z/OS V1R13

JES2: Batch modernization

Session objectives

- In this session we will introduce JES Batch Modernization changes
  - Introduce instream data set support in JCL PROCs and INCLUDEs
  - Introduce JOBRC support to set a jobs return code
  - Introduce support to evict jobs on a step boundary
  - Introduce Spin Any Spin support

Overview – JCL instream data sets

- Support added to allow instream data in JCL PROCs and INCLUDE sections
  - Works like instream data in normal JCL stream
  - Does not support generating //SYSIN DD *
- Support is based on where the job converts (z/OS 1.13)
  - Can run on downlevel system
- New SYSIN data sets are included in extended status DSLIST function
- New SYSIN data sets are NOT included in SPOOL Data Set Browse of JCLIN
  - Were not part of original JCL submitted
- New SYSIN are NOT transmitted to other nodes or offloaded
  - Were not part of original JCL submitted
- Works for batch jobs as well as started tasks

Usage and invocation – JCL instream data sets

- Embedding instream data in a JES2 procedure

```
//HELLO           PROC
//STEPA           EXEC  PGM=IEBGENER
//SYSIN                   DD    DUMMY
//SYSPRINT        DD    SYSOUT=A
//SYSUT2          DD    SYSOUT=A
//SYSUT1          DD    DATA
HELLO WORLD
/*
//        PEND
```

Overview – Job completion code

- Problem Statement / Need Addressed
  - Highest completion code of a job may not be the most meaningful
- Solution
  - New JCL keyword added to control reported completion code
  - Can select last step, specific step, or highest step completion code
- Benefit / Value
  - Can better determine success of a job without having to look at the job output
- New JOBRC keyword on JOB card
  - Possible values for JOBRC keyword
    - MAXRC – Default, job return code is the max of any step
    - LASTRC – Job return code is the return code of the last step
    - (STEP,*stepname*.*procstepname*) – Job return code is indicated step if it executes, otherwise same as MAXRC
- $T JOBCLASS(x) was enhanced with a new JOBRC operand
  - JOBRC=MAXRC|LASTRC
    - JOBRC keyword on job card takes precedence
- Extended status STTRMXRC and STVBMXRC (traditional) fields
  - Two new conditions on how a job can end
    - Converter error – The converter returned a bad return code and failed the job
    - System failure – The job was executing at the time of a system failure and was not re-queued for execution
  - Old maximum return code in verbose job STVBMXRC field
- $DJQ,CC= value
- ENF 70 field ENF70_MAXCC
- Uses new step end SSI to extract the step return code
  - New JES2 exit 58 called in step end SSI
  - Can be used to influence the step return code
- Updated HASP165 message text
  - *Jobname* ENDED AT *node reason*
  - Examples of *reason*:
    - MAXCC=*code* - JOBRC was not specified
      - Code is now always 4 digits (MAXCC=0000)
    - JOBRC=*code* - JOBRC was specified and affected the return code
    - MAXRC=*code* - JOBRC was specified but MAXRC was returned

- ABENDED S*xxx*,U*yyy*
- ABENDED *abend_code*, JOBRC=code
  - JOBRC=(STEP,*stepname*), step executed, but later step ABENDed

Usage and invocation – Job completion code

- JOBRC=LASTRC
  - This specification indicates to use the return code of the last executed step as the completion code for the job.
- JOBRC=(STEP,C.HLASM)
  - Use the return code for the C step in the HLASM procstepname as the completion code for the job.

Overview – Evict job on step boundary

- Problem Statement / Need Addressed
  - Need an orderly way to get jobs out of execution
- Solution
  - Implemented a new operand on the $E J command that forces a job out of execution when the current step ends
  - Job resumes execution from next step
- Benefit / Value
  - More orderly shutdown of system

Usage and invocation – Evict job on step boundary

- New JES2 command option
  - $EJ*xxxx*,STEP[,HOLD]
  - Deals with cross member requests
- Forces job out of execution when current step ends
- Job requeued for execution (and held if requested)
- Utilizes existing restart logic (continue restart) to perform function
  - Requires JES journal to be active (JOBCLASS(*x*) JOURNAL=YES)
- Uses new step end SSI to communicate with initiator to requeue job
  - New JES2 exit 58 called in step end SSI
  - Can be used to inhibit or trigger function

Overview – Spin any spin

- Problem Statement / Need Addressed
  - Existing JESLOG spin only deals with job logs and system message data sets
  - Other spin data sets may exist that can't be spun
- Solution
  - JES2 will provide the ability to 'spin' any spin SPOOL datasets
- Benefit / Value
  - Can free SPOOL space associated with log data sets created by long running jobs
- JESLOG function supports spinning a job's JESMSGLG and JESYSMSG
- Based on command, size, or time (interval or absolute time)
- Similar spin processing for any spin data set
- The SPIN= DD JCL is enhanced to support operands similar to JESLOG=
- $T JOB was enhanced with a new DDNAME= operand when specified with the SPIN operand.

