

(E)JES[®]

(E)JES vs. SDSF

July, 2010



PHOENIX Software International[®]

Phoenix Software International, Inc.®

831 Parkview Drive North

El Segundo, California 90245-4932

1-(310)-338-0400

<http://www.phoenixsoftware.com>

©Copyright 2005-2010 by Phoenix Software International. *EJES* is a registered trademark of Phoenix Software International, Inc. This information is intended for use by (E)JES customers only. Other usages are prohibited. Reproduction for internal use only is permitted, provided that copies are made without modification and with all proprietary notices preserved. All other trademarks are acknowledged and respected.

Contents

Functionality	5
Performance.....	9
Resource Utilization	9
Performance Testing.....	9
Other Benefits	12

Functionality

Users and system administrators with sufficient exposure to both (E)JES and SDSF universally agree that SDSF is the inferior product with a lot of "catching up" to do. The individuals making those observations tend to concentrate almost exclusively on ease-of-use and end-user functionality not available in SDSF.

Some of these functions include:

- Point-and-shoot sorting of columns. Point to column heading and press Enter to sort ascending, press Enter again to sort descending, and again to return to the original sort. The column underscore denotes directionality (See Figure 1: The display is sorted by JobName in ascending order).

Figure 1. Example of the Activity Display showing intelligent column formatting and point-and-shoot sorting

```

Jobs Resources Devices Tools Filter View Options Help
-----
ACTIVITY PHXHQ2(S60)   Paging 0   SIO 6   CPU 6/6                               Row 33 of 288
Command ===>                               Scroll ===> CSR
Cmd JobName  Real    CPU-Time                               Paging ExCP   CPU%   CPU-Utilization-
<-- ^^^^^^^^/----->
   BDT      548KB 00-00:00:00.04                               .00   .00
   BPXAS    708KB 00-00:00:00.00                               .00   .00
   BPXAS    748KB 00-00:00:00.00                               .00   .00
   BPXOINIT 756KB 00-00:00:57.51                               .00   .00
   BPXOINIT 756KB 00-00:01:27.84                               .00   .00
   BPXOINIT 776KB 00-00:01:16.43                               .00   .00
   CATALOG  2MB  00-01:13:31.43                               .00   .14   .08 *
   CATALOG  5MB  00-01:22:11.91                               .00   .00   .08 *
   CATALOG  4MB  00-00:35:38.94                               .00   .00   .07 *
   CICSA    34MB 00-00:00:27.49                               .00   .00   .04 *
   CONSOLE  3MB  00-00:21:06.19                               .00   .00   .01 *
   CONSOLE 15MB 00-00:15:04.77                               .00   .00   .01 *
   CONSOLE 18MB 00-00:18:16.11                               .00   4.95   .27 *
   C1CONDOR 38MB 00-00:00:15.32                               .00   .00   .00
   C3CONDOR 8MB  00-00:00:03.44                               .00   .00   .00
   C4CONDOR 61MB 00-00:07:50.85                               .00  13.26  1.54 *
F1=Help   F2=Split  F3=Exit   F4=Return  F5=RfindP  F6=Book   F7=Up
F8=Down   F9=Swap   F10=Left  F11=Right  F12=Cancel

```

- Intelligent column formatting for dates, time values (both absolute and elapsed), memory amounts, numeric values, etc. For example, the Real Memory column in Figure 1 calculates KB, MB, GB, TB, PB, or EB, whereas SDSF just lists the number of frames in use. Also (E)JES shows all accumulated CPU values in a self-adjusting format that—depending on arranged field width—details days, hours, minutes, seconds, and fractions. In contrast, SDSF formats the number of seconds, leaving further calculations as an exercise for the user (See the CPU-Time column in Figure 1).
- A line below the column titles that serves as a visual aid using graphics escape characters (on terminals that support them) on all tabular displays. The number of characters for each column marks the column width while arrows denote custom sort directions. A “<” at the left end of this line indicates more data can be seen by scrolling left and a “>” at the right signifies additional data to the right of the display. (See Figure 1 above.)
- Custom sorting using up to four columns, not two.

