z/OS

JES3 Introduction

Version 2 Release 1

SA32-1004-00

z/OS

JES3 Introduction

Version 2 Release 1

SA32-1004-00

Note
Before using this information and the product it supports, read the information in "Notices" on page 35.

This edition applies to Version 2 Release 1 of z/OS (5650-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

About this document

This document supports z/OS® (5650-ZOS). This document is intended for any JES3 complex that runs z/OS MVS™.

This document introduces you to the Job Entry Subsystem 3 (JES3).

Who Should Use This Document

Whether you are an installation manager, system programmer, application programmer, operator, student, are currently using JES3, or are just considering or planning a JES3 installation this document will benefit you. Read this document to get a perspective on JES3 and to get the maximum benefit from the other documents in the JES3 library.

Where to Find More Information

The following table lists documents that contain information related to the information contained in this document. The table shows the complete titles, the short titles, and order numbers that are not listed in *z/OS Information Roadmap*. See that document for all z/OS publications.

Most licensed documents were declassified in OS/390® V2R4 and are now included on the z/OS Online Library Collection, SKT2T-6700. The remaining licensed documents appear in unencrypted BookManager® softcopy and PDF form on the z/OS Licensed Product Library, LK2T-2499.

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or provide any other feedback that you have.

Use one of the following methods to send your comments:

1. Send an email to mhvrcfs@us.ibm.com.
2. Send an email from the "Contact us" web page for z/OS (http://www.ibm.com/systems/z/os/zos/webqs.html).
3. Mail the comments to the following address:
   IBM Corporation
   Attention: MHVRCFS Reader Comments
   Department H6MA, Building 707
   2455 South Road
   Poughkeepsie, NY 12601-5400
   US
4. Fax the comments to us, as follows:
   From the United States and Canada: 1+845+432-9405
   From all other countries: Your international access code +1+845+432-9405

Include the following information:

- Your name and address.
- Your email address.
- Your telephone or fax number.
- The publication title and order number:
  z/OS V2R1.0 JES3 Introduction
  SA32-1004-00
- The topic and page number that is related to your comment.
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

Do not use the feedback methods that are listed for sending comments. Instead, take one of the following actions:

- Contact your IBM service representative.
- Call IBM technical support.
- Visit the IBM Support Portal at z/OS support page (http://www.ibm.com/systems/z/support/).

z/OS Version 2 Release 1 summary of changes

See the following publications for all enhancements to z/OS Version 2 Release 1 (V2R1):

- *z/OS Migration*
- *z/OS Planning for Installation*
- *z/OS Summary of Message and Interface Changes*
- *z/OS Introduction and Release Guide*

Chapter 1. Introduction

A major goal of operating systems is to process jobs while making the best use of system resources. Thus, one way of viewing operating systems is as resource managers. Before job processing, operating systems reserve input and output resources for jobs. During job processing, operating systems manage resources such as processors and storage. After job processing, operating systems free all resources used by the completed jobs, making the resources available to other jobs. This process is called *resource management*.

There is more to the processing of jobs than the managing of resources needed by the jobs. At any instant, a number of jobs can be in various stages of preparation, processing, and post-processing activity. To use resources efficiently, operating systems divide jobs into parts. They distribute the parts of jobs to queues to wait for needed resources. Keeping track of where things are and routing work from queue to queue is called *workflow management*, and is a major function of any operating system.

With the z/OS MVS JES3 system, resource management and workflow management are shared between MVS and its Job Entry Subsystem 3 (JES3) component. Generally speaking, JES3 does resource management and workflow management *before* and *after* job execution, while MVS does resource and workflow management *during* job execution.

JES3 considers job priorities, device and processor alternatives, and installation-specified preferences in preparing jobs for processing job output. Features of the JES3 design include:

- Single-system image
- Workload balancing
- Availability
- Control flexibility
- Physical planning flexibility

Single-System Image

JES3 runs on either one processor or up to thirty-two processors in a sysplex. A *sysplex* is a set of MVS systems communicating and cooperating with each other through certain multisystem hardware components and software services to process customer workloads.

In a sysplex, your installation must designate one processor as the focal point for the entry and distribution of jobs and for the control of resources needed by the jobs. That processor, called the global processor, distributes work to the processors, called local processors.

It is from the global processor that JES3 manages jobs and resources for the entire complex, matching jobs with available resources. JES3 manages processors, I/O devices, volumes, and data. To avoid delays that result when these resources are not available, JES3 ensures that they are available before selecting the job for processing.

JES3 keeps track of I/O resources, and manages workflow in conjunction with the workload management component of MVS by scheduling jobs for processing on the processors where the jobs can run most efficiently. At the same time, JES3 maintains data integrity. JES3 will not schedule two jobs to run simultaneously anywhere in the complex if they are going to update the same data.

JES3 can be operated from any console that is attached to any system in the sysplex. From any console, an operator can direct a command to any system and receive the response to that command. In addition, any console can be set up to receive messages from all systems, or a subset of the systems in the sysplex. Thus, there is no need to station operators at consoles attached to each processor in the sysplex.

If you want to share input/output (I/O) devices among processors, JES3 manages the sharing. Operators do not have to manually switch devices to keep up with changing processor needs for the devices.

The JES3 architecture of a global processor, centralized resource and workflow management, and centralized operator control is meant to convey a single-system image, rather than one of separate and independently-operated computers.

Workload balancing

JES3 balances workload among processors in a way consistent with and in conjunction with the workload management (WLM) and system resources management (SRM) functions of MVS by considering the resource requirements of jobs. The method JES3 uses is the same whether one or several processors make up the configuration. Thus, addition of another processor does not mean a new operational and scheduling environment.

Availability

If a problem develops with the global processor, you can have one of the other processors assume the global functions. (Operators have JES3 commands to cause the switch.) Jobs running on other processors, including the one to become the new global processor, will typically be unaffected (except for a waiting period until global activities are resumed).

If part of JES3 fails, JES3 collects failure symptoms, records error data, and attempts recovery. All major JES3 components are protected by at least one special JES3 recovery routine. If recovery is unsuccessful, the failing component gets insulated from the rest of JES3, resources are freed, and the failing component will not be used again. If the component is not critical to overall JES3 processing, complete JES3 failure might be avoided.

Control Flexibility

Operating systems must be easy to control. Internal complexity must be offset by features that make the systems easy to operate, to monitor, and to change. JES3 has designed-in features for operators, application programmers, and system programmers:

- Operators have special JES3 commands. Some commands activate programs to handle I/O devices, while others obtain or change the status of jobs being processed. With multiple processing-unit systems, JES3 operators have less to do

than for an equal number of individual systems, because they can control the entire complex from a central point, and because JES3 decides where and when jobs will be processed.

JES3 applies installation policies, as defined in the JES3 initialization stream, to perform job scheduling, thus freeing the operator from this task.

Even though JES3 handles job flow control, there are operational controls in JES3 for use at the descretion of the operator to override JES3 decisions, and to take control in unusual situations.

- Application programmers have special JES3 control statements (similar to JCL statements). There are control statements to make some jobs run only after successful (or unsuccessful) processing of other jobs, and for specifying the time-of-day, week, month, or even the year when jobs should run.
- System programmers have special JES3 initialization statements to define the way JES3 is to manage devices and jobs. Operators or application programmers can override many of these initialization options on a job-by-job basis. JES3 gives system programmers a unique way of setting installation policy for device and job management. They define separate groups of processing rules. Application programmers select and apply the groups of rules in various ways to individual jobs.

Where JES3 does not provide the exact function that your complex requires, such as special printing requirements, the system programmers can write their own routines and make them part of the JES3 program. This is done through installation exits provided with JES3 and through dynamic support programs that can be added to JES3. For diagnostic purposes, JES3 provides traces and dumps. Traces are copies of data made during normal processing to help monitor activity in the system, and dumps are copies of data made at times of failure.

