z/OS Basics:
JES 201 - Differences Between JES2 and JES3

Tom Wasik
IBM JES2 Development

David Jones
IBM JES3 Development

Wednesday, August 10, 2011
Session 9724: 11:00 AM – 12:00 PM

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

IBM®
MVS
JES2
JES3
RACF®
z/OS®
zSeries®

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
UNIX is a registered trademark of The Open Group in the United States and other countries.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

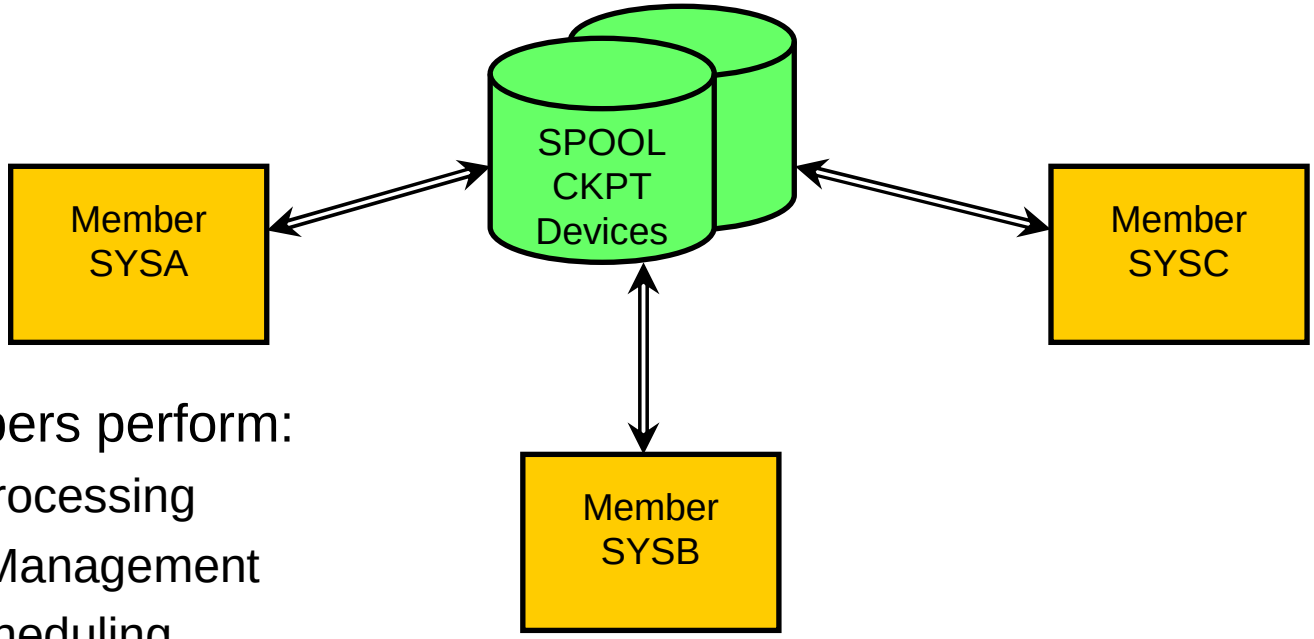
What does JES do?

- Manages SPOOL space to store JCL, SYSIN, SYSOUT
- Manages jobs (started tasks, TSO users, BATCH)
  - Does input processing for job
  - Manages conversion processing
  - Manages JES mode initiators
  - Selects jobs for execution
  - Processes SYSOUT
- Provides execution services and enquiry services
- Does remote processing (NJE/RJE/RJP)
- Performs these functions across multiple z/OS images
  - Integrates images into a single system to process work

JES2

- Common set of work queues stored in its checkpoint
  - Member adds to or selects work from this common queue
  - Checkpoint is time sliced among members
- Simple mechanisms for managing work
  - Resource management done by MVS
  - Depend on MVS (scheduling environment, etc) to determine eligibility to select jobs for execution
  - Jobs sit in initiators waiting for resources
- Peer to Peer relationship between members
  - Members select work that it can process
    - Little regard to other members
  - No single point of control
  - No critical member
- Primary communication via JES2 checkpoint data set

