RXBATCH V1.5

The REXX procedure RXBATCH provides enhanced usability for jobs using BPXBATCH or BPXBATSL. Using REXX RXBATCH to run BPXBATCH from TSO in batch mode provides the following advantages over using BPXBATCH native:

- You can use DD name STDIN for STDIN data as desired. You do not and should not use DD name STDPARM and having to provide the data as one long UNIX command.
- It allows you to use BPXBATSL with parameter SH to process a login shell if you are a BPX.SUPERUSER.
- This allows to work with DD names in UNIX commands as you stay in the same ASID.
- You can code your UNIX STDIN data using your preferred code page. This makes it easy to write the commands and they are much better readable.
- You can use the back slash ("\") if you need to continue the command on a next line. This works even if you are using record format FB for your JCL library.
- The combination of several advantages mentioned above allows to use MVS data sets or members, created in your code page as "/bin/sh" scripts for usage as STDIN data.
- Doing so and different from using the STDPARM DD name you can switch to other sub shells or interactive command processors, like using "su", "su -s" or the interactive "rexx".
- Unlike standard BPXBATCH utility RXBATCH still works fine when the user's home directory does not exist. BPXBATCH shows message **`FSUM1004 Cannot change to directory </home_dir>.`** and ends.

New functions in RXBATCH V1.1

- The ddname RXBPARM has been introduced for possible separation of pseudo envvars from real environment variables. A free combination of DD:RXBPARM and DD:STDENV (pseudo and real envvars) is allowed to provide full compatibility with the previous behavior.
- Set the home directory to "/" if the defined home directory for the user is not available.

New functions in RXBATCH V1.2

- Command **`Call Syscalls "OFF"`** has been added at the end of processing in order to avoid possible SA03 abends.

New functions in RXBATCH V1.3

- Internal support has been added to show variables used that have not been initialized already.
- Enhancements have been added to the internal SYSCALL command subroutine.
- The problems with final error messages written to STDOUT if an MVS data set or z/OS UNIX file is used have been fixed; all these messages are now written to the job message log.
- A message is provided if ddname STDOUT is not defined.

New functions in RXBATCH V1.4

- A problem has been corrected with the current working directory if the home directory does not exist.

New functions in RXBATCH V1.5

- If establishing the SYSCALL environment fails the error code is provided with the message shown.
- The names of temporary files used are made better unique to avoid using the same name twice in high parallel processing environments.

RXBATCH variables

The procedure supports a set of four RXBATCH variables to set some specific control switches for processing.

_RXBATCH_SWSU	Do (value 1) or do not (value 0) try to switch to superuser mode. The default is 0 (do not switch).
_RXBATCH_LOGIN	Use a login shell (value 1) or do not (value 0). The default is 1.
_RXBATCH_SL	Use BPXBATSL if running in superuser mode (value 0) or use it anyway (value 1). The default is 0.
_RXBATCH_CP	Specify a code page that all the STDIN data is created in. The default code page is IBM-1047.

RXBATCH sample job and output

Following you find a sample JCL for using RXBATCH:

```
//USSJOB   JOB ,'USS Commands',NOTIFY=&SYSUID.,REGION=0M,SYSTEM=xxxx
//* ------------------------------------------------------------------
//* Run UNIX shell commands in batch mode
//* ------------------------------------------------------------------
//  SET REXXLIB=&SYSUID..UNIX.REXX.EXEC          <=== SYSEXEC library
//* ------------------------------------------------------------------
//RXBATCH  EXEC PGM=IKJEFT01,PARM=RXBATCH
//SYSEXEC  DD DSNAME=&REXXLIB.,DISP=SHR
//STDIN    DD DATA,DLM=##
pwd; id
echo aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\
bbbbbbbbbbbbbbbbbbbbbb
echo "ccccccccccccccccccccc" | cat
cp //DD:TEST /dev/fd1
routine_does_not_exist
RC=$?; if [ $RC -ne 0 ]; then echo "\nRc=$RC"; exit 1; else exit 0; fi
##
//STDENV   DD DATA,DLM=##
PATH=/usr/local/bin:/bin
##
//RXBPARM  DD DATA,DLM=##
 _RXBATCH_SWSU=0       < 0= no switch (default), 1= switch to SU mode
_RXBATCH_LOGIN=0       < 0= no login shell, 1= login shell (default)
   _RXBATCH_SL=1       < 0= use BPXBATSL if in SU mode, 1= use always
_RXBATCH_CP=IBM-273    < code page of STDIN data, default is IBM-1047
##
//TEST     DD DATA,DLM=##
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
##
//STDOUT   DD SYSOUT=*,LRECL=136,RECFM=VB
//SYSTSIN  DD DUMMY
//SYSTSPRT DD DUMMY    < Use "SYSOUT=*" instead in case of problems
//*STDPARM  DD ...     < Do not add a STDPARM DD statement to this JCL
//* ------------------------------------------------------------------
```

Note: You can also define these RXBATCH specific settings as “pseudo” environment variables instead of using DD:RXBPARM. This is for your convenience and in case of using the old version already. There is no need to change any JCL statements. Although these settings are provided with DD name STDENV in this case, they are not activated as UNIX environment variables. The environment part of the JCL looks like the following:

```
//STDENV   DD DATA,DLM=##
PATH=/usr/local/bin:/bin
```

```
# --- "Pseudo" envvars -----------------------------------------------
 _RXBATCH_SWSU=0       < 0= no switch (default), 1= switch to SU mode
_RXBATCH_LOGIN=0       < 0= no login shell, 1= login shell (default)
   _RXBATCH_SL=1       < 0= use BPXBATSL if in SU mode, 1= use always
_RXBATCH_CP=IBM-273    < code page of STDIN data, default is IBM-1047
##
```

Following you find the joblog output that has been written to STDOUT:

```
/u/hering
uid=888(HERING) gid=2(SYS1) groups=1047(USSTEST)
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbb
ccccccccccccccccccc
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
routine_does_not_exist: FSUM7351 not found

Rc=127
```

Messages shown in the job message log...

```
 BPXF024I (HERING) BPXBATCH processing ended with RC= 256
 BPXF024I (HERING) Last UNIX status retrieved is: RC= 1
```

Using the specific return code handling in the last line of STDIN we get displayed the real UNIX return code. The UNIX environment is left with Rc=1 to avoid the sometimes confusing way BPXBATCH presents return codes (by multiplying the value received by 256). RXBATCH then sets the TSO return code to 8 in case of an error.