ="ESPOL_SWP_SP_AVAIL"
```

© 2004. Reproduction prohibited. Please inform Xephon of any infringement.

```
when TCOD=4090   Then text ="HSK_ROLL_EXP_SPA"
when TCOD=4200   Then text ="STL_CR_AS_BLW_TRGT"
when TCOD=4201   Then text ="STL_CR_AS_BLW_TRG2"
when TCOD=4202   Then text ="STL_CR_AS_BLW_TRG3"
when TCOD=4203   Then text ="STL_CR_REQ_BLW_PPS"
when TCOD=4510   Then text ="ALL_OK_REM_ISI_TAR"
when TCOD=4511   Then text ="ALL_OK_REM_ISI_GP"
when TCOD=4520   Then text ="HSK_SL_DEC_ISI_TAR"
when TCOD=4521   Then text ="HSK_SL_DEC_ISI_GP"
when TCOD=4530   Then text ="HSK_SL_REM_ISI_TAR"
when TCOD=4531   Then text ="HSK_SL_REM_ISI_GP"
when TCOD=4540   Then text ="OK1_INC_ISI_TAR"
when TCOD=4541   Then text ="OK1_INC_ISI_TAR_GR"
when TCOD=4550   Then text ="PA_DEC_ISI_TAR"
when TCOD=4551   Then text ="PA_DEC_ISI_TAR_GP"
when TCOD=4560   Then text ="PA_INC_ISI_TAR"
when TCOD=4561   Then text ="PA_INC_ISI_TAR_GR"
when TCOD=4570   Then text ="PA_ISI_NA_NET_VAL"
when TCOD=4571   Then text ="PA_ISI_GREC_NETVAL"
when TCOD=4580   Then text ="PA_ISI_NA_REC_VAL"
when TCOD=4581   Then text ="PA_ISI_GREC_RECVAL"
when TCOD=4590   Then text ="PA_REM_ISI_TAR"
when TCOD=4591   Then text ="PA_REM_ISI_TAR_GP"
when TCOD=4592   Then text ="PA_DEC_ISI_GDON"
when TCOD=4600   Then text ="PLOT_X_REM_ISI_TAR"
when TCOD=4601   Then text ="PLOT_X_REM_ISI_GP"
when TCOD=4610   Then text ="ROLL_EXP_REM_ISI"
when TCOD=4611   Then text ="ROLL_EXP_REM_ISIGP"
when TCOD=4620   Then text ="RV_HSK_INC_ISI_TAR"
when TCOD=4621   Then text ="RV_HSK_INC_ISI_GR"
when TCOD=4630   Then text ="SH_DEC_ISI_TAR"
when TCOD=4631   Then text ="SH_DEC_ISI_TAR_GP"
when TCOD=4640   Then text ="SH_REM_ISI_TAR"
when TCOD=4641   Then text ="SH_REM_ISI_TAR_GP"
when TCOD=4650   Then text ="TDH_ME_DEC_ISI_TAR"
when TCOD=4653   Then text ="TDH_ME_DEC_ISI_GP"
when TCOD=4660   Then text ="TDH_ME_REM_ISI_TAR"
when TCOD=4661   Then text ="TDH_ME_REM_ISI_GP"
when TCOD=4670   Then text ="TDH_UA_DEC_ISI_TAR"
when TCOD=4671   Then text ="TDH_UA_DEC_ISI_GP"
when TCOD=4680   Then text ="TDH_UA_REM_ISI_TAR"
when TCOD=4681   Then text ="TDH_UA_REM_ISI_GP"
when TCOD=4690   Then text ="WSM_DEC_ISI_TAR"
when TCOD=4691   Then text ="WSM_DEC_ISI_TAR_GP"
when TCOD=4700   Then text ="WSM_INC_ISI_TAR"
when TCOD=4701   Then text ="WSM_INC_ISI_TAR_GR"
when TCOD=4710   Then text ="WSM_REM_ISI_TAR"
when TCOD=4711   Then text ="WSM_REM_ISI_TAR_GP"
when TCOD=4720   Then text ="Hsk_cr_inc_ici_tar"
when TCOD=4721   Then text ="Hsk_cr_dec_ici_tar"
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
when TCOD=4722   Then text ="Hsk_cr_inc_ipi_tar"
when TCOD=4723   Then text ="Hsk_cr_dec_ipi_tar"
when TCOD=4724   Then text ="Hsk_cr_inc_ici_gp"
when TCOD=4725   Then text ="Hsk_cr_dec_ici_gp"
when TCOD=4726   Then text ="Hsk_cr_inc_ipi_gp"
when TCOD=4727   Then text ="Hsk_cr_dec_ipi_gp"
when TCOD=4728   Then text ="Hsk_cr_inc_rcs_tar"
when TCOD=4729   Then text ="Hsk_cr_inc_rps_tar"
when TCOD=473Ø   Then text ="Hsk_cr_rem_ipi_tar"
when TCOD=474Ø   Then text ="Chp_cr_inc_ici_tar"
when TCOD=4741   Then text ="Chp_cr_inc_rcs_tar"
when TCOD=4742   Then text ="Chp_cr_inc_rps_tar"
when TCOD=4743   Then text ="Chp_cr_inc_ipi_tar"
when TCOD=4744   Then text ="Chp_cr_inc_ipi_gp"
when TCOD=4745   Then text ="Chp_cr_rem_rcs_tar"
when TCOD=4746   Then text ="Chp_cr_rem_rps_tar"
when TCOD=4747   Then text ="inc_ipi_tar_blw_bw"
when TCOD=475Ø   Then text ="pa_cr_no_action"
when TCOD=4751   Then text ="paaup_cr_no_action"
when TCOD=4752   Then text ="palpd_cr_no_action"
when TCOD=476Ø   Then text ="pa_fst_outof_donor"
when TCOD=4761   Then text ="pa_fst_action"
when TCOD=4762   Then text ="pa_fst_begin"
when TCOD=4763   Then text ="pa_fst_end"
when TCOD=4764   Then text ="pa_fst_parms"
when TCOD=4765   Then text ="pa_fst_wsi_dnval_fd"
when TCOD=4766   Then text ="pa_fst_no_wsi_sdon"
when TCOD=4767   Then text ="pa_fst_wsi_no_bactn"
when TCOD=4768   Then text ="pa_fst_isi_dnval_fd"
when TCOD=4769   Then text ="pa_fst_no_isi_sdon"
when TCOD=477Ø   Then text ="pa_fst_isi_no_bactn"
when TCOD=4771   Then text ="pa_fst_no_bst_5as"
when TCOD=5Ø1Ø   Then text ="RUN_OK_REM_RPS_TAR"
when TCOD=5Ø2Ø   Then text ="PA_DEC_RPS_TAR"
when TCOD=5Ø3Ø   Then text ="PA_INC_RPS_TAR"
when TCOD=5Ø4Ø   Then text ="PA_REM_RPS_TAR"
when TCOD=5Ø5Ø   Then text ="PA_SET_RPS_TAR"
when TCOD=5Ø6Ø   Then text ="PC_REM_RPS_TAR"
when TCOD=5Ø7Ø   Then text ="SH_DEC_RPS_TAR"
when TCOD=5Ø8Ø   Then text ="SH_REM_RPS_TAR"
when TCOD=5Ø9Ø   Then text ="SH_SET_RPS_TAR"
when TCOD=51ØØ   Then text ="WSM_DEC_RPS_TAR"
when TCOD=511Ø   Then text ="WSM_INC_RPS_TAR"
when TCOD=512Ø   Then text ="WSM_REM_RPS_TAR"
when TCOD=513Ø   Then text ="WSM_SET_RPS_TAR"
when TCOD=55ØØ   Then text ="PA_DCM_INC_TAR"
when TCOD=55Ø1   Then text ="PA_DCM_NA_NOPROB"
when TCOD=55Ø2   Then text ="PA_DCM_NA_MAXVEL"
when TCOD=55Ø3   Then text ="PA_DCM_NA_MAXTARG"
when TCOD=55Ø4   Then text ="PA_DCM_NA_TAR_UNAB"
```

© 2004. Reproduction prohibited. Please inform Xephon of any infringement.

```
when TCOD=5505   Then text ="PA_DCM_NA_RECVAL"
when TCOD=5506   Then text ="PA_DCM_NA_SVC_INC"
when TCOD=5507   Then text ="PA_DCM_NA_IOSCDT"
when TCOD=5508   Then text ="PA_DCM_WLM_HUNG"
when TCOD=5510   Then text ="PA_DCM_GREC"
when TCOD=5515   Then text ="PA_DCM_NO_SCMT_ROW"
when TCOD=5516   Then text ="PA_DCM_DROP_SUBSYS"
when TCOD=5517   Then text ="PA_DCM_NEWSUB_ERR"
when TCOD=5518   Then text ="PA_DCM_GOALALG_ON"
when TCOD=5519   Then text ="PA_DCM_GOALALG_OFF"
when TCOD=5520   Then text ="HSK_DCM_BELOW_DEF"
when TCOD=5521   Then text ="HSK_DCM_NO_DELAY"
when TCOD=5522   Then text ="HSK_DCM_IOSCDT_ERR"
when TCOD=5530   Then text ="IOV_SUBSYS"
when TCOD=5531   Then text ="IOV_GREC_SYS"
when TCOD=5532   Then text ="IOV_GREC_LOC"
when TCOD=5533   Then text ="IOV_GREC_REM"
when TCOD=5534   Then text ="IOV_GREC_NETV_SYS"
when TCOD=5535   Then text ="IOV_GREC_NETV_LOC"
when TCOD=5536   Then text ="IOV_GREC_NETV_REM"
when TCOD=5537   Then text ="IOV_GDON_NETV_SYS"
when TCOD=5538   Then text ="IOV_GDON_NETV_LOC"
when TCOD=5539   Then text ="IOV_GDON_NETV_REM"
when TCOD=5540   Then text ="IOV_RREC_NETV"
when TCOD=5541   Then text ="IOV_RDON_NETV"
when TCOD=5542   Then text ="IOV_GDON_MIMP_SYS"
when TCOD=5543   Then text ="IOV_GDON_MIMP_LOC"
when TCOD=5544   Then text ="IOV_GDON_MIMP_REM"
when TCOD=5545   Then text ="IOV_NORECEIVER"
when TCOD=5546   Then text ="IOV_NODONOR"
when TCOD=5547   Then text ="IOV_RC"
when TCOD=5548   Then text ="IOV_IREC_SYS"
when TCOD=5549   Then text ="IOV_IREC_LOC"
when TCOD=5550   Then text ="IOV_IREC_REM"
when TCOD=5551   Then text ="IOV_IDON_SYS"
when TCOD=5552   Then text ="IOV_IDON_LOC"
when TCOD=5553   Then text ="IOV_IDON_REM"
when TCOD=5554   Then text ="IOV_DEC_TAR"
when TCOD=5555   Then text ="IOV_BAD_SUBSYS"
when TCOD=5556   Then text ="IOV_RDON_MIMP"
when TCOD=5557   Then text ="IOV_ADD_CHPID"
when TCOD=5558   Then text ="IOV_DELETE_CHPID"
when TCOD=5559   Then text ="IOV_AVAILABILITY"
when TCOD=6510   Then text ="HSK_SL_DEC_ICI_TAR"
when TCOD=6520   Then text ="HSK_SL_REM_ICI_TAR"
when TCOD=6530   Then text ="OK1_INC_ICI_TAR"
when TCOD=6540   Then text ="PA_DEC_ICI_TAR"
when TCOD=6550   Then text ="PA_INC_ICI_TAR"
when TCOD=6560   Then text ="PA_REM_ICI_TAR"
when TCOD=6570   Then text ="PLOT_X_REM_ICI_TAR"
```