JES2 MAS



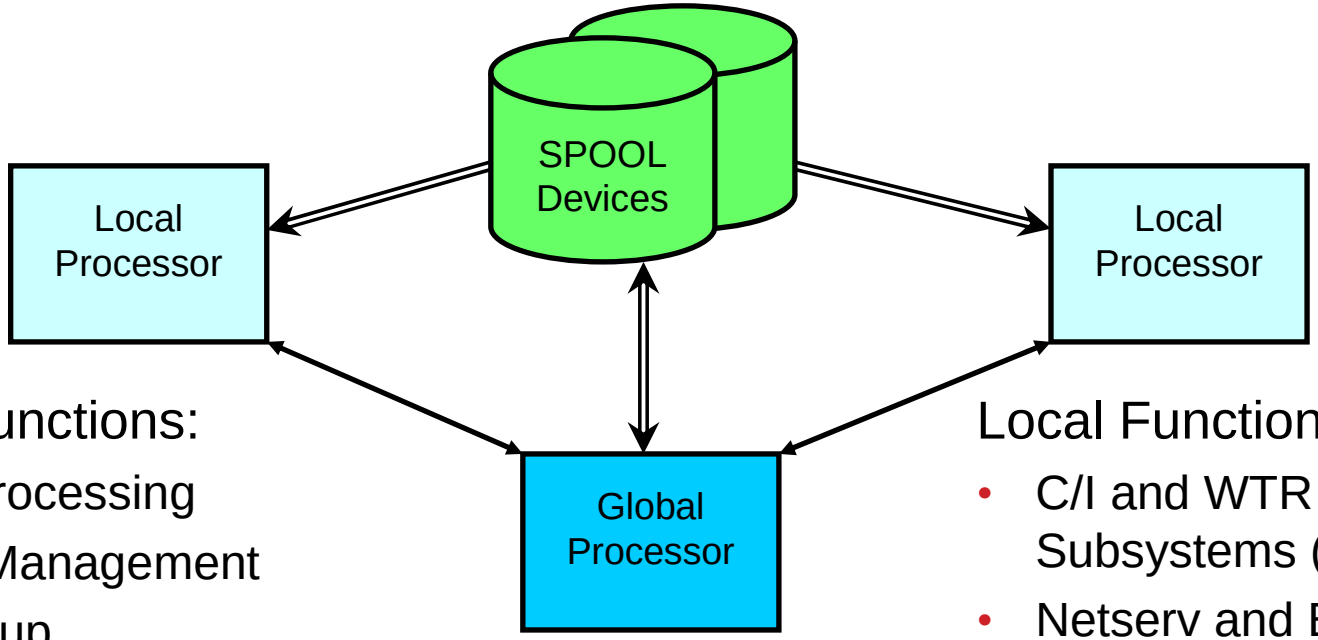
All members perform:

- Input processing
- Spool Management
- Job Scheduling
- SYSOUT scheduling
- SSI processing
- NJE/RJE

JES3

- JES3 does resource and workflow management before and after job execution.
- MVS does resource and workflow management during job execution.
- JES3 exercises centralized control through the global processor.
  - Global performs job selection, scheduling, and device allocation functions for the local processors.
  - Provides increased job scheduling control, deadline scheduling capabilities, and increased control with JES3 device allocation.

JES3 Global vs Local Function



Global Functions:

- Input processing
- Spool Management
- Job setup
- Job scheduling
- SYSOUT scheduling
- SSI processing
- C/I and WTR Functional Subsystems (FSS)
- Netserv and BDT

Local Functions:

- C/I and WTR Functional Subsystems (FSS)
- Netserv and BDT
- Some SSI processing

JES3 Global

- Introduces all jobs into the system
- Handles scheduling of JCL Conversion/Interpretation (C/I)
- Pre-execution setup of JES3 managed devices
- Schedules jobs to all main processors (locals)
- Has awareness of all jobs in execution
- Handles scheduling of SYSOUT data sets for writers
- Manages space on the shared-spool devices

JES Structure

- JES address space
  - Main task
    - Sub-dispatched (shared) with multiple processes
  - Subtasks to perform tasks that must MVS wait
- Auxiliary address space to own processes/objects
  - Allows JES address space to fail yet JES functions continue
- JESXCF services
  - Provide XCF communications between address spaces
    - Primary communication between JES3 global and locals
    - Mainly used for enquiry/posting function for JES2
  - Manages status of JES address spaces
    - Notifies other members of a JES failure