- Default sorting that affects all job-oriented panels. For example, 'sort time d' puts most recent jobs first.
- Browser FIND with ISPF-like positioning, cursor emphasis, FIND match highlighting, picture strings, and RFINDP function.
- Tabular display FIND that looks in all columns, not just the first one.
- ATTN key that breaks long FIND operation—no FINDLIM.
- Powerful search engine (pattern filtering) that can be applied to any sysout. Store and retrieve different pattern sets in a library.
- Tabular SELECT command that works like ISPF.
- Much more powerful filtering options. For example, 'fil ctrue;fil maxc=ab* or;fil +maxc=jcl*' makes jobs with abends & JCL errors appear highlighted in green on the STATUS display.
- Many more global filters (eight job names, eight owners, etc.)
- Job type filtering (e.g., OJOB, NTSU) applies consistently to all job- and group-oriented displays, not just DA. Temporary filters can be used for one display invocation only.
- "Snap 1st scrollable column" feature prevents columns from being "chopped" at the scroll point when scrolling tabular displays horizontally.
- Arrangement control for first column and scroll point as well as all other tabular columns.
- Better vertical scrolling functions (e.g., BOTTOM command makes viewing "live" data easier).
- Settings notices that provide an at-a-glance comprehensive summary of filters, options and view settings that might effect the current display.
- ISPF-like label support in all browsers.
- Enhanced multiple data set browsing capabilities including box mode for viewing data sets one at a time using PREV and NEXT to navigate between data sets.
- Scrollable fixed-width displays.
- Overtyping capability for more columns on data set level displays (e.g., DSHOLD or DSOUT).
- CPU utilization percentage on Process Status (PS) display for finding the heavy z/OS UNIX consumers.
- Ability to send arbitrary signals to z/OS UNIX processes, such as HUP to a DNS (See Figure 2)

Figure 2. Available line commands for the z/OS UNIX process display

```

Jobs Resources Devices Tools Filter View Options Help
-----
PSTATUS PHXHQ2(S60) Row 1 of 77
Command ==> Scroll ==> CSR
Cmd JobName JobID Status Owner ASID State CPU%
-----
BPX0INIT STC FileSysKrnlWait/Swap OMVS 002F MFI .00
+----- Line Commands -----+
| = Repeat UD Dump (SIGDUMP) | TCPIP 0024 1R .00
| == Block repeat UK Kill (SIGTERM) | TCPIP 0022 1R .12
| Ka MVS Cancel* UZ Force (SIGKILL) | TCPIP 0022 1R .12
| KDa MVS Cancel/Dump* UZZ Super kill | TCPIP 0022 1F .12
| | Unn Send any signal* | TCPIP 0022 1F .12
| | Uxx Send any signal* | WEBSRV 0038 HK .08
| * a A=ARM restart | SYSOPER 00AD HG .08
| * nn signal number (1-39) | SYSOPER 002B HK .00
| * xx signal name (e.g. HUP) | TCPIP 005D HKI .00
+-----+
| TCPIP 0062 HF .01
RXSERVE S0193771 FileSysKrnlWait/Swap TCPIP 005A 1FI .00
TN3270 S0193767 Running TCPIP 005C 1R .06
TN3270 S0193767 Running TCPIP 005C 1R .06
TN3270 S0193767 Running TCPIP 005C 1R .06
F1=Help F2=Split F3=Exit F4=Return F5=RfindP F6=Book F7=Up
F8=Down F9=Swap F10=Left F11=Right F12=Cancel

```

- Extended help on all messages; PF1 works as expected for an application under ISPF.
- Pop-up help windows for every tabular column that explain the column's contents. Overtypable columns have a second pop-up showing possible input choices (See Figure 3).