Physical Planning Flexibility

You can give I/O devices specific assignments, and then put the devices in convenient places. For example, you can put card readers, card punches, and printers where programs get submitted and where output is needed. You can put I/O devices with removable volumes near tape and disk libraries, and you can have tape and disk librarians receive requests for the volumes on nearby consoles. Thus you can put processors into areas free of the congestion that often surrounds peripheral units.

You can define your MVS/JES3 complex so that only appropriate data, such as system messages, is sent to certain specified devices. This device tailoring capability lets you set up your system so that much of the activity at the main operator's console can be reduced.

Chapter 2. JES3 Environments

The work environment for JES3 consists of processors and I/O devices. JES3 covers a range of data processing needs, partly because it can accommodate various combinations of processors and devices. JES3 is a manager of its environment, so if you understand that environment, it will be easier for you to understand JES3.

This topic describes the I/O devices used by JES3 and MVS and the following environments in which JES3 runs:

- Single-processor environment
- Multiprocessor environment
- Remote job processing environment
- JES3 networking environment
- APPC environment

A JES3 complex can involve any combination of these environments.

Single-Processor JES3 Environment

Figure 1 shows JES3 in a single-processor environment, also known as a *single-system sysplex*. Besides the processor, two categories of I/O devices are shown: JES3 devices (those used by JES3); and JES3- and MVS-managed devices (those used by jobs). The spool device is a direct-access storage device (DASD) that is treated in a special way by JES3, so it is shown and explained separately.

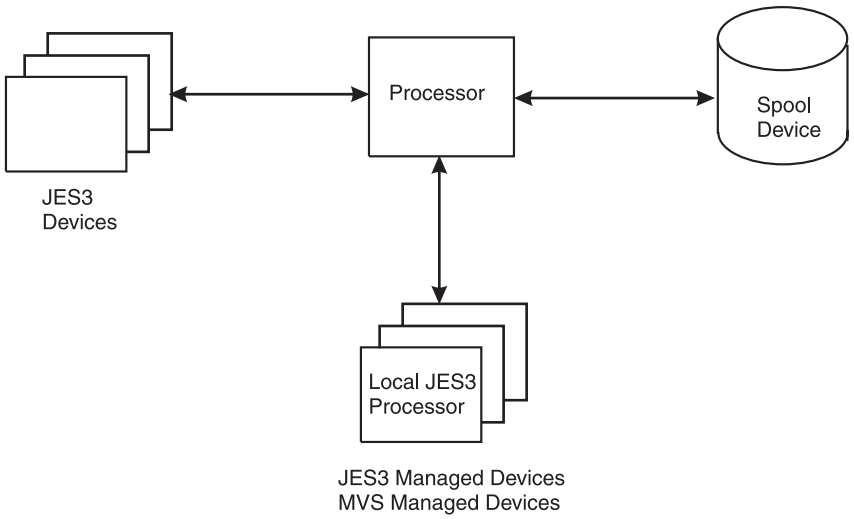

Figure 1. JES3 in a Single-Processor (Single-System Sysplex) Environment

The JES3 Processor

In installations with one processor, JES3 coexists in that processor with MVS and with running jobs. In many respects, JES3 is treated like a job by MVS. (MVS handles JES3 in much the same way it handles the jobs it receives from JES3 for processing). JES3 becomes a processable "job" during initialization, while jobs submitted to the system become processable by the actions of JES3.

Multiprocessor JES3 Environment

Multiprocessing is a way of doing work with two or more connected processors. In a multiprocessing environment, also known as a multisystem sysplex, JES3 allows up to 32 processors, also known as mains, to be configured into the complex. JES3 uses one processor (called the global) to do work and also to distribute work to up to 31 other processors (called locals). Some of the advantages of running JES3 in this environment are:

- Elimination of much of the overhead of scheduling work for and operating separate processors
- Sharing devices by processors, which means that the devices can be used more efficiently
- Movement of work to other processors, should one processor become overworked, need maintenance, or need to be removed from the complex for any reason.

Use of the Spool Device For Shared Data Storage

Most multiprocessing systems use shared data storage. In JES3 the spool device becomes the shared data storage. Thus, besides being a buffer (as for single-processor systems), in multiprocessing systems the spool device becomes a collection point for data to be distributed to individual processors (see Figure 2). Also, the spool device becomes a collection point for data coming from local processors routed to JES3 output devices connected to the global processor (see Figure 3 on page 7).