JES2 PCEs

- JES2 Main task processing is performed by PCEs
  - Sub-dispatchable units of work running under main task
- Each job phase is processed by a PCE type
  - Conversion, Execution, Output, purge
- JES2 functions are also implemented by PCEs
- Table pairs can be used to define and created PCEs
  - Tables are defined in modules loaded via JES2 LOADMOD statement
  - Can be dynamically created using $ADD LOADMOD command
- Table pairs can be used to extend other things in JES2
  - For example commands and initialization statements

JES3 DSPs

- JES3 is composed of routines called Dynamic Support Programs (DSPs).
  - DSPs are schedulable units that are processed the same as jobs.
- Resident DSPs are fixed parts of JES3 processing.
  - Dynamic Allocation (DYNAL)
  - Output Service (OUTSERV)
  - Modify commands processor (MODDRVR)
  - Main Device Scheduler (SETUP)
- Callable DSPs are invoked by the operator with *CALL,dspname.
  - Display
  - Writer (WTR)
  - Network Job Entry (NJE)
  - Dump Core (DC)
  - JES3 Monitoring Facility (JMF)
- DSPs that process pieces of work required by a job.
  - Converter/Interpreter (CI)
  - Main service (MAIN)
  - Purge (PURGE)
- To customize job processing, JES3 users can add DSPs, replace existing DSPs, or alter behavior with installation exit routines.

JES2 Input Processing

- Internal reader processing occurs in submitter address space
  - Request sent to JES2 address space to get a job number, job structure (in checkpoint), and initial SPOOL space
  - JCL is parsed, SYSIN data sets created, and SPOOLed in submitter address space
- Job completes input processing when
  - ENDREQ macro is issued
  - /*EOF card submitted
  - New job card is encountered
  - Data set is closed
- NJE/TCP processing is similar (occurs in NETSERV address space)
- BSC and SNA NJE, RJE, SPOOL Reload processing occurs in the JES2 address space (main task)
- Single submission streams are limited by ability to access the JES2 checkpoint
  - Parallel streams can submit much faster (10 parallel streams are 10x one single stream)

JES3 Input Processing

- Input processing occurs in JES3 global address space when:
  - Reader completes input of a job to spool.
    - Disk reader, tape reader, card reader including RJP workstation.
  - INTRDR spool data set is closed or ENDREQ macro used.
  - NETSERV spool data set is closed.
  - BDT spool data set is closed.
- Job Id(s) are created at:
  - ENDREQ
  - When control statements are processed from the spool data set.
- JES3 job control blocks are built and written to spool when the control statements are processed.

JES2 Conversion Processing

- Converts JCL into internal format needed to run job
- JES2 calls MVS converter in the JES2 address space
  - PCE selects job and sets up environment
  - Subtask used to call z/OS converter
- JCL interpretation occurs when job executes in target address space
  - Certain JCL error not detected until job is in execution
- Converter parms (defaults) based on JOBCLASS
  - Journal, BLP, SMF exits, PROCLIB, region, SWA ABOVE, etc
- TYPERUN=SCAN just performs conversion processing
  - Errors discovered during interpretation not detected

JES3 Converter/Interpreter (C/I)

- C/I takes place in the JES3 global address space (C/I DSP) or can be offloaded to C/I FSSes.
  - JES3 C/I invokes the MVS C/I function to process the JCL.
- JCL interpretation is done immediately after conversion.
  - In the C/I DSP or FSS rather than after job selection in an initiator.
  - Can provide quicker feedback of JCL errors.
- Converter parms (defaults) set from CIPARM statement based upon the input device.
  - Journal, BLP, SMF exits, PROCLIB, region, SWA ABOVE, etc
- TYPRUN=SCAN will also invoke the MVS interpreter.
  - Parameter and specification errors can be detected and corrected before a job is submitted for execution.

Functional Subsystems (FSS)

- Interface to offload function to separate address spaces
  - Reduces workload on JES main task
  - Original use for printer/writer support
  - Removes printer “driver” knowledge from JES and associates them with FSS software
  - JES managed interface
    - Controlled by JES commands

- JES3 extended FSS to support Converter/Interpreter service
  - Allows conversion to be processed outside JES3 address space
    - Runs on both the Global and Local processors

MVS Allocation

- A job's resource requirements are not known until a system initiator begins the step allocation process.
  - The job's requirements are satisfied one step at a time.
- Attempts to satisfy requirements are in contention with every other job step currently executing in the SYSPLEX.
- Jobs waiting for resources hold other available resources (an initiator, an address space, data sets, devices).
- Waiting jobs complicate the task of determining how many JES initiators are needed to keep the system full.