```
    when TCOD=6580   Then text ="SH_DEC_ICI_TAR"
    when TCOD=6590   Then text ="SH_REM_ICI_TAR"
    when TCOD=6600   Then text ="SWAPIN_DEC_ICI_TAR"
    when TCOD=6610   Then text ="SWAPIN_REM_ICI_TAR"
    when TCOD=6620   Then text ="WSM_DEC_ICI_TAR"
    when TCOD=6630   Then text ="WSM_INC_ICI_TAR"
    when TCOD=6640   Then text ="WSM_REM_ICI_TAR"
    when TCOD=7010   Then text ="PA_DEC_RCS_TAR"
    when TCOD=7020   Then text ="PA_INC_RCS_TAR"
    when TCOD=7030   Then text ="PA_REM_RCS_TAR"
    when TCOD=7040   Then text ="PA_SET_RCS_TAR"
    when TCOD=7050   Then text ="PC_REM_RCS_TAR"
    when TCOD=7060   Then text ="RA_UP_SQUEEZE"
    when TCOD=7070   Then text ="RUN_OK_REM_RCS_TAR"
    when TCOD=7080   Then text ="SH_DEC_RCS_TAR"
    when TCOD=7090   Then text ="SH_REM_RCS_TAR"
    when TCOD=7100   Then text ="SH_SET_RCS_TAR"
    when TCOD=7110   Then text ="SWAPIN_REM_RCS_TAR"
    when TCOD=7120   Then text ="SWAPIN_SET_RCS_TAR"
    when TCOD=7130   Then text ="WSM_DEC_RCS_TAR"
    when TCOD=7140   Then text ="WSM_INC_RCS_TAR"
    when TCOD=7150   Then text ="WSM_REM_RCS_TAR"
    when TCOD=7160   Then text ="WSM_SET_RCS_TAR"
    when TCOD=7510   Then text ="OTL_USE_DISC_CENT"
    when TCOD=7520   Then text ="WSM_DEC_MPL"
    when TCOD=7521   Then text ="WSM_DEC_MPL_GP"
    when TCOD=7530   Then text ="WSM_END_A2B_CNT"
    when TCOD=7540   Then text ="WSM_END_A2B_PSTOR"
    when TCOD=7550   Then text ="WSM_END_OK1"
    when TCOD=7560   Then text ="WSM_END_OK1_BY_STL"
    when TCOD=7570   Then text ="WSM_END_OK1_RUN_OK"
    when TCOD=7580   Then text ="WSM_END_PHASE_CHG"
    when TCOD=7590   Then text ="WSM_END_SWAPIN"
    when TCOD=7600   Then text ="WSM_END_TRYLRU"
    when TCOD=7610   Then text ="WSM_NA_MP1"
    when TCOD=7620   Then text ="WSM_NA_NET_VAL"
    when TCOD=7630   Then text ="WSM_NA_NPCR_VAL"
    when TCOD=7640   Then text ="WSM_STRT_A2B_CNT"
    when TCOD=7650   Then text ="WSM_STRT_A2B_PSTOR"
    when TCOD=7660   Then text ="WSM_STRT_OK1"
    when TCOD=7670   Then text ="WSM_START_OTL_IN"
    when TCOD=7680   Then text ="WSM_STRT_PHASE_CHG"
    when TCOD=7690   Then text ="WSM_STRT_SWAPIN"
    when TCOD=7700   Then text ="WSM_STRT_TRYLRU"
    when TCOD=7710   Then text ="WSM_USE_DISC_CENT"
    when TCOD=7720   Then text ="WSM_USE_DISC_EXP"
    when TCOD=8010   Then text ="PA_CAP_DECS"
    when TCOD=8020   Then text ="PA_CAP_INCS"
    when TCOD=8025   Then text ="PA_CAP_GETMAIN"
    when TCOD=8030   Then text ="PA_DRGROUP_ADD"
```

```
when TCOD=8040   Then text ="PA_DRGROUP_DELETE"
when TCOD=8050   Then text ="PA_DRGROUP_MRK_DEL"
when TCOD=8055   Then text ="PA_DRGROUP_MRK_ALL"
when TCOD=8060   Then text ="PA_DRGROUP_EXCHG"
when TCOD=8070   Then text ="PA_DRGROUP_MAX_INC"
when TCOD=8075   Then text ="PA_DRGROUP_MAX_NI"
when TCOD=8080   Then text ="PA_DRGROUP_MAX_DEC"
when TCOD=8090   Then text ="PA_DRGROUP_ADD_INT"
when TCOD=8095   Then text ="PA_DRGROUP_TEST"
when TCOD=8500   Then text ="HSK_FROM_SPC_IODP"
when TCOD=8510   Then text ="HSK_TO_SPC_IODP"
when TCOD=8520   Then text ="HSK_XFROM_SPC_IODP"
when TCOD=8525   Then text ="HSK_UNBUNCH_IOPRTY"
when TCOD=8530   Then text ="PA_IMDO_DON"
when TCOD=8540   Then text ="PA_IMDU_DON"
when TCOD=8550   Then text ="PA_IMD_DON_NETVAL"
when TCOD=8560   Then text ="PA_IMD_GDON_NETVAL"
when TCOD=8565   Then text ="PA_IMD_GREC_NETVAL"
when TCOD=8570   Then text ="PA_IMD_RDON_NETVAL"
when TCOD=8573   Then text ="PA_IMD_REC_NETVAL"
when TCOD=8576   Then text ="PA_IMD_RREC_NETVAL"
when TCOD=8580   Then text ="PA_IMD_SEC_DON"
when TCOD=8590   Then text ="PA_IMU_DON_NETVAL"
when TCOD=8595   Then text ="PA_IMU_DON_SEC_REC"
when TCOD=8600   Then text ="PA_IMU_GDON_NETVAL"
when TCOD=8605   Then text ="PA_IMU_GREC_NETVAL"
when TCOD=8610   Then text ="PA_IMU_RDON_NETVAL"
when TCOD=8613   Then text ="PA_IMU_REC_NETVAL"
when TCOD=8616   Then text ="PA_IMU_RREC_NETVAL"
when TCOD=8620   Then text ="PA_IMUO_REC"
when TCOD=8630   Then text ="PA_IMUUA_REC"
when TCOD=8635   Then text ="PA_IMUUB_REC"
when TCOD=8640   Then text ="PA_IMU_SEC_REC"
when TCOD=8650   Then text ="PA_IMU_TO_SPC_DP"
when TCOD=8660   Then text ="PA_IO_DECP_DON"
when TCOD=8670   Then text ="PA_IO_DECP_SEC"
when TCOD=8690   Then text ="PA_IO_DON_DEPEN"
when TCOD=8720   Then text ="PA_IO_GREC_NETVAL"
when TCOD=8730   Then text ="PA_IO_GREC_RECVAL"
when TCOD=8740   Then text ="PA_IO_INCP_DON"
when TCOD=8750   Then text ="PA_IO_INCP_REC"
when TCOD=8760   Then text ="PA_IO_INCP_SEC"
when TCOD=8850   Then text ="PA_IO_NA_NO_DONOR"
when TCOD=8870   Then text ="PA_IO_NA_SPC_DP"
when TCOD=8880   Then text ="PA_IO_RDON_CAND"
when TCOD=8890   Then text ="PA_IO_REC_DEPEN"
when TCOD=8900   Then text ="PA_IO_REC_NETVAL"
when TCOD=8910   Then text ="PA_IO_REC_RECVAL"
when TCOD=8920   Then text ="PA_IO_RREC_NETVAL"
when TCOD=8930   Then text ="PA_IO_RREC_RECVAL"
when TCOD=8933   Then text ="PA_IO_SERVED_GDON"
when TCOD=8936   Then text ="PA_IO_SERVED_GREC"
when TCOD=8938   Then text ="PA_IO_TO_SPC_DP"
when TCOD=8940   Then text ="PA_IO_UNC_DON"
when TCOD=8950   Then text ="PA_IO_UNC_REC"
when TCOD=8960   Then text ="PA_IO_UNC_SEC_DON"
when TCOD=8970   Then text ="PA_IO_UNC_SEC_REC"
when TCOD=8975   Then text ="PA_IO_NA_TOO_SOON"
when TCOD=8980   Then text ="PA_IO_NA_NO_CLUST"
when TCOD=8985   Then text ="PA_IO_NA_REC_INEL"
when TCOD=8990   Then text ="PA_IO_IMPLEMENT"
when TCOD=9010   Then text ="PA_DEC_BP_TAR"
when TCOD=9020   Then text ="PA_INC_BP_TAR"
when TCOD=9030   Then text ="PA_BP_NA_NET_VAL"
when TCOD=9040   Then text ="PA_BP_NA_REC_VAL"
when TCOD=9050   Then text ="PA_BP_TAR_UNAB"
when TCOD=9060   Then text ="PA_BP_NA_EXIT_FAIL"
when TCOD=9170   Then text ="WSM_DEC_BP_TAR"
when TCOD=9180   Then text ="PA_QMPL_NA_REC"
when TCOD=9190   Then text ="PA_QMPL_NA_STOR"
when TCOD=9191   Then text ="PA_QMPL_AUX_STOR"
when TCOD=9195   Then text ="PA_QMPL_NA_RUA0"
when TCOD=9200   Then text ="PA_QMPL_NA_MPL"
when TCOD=9202   Then text ="PA_QMPL_NA_IDLE"
when TCOD=9205   Then text ="PA_QMPL_NA_QUEUE"
when TCOD=9210   Then text ="PA_QMPL_NA_PEND"
when TCOD=9220   Then text ="PA_QMPL_NA_UNMGED"
when TCOD=9230   Then text ="PA_QMPL_NA_REC_RR"
when TCOD=9240   Then text ="PA_QMPL_NA_REC_GR"
when TCOD=9245   Then text ="PA_QMPL_NA_SYSLOC"
when TCOD=9246   Then text ="PA_QMPL_NA_NOSYS"
when TCOD=9247   Then text ="PA_QMPL_NA_SMANG"
when TCOD=9250   Then text ="PA_INC_QMPL_GR"
when TCOD=9251   Then text ="PA_DEC_QMPL_GR"
when TCOD=9260   Then text ="PA_INC_QMPL_RR"
when TCOD=9261   Then text ="PA_DEC_QMPL_RR"
when TCOD=9270   Then text ="PA_QMPL_NA_NETVAL"
when TCOD=9280   Then text ="PA_QMPL_NA_NO_REQ"
when TCOD=9285   Then text ="PA_QMPL_NA_GSMAX"
when TCOD=9295   Then text ="ra_inc_qmpl_aff"
when TCOD=9296   Then text ="PA_QMPL_LIMIT_NUM"
when TCOD=9297   Then text ="PA_QMPL_IMPACT_PER"
when TCOD=9298   Then text ="PA_QMPL_CPU_DON"
when TCOD=9299   Then text ="PA_QMPL_INC_GSMAX"
when TCOD=9300   Then text ="PA_PPP_DECP_DON"
when TCOD=9301   Then text ="PA_PPP_POT_REC"
when TCOD=9305   Then text ="PA_LMP_WT_CHANGE"
when TCOD=9306   Then text ="PA_LMP_GWT_CHANGE"
when TCOD=9307   Then text ="PA_LMP_RWT_CHANGE"
when TCOD=9308   Then text ="PA_LMP_DON_NO_CAP"
when TCOD=9309   Then text ="PA_LMP_DIAG_FAIL"
when TCOD=9310   Then text ="PA_LMP_REC_RECVAL"
when TCOD=9311   Then text ="PA_LMP_GREC_RECVAL"
when TCOD=9312   Then text ="PA_LMP_RREC_RECVAL"
when TCOD=9313   Then text ="PA_LMP_REC_NETVAL"
when TCOD=9314   Then text ="PA_LMP_GREC_NETVAL"
when TCOD=9315   Then text ="PA_LMP_RREC_NETVAL"
when TCOD=9316   Then text ="PA_LMP_DON_NETVAL"
when TCOD=9317   Then text ="PA_LMP_GDON_NETVAL"
when TCOD=9318   Then text ="PA_LMP_RDON_NETVAL"
when TCOD=9319   Then text ="PA_LMP_DON_INV"
when TCOD=9320   Then text ="PA_LMP_REC_MAX_WT"
when TCOD=9321   Then text ="PA_LMP_REC_TIMEINT"
when TCOD=9322   Then text ="PA_LMP_REC_INV"
when TCOD=9323   Then text ="PA_LMP_DON_NETVOK"
when TCOD=9324   Then text ="PA_LMP_GDON_NETVOK"
when TCOD=9325   Then text ="PA_LMP_RDON_NETVOK"
when TCOD=9326   Then text ="PA_CPU_ONLINE_REQ"
when TCOD=9327   Then text ="PA_CPU_OFFLINE_REQ"
when TCOD=9328   Then text ="PA_LMP_DON_CAND"
when TCOD=9329   Then text ="PA_LMP_RECVAL_OK"
when TCOD=9330   Then text ="PA_LPCAP_PMAW"
when TCOD=9331   Then text ="PA_LPCAP_PATTERN"
when TCOD=9332   Then text ="PA_LPCAP_CAP_ON"
when TCOD=9333   Then text ="PA_LPCAP_CAP_OFF"
when TCOD=9334   Then text ="PA_LPCAP_ON_ERR"
when TCOD=9335   Then text ="PA_LPCAP_OFF_ERR"
when TCOD=9336   Then text ="PA_LPCAP_NODATA"
when TCOD=9337   Then text ="PA_LPQUERY_ERR"
when TCOD=9338   Then text ="PA_LPCAP_CONFIGCAP"
when TCOD=9339   Then text ="PA_LPCAP_FIX_PMAW"
when TCOD=9340   Then text ="PA_LPCAP_FIX_OFF"
when TCOD=9341   Then text ="PA_LPCAP_FIX_ON"
when TCOD=9342   Then text ="PA_LMP_GREC_RECOK"
when TCOD=9343   Then text ="PA_LMP_RREC_RECOK"
when TCOD=9344   Then text ="PA_LMP_REC_CAND"
when TCOD=9345   Then text ="PA_LPCAP_PATTERN2"
when TCOD=9398   Then text ="PA_LMP_TEST"
when TCOD=9399   Then text ="PA_LMP_TEST1"
when TCOD=9401   Then text ="PA_LPD204_ERR"
when TCOD=9402   Then text ="PA_LMP_REC_LOWUTIL"
when TCOD=9403   Then text ="PA_PPP_MU_BLKD_PER"
when TCOD=9404   Then text ="PA_GSL_HIGH_DELAY1"
when TCOD=9405   Then text ="PA_GSL_HIGH_DELAY2"
when TCOD=9406   Then text ="PA_GSL_LPAR_TIMES"
when TCOD=9407   Then text ="PA_CA2_BLKD_PER_NS"
when TCOD=9408   Then text ="PA_CA2_BLKD_PER_CM"
when TCOD=9501   Then text ="RA_PAE_MOV_UBA"
when TCOD=9502   Then text ="RA_PAE_MOV_BDEV"
when TCOD=9531   Then text ="SPV_PAE_INV_DEVNUM"
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
     when TCOD=9532    Then text ="SPV_PAE_PLIST_INVD"
   end
     Return text
```

FURTHER READING

- *OS/390 Workload Manager Implementation and Exploitation* (SG24-5326) (Chapter 2).
- *z/OS MVS Planning: Workload Management* (SA22-7602)
- *z/OS MVS Programming: Workload Management Services* (SA22-7619)
- *System/390: Workload Manager Performance Studies* (SG24-4352) (Chapter 2).

Mile Pekic
Systems Programmer (Serbia and Montenegro) © Xephon 2004

Copying and changing datasets quickly

DUPLA is a CLIST that can be used by all TSO/ISPF users who need to make copies of datasets quickly, without searching and editing the source JCL libraries. DEDUPLA then allows them quickly to alter the copies obtained.

USE

It is used to create a copy of a sequential, partitioned, VSAM, ISAM, or DA dataset, or append a suffix to the original dataset name. It is also possible, with a single line command, to:

- Extend partitioned directories.
- Extend PS/PO/VS primary space allocation.
- Create an Extended Partitioned (PO/E) from a PDS.

- Choose another volume for the target dataset.
- Submit JCL with or without editing the skeleton.
- Change the chosen suffix for the copy.

At any time, you can use DEDUPLA to easily cancel the old entries and rename the new objects.

ENVIRONMENT

It works with MVS/ESA or OS/390 until Release 2.10, and TSO/ISPF/DM up to Release 4.5.

RESTRICTIONS

It works fine only with catalogued datasets. The resulting copies are catalogued too, using the same standard catalog as the input has. In SMS environments, the target dataset may be redirected through ACS routines to other volumes (part of a storagegroup), if the chosen suffix matches storageclass rules. In this case, you cannot force the target VOLSER.

COMMANDS

See section DUPLA CLIST:

```
DUPLA datasetname <parameters>
```

See section DEDUPLA CLIST:

```
DEDUPLA datasetname <parameters>
```

The best results are obtained using DUPLA/DEDUPLA from a 3.4 ISPF DSlist, but you can use them from any TSO command or option line in ISPF.

DUPLA CLIST

A dataset or VSAM cluster name is required.

You cannot use DUPLA with DATA or INDEX VSAM objects.

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

If you supply a dataset name without quotes, &SYSPREF is added in front of the dataset name, unless you are in DSLIST (ISPF 3.4). Look at the example below:

```
DSLIST - Datasets Matching EE.SCDS                             Row 1 of 8
Command ===>                                               Scroll ===> PAGE

Command - Enter "/" to select action          Message           Volume
 -------------------------------------------------------------------------
dupla    EE.SCDS                                                  *VSAM*
         EE.SCDS.AL281002                                         MIGRAT1
         EE.SCDS.AL281002.DATA                                    MIGRAT1
         EE.SCDS.COPIA                                            MIGRAT2
         EE.SCDS.COPIA.DATA                                       MIGRAT2
         EE.SCDS.DATA                                             SMPR00
         EE.SCDS.DUPLICAT                                         *VSAM*
         EE.SCDS.DUPLICAT.DATA                                    SMPR00
************************** End of Dataset list ************************
```

The optional parameters are:

- DEBUG – to see all CLIST messages.
- DUP (default ‘DUPLICAT’) – target dataset name suffix.
- PCT (default ‘0’) – additional primary space for the target dataset. PCT is ignored for ISAM and DA object (only the clone-copy is allowed by ADRDSSU). For partitioned datasets, the PCT value refers to the number of directory blocks. You can set this value from 0 to 9999. If the limit is exceeded, the value is set to the maximum allowed (9999). For example setting &PCT(100) means giving an additional 100% of space to the original one; in other words the output space is twice the input.
- PDSE (default ‘PDS’) – PDS means a ‘classic’ partitioned dataset. If you specify ‘LIBRARY’, ‘PDSE’, ‘Y’, or ‘YES’, the target dataset is converted to PO-E. The parameter is ignored when input is not partitioned. In OS/390 2.10, a PO-E dataset may even reside on non-SMS volumes. (In previous releases, the PO-E dataset could reside only on SMS volumes.)
- SUB (default ‘YES’) – specify whether you wish to submit the job without editing the JCL (enter ‘NO’ or ‘N’ to edit

JCL). Obviously, you must enter Command ===> SUB to submit the job if you choose sub(NO).

- VOL (default as the input volser) – target volume, if not SMS-managed. If you supply an output volser, unit-type will be changed to 'SYSDA'. If your installation does not support 'SYSDA' or the chosen VOLSER is not associated with SYSDA, edit the JCL with SUB(NO) and change it manually.

  For example:

  ```
  TSO %DUPLA 'MYDSN' SUB(NO) DUP(CPY) VOL(IPLVOL) PCT(3Ø) PDSE(Y)
  ```

  The use of the % sign before DUPLA reduces the time taken in searching for the CLIST.

  At the end, we obtain a PO-E dataset called MYDSN.CPY on volume IPLVOL, with 30% more primary space than the original input; we chose to edit the JCL and do a manual SUBMIT.

  If input is not a partitioned dataset, parameter PDSE is ignored.

  If you do not use 'quotes' and you are not in 3.4 DSLIST, &SYSPREF is added in front of the dataset name you've supplied. For example MYDSN was interpreted as 'userid-tso.MYDSN'. To see your actual &SYSPREF value, issue the command TSO PROFILE LIST. When null, no first qualifier is added.

The exit codes are:

- 0 – request successfully completed.
- 4 – dataset not on DASD (ml2): request terminated by user.
- 8 – dataset hrecall ml2 failed.
- 12 – dsorg not recognized by listdsi.
- 14 – VSAM, error processing LISTCAT.

- 16 – VSAM entry space-type error.
- 18 – VSAM entry space-sec error.
- 20 – VSAM component, but not cluster entry.
- 22 – listdsi sysreason not 0, error.
- 24 – input dataset not catalogued.
- 26 – target dataset name already catalogued.

DEDUPLA CLIST

A dataset or VSAM cluster name is required.

You can use DEDUPLA with DATA or INDEX VSAM single objects. In this case, only the requested entry is altered. So, we suggest that you use DEDUPLA simultaneously on all parts (in 3.4 DSLIST). Look at the example below:

```
DSLIST - Datasets Matching EE.ACDS                             Row 1 of 4
Command ===>                                                Scroll ===> PAGE

Command - Enter "/" to select action            Message           Volume
-------------------------------------------------------------------------
         EE.ACDS                                                  *VSAM*
         EE.ACDS.DATA                                             SMPRØØ
dedupla  EE.ACDS.DUPLICAT                                         *VSAM*
=        EE.ACDS.DUPLICAT.DATA                                    SMPRØ1
************************** End of Dataset list *************************
```

If you supply a dataset name without quotes, &SYSPREF is added in front of the dataset name, unless you are in DSLIST.

The optional parameters are:

- DEBUG – to see all CLIST messages.
- DUP (default ‘DUPLICAT’) – a single qualifier in DSNAME.

Warning, altering an &DUP VSAM component affects only the entry selected. For example, if you alter an index entry, the data and cluster entries maintain the qualifier &DUP. So, you have to manually ‘deduplicate’ each individual part of a VSAM cluster.

eg:

```
TSO %DEDUPLA 'MYDSN.CPY.AL.CPY' DUP(CPY)
```

At the end, we obtain an object named MYDSN.CPY.AL, because only the LAST qualifier .CPY has been eliminated.

```
TSO %DEDUPLA 'MYDSN.CPY.AL' DUP(CPY)
```

At the end, we obtain an object named MYDSN.AL, because the qualifier .CPY has been eliminated.

If you do not use ‘quotes’ and you are not in 3.4 DSLIST, &SYSPREF is added in front of the dataset name you’ve supplied. For example MYDSN was interpreted as ‘userid-tso.MYDSN’.

To see your actual &SYSPREF value, issue the command TSO PROFILE LIST. When null, no first qualifier is added.

The exit codes are:

- 0 – request successfully completed.
- 4 – dataset not renamed, because the user chose to maintain the existing one.
- 8 – entry not altered.
- 10 – existing dataset not deleted.
- 12 – no dataset qualifier matches &DUP parameter.
- 14 – the supplied qualifier is not correct.

INSTALLATION AND CUSTOMIZATION

DEDUPLA CLIST and DUPLA CLIST must reside in a //SYSPROC concatenated dataset, eg ISP.UISPCLIB.

There are 10 ISPF pop-up panel members. They are DEDOPDEL, DEDOPEXI, DEDOPNDL, DUPOPHSM, DUPOPML2, DUPOPPCT, DUPOPRCD, DUPOPSUB, DUPOPWNG, and DUPOP20.

These ten must reside in a //ISPPLIB concatenated dataset, eg ISP.UISPPLIB.

There are five ISPF skeleton members. They are DUPLICDA, DUPLICIS, DUPLICPO, DUPLICPS, and DUPLICVS.

These five must reside in a //ISPSLIB concatenated dataset, eg ISP.UISPSLIB.

There are three ISPF message members. They are DED00, DUP00, and DUP01.

These must reside in a //ISPMLIB concatenated dataset, eg ISP.UISPMLIB.

DEDUPLA CLIST

```
PROC 1 DATASET DUP(DUPLICAT) DEBUG
/*- SETUP FOR DEBUG IF REQUESTED -----------------------------------*/
  CONTROL NOMSG NOLIST NOFLUSH END(ENDO) NOCONLIST NOPROMPT
  IF &DEBUG  =  DEBUG THEN +
   CONTROL MSG LIST NOFLUSH END(ENDO) PROMPT SYMLIST CONLIST
/*- END OF SETUP ---------------------------------------------------*/
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*                                                                 */
/*  DEDUPLA: ALTER NEWNAME OF A .DUPLICAT ENTRY (SEE DUPLA CLIST)    */
/*           TO ELIMINATE THE LAST QUALIFIER MATCHING &DUP PARAMETER */
/*                                                                 */
/*  WARNING! ALTERING A .DUPLICAT VSAM COMPONENT AFFECTS ***ONLY***  */
/*           THE SAME ENTRY SELECTED. EG, IF YOU ALTER AN INDEX      */
/*           ENTRY, THE DATA AND CLUSTER ENTRIES MAINTAIN THE QUALIF */
/*           .DUPLICAT; SO, YOU HAVE TO MANUALLY 'DEDUPLATE' ALL     */
/*           THE SINGLE PARTS OF A VSAM CLUSTER.                     */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* EXIT CODES:   Ø  REQUEST SUCCESFULLY COMPLETED                    */
/*               4  DATASET NOT RENAMED, BECAUSE USER CHOOSE         */
/*                  TO MAINTAIN THE EXISTING ONE                     */
/*               8  ENTRY NOT ALTERED                                */
/*              1Ø  EXISTING DATASET NOT DELETED                     */
/*              12  NO DATASET QUALIFIER MATCHES &DUP PARM           */
/*              14  THE SUPPLIED QUALIFIER IS NOT CORRECT            */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
  SET &RIG1 = &STR(ENTRY SUCCESFULLY ALTERED)
  SET &LL = &LENGTH(&DATASET)
    SET &APICE = &STR(&SUBSTR(1,&DATASET)
    IF &APICE = &STR(') THEN DO
        SET &DATASET = &STR(&SUBSTR(2:&LL-1,&DATASET)
        SET &LL = &LENGTH(&DATASET)
```