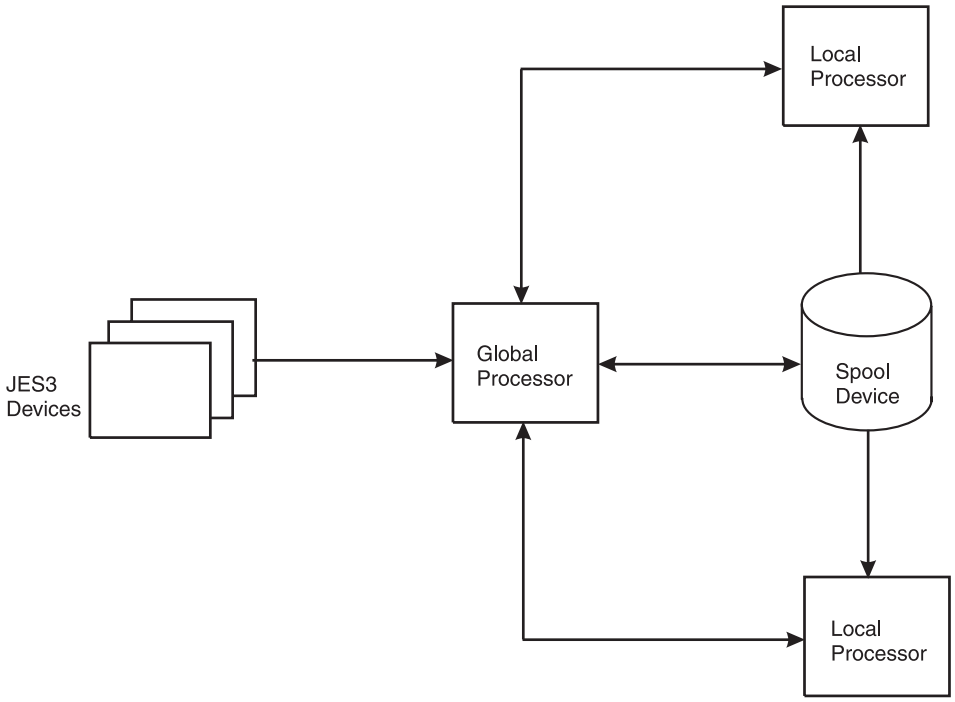

Figure 2. Spooling as a Method of Distributing Work to Processors

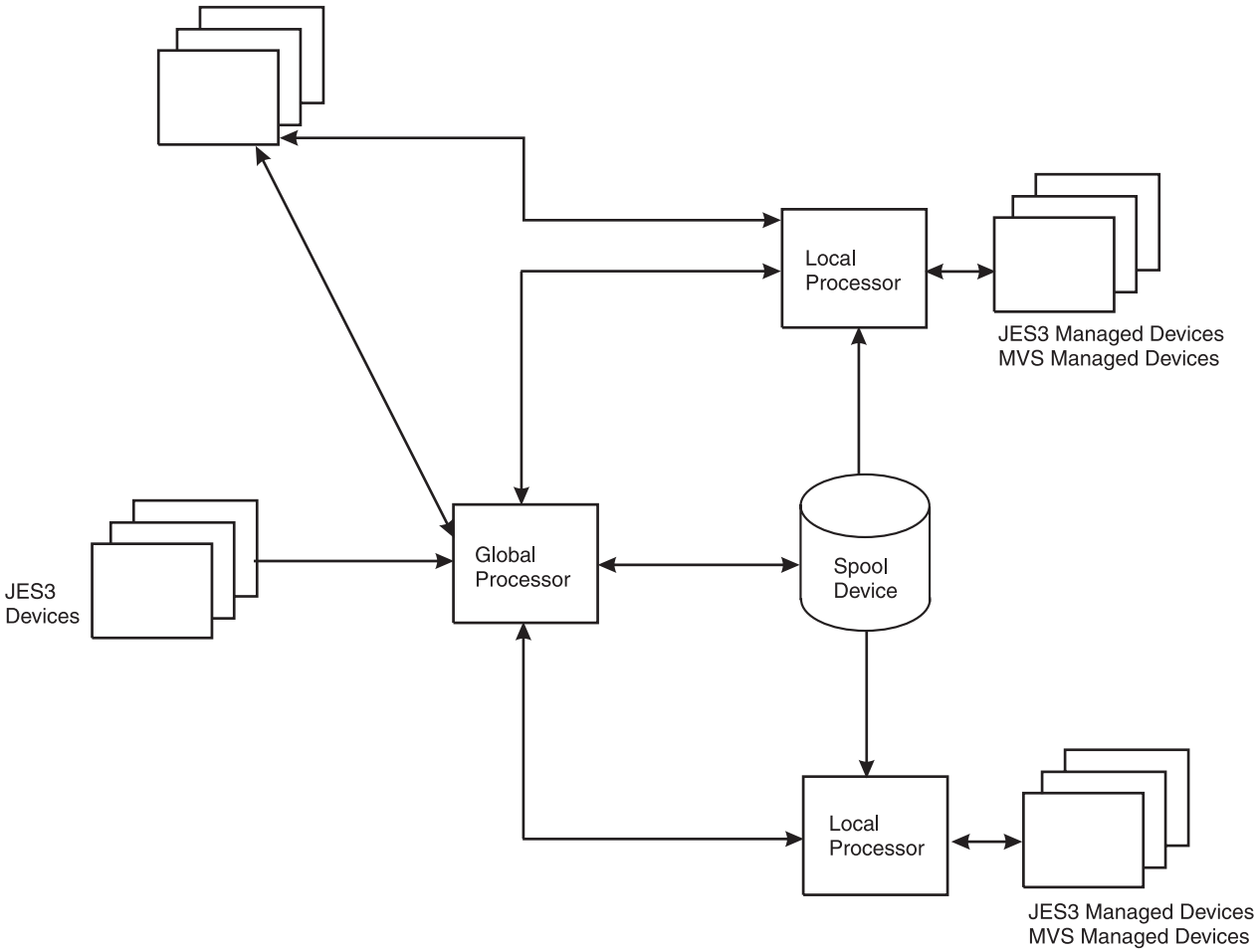

Figure 3. Spooling as a Method of Collecting Job Output

Processor Communications

In a JES3 multisystem sysplex environment, one processor is called the global processor while the other processors are called the local processors. In a JES3 complex there can be a maximum of 32 processors, one global and up to 31 locals. The global processor and the local processors communicate using the MVS cross-system coupling facility (XCF). JES3 devices (the devices from which JES3 reads jobs, and on which JES3 stores jobs awaiting processing, stores job output, and writes job output) are connected to the global processor. Some JES3 devices that write job output can also be connected to the local processors (not shown in Figure 3).

Figure 4 on page 8 shows a JES3 multiprocessing system that has three processors (that communicate through XCF signalling). The global processor runs most of JES3, for it is from the global processor that JES3 allocates resources to jobs and sends jobs to local processors for processing. In addition, JES3 can also pass jobs to MVS in the global processor for processing.

Each local processor has a complete MVS system and JES3 routines for spooling and for communication with the global processor. When MVS in a local processor needs a job, JES3 in that local processor sends the job request to JES3 in the global

processor through XCF signalling. JES3 in the global processor returns identifying information about a job, and JES3 in the local processor uses that information to retrieve the job from the spool device.

If a problem arises on the global processor while JES3 is running, you can transfer global functions to a local processor. This transfer is called dynamic system interchange (DSI). The processor you choose must be capable of assuming the same workload as the previous global.

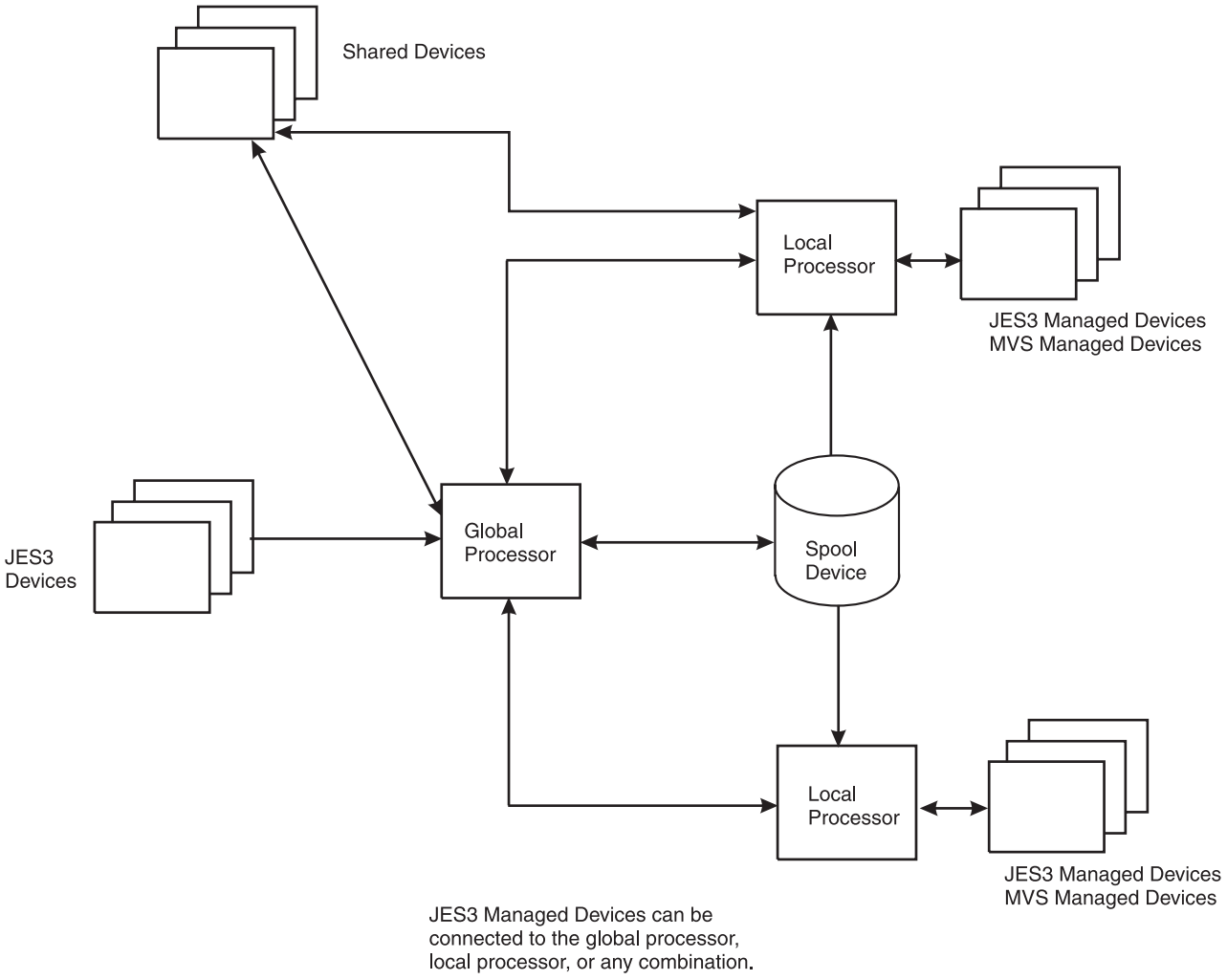

Figure 4. JES3 Multiprocessing in a Sysplex

Devices

In a JES3 complex, the devices in the system configuration are identified to JES3 when the system programmer creates a JES3 initialization stream defining the devices. There are five categories of devices:

- JES3 devices
  - JES3 Devices for Reading Jobs
  - JES3 Devices for Writing Job Output
  - The Spool Device
- JES3-managed devices

- MVS-managed devices
- Jointly-managed devices
- Shared devices

JES3 Devices

JES3 I/O devices are 'owned' by JES3. With these devices, JES3 reads jobs, stores jobs awaiting processing, and writes job output. JES3 devices are typically:

- card readers
- punches
- printers
- magnetic tape devices

System programmers identify JES3 devices during JES3 initialization. While JES3 is running, operators control JES3 devices with JES3 commands.