JES2 Job Scheduling

- Device/Data set scheduling managed by MVS
  - Job starts in initiator and waits for needed resources
  - GRS ENQ at allocation performs serialization and reports contention
- Allocations managed at a STEP level
  - No issues if one step creates data sets used by later steps
  - Steps that are skipped (due to conditional JCL) do not reserve resources
- System affinity and Scheduling environment used
  - Controls what jobs can be selected for execution
- Some balancing done for WLM initiators
  - Keep same percent busy on all members

JES3 Main Device Scheduling (MDS)

- A device management facility that can wholly or partially support the MVS allocation process.
  - Useful on a single system (global only) or multi-image (global-local) complex.
  - Optional and can be defined with JES initialization statements, JCL control statements, and JES3 operator commands.
- Satisfies the job's I/O resource requirements before and during job execution to prevent allocation delays.
  - Job's resources (devices, volumes, and data sets) are setup before execution.
  - Maximizes use of devices across the system.
  - An initiator should never be idle waiting for a job's resources.

JES3 MDS (continued)

- Considers the resources for all steps of a job and across all mains in the JESplex.
  - Has more information than is available to MVS allocation.
  - Can schedule combinations of jobs that will execute on a main without contention of resources attached to the main.
- Cooperates with Workload Management (WLM) to ensure scheduling environments for jobs are honored.
- With setup before job execution, dependencies between jobs or steps in a job cannot be determined.
  - Important for cataloging and passing data sets.
  - **JES3 assumes all steps will execute** and cannot determine if conditional job steps are skipped.

JES3 Generalized Main Scheduling (GMS)

- Selects and schedules a job for execution when an MVS initiator requests work.
- GMS features can be used to control when jobs execute.
  - Deadline scheduling
    - JES3 periodically increases a job's selection priority in an attempt to run the job by a specified deadline.
  - Dependent Job Control (DJC)
    - A "network" of jobs can be executed in a specific order or in parallel as determined by job dependencies.
    - //*NET control statements are used to define a DJC network and the dependencies between jobs in the network.
    - Can manage data dependencies, device utilization, or the job stream.

WLM–Managed Initiators

- Initiators started and managed by WLM
- Initiators associated with WLM service class
  - Only select work for a specific service class
  - Job class can influence service class assignment
- WLM starts initiators based on
  - System capacity (WLM tries to balance work across SYSPLEX)
  - Whether service class is meeting goals
  - Relative importance of the service class
- JES tells WLM on each system how many jobs are waiting
  - Based on service class, resource availability, where jobs can run
- JES decides what job to start in each initiator
- Specified by MODE=WLM on
  - JOBCLASS for JES2
  - Job class group for JES3

JES2–Managed Initiators

- Type of initiator used based on a JOBCLASS MODE=
  - Applies to all members of the MAS
  - MODE=WLM cannot be selected by JES2 initiators
- MODE=JES JOBCLASS uses JES2 initiators
  - Initiators started and managed by operator commands
  - Number of initiators defined at initialization
  - Initiators select jobs based on a ordered list of job classes.
- Start job command, $S J(nnn), causes WLM to start an initiator to run a specific job
  - Job can be in a WLM or JES mode class

JES3–Managed Initiators

- Associated with a job class group.
  - GROUP initialization statement.
- Expressed in terms of resources, not goals.
  - How many to start, what systems, when to start/stop defined with EXRESC parameter of GROUP initialization statement.
- Started/stopped by JES3 based on definitions and the backlog of jobs.
  - Also with *MODIFY,G command.
- Jobs selected within the job class group.
  - Selected based upon priority within the group.
- Workload balancing is difficult to define and static in nature.

Why WLM–managed over JES–managed?

- Fewer and simpler externals are needed to control WLM-managed initiators and to perform workload balancing.
- Managed according to the service classes and performance goals specified in the WLM policy.
- Externals reflect customer expectations typically in terms that are found in service level agreements.
- Workload balancing is automatic as the number of initiators running is based on performance goals and the importance of batch work with respect to other work.
- Dynamic, goal oriented initiator management allows the system to adapt to changing conditions and how well the work is meeting its performance goals.