```
                           ENDO
  SET &MM = &LENGTH(&DUP)
  SET &LLL=Ø
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* SEARCH FOR &DUP IN DATASET NAME                                     */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
SYSI: +
  SET &L  = &SYSINDEX(.&DUP,&DATASET,&LLL+1)
  IF &L = Ø AND &LLL=Ø THEN GOTO FUOR
  IF &L NE Ø THEN DO
             SET &LLL=&L
             GOTO SYSI
                ENDO
  SET &L=&LLL+1
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/* SET NEW DSNAME WITHOUT &DUP (LAST OCCURRENCE OF)                    */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
  IF &L+&MM GE &LL THEN SET NEW=&SUBSTR(1:&L-2,&DATASET)
  ELSE SET NEW=&SUBSTR(1:&L-2,&DATASET)&SUBSTR(&L+&MM:&LL,&DATASET)
  IF &DEBUG=DEBUG THEN +
     CONTROL NOMSG NOLIST NOFLUSH END(ENDO) NOCONLIST NOPROMPT
 IF &STR(&SYSDSN('&NEW')) = &STR(INVALID DATASET NAME, '&NEW') THEN DO
      ISPEXEC SETMSG MSG(DEDØØ6)
                    EXIT CODE(14)
                    ENDO
  ELSE SET &RIG2 = &STR(RENAMED AS ... &NEW)
  IF &DEBUG=DEBUG THEN +
     CONTROL MSG LIST NOFLUSH END(ENDO) PROMPT SYMLIST CONLIST
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*  TRYING TO RENAME DATASET...                                        */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
ALTERD: +
             CONTROL MSG
  IF &DEBUG = DEBUG THEN SET &SYSLIST=OFF
       SET &SYSOUTTRAP=999
        ALTER '&DATASET' NEWNAME('&NEW')
         SET &RC = &LASTCC
       SET &SYSOUTTRAP=Ø
  IF &DEBUG = DEBUG THEN SET &SYSLIST=ON
       SET  F=&SYSOUTLINE
             DO UNTIL &F = Ø
            SET TERP=&STR(&&SYSOUTLINE&F)
            SET SY&F=&STR(&TERP)
            SET &F=&F-1
           ENDO
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*  ALTER FAILED WITH A RC=8, DISPLAY MESSAGE IN POP-UP WINDOW         */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
        IF &RC = 8 THEN DO
  SET &L  = &SYSINDEX(&STR(IDC3Ø13I DUPLICATE DATA SET),&SY1)
  IF &L NE Ø THEN DO
                  SET &SY4=&STR(PRESS ENTER TO CANCEL &NEW.)
                  SET &SY5=&STR(OR PF3 TO TERMINATE.)
                ENDO
             CONTROL NOMSG
  ISPEXEC ADDPOP ROW(1Ø)
  ISPEXEC DISPLAY PANEL(DEDOPEXI)
  IF &LASTCC EQ 8 AND &L NE Ø THEN DO
      ISPEXEC SETMSG MSG(DEDØØ2)
                    EXIT CODE(4)
                    ENDO
  ISPEXEC REMPOP
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*  ALTER FAILED, TRYING TO CANCEL EXISTING OBJECT WITH THE SAME NAME*/
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
                      IF &L NE Ø THEN DO
       CONTROL MSG
  IF &DEBUG = DEBUG THEN SET &SYSLIST=OFF
       SET &SYSOUTTRAP=999
       DELETE '&NEW'
       SET &DELCC = &LASTCC
       SET &SYSOUTTRAP=Ø
       SET F=&SYSOUTLINE
             DO UNTIL &F = Ø
            SET TERP=&STR(&&SYSOUTLINE&F)
            SET SX&F=&STR(&TERP)
            SET &F=&F-1
           ENDO
             CONTROL NOMSG
  IF &DEBUG = DEBUG THEN SET &SYSLIST=ON
       IF &DELCC NE Ø THEN DO
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*  IF DELETE FAILS, SEND A MESSAGE AND TERMINATE CLIST.             */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
  ISPEXEC ADDPOP ROW(1Ø)
  ISPEXEC DISPLAY PANEL(DEDOPNDL)
      ISPEXEC SETMSG MSG(DEDØØ4)
  EXIT CODE(1Ø)
                          ENDO
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*  IF DELETE WORKS, TRY AGAIN TO ALTER THE ENTRY.                   */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
  SET &RIG1 = &STR(DATASET &NEW. DELETED ...)
       GOTO ALTERD
                                    ENDO
             ELSE     DO
      ISPEXEC SETMSG MSG(DEDØØ3)
                    EXIT CODE(8)
                    ENDO
                      ENDO
```

```
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
/*  IF ALTER WORKS, SEND AN INFORMATIVE MESSAGE AND CLOSE WITH RC Ø  */
/* . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . */
       IF &RC = Ø THEN DO
  ISPEXEC ADDPOP ROW(6)
  ISPEXEC DISPLAY PANEL(DEDOPDEL)
  ISPEXEC REMPOP
                     ENDO
      ISPEXEC SETMSG MSG(DEDØØ1)
   EXIT CODE(Ø)
 FUOR: +
      ISPEXEC SETMSG MSG(DEDØØ5)
   EXIT CODE(12)
/*-----------------------------------------------------------------*/
```

DUPLA CLIST

```
PROC 1 DATASET +
       DUP(DUPLICAT) +
       VOL(VOLSER) +
       SUB(Y) +
       PCT(Ø) +
       PDSE(PDS) +
       DEBUG
/*- SETUP FOR DEBUG IF REQUESTED ----------------------------------*/
  CONTROL NOMSG NOLIST NOFLUSH END(ENDO) NOCONLIST NOPROMPT
  IF &DEBUG  =  DEBUG THEN +
   CONTROL MSG LIST NOFLUSH END(ENDO) PROMPT SYMLIST CONLIST
/*- END OF SETUP --------------------------------------------------*/
/*-----------------------------------------------------------------*/
/* DUPLA: DATASET CLONE UTILITY.                                    */
/*        IT MAKES A COPY OF A DATASET IS/DA/PO/PS/VS               */
/*        WITH A &PCT ADDITIONAL PRIMARY SPACE (FOR PS/PO/VS) AND   */
/*        WITH A &PCT ADDITIONAL DIRECTORY SPACE (ONLY FOR PO).     */
/*        OBVIOUSLY, YOU MAY CHANGE THIS PERCENTAGE                 */
/*        (SEE &PCT VARIABLE: DEFAULT ØØ% ADDITIONAL PRIMARY SPACE). */
/*        IT ALSO MAKES A PO-E COPY FROM A PO INPUT (&PDSE PARM)    */
/* HOW TO USE: DUPLA (LEFT TO 3.4 DATASET NAME), OR                 */
/*             TSO DUPLA 'MYDSN' <PARMS> (FROM CMD LINE)            */
/* OPTIONAL PARAMETERS:                                             */
/*  DEBUG: TO SEE ALL CLIST MESSAGES                                */
/*  DUP (DEFAULT 'DUPLICAT'): TARGET DSN SUFFIX                     */
/*  PCT (DEFAULT 'Ø'):  TARGET DSN ADDITIONAL PRIMARY SPACE         */
/*                      PCT IS IGNORED FOR ISAM AND DA OBJECT       */
/*                      (ONLY CLONE-COPY ALLOWED BY ADRDSSU)        */
/*                      WHEN &PCT VALUE EXCEED 9999, IT IS SET TO   */
/*                      MAXIMUM LIMIT (9999)                        */
/*  PDSE (DEFAULT 'PDS'): IF YOU SPECIFY 'LIBRARY' OR 'PDSE' OR 'Y', */
/*                      TARGET DATASET IS CONVERTED TO PO-E;        */
/*                      PARM IGNORED IF INPUT IS NOT PARTITIONED.    */
/*  SUB (DEFAULT 'YES'): SPECIFY WHETHER YOU WISH TO SUBMIT THE JOB  */
/*                      WITHOUT EDITING JCL (ENTER 'NO' OR 'N' TO    */
/*                      EDIT JCL). OBVIOUSLY YOU MUST ENTER 'SUB'    */
/*                      TO SUBMIT THE JOB IF YOU CHOOSE SUB(NO).     */
/*  VOL (DEFAULT THE INPUT VOLSER): TARGET VOLUME IF NOT SMS-MANAGED */
/*                      IF YOU SUPPLY OUTPUT VOLSER, UNIT-TYPE WILL  */
/*                      BE CHANGED TO 'SYSDA' ESOTERIC. IN CASE YOUR */
/*                      INSTALLATION DOES NOT SUPPORT 'SYSDA', EDIT  */
/*                      AND CHANGE IT MANUALLY.                      */
/*                                                                   */
/*  EG: TSO %DUPLA 'MYDSN' SUB(YES) DUP(CPY) VOL(IPLVOL) PCT(Ø)      */
/*      IF YOU DO NOT USE QUOTES AND YOU ARE NOT IN 3.4 DSLIST,      */
/*      &SYSPREF IS ADDED BEFORE THE DSNAME YOU'VE SUPPLIED.         */
/*-------------------------------------------------------------------*/
/* EXIT CODES:   Ø  REQUEST SUCCESFULLY COMPLETED                    */
/*               4  DATASET NOT ON DASD (ML2): REQUEST TERMINATED    */
/*                  BY USER                                          */
/*               8  DATASET HRECALL ML2 FAILED                       */
/*              12  DSORG NOT RECOGNIZED BY LISTDSI                  */
/*              14  VSAM, ERROR PROCESSING LISTCAT                   */
/*              16  VSAM ENTRY SPACE-TYPE ERROR                      */
/*              18  VSAM ENTRY SPACE-SEC  ERROR                      */
/*              2Ø  VSAM COMPONENT, BUT NOT CLUSTER ENTRY            */
/*              22  LISTDSI SYSREASON NE Ø, ERROR                    */
/*              24  INPUT DATASET NOT CATALOGED                      */
/*              26  TARGET DATASET NAME ALREADY CATALOGED            */
/*-------------------------------------------------------------------*/
   IF &PCT > 9999 THEN SET &PCT = 9999
       SET &LL = &LENGTH(&DATASET)
       SET &LD = &LENGTH(&DUP)+1
  SET &APICE = &STR(&SUBSTR(1,&DATASET)
   IF &APICE = &STR(') THEN DO
       SET &DATASET = &STR(&SUBSTR(2:&LL-1,&DATASET)
       SET &LL = &LENGTH(&DATASET)
                           ENDO
   ELSE IF &SYSPREF NE    THEN DO
       SET &DATASET = &SYSPREF..&DATASET
       SET &LL = &LENGTH(&DATASET)
                           ENDO
/*-------------------------------------------------------------------*/
/* USE LISTDSI TO OBTAIN INFORMATION ABOUT THE ORIGINAL DATASET.     */
/* DATASETS MIGRATED TO A NON-DASD DEVICE ARE NOT RECALLED.          */
/* TO OBTAIN MORE INFORMATION ABOUT THE LISTDSI COMMAND,             */
/* AND USE OTHER VARIABLES TO MAKE IMPROVEMENTS TO YOUR OWN 'DUPLA'  */
/* CLIST, PLEASE READ THE 'OS/39Ø TSO/E CLISTS' MANUAL.              */
/*-------------------------------------------------------------------*/
INFOS: +
  LISTDSI  '&DATASET' DIRECTORY  /* SMSINFO & RECALL NOT USED */
 SET &§1 = &STR(&DATASET)
 SET &§2 = &SYSVOLUME
 SET &§3 = &SYSUNIT
 SET &§4 = &SYSDSORG
 SET &§5 = &SYSRECFM
 SET &§6 = &SYSLRECL
 SET &§7 = &SYSBLKSIZE
 SET &§8 = &SYSKEYLEN
 SET &§9 = &SYSALLOC
 SET &W1 = &SYSUSEDPAGES
 SET &W2 = &SYSPRIMARY
 SET &W3 = &SYSSECONDS
 SET &W4 = &SYSUNITS
 SET &W5 = &SYSEXTENTS
 SET &W6 = &SYSCREATE
 SET &W7 = &SYSREFDATE
 SET &W8 = &SYSEXDATE
 SET &W9 = &SYSPASSWORD
 SET &X1 = &SYSRACFA
 SET &X2 = &SYSDSSMS
 SET &X3 = &SYSDATACLASS
 SET &X4 = &SYSSTORCLASS
 SET &X5 = &SYSMGMTCLASS
 SET &Y1 = &SYSUPDATED
 SET &Y2 = &SYSTRKSCYL
 SET &Y3 = &STR(&SYSBLKSTRK)
 SET &Y4 = &SYSADIRBLK
 SET &Y5 = &STR(&SYSUDIRBLK)
 SET &Y6 = &SYSMEMBERS
 SET &Y7 = &SYSREASON
 SET &Y8 = &STR(&SYSMSGLVL1)
 SET &Y9 = &STR(&SYSMSGLVL2)
 SET &DSICC = &LASTCC
/*-----------------------------------------------------------------*/
/*  THIS CLIST PORTION MAKES THE NEW DATASET &PCT LARGER THAN       */
/*  THE ORIGINAL ONE (EXCEPT VSAM), WITH THE MINIMUM OF 1 UNIT MORE. */
/*  EXCEPTION: IF YOU SPECIFY PCT(Ø), DUPLA MAKES A PERFECT COPY    */
/*             (INPUT=OUTPUT) WITHOUT ADDED SPACE.                  */
/*-----------------------------------------------------------------*/
 IF &§4 = PS OR &§4 = PO THEN DO
 SET &NEWP=&EVAL((&W2*&PCT./1ØØ)+&W2)
 IF &NEWP=&W2 AND &PCT NE Ø THEN SET &NEWP=&W2+1
                             ENDO
 IF &§4 = PO THEN DO
 IF &Y4 = &STR(NO_LIM) THEN SET &Y4=1
 SET &NEWD=&EVAL((&Y4*&PCT./1ØØ)+&Y4)
 IF &NEWD=&Y4 AND &PCT NE Ø THEN SET &NEWD=&Y4+1
                                         ENDO
 IF &§4 = IS OR &§4 = DA THEN +
    IF &PCT NE Ø THEN DO
  ISPEXEC ADDPOP ROW(1Ø)
  ISPEXEC DISPLAY PANEL(DUPOPPCT)
  ISPEXEC REMPOP
                  ENDO
 IF &W4=CYLINDER THEN SET &W4=CYL
 IF &W4=TRACK    THEN SET &W4=TRK
 IF &W4=BLOCK    THEN SET &W4=&§7
/*--------------------------------------------------------------------*/
/*  DISPLAY A WARNING WHEN DATASET LENGTH EXCEEDS MAXIMUM ALLOWED.    */
/*  WHEN DATASET LENGTH EXCEEDS 44 BYTES INCLUDING THE                */
/*  TARGET SUFFIX (.DUPLICAT), IT GENERATES A 'JCL ERROR'             */
/*  (DSN MAX LENGTH IS 44 BYTE).                                      */
/*  IF USER WISHES TO CONTINUE, HE HAS TO MANUALLY CHANGE             */
/*  THE TARGET DSNAME, OR RERUN DUPLA CHANGING THE DEFAULT SUFFIX,    */
/*  USING THE PARAMETER 'DUP'.                                        */
/*  EG: DUPLA 'MYDSN' DUP(CLONE)                                      */
/*--------------------------------------------------------------------*/
 SET &LT = &LD + &LL
 IF &LT > 44 THEN DO
  ISPEXEC ADDPOP ROW(1Ø)
  ISPEXEC DISPLAY PANEL(DUPOPWNG)
  SET &TL=YES
  SET &SUB=NO
  ISPEXEC REMPOP
              ENDO
/*--------------------------------------------------------------------*/
/*  IF LISTDSI &SYSREASON = 25, THEN DATASET IS NOT ON DASD (ML2).    */
/*  ASK USER FOR RECALL; AT END RETURN TO LISTDSI TO OBTAIN DSORG,    */
/*  OR EXIT DUPLA.                                                    */
/*--------------------------------------------------------------------*/
  IF &Y7 = 25 THEN       DO
  ISPEXEC ADDPOP ROW(1Ø)
  ISPEXEC DISPLAY PANEL(DUPOPML2)
  IF &LASTCC NE 8 THEN DO
       ISPEXEC SETMSG MSG(DUPØØ1)
                   EXIT CODE(4)
                   ENDO
  ELSE DO
  ISPEXEC REMPOP
  HRECALL '&DATASET' WAIT EXTENDRC
  SET &RC = &LASTCC
  IF &RC NE Ø THEN DO
  ISPEXEC ADDPOP ROW(1Ø)
  ISPEXEC DISPLAY PANEL(DUPOPHSM)
  ISPEXEC REMPOP
  ISPEXEC SETMSG MSG(DUPØØ3)
  EXIT CODE(8)
               ENDO
  GOTO INFOS
     ENDO
                     ENDO
```