In JES3 SP5.2.1, the MVS Multiple Console Support (MCS) sysplex operations environment is enabled. MCS function replaces that previously provided by JES3, including:

- JES3 support for operator console devices
- JES3 command routing
- JES3 message formatting and queuing

JES3 continues to provide console support for RJP workstation consoles and networking.

JES3 Devices for Reading Jobs

The individual jobs that are the input to JES3 are combinations of job control language (JCL) statements, data accompanying the JCL statements, and JES3 control statements. The set of jobs to be processed is called an input stream. JES3 can read input streams from card readers, magnetic tape units, or magnetic disk units. (There are JES3 programs called *utility programs* that operators can use to transfer input streams or any other kinds of data from one storage medium to another.)

Jobs themselves can create input streams. That is, the output of one job could very well be an input stream containing other jobs. This eliminates the need to reenter job-created input streams in the conventional manner using an I/O device. A special *JES3 internal reader program* can transfer job-produced input streams directly to the JES3 devices for reading jobs.

JES3 Devices for Writing Job Output

Data produced by jobs during processing is called *system output data*. JES3 stores system output data temporarily on the spool device. After job processing JES3 prints or punches the system output data to the appropriate output device, which was already identified during system initialization. Operators have JES3 commands for controlling the writing of system output data.

Supporting Advanced Function Presentation (AFP) Printers

Advanced function presentation printers (AFPs), such as the 3900 and 3827, provide all points addressability. All points addressability allows every point on the page to be available for reference. These points on the page include graphics and a range of font types and sizes unavailable to impact printers.

AFP printers are controlled by Print Service Facility (PSF), which interacts with JES3. JES3 and PSF work together by communicating through the functional subsystem interface (an extension of JES3 device control that removes the need for JES3 to be dependent on specific characteristics of the printing device). This independence allows JES3 to use AFP printers in **full function mode** and **page mode**.

Full function mode is defined as using those functions of the printer that produce page mode output. The concept of **page mode** permits printed pages to contain both text data and graphic presentations. The user can define and request attributes such as **segments** (predefined portions of a page), **overlays** (predefined page templates), **images** (pictures and graphics), and **type fonts** (collections of unique or stylized characters). For example, the graphical material can include a wide range of print font types and sizes for use in text headings, logos, and imbedded artwork; shading of textural and user-produced graphics; and graph plotting, some of which you can see in this book. [1]

The Spool Device

In a system that has one processor, the spool device serves:

1. as a buffer between input devices and routines that read input data, and between routines that write output data and output devices
2. as a storage place for the control blocks and data that JES3 builds to process jobs.

A spool device must be a DASD. Transfer of job data to and from the spool device is called *spooling*. (One spool device is shown in Figure 1 on page 5, but more might be used.) *Spooling* refers only to use of a DASD(s) by JES3 for storage of jobs and job-related data. Use of the same device(s) for other purposes is *not* spooling.

Spooling allows reading or writing to take place continuously at or near the full speed of I/O devices. Without spooling, there would be frequent delays (and processing overhead) during reading or writing of data.

For example, JES3 would have to read a job, process the job, and then wait while the job reads its input data (if there is any) before reading the next job. With spooling, JES3 reads an input stream containing many jobs and their input data, and sends the input stream to the spool device. JES3 then retrieves jobs individually and processes them, returning the jobs to the spool device to await processing. During processing, jobs read their input data from the spool device and write their output data to the spool device, thereby running faster.

JES3-Managed Devices

JES3-managed devices are the devices that JES3 allocates to jobs. Jobs use these devices during processing. As they do for JES3 devices, system programmers identify JES3-managed devices during JES3 initialization.

MVS-Managed Devices

MVS-managed devices are the devices that MVS allocates to jobs. Jobs use these devices during processing just like they use JES3-managed devices. System programmers do not identify MVS-managed devices during JES3 initialization. Rather they must follow MVS rules and identify the devices to MVS during MVS

1. All the illustrations and headings in this book are imbedded within the text files, and all print as a single image on IBM's AFP printers.

initialization. Because JES3 is unaware of MVS-managed devices, JES3 ignores them when allocating devices to jobs. Thus the devices will be left for allocation by MVS.

As you can see, you can choose whether you want JES3 to allocate I/O devices. At one extreme, you could have JES3 allocate all devices to jobs before it forwards the jobs to MVS for processing. At the other extreme, you could have JES3 forward jobs to MVS without allocated devices, and MVS would allocate devices to the jobs. Or, you could have JES3 allocate some devices to jobs, and have MVS allocate others. The key is whether system programmers define the devices as JES3-managed devices or MVS-managed devices during the JES3 initialization process. (JES3 allocates devices to jobs, while MVS allocates devices to job steps.)

Jointly-Managed Devices

DASDs with volumes that you cannot physically remove (called permanently-resident devices) can be managed jointly by JES3 and MVS. You could have JES3 allocate these to some jobs and have MVS allocate them to other jobs. System programmers should define these as *both* JES3-managed and MVS-managed devices.

Shared Devices

If an I/O device is not a DASD, it can be both a JES3 device or a JES3-managed device. Then you could have JES3 use the device for special purposes (such as for labeling tapes), and jobs could use the device the rest of the time. System programmers would define shared I/O devices by making them *both* JES3 devices and JES3-managed devices.

Remote Job Processing JES3 Environment

With remote job processing (RJP), you can submit work to JES3 from locations significantly distant from the JES3 global processor. The points of origin for RJP jobs are called workstations (see Figure 5 on page 12). A workstation can be a single I/O device, a number of separate devices, or one of the allowable processors with its devices.