Figure 3. Pop-up help window for overtypable Disposition column

```

Jobs Resources Devices Tools Filter View Options Help
-----
HOLD 365S 90J 5T 3,261,179 Records (0 Sched) Row 1 of 109
Command ==> Scroll ==> CSR
Cmd JobName JobID Status MaxComp C Pri Dest ODisp Records
-----
DAVEJES2 A0000026 QUEUED C 96 DEST HOLD 38,942
DAVEJES2 A0000017 QUEUED C 144 REMOTE HOLD 1,492
QUEUED C 144 REMOTE HOLD 1,492
QUEUED C 144 REMO +--- Output Disposition ---+
QUEUED C 144 REMO | "HOLD", "KEEP", "LEAVE", |
| "PURGE", or "WRITE". |
TESTSPNJ JOB06043 QUEUED AB S0C1 H 144 LOCA +-----+
TESTSPNJ JOB06045 QUEUED AB S0C1 H 144 LOCA
TESTSPNJ JOB06047 QUEUED AB S0C1 H 144 LOCAL HOLD 99
EEJJOBNA EEJJOBID QUEUED H 144 LOCAL HOLD 3
TESTSPNJ JOB06049 QUEUED AB S0C1 H 144 LOCAL HOLD 99
EEJJOBNA EEJJOBID QUEUED H 144 LOCAL HOLD 3
QUEUED H 128 LOCAL HOLD 4,000
EJWSJES3 A0000012 QUEUED P 144 LOCAL HOLD 108
EJWSJES3 A0000024 QUEUED P 144 LOCAL HOLD 1,169
EJWSJES3 A0000020 QUEUED T 144 LOCAL HOLD 1,169
D10JHM1P JOB88822 QUEUED CC 0000 H 144 LOCAL HOLD 259
F1=Help F2=Split F3=Exit F4=Return F5=RfindP F6=Book F7=Up
F8=Down F9=Swap F10=Left F11=Right F12=Cancel

```

- Much more powerful extract functions, including automatic derivation of DCB attributes, carriage control translation, dynamic output descriptors, extract of tabular display contents, etc.

Performance

You would think increased functionality and usability would diminish performance. In fact, (E)JES outperforms SDSF in an “apples to apples” comparison. This is because the (E)JES development team focuses on performance. Nearly every release contains key performance enhancements often as a result of leveraging the latest IBM System z and z/OS technologies.

Resource Utilization

SDSF uses JES-provided facilities to gather its information. Because of this design choice, some batch and/or TSO/E end user-requested processing might actually run under work units dispatched within the JES address space. Depending on the priority of your JES subsystem, this processing might compete with, or possibly even preempt, important work on the system such as that being performed by transaction or database servers.

(E)JES gathers all information directly from work units dispatched within the end user’s address space. No JES work units are used.

SDSF gathers information independently for each user request via JES services. On remote systems, such as on a JES3 local processor, (E)JES gathers information using the CAS address spaces. (E)JES uses sophisticated request/response “coat tailing” algorithms to minimize cross-system activity for XCF and XES. This process improves efficiency by reducing the number of signals sent over coupling links and the CPU consumption associated with satisfying each request independently.

On most requests, nearly 100% of the CPU resources used by EJES are eligible for redirection to an available System z Integrated Information Processor (zIIP). This results in faster (E)JES performance for sub-capacity systems because zIIP engines always run at full speed. Another advantage to zIIP redirection is that these processors do not incur capacity-based software license charges. SDSF does not support zIIPs.

The I/O channel programs (E)JES uses for accessing SPOOL, the CAS server global data set (for JES3), and all TP Monitor proprietary files use advanced channel programming techniques. (E)JES uses High Performance FICON (zHPF) if available. If ZHPF is not available (E)JES uses the highest-performing Command Mode channel program currently available. These channel programs perform I/O significantly faster than JES SPOOL I/O services and any other “ordinary” channel programs.