```
/*-------------------------------------------------------------------*/
/* CHECK INPUT DATASET. IF NOT CATALOGUED, EXIT CODE 24.             */
/*-------------------------------------------------------------------*/
   IF &SYSDSN('&DATASET') NE OK THEN DO
       ISPEXEC SETMSG MSG(DUPØ1Ø)
                    EXIT CODE(24)
                                  ENDO
/*-------------------------------------------------------------------*/
/* CHECK TARGET DATASET NAME. IF ALREADY EXISTS, EXIT CODE 26.       */
/*-------------------------------------------------------------------*/
   IF &SYSDSN('&DATASET..&DUP') EQ OK THEN DO
       ISPEXEC SETMSG MSG(DUPØ11)
                    EXIT CODE(26)
                                  ENDO
/*-------------------------------------------------------------------*/
/*  IF LISTDSI &SYSREASON EQ 12, IT IS A VSAM DATASET.               */
/*  WE EXTRACT SPACE ALLOCATION BY TRAPPING 'LISTCAT' OUTPUT.        */
/*  WARNING: IF IT IS NOT A 'CLUSTER' ENTRY, EXIT WITH RC=2Ø.        */
/*-------------------------------------------------------------------*/
  IF &Y7 EQ 12 THEN   DO
         SET &SYSOUTTRAP=9999
         IF &DEBUG = DEBUG THEN SET &SYSLIST=OFF
         ELSE CONTROL MSG
         LISTC ENT('&DATASET') ALL
         SET &SYSOUTTRAP=Ø
         IF &DEBUG = DEBUG THEN SET &SYSLIST=ON
         ELSE CONTROL NOMSG
         SET &OUTLINE=&SYSOUTLINE
 SEARC: +
 IF &OUTLINE <= &N THEN DO
                     ISPEXEC ADDPOP ROW(1Ø)
                     ISPEXEC DISPLAY PANEL(DUPOP2Ø)
                     ISPEXEC SETMSG MSG(DUPØØ8)
                     EXIT CODE(2Ø)
                     ENDO
 SET &N=&N+1
 SET AL=&&SYSOUTLINE&N
 IF &N=1 THEN SET &POP2Ø=&STR(&AL)
 SET AL=&STR(&SUBSTR(1:1Ø,&AL))
 SET &L = &SYSINDEX(CLUSTER,&STR(&AL))
 IF &L EQ Ø THEN GOTO SEARC
/*-------------------------------------------------------------------*/
/*  VSAM CLUSTER ENTRY FOUND. PROCEED.                               */
/*-------------------------------------------------------------------*/
 CERCA: +
 IF &OUTLINE <= &N THEN DO
        ISPEXEC SETMSG MSG(DUPØØ5)
                    EXIT CODE(14)
                    ENDO
 SET &N=&N+1
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
SET AL=&&SYSOUTLINE&N
SET &W4=NIENTE
  SET &L = &SYSINDEX(SPACE-TYPE,&STR(&AL))
  IF &L EQ Ø THEN GOTO CERCA
  SET &L = &SYSINDEX(KILOBYTE,&STR(&AL))
  IF &L NE Ø THEN SET &W4=KB
  SET &L = &SYSINDEX(MEGABYTE,&STR(&AL))
  IF &L NE Ø THEN SET &W4=MB
  SET &L = &SYSINDEX(CYLINDER,&STR(&AL))
  IF &L NE Ø THEN SET &W4=CYL
  SET &L = &SYSINDEX(TRACK,&STR(&AL))
  IF &L NE Ø THEN SET &W4=TRK
  IF &W4=NIENTE THEN  DO
       ISPEXEC SETMSG MSG(DUPØØ6)
                    EXIT CODE(16)
                    ENDO
SEARCH2: +
SET &N=&N+1
SET AL=&&SYSOUTLINE&N
  SET &L = &SYSINDEX(SPACE-PRI,&STR(&AL))
  IF &L EQ Ø THEN GOTO SEARCH2
  DO UNTIL &L > 32
  SET &I = &L
  SET &L = &SYSINDEX(-,&STR(&AL),&I+1)
  ENDO
  SET &OLDP=&STR(&SUBSTR(&I+1:31,&STR(&AL)))
SEARCH3: +
SET &N=&N+1
SET AL=&&SYSOUTLINE&N
  SET &L = &SYSINDEX(SPACE-SEC,&STR(&AL))
  IF &L EQ Ø THEN     DO
       ISPEXEC SETMSG MSG(DUPØØ7)
                    EXIT CODE(18)
                    ENDO
  DO UNTIL &L = Ø
  SET &I = &L
  SET &L = &SYSINDEX(-,&STR(&AL),&I+1)
  ENDO
  SET &SECP=&STR(&SUBSTR(&I+1:31,&STR(&AL)))
  SET &NEWP=&OLDP
  IF &PCT = Ø THEN GOTO ALLO
IF &W4=TRK OR &W4=CYL THEN DO
   SET &NEWP=&EVAL((&NEWP*&PCT./1ØØ)+&NEWP)
                   ENDO
IF &W4=MB THEN DO
   SET &NEWP=&EVAL(((&NEWP*&PCT./1ØØ)+&NEWP)/21)
   SET &W4=TRK
             ENDO
IF &W4=KB THEN DO
   SET &NEWP=&EVAL(((&NEWP*&PCT./1ØØ)+&NEWP)/21)
```

```
    IF &NEWP < 2ØØØ THEN SET &NEWP=2
    ELSE SET &NEWP=&NEWP/1ØØØ
    SET &W4=TRK
              ENDO
    IF &NEWP <= &OLDP THEN SET &NEWP=&OLDP+1
   ENDO
/*-----------------------------------------------------------------*/
/*  IF LISTDSI &SYSREASON NE Ø, SEND A MESSAGE TO INFORM USER, AND   */
/*  TRY TO PROCEED; IF USER PRESSED PF3, DUPLA TERMINATES WITH RC=4  */
/*  A REASON=12 IS GENERATED FOR VSAM DATASET, IT IS VERIFIED LATER. */
/*-----------------------------------------------------------------*/
 IF  &Y7 NE Ø AND &Y7 NE 12 THEN   DO
  ISPEXEC ADDPOP ROW(1Ø)
  ISPEXEC DISPLAY PANEL(DUPOPRCD)
  IF &LASTCC EQ 8 THEN DO
        ISPEXEC SETMSG MSG(DUPØØ9)
                     EXIT CODE(22)
                     ENDO
                   ENDO
/*-----------------------------------------------------------------*/
/*  PREPARE JCL SKELETON FOR SUCCESSIVE SUBMIT.                      */
/*  WARNING: IF YOU WISH TO DIFFERENTLY MANAGE 'ISPFILE' DD, YOU     */
/*           NEED TO MODIFY THE 'ALLOC' COMMAND (LINES 316-318)      */
/*           THE 'ISPEXEC EDIT' COMMAND (LINE 331) AND THE 'SUB'     */
/*           COMMAND (LINE 335), EG USING YOUR OWN LOGON PROCEDURE  */
/*           PARTITIONED DATASET 'ISP.UISPOLIB'.                     */
/*           USING DUPLA AS IS, AT END FREES 'ISPFILE' DD.           */
/*-----------------------------------------------------------------*/
ALLO: +
  ALLOC F(ISPFILE) DA('&SYSPREF..TEMPLIB.PO') NEW REU +
   LRECL(8Ø) BLKSIZE(Ø) DSORG(PO) RECFM(F B) +
   SPACE(1,1) TRACKS DELETE DIR(1)
  IF &PDSE=PDSE OR &PDSE=Y OR &PDSE=YES THEN +
                     SET &PDSE=LIBRARY
  IF &VOL=VOLSER THEN SET &VOL=&§2
  ELSE SET &§3=SYSDA
  IF &SYSDSORG= THEN   DO
        ISPEXEC SETMSG MSG(DUPØØ4)
                     EXIT CODE(12)
                     ENDO
  ISPEXEC FTOPEN
  ISPEXEC FTINCL  DUPLIC&SYSDSORG
  ISPEXEC FTCLOSE NAME(JCLDUPL)
  IF &SUB = NO OR &SUB = N THEN +
  ISPEXEC EDIT DATASET('&SYSPREF..TEMPLIB.PO(JCLDUPL)')
  ELSE   DO
         SET &SYSOUTTRAP=9999
         IF &DEBUG = DEBUG THEN SET &SYSLIST=OFF
         ELSE CONTROL MSG
  SUB '&SYSPREF..TEMPLIB.PO(JCLDUPL)'
```