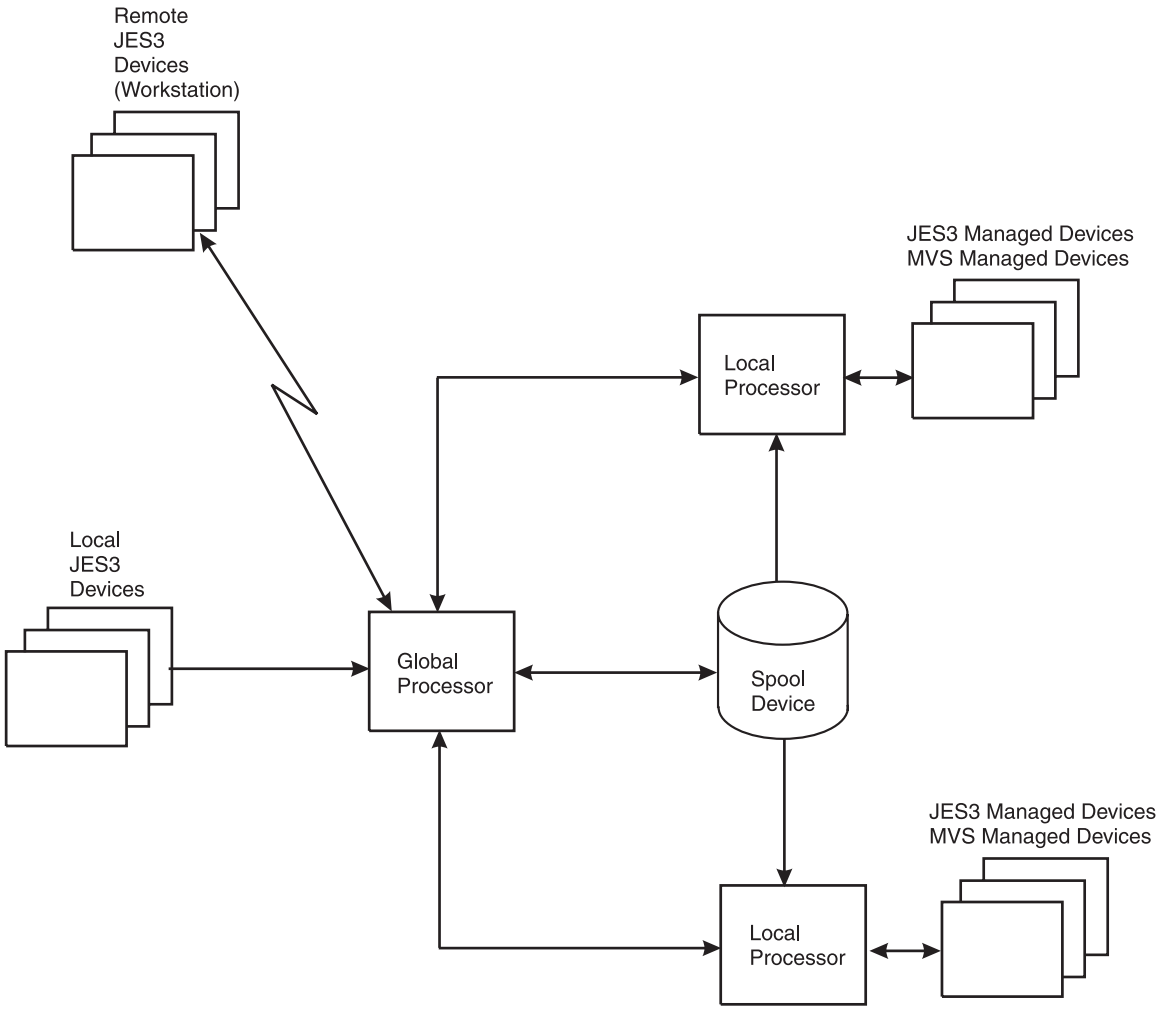

Figure 5. Remote Job Processing with JES3

Data travels between workstations and the JES3 global processor over communication lines or adapters that substitute for communication lines. JES3 processes the jobs it gets from workstations as if it had received the jobs locally. JES3 can write output of remotely-entered jobs on local devices or it can transmit the output to the originating workstation. JES3 can also transmit output to any other workstation connected to the global processor.

With communication lines, special rules exist to identify:

- the direction of data flow
- where data begins and ends
- how much data is being transmitted
- whether data or control information is being sent

Additionally, precautions have to be taken to ensure that data is not altered in transit. There must also be a means of identifying work stations and for routing data to the workstations. The rules for doing this are called protocols. JES3 offers installations two kinds of protocols:

- Binary synchronous communication (BSC) multi-leaving protocols, where a separate communication line is needed for each device at a workstation.

- IBM® systems network architecture (SNA) protocols, where many devices can share a communication line.

Remote job processing with BSC multi-leaving protocols is called BSC remote job processing (RJP), and that with SNA protocols is called SNA remote job processing (RJP).

BSC Remote Job Processing

Features of BSC remote job processing include:

- Password protection. You can restrict access to an RJP line, a device by using an 8-character password, or both.
- Message routing. JES3 sends job-started, job-ended, and abnormal-ending messages to workstations where the corresponding jobs originated.
- Support for programmable and non-programmable devices. JES3 handles programmable devices in an interleaved manner. (There can be concurrent activity for up to 15 devices at a workstation: seven input devices, seven output devices, and one console.) JES3 handles non-programmable devices in a non-interleaved manner. (Only one device at a time can be active at a workstation.)
- Output suspension for non-programmable devices. An operator can interrupt printing to use a card reader.
- Compressed data support. JES3 accepts compressed inbound data (compression is removal of repeated characters from data).
- Connection through leased or dial-up communication lines.
- Remote console support. You can make a device at a workstation a full-function console, a console only for work originating at that workstation, or an inquiry-only console. If one of the devices typically used for consoles does not exist at a workstation, an operator can enter commands on a card reader and receive messages on a printer.
- Inquiry by data set origin or destination. An operator at a work station can find out if there is output for a workstation by specifying the data set origin or destination in a command. An operator at a local console can inquire about any workstation.
- Error recovery and error statistics. JES3 automatically attempts error recovery and creates reports of line error statistics.
- Message queuing for signed-off consoles. If a console at a workstation cannot accept a transmission, JES3 will optionally hold the messages for later transmission.
- System management facilities (SMF) recording. JES3 creates SMF records for every workstation signon, signoff, and line start. (SMF is an MVS feature that supplies processing data useful for purposes such as job accounting.)
- Interface with remote terminal processor (RMT) programs. These are programs for managing devices connected to processors at workstations.

SNA Remote Job Processing

Features of SNA remote job processing include:

- Password protection. You can restrict access to an RJP workstation by using an 8-character password.
- Message routing. JES3 sends job started, job ended, and abnormal termination messages to workstations where the corresponding jobs originated.
- SNA data transmission protocols.

- Connection through leased or dial-up communication lines.
- Multiple logical unit support. You can have concurrent data transmission between JES3 and the console, readers, and printers at a workstation.
- Compression and compaction. JES3 can compress inbound data, and it can compress or compact outbound data. Compression is removal of repeated characters from data; compaction is substitution of characters according to installation-specified rules.
- ASCII support. JES3 can handle ASCII or EBCDIC character sets, but does not compress or compact ASCII data.
- Block size selection. Remote workstations operators can choose compression, compaction, the ASCII character set, and block sizes during logon.
- Remote console support. You can make a console at a workstation a full-function operator console, a console only for work originating at that workstation, or an inquiry-only console. If one of the devices typically used for consoles does not exist at a workstation, an operator can enter commands on a keyboard and receive messages on a printer.
- Inquiry by data set origin or destination. An operator at a workstation can find out if output for the workstation by specifying a data set origin or destination in a command. An operator at a local console can inquire about any workstation.
- Error recovery and error statistics. JES3 automatically attempts error recovery and creates reports of line error statistics.
- Message queuing for signed-off consoles. If a remote console cannot accept a transmission, JES3 will optionally hold its messages for later transmission.
- System management facilities (SMF) recording. JES3 creates SMF records for every workstation signon, signoff, and line start. (SMF is an MVS feature that supplies processing data useful for purposes such as job accounting.)

JES3 Networking

JES3 network job entry (NJE) allows JES3 users at one location to send jobs to another JES3 location for execution, to send output (SYSOUT) data to another JES3 location for processing, and to send jobs, commands, and messages to another JES3 location or to a non-JES3 location.

Therefore, JES3 can become a part of a network comprised of a Job Entry Subsystem 2 Network Job Entry (JES2 NJE) configuration, a Virtual Machine/Remote Spooling Communications Subsystem (VM/RSCS) configuration, a VSE/POWER system, as well as other JES3 complexes.

Note: Networking is not to be confused with remote job processing (RJP), since the concept of nodes does not really apply in an RJP environment. Users are connected to the computer from a remote distance, but are connected directly and not through another computer serving as a node. For information about RJP, see section “Remote Job Processing JES3 Environment” on page 11.