Usage and invocation – Spin any spin

- An additional operand on SPIN=(UNALLOC) specification in JCL
  - Similar in function to JESLOG= keyword on job card
    - SPIN=(UNALLOC,*value*) where value is one of:
    - '*hh:mm*' – Spin data set at specified time
    - '+*hh:mm*' – Spin data set at interval specified
    - *nnn* [K|M] – Spin every nnn records
    - NOCMD – Cannot spin data set by command (current processing)
    - CMDONLY – Spin only when a command is issued (default)
- Also supported in dynamic allocation and TSO ALLOC
- $T JQ(xxx),SPIN,DDNAME=*ddname* – command can spin data set on demand
  - If you omit DDNAME= all active spin data sets will be spun

Usage and invocation – SSI 82 updates

- Node information SSI – sub-function of JES properties SSI (SSI 82)
  - Enhanced to provide information about NJE nodes from all active members of JES2 MAS
  - Node information is available from JES2 MAS members starting from z/OS 1.11 (requires coexistence APAR on z/OS 1.11 and z/OS1.12)
- New function is exploited by SDSF
- For more information, see publication MVS Using the Subsystem Interface
  - The node information sub-function of the JES properties SSI (mapping macro is IAZJPNJN) was enhanced to support requesting information from other members in MAS.
  - This information is available on z/OS 1.13 from any z/OS 1.11 MAS member and later. The information is provided via node data gatherer code that is common for all releases starting from z/OS 1.11. To support requests from z/OS 1.13, you must apply the node data gatherer APAR on your z/OS 1.11 or 1.12 system.
  - This new function is exploited by the SDSF. There are no external changes, however new function simplifies SDSF installation and configuration in a multi-system environment.

Migration and coexistence considerations

- A down level member will not be allowed to join the MAS and a HASP720 message will be issued in the following situations:
  - A SPOOL migration is active or pending
  - An Extend SPOOL is active or pending
  - A SPOOL volume has STATUS=MAPPED
  - A SPOOL prefix has been defined using generics
- To request an Extend SPOOL data set or SPOOL Migration, all members of the MAS must be at JES2 z/OS 1.13
- To request a SPOOL Migration the MAS must be running in z11 checkpoint mode
- From JES2 z/OS 1.9 or 1.10
  - Can all member warm to z/OS 1.13
  - No coexistence support
  - Fall back implications
    - Some new data structures created by z/OS 1.13 JES2 may result in problems in z/OS 1.10 and prior
    - Prior to z/OS 1.10 may not be able to use SPOOL volumes with non-standard data set names
- From JES2 z/OS 1.11 or z/OS 1.12
  - COMPAT APAR OA31806 is needed on a z/OS 1.11, or z/OS 1.12 member to coexist in a MAS with z/OS 1.13
  - APAR also highly recommended for fall back as well
    - Some new data structures created by z/OS 1.13 JES2 may result in problems if OA31806 is not installed.
- Applications that directly access SPOOL data without using published SSI interfaces will encounter severe problems accessing data for migrating/migrated volumes.
- If you do I/O to SPOOL directly you will break
  - I/Os need to be serialized to the data migrator
  - Need to notify data migrator of intent to do I/O
  - Need to know which device (source or target) to direct I/O to

Session summary

- All customers have some JCL needed to run their business.
  - Batch Modernization enhancements will help simplify the creation of that JCL library. It is expected that application programmers as well as system programmers will benefit from these changes.

Appendix - References

- Publications
  - z/OS V1R13.0 JES Application Programming – SA23-2240-03
  - z/OS V1R13.0 JES2 Commands – SA22-7526-12
  - z/OS V1R13.0 JES2 Initialization and Tuning Guide – SA22-7532-11
  - z/OS V1R13.0 JES2 Initialization and Tuning Reference – SA22-7533-11
  - z/OS V1R13.0 JES2 Installation Exits – SA22-7534-13
  - z/OS V1R13.0 JES2 Macros – SA22-7536-11
  - z/OS V1R13.0 JES2 Messages – SA22-7537-11
  - z/OS V1R13.0 MVS Using the Subsystem Interface - SA22-7642-11