```
         SET &SYSOUTTRAP=Ø
        SET  F=&SYSOUTLINE
         DO UNTIL &F = Ø
        SET TERP=&STR(&&SYSOUTLINE&F)
        SET SJ&F=&STR(&TERP)
        SET &F=&F-1
       ENDO
         IF &DEBUG = DEBUG THEN SET &SYSLIST=ON
         ELSE CONTROL NOMSG
         ISPEXEC ADDPOP ROW(1Ø)
         ISPEXEC DISPLAY PANEL(DUPOPSUB)
         ISPEXEC REMPOP
       ENDO
  ISPEXEC SETMSG MSG(DUPØØ1)
  FREE F(ISPFILE)
  EXIT CODE(Ø)
/*-----------------------------------------------------------------*/
```

MEMBER DEDOPDEL

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(GREEN)
 £ TYPE(TEXT) COLOR(TURQ)
)BODY WINDOW(7Ø,3)
$ &RIG1
$ &SY1
£ &RIG2
)END
```

MEMBER DEDOPEXI

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(YELLOW)
 # TYPE(TEXT) COLOR(RED)
 £ TYPE(TEXT) COLOR(TURQ) HILITE(BLINK)
)BODY WINDOW(75,5)
$ &SY1
$ &SY2
$ &SY3
# &SY4
# &SY5
)END
```

MEMBER DEDOPNDL

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(GREEN)
 £ TYPE(TEXT) COLOR(TURQ) HILITE(BLINK)
)BODY WINDOW(75,5)
£ Error processing dataset &NEW.:
$ &SX1.
$ &SX2.
$ &SX3.
$ &SX4.
)END
```

MEMBER DUPOPHSM

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(YELLOW)
 ç TYPE(TEXT) COLOR(TURQ)
 £ TYPE(TEXT) COLOR(GREEN) HILITE(BLINK)
)BODY WINDOW(36,3)
£ HSM recall error, code=&RC
$ &DATASET.
$ press ENTER or a PFK to terminate
)END
```

MEMBER DUPOPML2

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(RED) HILITE(BLINK)
 [ TYPE(TEXT) COLOR(TURQ)
 ] TYPE(TEXT) COLOR(YELLOW)
 £ TYPE(TEXT) COLOR(GREEN)
)BODY WINDOW(5Ø,5)
] WARNING!
] &Y9
[  Press£PF3[to£recall
[ (you must$wait[on tape mount)
[ or£ENTER[to terminate DUPLA.
)END
```

MEMBER DUPOPPCT

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(YELLOW)
 # TYPE(TEXT) COLOR(RED)
 £ TYPE(TEXT) COLOR(TURQ)
)BODY WINDOW(56,3)
$ You cannot specify PCT parm to extend this dataset:
£ &DATASET
$ DSORG=&§4, you can only make a copy of it.
)END
```

MEMBER DUPOPRCD

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(RED) HILITE(BLINK)
 [ TYPE(TEXT) COLOR(TURQ)
 ] TYPE(TEXT) COLOR(YELLOW)
 £ TYPE(TEXT) COLOR(GREEN)
)BODY WINDOW(75,2)
$ Warning! [LISTDSI SYSREASON ne Ø:
] &Y9
)END
```

MEMBER DUPOPSUB

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(YELLOW)
 £ TYPE(TEXT) COLOR(TURQ)
)BODY WINDOW(65,3)
$ &SJ1
£ At end check job output: when RC=Ø, you've got
$ &DATASET..&DUP
)END
```

MEMBER DUPOPWNG

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(YELLOW)
 ç TYPE(TEXT) COLOR(TURQ)
 £ TYPE(TEXT) COLOR(GREEN) HILITE(BLINK)
)BODY WINDOW(55,4)
£ WARNING! Target DSN is too long:
$ &DATASET..&DUP
ç To avoid a JCL ERROR, please modify the JCL card
$ (press ENTER or a PFK to edit JCL)
)END
```

MEMBER DUPOP20

```
)ATTR DEFAULT(%+_)
 $ TYPE(TEXT) COLOR(YELLOW)
 # TYPE(TEXT) COLOR(PINK)
 £ TYPE(TEXT) COLOR(TURQ) HILITE(BLINK)
)BODY WINDOW(72,2)
# &POP2Ø
£ This is not a VSAM CLUSTER entry,$DUPLA failed with rc=2Ø
)END
```

DUPLICDA

```
)CM
)CM   MEMBER DUPLICDA
)CM
)CM   SKELETON TO SUBMIT ADRDSSU
)CM   TO DUPLICATE A 'DA' DATASET
)CM   PLEASE, CHANGE THE JOBCARD TO MEET YOUR INSTALLATION STANDARDS
)CM
//DUPLADA JOB  MSGLEVEL(1,1),NOTIFY=&SYSUID
//COPYDA  EXEC PGM=ADRDSSU,REGION=3ØØØK
//SYSPRINT DD SYSOUT=*
)SEL &TL  EQ YES
//***************************************************************
//* WARNING: DSN=&DATASET..&DUP
//* EXCEED 44 CHARS IN LENGTH. TO AVOID JCL ERROR, PLEASE
//* MODIFY THE DSNAME IN THE 'RENUNC' CARD ...
//***************************************************************
)ENDSEL
//SYSIN DD *
  COPY -
  DS(INC(&DATASET)) -
  INDYNAM(&§2) -
  OUTDYNAM(&VOL,&§3) -
  CANCELERROR       -
  CATALOG FORCE     -
  RENUNC(&DATASET,+
         &DATASET..&DUP)
/*
//
```

DUPLICIS

```
)CM
)CM   MEMBER DUPLICIS
)CM
)CM   SKELETON TO SUBMIT ADRDSSU
)CM   TO DUPLICATE AN 'IS' DATASET
)CM   PLEASE, CHANGE THE JOBCARD TO MEET YOUR INSTALLATION STANDARDS
)CM
//DUPLAIS JOB  MSGLEVEL(1,1),NOTIFY=&SYSUID
//COPYIS  EXEC PGM=ADRDSSU,REGION=3ØØØK
//SYSPRINT DD SYSOUT=*
)SEL &TL  EQ YES
//***************************************************************
//* WARNING: DSN=&DATASET..&DUP
//* EXCEED 44 CHARS IN LENGTH. TO AVOID JCL ERROR, PLEASE
//* MODIFY THE DSNAME IN THE 'RENUNC' CARD ...
//***************************************************************
```

© 2004. Xephon USA telephone (214) 340 5690, fax (214) 341 7081.

```
)ENDSEL
//SYSIN DD *
  COPY -
  DS(INC(&DATASET)) -
  INDYNAM(&§2) -
  OUTDYNAM(&VOL,&§3) -
  CANCELERROR       -
  CATALOG FORCE     -
  RENUNC(&DATASET,+
         &DATASET..&DUP)
/*
//
```

DUPLICPO

```
)CM
)CM   MEMBER DUPLICPO
)CM
)CM   SKELETON TO SUBMIT IEBCOPY
)CM   TO DUPLICATE A 'PO' DATASET
)CM   AND EXTEND ITS PRIMARY AND DIRECTORY SPACE.
)CM   PLEASE, CHANGE THE JOBCARD TO MEET YOUR INSTALLATION STANDARDS
)CM
//DUPLAPO JOB  MSGLEVEL(1,1),NOTIFY=&SYSUID
//STEP1     EXEC PGM=IEBCOPY,REGION=2ØØØK
//SYSPRINT DD SYSOUT=*
//SYS1    DD DISP=SHR,
//        DSN=&DATASET
//SYS2    DD DISP=(NEW,CATLG),UNIT=&§3,VOL=SER=&VOL,
//       LIKE=&DATASET,DSNTYPE=&PDSE,
//       SPACE=(&W4,(&NEWP,&W3,&NEWD)),
)SEL &TL  EQ YES
//***********************************************************
//* WARNING: DSN=&DATASET..&DUP
//* EXCEED 44 CHARS IN LENGTH. TO AVOID JCL ERROR, PLEASE
//* CUT THE DSN IN THE FOLLOWING CARD:
)ENDSEL
//       DSN=&DATASET..&DUP
//SYSIN     DD *
   COPY INDD=SYS1,OUTDD=SYS2
/*
//
```

DUPLICPS

```
)CM
)CM   MEMBER DUPLICPS
```

```
)CM
)CM   SKELETON TO SUBMIT ICEGENER
)CM   TO DUPLICATE A 'PS' DATASET
)CM   AND EXTEND ITS PRIMARY SPACE.
)CM   PLEASE, CHANGE THE JOBCARD TO MEET YOUR INSTALLATION STANDARDS
)CM
//DUPLAPS JOB  MSGLEVEL(1,1),NOTIFY=&SYSUID
//STEPPS    EXEC PGM=ICEGENER,REGION=2000K
//SYSPRINT DD SYSOUT=*
//SYSUT1    DD DISP=SHR,
//          DSN=&DATASET
//SYSIN     DD DUMMY
//SYSUT2    DD DISP=(NEW,CATLG),UNIT=&§3,VOL=SER=&VOL,
//          LIKE=&DATASET,
//          SPACE=(&W4,(&NEWP,&W3)),
)SEL &TL  EQ YES
//*************************************************************
//* WARNING: DSN=&DATASET..&DUP
//* EXCEED 44 CHARS IN LENGTH. TO AVOID JCL ERROR, PLEASE
//* CUT THE DSNAME IN THE FOLLOWING CARD:
)ENDSEL
//          DSN=&DATASET..&DUP
```

DUPLICVS

```
)CM
)CM   MEMBER DUPLICVS
)CM
)CM   SKELETON TO SUBMIT IDCAMS
)CM   TO DUPLICATE A 'VSAM' DATASET.
)CM   PLEASE, CHANGE THE JOBCARD TO MEET YOUR INSTALLATION STANDARDS
)CM
//DUPLAVS JOB  MSGLEVEL(1,1),NOTIFY=&SYSUID
//STEPVS    EXEC PGM=IDCAMS,REGION=2000K
//SYSPRINT DD  SYSOUT=*
//INP DD DISP=SHR,DSN=&DATASET
//OUT DD DISP=(NEW,CATLG),UNIT=&§3,VOL=SER=&VOL,
//        LIKE=&DATASET,SPACE=(&W4,(&NEWP,&SECP)),
)SEL &TL  EQ YES
//*************************************************************
//* WARNING: DSN=&DATASET..&DUP
//* EXCEED 44 CHARS IN LENGTH. TO AVOID JCL ERROR, PLEASE
//* CUT THE DSNAME IN THE FOLLOWING CARD:
)ENDSEL
//        DSN=&DATASET..&DUP
//SYSIN     DD  *
     REPRO INFILE(INP) OUTFILE(OUT) -
           REPLACE REUSE
/*
//
```

MEMBER DED00

```
DED001 'RC=0 Request satisfied'       .ALARM=NO
'Dataset successfully renamed, .&DUP cut off'
DED002 'RC=4 DS not renamed' .ALARM=YES
'User choose to maintain the actual &NEW'
DED003 'RC=8 Entry not altered'        .ALARM=YES
'Please, retry the operation and control IDC messages'
DED004 'RC=10 DS not deleted'         .ALARM=YES
'&NEW not deleted due to an error'
DED005 'RC=12 .&DUP not matched' .ALARM=YES
'.&DUP qualifier not found, please check and correct your request'
DED006 'RC=14 .&DUP wrong' .ALARM=YES
'&DUP is only a part of a qualifier, please supply a complete
qualifier'
```