Each complex in the network is called a **node** and is identified by a unique node name. System programmers define nodes by name during JES3 initialization. The name of a JES3 node will always represent a global processor *and* all of its local processors within the JES3 complex. (The only time a node name can represent a single JES3 processor is when JES3 at that node has no local processors.)

When defining a JES3 networking environment, your JES3 complex becomes the *home node* and part of a job entry network that can include one or more additional systems or complexes - the *remote nodes*. Nodes communicate with each other on an equal basis.

Some features of JES3 networking are:

- Store-and-forward. The spool device at each node holds jobs in transit. JES3 automatically routes jobs to their destinations, passing them from node to node when necessary. (JES3 delays forwarding until an entire job reaches a node.) At each node, JES3 checks whether a job is to be passed on, so all a node has to 'know' is where the job goes next.
- Output and job routing. JES3 can send output and jobs to one or more nodes.
- Operator commands for network control. Operators can alter the status of nodes or change routing tables. Also, operators can enter commands at any node, and JES3 will pass the commands to other nodes for processing. JES3 will return responses to commands to consoles at originating nodes. Also, operators can use commands to add or delete a node from the network.
- Data compression. JES3 removes repeated characters from data to improve transmission efficiency.
- Collecting accounting information. JES3 collects job-forwarding accounting data at each transmitting node.
- Tracing facilities. Problem diagnosis is made easier because JES3 traces (creates records of) transmission activity.

A way you can link systems to form a network is shown in Figure 6 on page 16. Data can travel between nodes over binary synchronous communications (BSC) lines, over systems network architecture (SNA) communication lines, or over transmission control protocol/internet protocol (TCP/IP) communications lines.

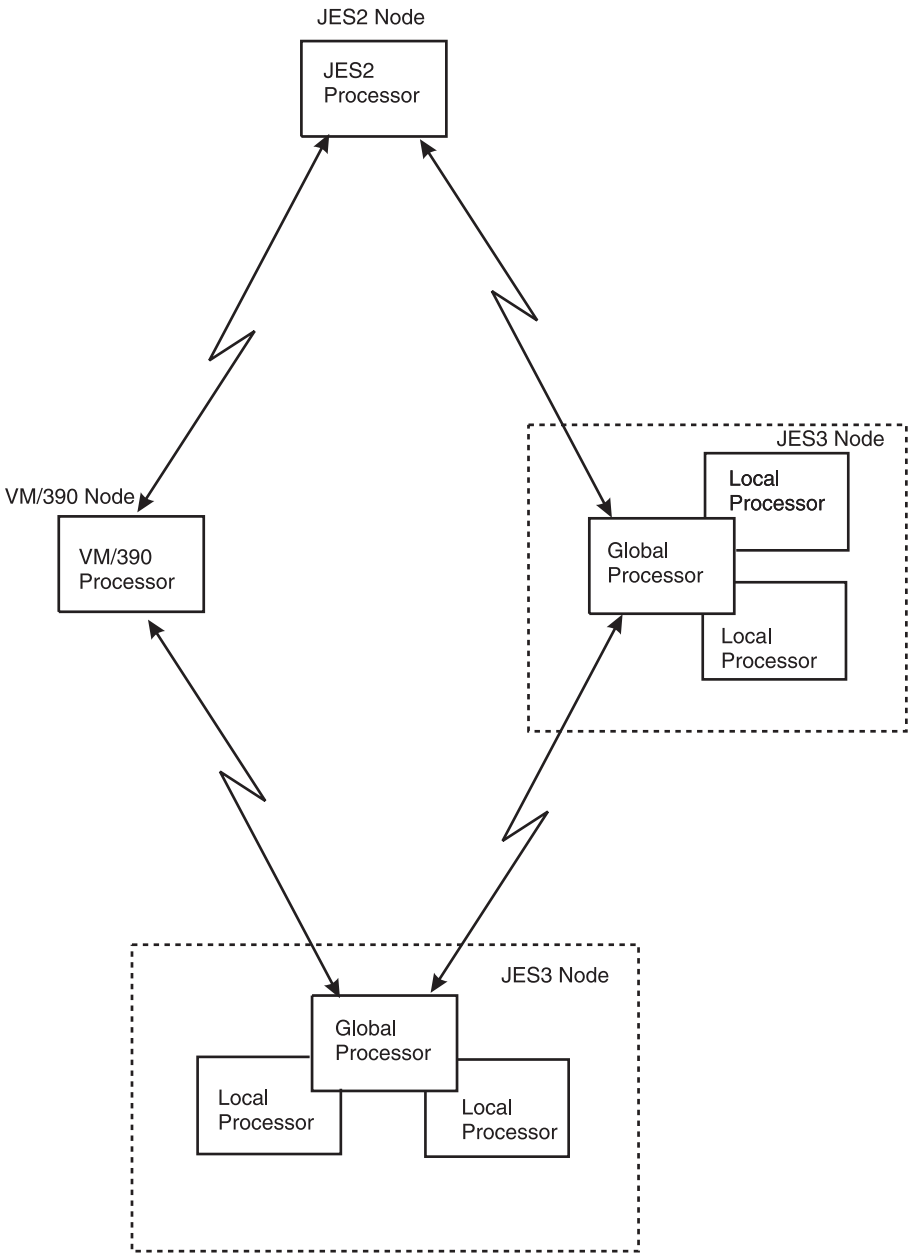

Figure 6. A Network Environment with JES3 and Other Systems

JES3 Networking Protocols

JES3 provides three types of networking protocols, binary synchronous communication (BSC) protocols, systems network architecture (SNA) protocols, and transmission control protocol/internet protocol (TCP/IP) protocols. Each type enables a complex to participate in a data communications network, and to pass jobs, commands, messages, and system output (SYSOUT) data between nodes in that network.

The JES3 SNA/NJE protocol in combination with MVS/BDT Version 2 provides a JES3 complex with systems network architecture/network job entry (SNA/NJE) capability.

Figure 7 shows an overview of JES3 networking where four JES3 complexes (or nodes) are connected by teleprocessing lines using both SNA and BSC protocols.