JES2 Job Limits and Affinities

- JOBCLASS limits exist on a JESPLEX and member level
  - Number of concurrent jobs that can be active in JOBCLASS
  - Applies to JES and WLM mode JOBCLASSes (JOBs)
  - Limits affect number of available jobs reported to WLM
    - Impacts number of initiators WLM starts
- JOBCLASS affinity controls member where class is active
  - Lists systems that can select from the job class
  - Holding class same as null affinity list
  - Applies to JES and WLM mode JOBCLASSes
  - Affect number of available jobs reported to WLM
- Service class affinity limits where service class is active
  - Service class only registered if member in affinity list
  - WLM only starts initiators if service class is active

JES3 Job Class Groups and Job Classes

- Job class group is a named set of resource assignment rules to be applied to a group of job classes.
  - Essentially job class groups define the initiators available and where.
  - GROUP definition includes the mode (JES3 or WLM), the mains the initiators can run on, and a name that can be used to assign job classes to the GROUP.
- Job class is a named set of job processing and scheduling rules.
  - CLASS definition specifies a GROUP, the systems where the job class can be scheduled to run on, and many other options that are used to control the jobs scheduled within the class.
  - 8-character job class names can be defined and used on the //*MAIN statement.

JES3 Job Resources

- Device Fencing/Pooling
  - Can reserve devices for use only by jobs within a job class group or DJC network.
- Spool Partitioning
  - Can specify the spool partition to be used for jobs in a job class.
  - More on spool partitioning in a moment.

JES3 Job Class Limits

- Class limits can control the number of jobs in execution.
  - TDEPTH to limit the total number of jobs in a class that can run in the entire JESplex.
  - MDEPTH to limit the number of jobs in a class that can run on a particular main.
  - TLIMIT to limit the total number of jobs in a class that can run in a JESplex based on the number of jobs running in that JESplex in another class.
  - MLIMIT to limit the number of jobs in a class that can run on a particular main based on the number of jobs running on that main in another class.
- Limits apply to JES3 and WLM managed job class groups.
  - Class limits are applied to each service class separately during sampling.

SYSOUT Grouping

- A SYSOUT group is defined as the set of data sets that prints between a set of job separator pages
  - Always for the same job and security information
  - For SAPI, between group begin and group end indicators
  - For SDSF, data sets in a row on the O or H panel
- JES2 does early binding
  - Groups data sets into JOEs during OUTPUT phase
- JES3 does late binding
  - No groups until writer select output
    - Output selected that match writer's criteria

JES2 SYSOUT Processing

- JES2 performs grouping during the OUTPUT phase
  - Data sets grouped into a JOE (Job Output Element)
  - Data sets in a group have same class, destination, security info, etc
- SPIN data sets are never grouped with other data sets
- Can influence (prevent) grouping using JCL (GROUPID=)
- Re-grouping only done with SAPI
  - PSO for certain held data sets
- SYSOUT cloning can create multiple copies
  - /*JOBPARM COPIES=
  - Allows multiple copies of entire job output
    - ABC – ABC vs AABBCC
  - Cannot be re-grouped
    - Problem for SAPI/PSO

JES3 Output Service

- JES3 has Output Service Elements (OSE) to maintain output characteristics and processing options for SYSOUT data sets within a job.
  - Each OSE represents 1 to 16 data sets with similar output characteristics within the job.
- “Best fit” approach to matching SYSOUT data sets, via OSEs, to the characteristics of a writer when requested.
  - Can be considered late binding.
- A SAPI output group is based upon the OSEs and may consist of 1 to 16 SYSOUT data sets.
- SYSOUT attribute defaults can be associated with the defined SYSOUT class.

JES2 SPOOL

- Up to 253 SPOOL volumes supported
  - 1 single extent SPOOL data set per volume
  - Maximum data set size 1M tracks
- Basic unit of SPOOL allocation is the TRACK GROUP
  - 1-20 tracks per track group
  - Size can differ from one volume to another
- Each member performs allocations for job/processes on that member
  - Cache of 255 entries for when checkpoint is not owned
- SPOOLs can be added (started) or deleted (drained) via operator command

JES2 SPOOL

- SPOOL affinity limits what SPOOLs a member can allocate SPOOL from
  - Can associate volumes with systems running a particular workload
  - Guideline rather than a hard limit
- SPOOL fencing can limit how many volumes a job can be spread over
  - Can limit impact of loss of SPOOL volume
  - Has negative performance implications
  - Specify the number of volumes that can be used

JES3 SPOOL

- Limited only by the amount of DASD used.
  - Up to 1024 SPOOL extents (i.e. spool data sets).
- Track Group is the basic unit of spool space allocation.
  - Each is a group of spool records.
  - Spool record size = JES3 buffer size.
- SPOOL partitioning
  - Logical groups of spool data sets.
  - Can isolate spool data in separate partitions to improve spool performance, spool recovery procedures, and spool space management.
  - Can specify the partitions JES3 is to use for each processor, for each job class, and for each SYSOUT class.
- JES3 supports dynamic SPOOL add – z/OS V1.13.