MEMBER DUP00

```
DUP001 'RC=0 Request satisfied'       .ALARM=NO
'If executed, job DUPLA&SYSDSORG generates a .&DUP dataset from input'
DUP002 'RC=4 DS not on DASD' .ALARM=YES
'Selected dataset is HSM ML2 migrated, you choose not to recall it'
DUP003 'RC=8 HRECALL failed'           .ALARM=YES
'HRECALL for &DSNAME on ML2 failed'
DUP004 'RC=12 DSORG not allowed'   .ALARM=YES
'&SYSDSORG DSORG not allowed. DUPLA accepts DA/IS/PO/PS/VS DSORGs'
DUP005 'RC=14 VSAM Error processing LISTCAT' .ALARM=YES
'SPACE-TYPE information not found for selected dataset'
DUP006 'RC=16 VSAM Error processing LISTCAT' .ALARM=YES
'SPACE-TYPE unexpected error for selected dataset'
DUP007 'RC=18 VSAM Error processing LISTCAT' .ALARM=YES
'SPACE-SEC information not found for selected dataset'
DUP008 'RC=20 VSAM entry is not cluster' .ALARM=YES
'You cannot duplicate a non-cluster VSAM entry'
DUP009 'RC=22 LISTDSI error=&Y7' .ALARM=YES
'Error processing LISTDSI command, reason=&Y7'
```

MEMBER DUP01

```
DUP010 'RC=24 Input DS not catalogued' .ALARM=YES
'&DATASET is not catalogued or not migrated'
DUP011 'RC=26 Output DS already exist' .ALARM=YES
'&DATASET..&DUP is already catalogued, change DUP(suffix)'
```

Alberto Mungai
Senior Systems Programmer (Italy)

© Xephon 2004

System LX and cross-memory services

In our day-to-day work as systems programmers, we have to face product installations requiring cross-memory services, and some questions arise, like:

- How many system LXs (linkage indexes) do I have to reserve for this product?
- How many system and non-system LXs are in use right now?
- Who is connected with whom (from a cross-memory point of view)?
- Who is the owner of a system LX?

To answer those questions I recently wrote a small batch program that displays all cross-memory connections.

Some explanations around system and non-system LXs:

- There is a maximum of 2048 LXs (it was 1024 before z/OS 1.3).
- The number of non-system LXs = 2048 - (NSYSLX + system reserved LXs).
- NSYSLX is a parameter in IEASYS*xx*. The default is 165 (it was 55 before z/OS 1.3). You can determine your actual value by looking at the halfword in the SVT + X'146' (SVTNSLX in the data area) (CVT -> SVT). An IPL is required to change this parameter.
- What are those system reserved LXs? Probably system LXs used by system address spaces like PCAUTH, RASP, TRACE, XCFAS, GRS, CONSOLE, etc.
- The number of system and non-system LXs are now monitored by MVS and messages arise when there is an LX shortage (IEA063E, IEA065E, IEA066I). An LX shortage is also indicated when you start experiencing abends S053 with return code 0111 or 0112.

- In the case of an LX shortage, if this program is not ready to run, take a dynamic dump of PCAUTH (asid 2), including common and private areas, and then use IPCS, as described in IBM APAR INFO II08563, to determine who is causing the trouble.

Note 1: the idea for this program was taken from the IBM INFO ABAR II08563.

Note 2: this program uses two ‘in-house’ macros:

- INITL to start the program (get some memory for save area, chaining of save areas, register equates).
- RCNTL at the end of the program (restore registers, free off save area, and return).

You can substitute them with your own macros.