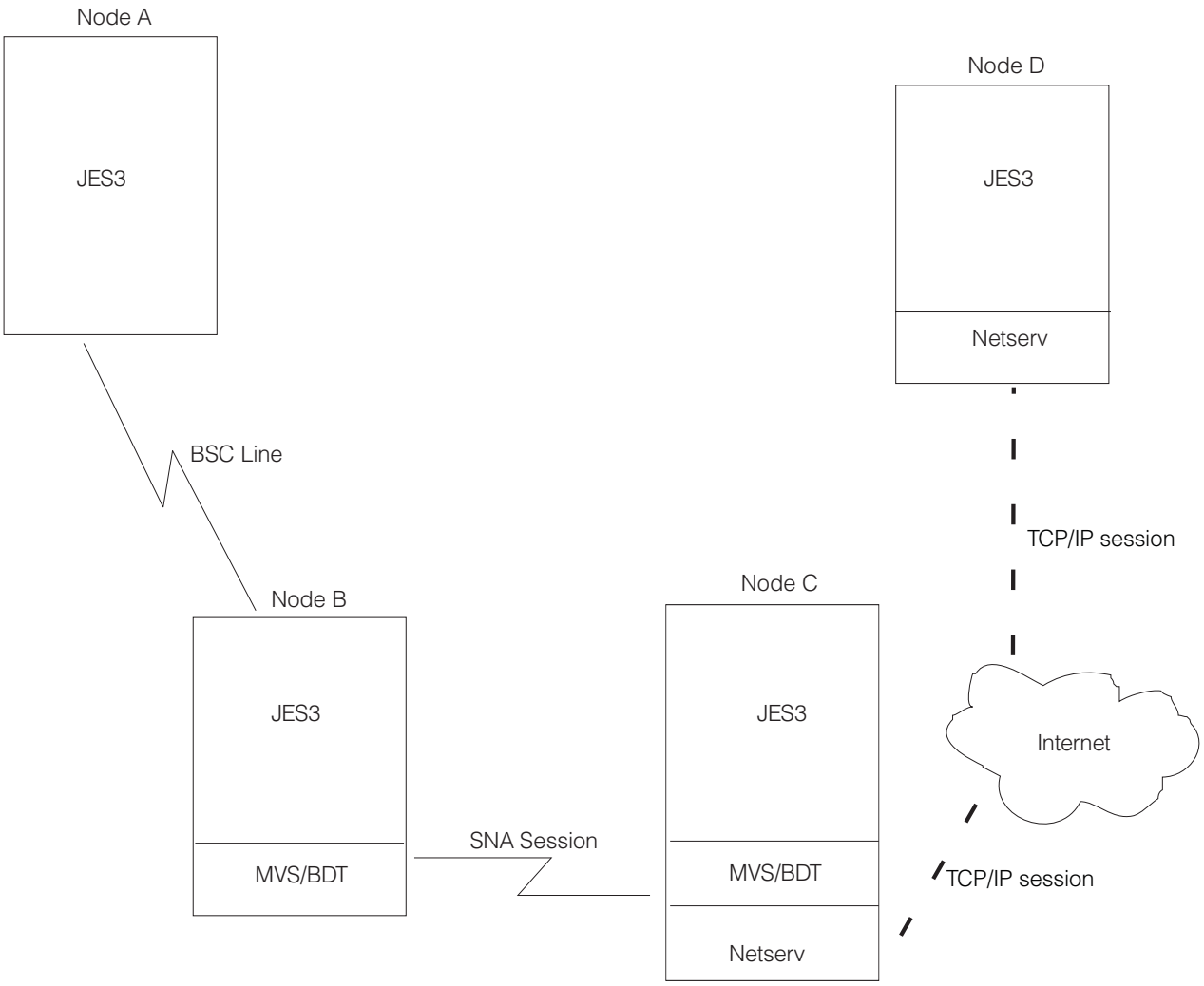

Figure 7. Overview of JES3 Networking with SNA, BSC and TCP/IP Protocols

The JES3 networking capability allows a JES3 complex to use SNA, BSC or TCP/IP communications protocols to:

- Send a job to another node in the network to be processed.
- Receive a job from another node in the network and either process the job or send it to another node.
- Send the SYSOUT data produced by a job to another node in the network.
- Receive SYSOUT data sent from another node and either store or print the data or send it to another node.
- Reroute jobs or SYSOUT data to another node in the network.

A JES3 complex that uses the BSC/NJE, SNA/NJE, or TCP/IP/NJE protocol might also communicate with nodes that use one of the three protocols. This means that the NJE network might consist of SNA/NJE nodes only, BSC/NJE nodes only,

TCP/IP/NJE only or a mixture of the two or three. Installations that plan to convert nodes from BSC protocol to SNA or TCP/IP protocol need not convert all nodes at the same time.

When an end user submits an NJE job, the user does not need to know the type of communications protocol that JES3 will use to send the job to another node. JES3 determines the proper routing and proper protocol.

With the MVS XMIT JCL statement, a user can send non-MVS JCL (such as VSE/Power JECL), as well as MVS JCL statements, to another node for processing.

Support for Advanced Program-to-Program Communication (APPC)

z/OS MVS supports Advanced Program-to-Program Communication (APPC). APPC allows MVS systems to participate in a cooperative processing environment. APPC is an implementation of the Systems Network Architecture (SNA) LU 6.2 protocol that permits interconnected systems to communicate and share the processing of programs.

APPC/MVS is a Virtual Telecommunications Access Method (VTAM®) application that extends APPC support to the z/OS MVS operating system. Although APPC/VTAM previously provided some LU 6.2 capability, APPC/MVS, in cooperation with APPC/VTAM, provides full LU 6.2 capability to programs running in MVS. With APPC/MVS, installations can create cross-system applications that exploit the unique strengths of different computer architectures.

Application programs that use APPC communication calls to perform cross-system requests are called **transaction programs**. Transaction programs can communicate with other transaction programs that are on local or remote systems.

JES3 supports APPC/MVS. As a result, APPC transaction programs can exploit certain JES3 functions, such as JES3 output service processing and complex-wide data set awareness.

Transaction programs do not run as JES3 jobs; instead, transaction programs are associated with APPC initiators, which are long-running jobs that can sponsor many transaction programs. APPC initiators utilize JES3 functions on behalf of transaction programs.

Chapter 3. JES3 from a system operator point of view

Most installation standards for how JES3 is to manage devices and jobs are defined in the JES3 initialization stream created by system programmers. Some initialization statements define system input devices and system output devices. Others tell JES3 where to read jobs from, how to schedule the jobs for processing, and where to write data produced by the jobs.

There could be times when operators might want to change what system programmers specified on JES3 initialization statements. For example, there might be times when printers or other devices should be made unavailable to JES3 or when jobs should be canceled. Or, an operator might want to get the status of a job or device, and then modify that status.

Operators can alter JES3 processing by using JES3 operator commands. Console authority levels govern which commands or sets of commands operators can enter at specific consoles. Installations can control commands that are allowed from specific consoles by using facilities such as SAF authorization checks and RJP console authority levels.

JES3 Operator Commands

All JES3 operator commands consist of (1) a command verb (commonly called the command name), (2) the name of the affected device, processor, JES3 component or job, and (3) parameters for indicating what is to be done. There are two general types of JES3 commands:

- Commands for getting status information
- Commands for changing the status

JES3 operators need not enter commands repetitively. A single command will get the status of a specific device, devices with certain volume serial numbers, all JES3-managed devices, or all devices. Likewise, a single command can put a specific device online or offline, or it can affect all devices within a range of device numbers, all devices for a specific device control unit, or all devices.

Resource Management for Operators

JES3 operators can change I/O device assignments. Requests for these kinds of changes typically come from system programmers. However, operators might decide changes are needed after seeing a sustained increase or decline in I/O activity, or after receiving JES3 messages. Their actions might be to start or stop I/O devices, or to make I/O devices available or unavailable to JES3 (or to MVS) for allocation to jobs.

In addition to changing I/O assignments, operators can react to a slow-running system by using commands to alter JES3 processing. For instance, operators can change the processor on which converter/interpreter (C/I) processing is performed. (C/I processing is the phase when JES3 converts information on JCL statements into internal control blocks for use by the system.) This capability gives operators an active role in monitoring and controlling system activity.

Starting and Stopping JES3 Devices

As discussed previously, JES3 devices are the ones JES3 uses for reading job input and for writing job output. JES3 disk reader, tape reader, and output writer programs do the I/O. As the need arises, operators can start or stop I/O activity by using JES3 commands, which cause JES3 reader and writer programs to be activated or deactivated. I/O activity will stop when a reader or writer program runs out of data, or when an operator enters a command to stop the reading or writing. If I/O stops for lack of data, JES3 can resume I/O when more data is introduced.

An operator who stops reading or writing with a command can resume the reading or writing with a command.

Placing Devices Online or Offline

With a single JES3 command, an operator can change the status of one I/O device, all I/O devices connected to a processor, or all I/O devices on all processors. This is possible because the devices can be identified in JES3 commands by name, by number, or by a range of numbers.

Workflow Management for Operators

Operators can alter how JES3 processes jobs, but they might not get explicit JES3 messages or instructions from programmers for some situations. Messages generally indicate that a problem or potential problem was recognized by JES3. System programmers might know overall job mix, but might not see day-to-day job mixes that affect intermittently system efficiency.