JES3 SPOOL Partitioning Advantages

- Limit impacts of spool data set failures.
- Limit competition between mains for each partition to improve system performance.
  - Spread the use of spool partitions across jobs, job classes, and SYSOUT classes,
- Tailor spool space allocation to the requirements of jobs using that partition to improve performance.
  - Track groups size can be specified for each partition.
- Isolate critical work to specific partitions to ensure that spool space is available for critical jobs and users.

JES2 SPOOL I/O

- Each address space does EXCP(VR) to read/write SPOOL
  - Each JES data set does its own EXCP(VR)
  - Writes to SYSOUT done asynchronously
  - I/Os can be done in parallel with PAV
    - Even within a single address space
- Uses 31 bit private buffers in application address space
- Supports 31 bit callers (ACB/RPL interface)
- Allocations can use XTIOT – z/OS V1.13

JES3 SPOOL I/O

- JES3 does its own STARTIOs for spool reads/writes.
  - Data buffers are copied from/to ECSA or JES3 AUX address space
    - Number of buffers in each area and number fixed is specified for each main.
  - When buffers are queued, check is done to see if I/O active to the extent:
    - If not active and needed resources are available, a new I/O is started for the extent.
    - If active, new buffers are processed as part of completing the active I/O.
  - One active I/O per extent per JES3 main.
- Supports 31 bit callers (ACB/RPL interface) – z/OS V1.13
- Allocations can use XTIOT – z/OS V1.13

Miscellaneous Features

JES2

- NJE
  - BSC – Native
  - SNA – Native
  - TCP/IP – Native common
  - Dynamic and static routing (NPM)
    - Dynamic not as interesting
  - Automatic restart (NRM)
- RJE
  - Native BSC and SNA
- SPOOL offload
  - NJE format dump

JES3

- NJE
  - BSC – Native
  - SNA – BDT
  - TCP/IP – Native common
  - Static routing
- RJP
  - Native BSC and SNA
- DUMP job
  - Internal control block format

Miscellaneous Features

JES2

- Performance Monitoring
  - JES2 Health Monitor
  - PERFDATA command
    - WTO records
- INIT deck
  - Processed on every start
- Instream data set max LRECL 32K
- DESTID and DEST processing
- ENF 70 job transition notification

JES3

- Performance Monitoring
  - JES3 Monitor DSP
  - JMF – JES3 monitoring facility
    - SMF records
- INISH deck
  - Processed on certain starts
  - Offline inish deck checker
- Instream data set max LRECL spool buffer size-46
- DLOG (consolidated SYSLOG)

JCL

- JCL that works differently depending on JES
  - DD or OUTPUT COPIES=1-255 (JES2) vs 0-255 (JES3)
  - DD DLM= processing
  - HOLD= processing
  - JOB accounting field
  - NULL statement processing

JCL

JES2 Unique

- /* JECL
- DD SEGMENT=
- JOB TYPRUN=COPY
- JOB TYPRUN=JCLHOLD
- OUTPUT GROUPID=
- OUTPUT INDEX=
- OUTPUT LINDEX=
- OUTPUT LINECT=
- OUTPUT OUTDISP=

JES3 Unique

- //* JECL
- EXEC PGM=JCLTEST
- EXEC PGM=JSTTEST
- OUTPUT OVFL=
- OUTPUT THRESHLD=
- XMIT SUBCHARS=

JES3 DLOG

- **SYSLOG** individually records command and message traffic for each system in MVS format.
- **JES3 DLOG** centrally records command and message traffic for systems in a JES3 complex in JES3 format.
  - Written to SYSLOG on the global processor.
  - Precursor to OPERLOG.
- **OPERLOG** centrally records command and message traffic for systems in a sysplex in Message Data Block (MDB) format.
  - “IBM recommends use of OPERLOG on all systems.”

Questions?