REXMEM

```
REXMEM CSECT
REXMEM AMODE   31
REXMEM RMODE   24
***************************************************************
* This program is written to help in determine who is working *
* in cross-memory mode (AR mode for example).                 *
***************************************************************
* The idea of this program is taken from IBM APAR II08563.
*
* Starting from the ASCB chain (out of the ASVT), we can
* follow this logic :
*
* 1) At offset X'150' into the ASCB is the address of the ASSB for
*    this address space.
* 2) At offset X'48' into the ASSB is the address of the XMSE.
*    The XMSE resides in the extended private area of ASID 2.
*    NOTE:  for OS/390 R1.3 and above, at offset +X'1C'
*    into the XMSE is the job name and home ASID at the time
*    this XMSE was created (ie the "owner" of this XMSE).
*    This information persists even after address space
*    termination in the case of nonreusable address spaces
*    (see below).
* 3) At offset X'4' into the XMSE is the address of the SETC.
*    The SETC also resides in extended private in ASID 2.
*    If the high order bit of the byte at SETC+X'6' is ON, this
*    address space has (or had) an entry table connected to a
*    system LX.  In effect, it has (had) a cross memory connection
*    to ALL other ASIDs (present and future) in the system.  There
*    is no need to proceed further in this event ...
*    A side effect of using system LX is that the corresponding
*    ASVT slot, once the address space terminates, will be non-reusable
*    for the life of the IPL.
* 4) If the address space does NOT have an ET connected to a
*    system LX, then there are two halfword fields of interest
*    in the SETC. SETC+X'14' contains the number of "TO"
*    connections with this address space, while SETC+X'16'
*    contains the number of "FROM" connections. The sum of these
*    two numbers is the total number of cross memory connections
*    for this address space.
* 5) SETC+X'2Ø' is the start of an array of fullwords. Each
*    fullword in this array has the following characteristics:
*    - If the low order bit (bit 31) is ON, this entry is not
*      currently in use and should not be examined further.
*    - Otherwise, if the high order bit (bit Ø) is ON, this entry
*      describes a "TO" connection.  If bit Ø is OFF, this entry
*      describes a "FROM" connection.  In either event, the
*      remainder of the entry is a pointer to the XMSE for the
*      connected address space.
* 6) It is quite possible for an active address space to have a
*    "cross memory" connection with itself. In this case, the
*    XMSE address seen in the array will be the same as the one
*    in the ASSB above (excluding the high order bit). Otherwise,
*    at XMSE+X'8' is a (doubleword) STOKEN for the associated
*    address space.
* 7) Be aware that, if you are examining an active address space,
*    one or more of the connected address spaces may already have
*    terminated.  If this has happened, you will be able to find
*    the ASCB (if running MVS R3) or ASSB (if running MVS R4 or
*    above) on the memory delete queue.
*    Once an address space has terminated, the jobname, task name,
*    or userid associated with it is no longer available prior to
*    OS/39Ø release 3. At OS/39Ø release 3 and later, the
*    jobname and ASID are found in the XMSE at offsets +X'1C'
*    and +x'24' respectively.
***************************************************************
* Environment :                                               *
* This program should work from OS/39Ø 1.3 and up.            *
* It was fully tested under Z/OS 1.4.                         *
***************************************************************
* Warning : Part of this program goes into PCAUTH's private   *
*           area and retrieves some information using CROSS-  *
*           MEMORY. So it should be link-edited with AC(1)    *
*           and loaded from an authorized library.            *
***************************************************************
* Main logic :                                                *
* CVT ---> ASVT ---> ASCB chain.                              *
* For each ASCB :
*     ASCB ---> ASSB ---> XMSE (in PCAUTH private)          *
*     XMSE ---> SETC (in PCAUTH private)                    *
*     SETC --> row of XMSE(s) for connected address space(s)*
*************************************************************
* INPUT  : - Nothing                                        *
*                                                           *
* OUTPUT : - The DD LISTXME1 (FBA lrecl 133) contains       *
*            the detail of the each ASCB in term of cross-   *
*            memory.                                        *
*          - The DD LISTXME2 (FBA lrecl 133) contains       *
*            the detail of each connection.                 *
*************************************************************
* JCL to execute this program  :                            *
*  //XMEMINFO  EXEC PGM=REXMEM                              *
*  //STEPLIB   DD   DISP=SHR,DSN=my.load                    *
*  //LISTXME1  DD   SYSOUT=*                                *
*  //LISTXME2  DD   SYSOUT=*                                *
*************************************************************
* Lked attributs :                                          *
* Amode 31                                                  *
* Rmode 24                                                  *
* AC     1                                                  *
*************************************************************
* This program will return the following information        *
* in the LISTXME1 DD :                                      *
*    one line for each valid address space in the system with*
*    AscbAddr, Jobname, Asid, AscbAste, AscbLtov, AscbAtov,  *
*    AscbEtc, AscbEtcn, AscbLxr, AscbAxr, AssbXmse, XmseSetc,*
*    use of system LX (if any), number of 'to' and 'from'    *
*    connections (if any).                                  *
*                                                           *
* This program will return the following information        *
* in the LISTXME2 DD :                                      *
*    for each address space involved in cross-memory        *
*    connection :                                           *
*    . one line with Jobname and asid number                *
*    . one line for each connection that this address space  *
*      maintain with other asid, with                       *
*      type of connection ('to' or 'from'), XmseAddr, jobname*
*      and asid of connected address space.                 *
*************************************************************
         EJECT
*********************************************************************
* Return codes :                                                    *
* Ø  : OK                                                           *
* 4  :                                                              *
* 8  : Problem in scanning the XMSE chain in PCAUTH's EPVT.         *
* 12 : Problem to obtain an ALET (cross-mem)                        *
* 16 : We didn't find PCAUTH                                        *
* 2Ø : Error opening LISTXME1 or LISTXME2 out file                   *
* 24 : Program not authorized                                        *
**********************************************************************
* Conventions :                                                      *
* $ Prefixed fields are part of output lines                         *
* # Prefixed fields are flags                                        *
**********************************************************************
* Register usage :                                                   *
*                                                                    *
* RØ  : reserved                                                     *
* R1  : reserved for macros                                          *
* R2  : reserved for trt instruction                                 *
* R3  : first base register                                          *
* R4  : not used                                                     *
* R5  : not used                                                     *
* R6  : not used                                                     *
* R7  : not used                                                     *
* R8  : not used                                                     *
* R9  : work register                                                *
* R1Ø : work register                                                *
* R11 : work register                                                *
* R12 : work register                                                *
* R13 : reserved as savearea pointer                                 *
* R14 : reserved as link register (return address)                   *
* R15 : reserved for return code                                     *
**********************************************************************
**********************************************************************
         EJECT
**********************************************************************
* Some housekeeping. R3, base register.                              *
**********************************************************************
         INITL 3,EQU=R
         EJECT
**********************************************************************
* Main logic                                                         *
**********************************************************************
         BAS   R14,VERIF_AUTH       Authorized ?
         TM    #PGMFLAG,#NOTAUTH    Flag authorized ?
         BO    RETURN               No, terminate processing rc=24
         BAS   R14,OPENDCBS         Open OUTPUT file
         TM    #PGMFLAG,#OPENERR    Open error ?
         BO    RETURN               Yes, terminate processing rc=2Ø
         BAS   R14,SEARCH_PCAUTH    Search for PCAUTH address space
         TM    #PGMFLAG,#PCANOTF    Found it ?
         BO    CLOSE                No, terminate processing rc=16
         BAS   R14,WRITE_TITLE      Let's write a title on output
         MODESET KEY=ZERO,MODE=SUP Superman suit
         BAS   R14,ALESERV_ADD      Get an ALET for PCAUTH
         TM    #PGMFLAG,#ALETNOK    ok ?
         BO    CLOSE                No, terminate processing rc=12
         BAS   R14,ASVT_SCAN       Let's do the job
         BAS   R14,ALESERV_DEL     Delete access to PCAUTH
         MODESET KEY=NZERO,MODE=PROB  Go back to mortal world
CLOSE    BAS   R14,CLOSDCBS        Close all DCBs
         B     RETURN              Bye
         EJECT
**********************************************************************
* This routine checks whether we are APF authorized.                 *
**********************************************************************
VERIF_AUTH DS  ØH
         BAKR  R14,Ø                   Push environment into stack
         TESTAUTH FCTN=1               Let see if we are authorized
         LTR   15,15                   If yes,
         BZ    PR1ØØØ8                 return
         OI    #PGMFLAG,#NOTAUTH       If not, indicate so
         WTO   'REXMEMØ1 program not authorized (APF).    ',ROUTCDE=11
PR1ØØØ8  DS    ØH
         PR                            Pop stack and return to caller
       EJECT
**********************************************************************
* This routine opens all DCBs that we need in this program           *
* R11 used as work register.                                         *
**********************************************************************
OPENDCBS DS    ØH
         BAKR  R14,Ø               Push environment into stack
         USING IHADCB,R11          Base For DCB dsect
         OPEN  (LISTXME1,OUTPUT)
         LA    R11,LISTXME1        R11 = DCB addr
         TM    DCBOFLGS,X'1Ø'      Good open ?
         BO    OPENXME2            Yes, go to next open
         WTO   'REXMEMØ2 error opening LISTXME1 out file.',ROUTCDE=11
         OI    #PGMFLAG,#OPENERR   Set OPEN_ERROR flag
OPENXME2 DS    ØH
         OPEN  (LISTXME2,OUTPUT)
         LA    R11,LISTXME2        R11 = DCB addr
         TM    DCBOFLGS,X'1Ø'      Good open ?
         BO    OPEN_OK             Yes, go to process
         WTO   'REXMEMØ2 error opening LISTXME2 out file.',ROUTCDE=11
         OI    #PGMFLAG,#OPENERR   Set OPEN_ERROR flag
OPEN_OK  DS    ØH
         PR                        Pop stack and return to caller
         DROP  R11                 Free R11
         EJECT
**********************************************************************
* This routine searches the ASVT for the PCAUTH's ASCB.              *
* From there, we get its ASSB address used further on.               *
* R9, R1Ø, R11, R12 used as work registers.                          *
**********************************************************************
SEARCH_PCAUTH  DS ØH
```

```
         BAKR  R14,Ø                  Push environment into stack
         L     R1Ø,16                 GET CVT ADDRESS
         USING CVT,R1Ø                ESTABLISH ADDRESSABILITY
         L     R1Ø,CVTASVT            GET ASVT ADDRESS
         USING ASVT,R1Ø               ESTABLISH ADDRESSABILITY
         L     R12,ASVTMAXU           GET MAX NUMB # SPACE FOR LOOP
         LA    R11,ASVTENTY           GET # OF FIRST ENTRY
ASCBLOP1 DS    ØH
         TM    Ø(R11),ASVTRSAV        VALID ASCB ?
         BO    RUNLOP1                NO, CHECK NEXT ASVT ENTRY
         L     R9,Ø(R11)              GET ASCB #
         USING ASCB,R9                ESTABLISH ADDRESSABILITY
         L     R2,ASCBJBNI            GET # INITIATED JOBNAME
         CLC   Ø(8,R2),=C'PCAUTH  '   IS IT OUR ADDRESS SPACE ?
         BE    BINGO                  YES, GOT IT
         L     R2,ASCBJBNS            GET # START/MOUNT/LOGON NAME
         CLC   Ø(8,R2),=C'PCAUTH  '   IS IT OUR ADDRESS SPACE ?
         BE    BINGO                  YES, GOT IT
RUNLOP1  DS    ØH
         LA    R11,4(,R11)            NEXT ASVT ENTRY
         BCT   R12,ASCBLOP1           CONTINUE TILL ASVTMAXU REACHED
         WTO   'REXMEMØ3 PCAUTH not found.     ',ROUTCDE=11
         OI    #PGMFLAG,#PCANOTF      ADDRESS SPACE NOT FOUND FLAG
         PR                           Pop stack and return to caller
*
BINGO    DS    ØH                     It is our address space
         MVC   ASSBPC#,ASCBASSB       Save PCAUTH's ASSB addr
         PR                           Pop stack and return to caller
         DROP  R9
         DROP  R1Ø
         EJECT
**********************************************************************
* This routine writes the titles on the Output line.                 *
**********************************************************************
WRITE_TITLE    DS ØH
         BAKR  R14,Ø                  Push environment into stack
         MVC   $XM1ASCB,=CL8'AscbAddr'
         MVC   $XM1JBNA,=CL8'Jobname'
         MVC   $XM1ASID,=CL4'Asid'
         MVC   $XM1ASTE,=CL8'AscbAste'
         MVC   $XM1LTOV,=CL8'AscbLtov'
         MVC   $XM1ATOV,=CL8'AscbAtov'
         MVC   $XM1ETC,=CL4'Etc'
         MVC   $XM1ETCN,=CL4'Etcn'
         MVC   $XM1LXR,=CL4'Lxr'
         MVC   $XM1AXR,=CL4'Axr'
         MVC   $XM1XMSE,=CL8'AssbXmse'
         MVC   $XM1SETC,=CL8'XmseSetc'
         MVC   $XM1TO,=CL4' To'
         MVC   $XM1FROM,=CL4'From'
         BAS   R14,WRITE_LISTXME1_LINE
         PR                        Pop stack and return to caller
         EJECT

***********************************************************************
* This routine gets an ALET for the target address space (PCAUTH in  *
* our case).                                                          *
* R12 used as work register.                                          *
***********************************************************************
ALESERV_ADD    DS ØH
         BAKR  R14,Ø                  Push environment into stack
         USING ASSB,R12
         L     R12,ASSBPC#            PCAUTH's ASSB addr, needed for
*                                     ASSBSTKN addressability
*
          ALESERV ADD,STOKEN=ASSBSTKN,ALET=MYALET,CHKEAX=NO
*
         LTR   R15,R15                Let's see rc
         BZ    PR147852               Ø, ok
         ST    R15,HEX1               Otherwise send a message
         BAS   R14,CONVERT_TO_CHAR    with ALESERV return code
         MVC   WTO2+18(8),HEX2
         WTO   'REXMEMØ4  unable to obtain ALET',ROUTCDE=11
WTO2     WTO   '          XXXXXXXX is the return code',ROUTCDE=11
         OI    #PGMFLAG,#ALETNOK      Post flag and go out
PR147852 DS    ØH
         PR                           Pop stack and return to caller
         DROP  R12
         EJECT
***********************************************************************
* This routine drives the logic to loop through the ASVT in search   *
* of valid ASCB.                                                      *
* R1Ø, R11, R12 used as work register.                                *
***********************************************************************
ASVT_SCAN      DS ØH
         BAKR  R14,Ø                  Push environment into stack
         L     R1Ø,16                 Get CVT address
         USING CVT,R1Ø                Establish addressability
         L     R1Ø,CVTASVT            Get ASVT address
         USING ASVT,R1Ø               Establish addressability
         L     R12,ASVTMAXU           Get max numb adspc for loop
         LA    R11,ASVTENTY           Get addr of first entry
ASCBLOOP DS    ØH
         TM    Ø(R11),ASVTRSAV        Valid ASCB ?
         BO    RUNLOOP                No, check next ASVT entry
         BAS   R14,PROCESS_ASCB       Let's do the job for one ASCB
RUNLOOP  DS    ØH
         LA    R11,4(,R11)            Next ASVT entry
         BCT   R12,ASCBLOOP           Continue till ASVTMAXU reached
         PR                           Pop stack and return to caller
         DROP  R1Ø
         EJECT
**********************************************************************
* This routine drives the logic to process one valid ASCB.           *
* At entry R11 is a pointer in ASVT where we can get the ASCB addr.  *
* R2, R9, R1Ø are used as work registers.                            *
**********************************************************************
PROCESS_ASCB   DS ØH
         BAKR  R14,Ø                  Push environment into stack
         L     R9,Ø(,R11)             Get ASCB addr
         USING ASCB,R9                ESTABLISH ADDRESSABILITY
         ST    R9,HEX1
         BAS   R14,CONVERT_TO_CHAR
         MVC   $XM1ASCB,HEX2          Ascb addr in output line
         L     R2,ASCBJBNI            Get @ of initiated jobname
         LTR   R2,R2                  Valid ?
         BNZ   MOVEJBNA               No, let's see other field
         L     R2,ASCBJBNS            Get @ stc/logon/mount jobname
MOVEJBNA DS    ØH
         MVC   $XM1JBNA,Ø(R2)         Jobname in output line
         MVC   $XM2JBNA,Ø(R2)
         MVC   HEX1,ASCBASID          Asid number
         BAS   R14,CONVERT_TO_CHAR
         MVC   $XM1ASID,HEX2
         MVC   $XM2ASID,HEX2
         MVC   HEX1,ASCBASTE          Address space second table
         BAS   R14,CONVERT_TO_CHAR
         MVC   $XM1ASTE,HEX2
         MVC   HEX1,ASCBLTOV          Linkage table origin
         BAS   R14,CONVERT_TO_CHAR    (addr in PCAUTH)
         MVC   $XM1LTOV,HEX2
         MVC   HEX1,ASCBATOV          Authorization table
         BAS   R14,CONVERT_TO_CHAR    (addr in PCAUTH)
         MVC   $XM1ATOV,HEX2
         MVC   HEX1,ASCBETC           Num of entry tables currently
         BAS   R14,CONVERT_TO_CHAR    owned by this address space &
         MVC   $XM1ETC,HEX2           Number of connections to entry
         MVC   $XM1ETCN,HEX2+4        tables
         MVC   HEX1,ASCBLXR           Number of linkage indexes reservd
         BAS   R14,CONVERT_TO_CHAR
         MVC   $XM1LXR,HEX2
         MVC   $XM1AXR,HEX2+4         Num of authorization indexes rsvd
         L     R1Ø,ASCBASSB           Address space secondary block
         USING ASSB,R1Ø
         MVC   HEX1,ASSBXMSE          Cross-memory services block
         BAS   R14,CONVERT_TO_CHAR
         MVC   $XM1XMSE,HEX2
         MVC   XMSECUR#,ASSBXMSE      Save XMSE addr for future use
         CLC   ASSBXMSE,=XL4'ØØ'      XMSE = Ø -> no cross-memory
         BE    NOXMSE
         BAS   R14,PROCESS_XMSE        Let's dive into PCAUTH's private
NOXMSE   DS    ØH
         BAS   R14,WRITE_LISTXME1_LINE
         PR                            Pop stack and return to caller
         DROP  R9
         EJECT
```

Editor's note: this article will be concluded next month.

Michel Joly
Systems Programmer (France)

© Xephon 2004

Please note that the correct contact address for Xephon Inc is PO Box 550547, Dallas, TX 75355, USA. The phone number is (214) 340 5690, the fax number is (214) 341 7081, and the e-mail address to use is info@xephon.com.

NEON Enterprise Software has announced Database Director Persist (D2 Persist), its online reorganization tool for IMS databases that eliminates outages of business applications during database reorganizations.

Previously, IMS database reorganizations always involved a period when business applications and data were unavailable. With D2 Persist, critical applications remain available, enabling business continuity requirements to be met.

For further information contact:
NEON Enterprise Software, 14100 SW Fwy, Suite 400, Sugar Land, TX 77478, USA.
Tel: (281) 491 4200.
URL: www.neonesoft.com/product_mcdd.html.

***

BMC Software has announced SmartDBA System Administration for IMS, a management product that plugs into its SmartDBA console. SmartDBA supports most databases including DB2 and DB2 Universal Database, IMS, Microsoft SQL Server, Oracle, and Sybase.

SmartDBA System Administration for IMS is not intended as an IMS DBA replacement tool. Instead, it is designed to cross-train DBAs and IT staff on the administration of different database platforms.

For further information contact:
BMC Software, 2101 City West Blvd, Houston, TX 77042, USA.
Tel: (713) 918 8800.
URL: www.bmc.com/supportu/hou_Support_ProdVersion/0,3648,19097_0_102903_0,00.html.

***

Embarcadero Technologies has announced Version 3.2 of Embarcadero Job Scheduler, its cross-platform job management tool that automates database maintenance and other routine tasks. It now supports MySQL.

Job Scheduler 3.2 streamlines automated job runs to help prevent failures that may otherwise result from maintaining databases in complex IT environments. Bolstered wildcard support reduces the need to add or remove files individually. Improved enterprise job filtering increases the efficiency of DBAs by allowing them to identify and tackle important jobs first.

For further information contact:
Embarcadero Technologies, 100 California Street, 12th Floor, San Francisco, CA 94111, USA.
Tel: (415) 834 3131.
URL: www.embarcadero.com/news/press_releases/job_scheduler_32.html.

***

GT Software has announced its Ivory Web Services solution, which enables developers to include mainframe applications in their Service-Oriented Architecture (SOA) graphically and without programming.

There are two components – Ivory Studio and Ivory Server. Ivory Studio is a PC-based development application that enables a company to create and publish Web services from existing mainframe assets. Ivory Server is a SOAP Server for Web service deployment.

For further information contact:
GT Software, 1314 Spring Street NW, Atlanta, GA 30309-2810, USA.
Tel: (404) 253 1300.
URL: http://www.gtsoftware.com/products/ivory.php.