Performance Testing

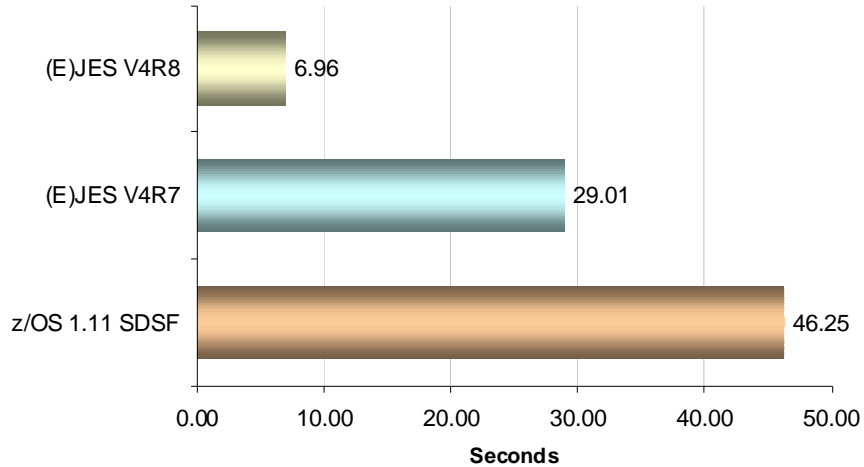
The test below compares SDSF in a z/OS 1.11 JES2 environment to both (E)JES V4R7 (the past release) and (E)JES V4R8 (the current release).

- Enter SDSF. Type 'PREFIX **', 'OWNER', 'DEST', 'APPC ON', and 'H ALL'.
- Begin measuring CPU. Press <Enter> 100 times.
- Note CPU start time and end time and record the difference.

- Enter (E)JES. Type 'SRESET', 'YALL', 'HOLD' and 'SORT OFF'.

- Begin measuring CPU. Press <Enter> 100 times.
- Note CPU start time and end time and record the difference.

Figure 5. (E)JES/SDSF performance comparison: Refresh Hold display



(E)JES performs better than SDSF in almost every case using out-of-the-box settings. In the specific test scenario shown above, on the HOLD panel using a 72x80 display under z/OS 1.11, (E)JES V4R7 was 37% faster and (E)JES V4R8 was 85% faster.

Additionally, our sample tests show SDSF to be more resource-intensive when browsing or searching through data in most cases. For reference, here are some timings using SDSF and (E)JES V4R8 in a z/OS 1.11 JES2 environment:

Timing Test	SDSF		(E)JES V4R8	
	ExCP	CPU	ExCP	CPU
Syslog top then locate 20 times	2,960	1.89	120	1.37
Browser FIND on large data set 3 times	1,761	23.1	1,088	10.12
Scroll Operlog 'DOWN 9999' ten times	0	24.27	0	4.31

NOTES:

- All test results in this document are examples and are not guaranteed. Your results may vary.
- In order to properly compare resource utilization in the two products consider the following:
 - SDSF offloads work to JES2 and/or JES3 to process some requests. Some of the CPU time and EXCP's used by JES will not be charged to the TSO user, especially under JES3. These values must be measured and aggregated.
 - Any comparisons performed should be consistently run in either TSO foreground or under ISPF in order to get valid comparison data. ISPF adds a layer of overhead that will add to resource utilization.

- The number of rows and columns of data displayed and the particular fields displayed should be the same in both tests as these factors affect resource utilization.
- If zIIPs are online they should be varied offline. SDSF does not support zIIPs and it would be unfair for (E)JES to utilize an underutilized zIIP, which may be capable of higher MIPS and which would report back a normalized CPU time rather than an actual CPU time.

Other Benefits

(E)JES exploits 64-bit (aka above the "bar") storage while SDSF still uses only 31-bit storage. For obvious reasons, z/OS tends to allocate more real storage to 64-bit exploiters than to 31-bit exploiters when the storage resource is highly available.

(E)JES also runs in multiple environments in addition to TSO/ISPF, including CICS, as a stand-alone VTAM application, or using a Windows workstation interface. Shops can save TSO resources by offloading users to these other environments.