With JES3, operators have ways of searching for potential workflow problems. They can use commands to learn:

- What jobs JES3 has read, but has not begun processing
- What resources are required for a job
- What jobs JES3 has processed and has scheduled for processing
- What jobs are processing on each processor
- Job names, job numbers, job priorities, region sizes, whether jobs are being held up, and what JES3 routines have yet to process jobs
- How much output data is ready for printing or punching
- Determine why a job is not being scheduled for processing

Operators can then use JES3 commands to:

- Suspend further processing of jobs.
- Change the attributes of a job.
- Cancel jobs.
- Resume processing of suspended jobs.
- Limit the number of jobs in execution.
- Change (dynamically) how JES3 is to process and manage the job workload.

Utility Programs for Operators

JES3 offers utility programs, or utilities, for handling the routine work related to running JES3. Utilities do not affect how JES3 processes jobs. Rather, they affect the input to jobs or the output from jobs. To learn more about the various utilities that are available, see *z/OS JES3 Commands*.

Chapter 4. JES3 from an application programmer point of view

Application programmers identify resources their programs need by using job control language (JCL) statements. Generally speaking, JES3 examines JCL statements on behalf of MVS. That is, MVS requires JES3 to prepare jobs for processing, and the information JES3 needs to do that is contained in JCL statements.

JES3 has its own statements called control statements, for defining special processing that application programmers want JES3 to do for their jobs.

The combination of JCL, JES3 control statements and the MVS workload management mechanisms ensure that jobs are scheduled to those systems that have the resources the jobs need.

JES3 Control Statements

JCL statements and JES3 control statements serve the same purpose: they identify resources and services to be used for individual jobs. JES3 control statements are optional, being needed only for the special processing options they represent. When JES3 control statements are used, they are to be used *in addition to* JCL statements (see Figure 8).

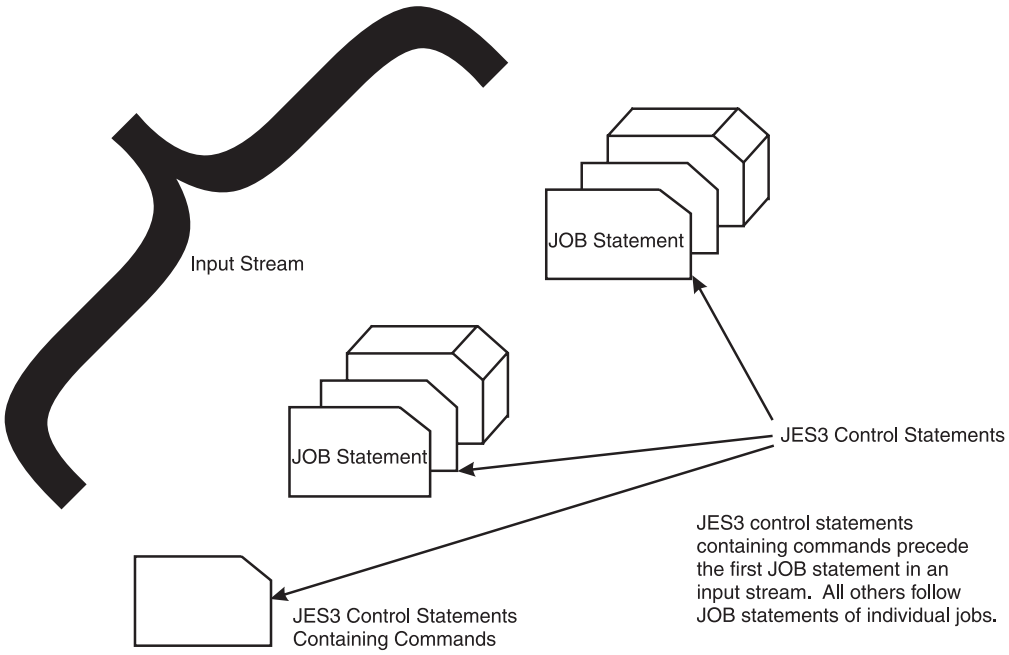

Figure 8. Placement of JES3 Control Statements

With JES3 control statements, application programmers can:

- Use operator commands
- Send messages to operators (such as special instructions for handling printed or punched data)

- Have secondary input sources containing JCL statements, data, or both.
- Route jobs to specific processors
- Route job output to specific devices
- Define limits for output data, and tell JES3 what to do when the limits are exceeded (examples of limits are number of lines to be printed or number of cards to be punched)

Two other things that programmers can use JES3 control statements for are:

- Deadline scheduling, for getting jobs processed before a specified deadline
- Dependent job control, for getting jobs processed only upon successful or unsuccessful processing of other jobs

Deadline Scheduling

Deadline scheduling is a way of scheduling jobs by time of day or by week, month, or year. Job priorities remain in force, but as deadlines approach, JES3 increases the priorities, thereby increasing the likelihood the jobs will be processed before a certain deadline.

System programmers must prepare for deadline scheduling with JES3 initialization statements that tell when a priority is to be increased, and by how much. Then application programmers can put deadlines on JES3 control statements they include with the jobs.

Dependent Job Control

Dependent job control (DJC) is a means of coordinating the processing of jobs. With DJC, one job will not run unless some other job has run first. (DJC has an MVS counterpart for conditional execution of job steps.)

DJC requires no advance preparation with JES3 initialization statements. Application programmers can specify all relationships between jobs on JES3 control statements they submit with the affected jobs.

Priority Aging

When all initiators are busy, throughput of certain jobs might fall below normal expectations. To help in these situations, JES3 uses the additional scheduling function of priority aging. **Priority aging** can help ensure that jobs that have been waiting to run have a chance of being selected to run before those jobs that just entered the system. By using priority aging, an installation can increase the priority of a waiting job. The longer the job waits, the higher its priority becomes, up to a limit, and the greater its chances of being selected to run.

Chapter 5. JES3 from a system programmer point of view

One way of visualizing JES3 is as a control program that performs job entry and job exit services. Another way of visualizing JES3 is as a language. Saying that JES3 is a control program is analogous to saying FORTRAN is a compiler. In reality, there is a FORTRAN language and a FORTRAN compiler. Similarly, there is a JES3 language and there is a JES3 program. Just as a FORTRAN programmer uses the FORTRAN language to define work the compiler is to do, JES3 system programmers use the JES3 language to define the work that JES3 is to do.

The JES3 language is made up of initialization statements, control statements, and commands. Operators and application programmers use parts of the language, but it is system programmers who define the total JES3 environment. Often, what is specified on JES3 initialization statements can be overridden with JES3 commands or control statements. While system programmers cannot themselves use commands or control statements, they might have to instruct operators or application programmers on when and how they should be used.

A further level of defining how JES3 should do its work comes in the form of installation exits supplied with JES3. For example, in the area of controlling the print output from jobs, a system programmer can:

- Define initial values in the initialization stream
- Allow the operator to make certain modifications to the initial values
- Write an installation exit routine to further modify or control what should happen to the output.

As in other sections, specific commands, control statements, and initialization statements are not mentioned. Commands are described in *z/OS JES3 Commands*. JES3 control statements are described in *z/OS MVS JCL Reference*. Initialization statements are described in *z/OS JES3 Initialization and Tuning Reference*. What is described here are some of the ways that system programmers can control processing by JES3 and, where applicable, where a designed-in flexibility exists for defining and redefining how JES3 is to process jobs.

Resource Management for System Programmers

System programmers use JES3 initialization statements to define resources to JES3. As part of resource definition, the system programmer establishes the policies that govern the use of installation resources. The process of resource definition is one of:

- Identifying the resources that are available to JES3
- Defining how JES3 is to use the resources

The resources that a system programmer defines include processors, I/O devices, network nodes, and remote workstations. In addition to defining resources the system programmer can use initialization statements to enable or disable specific JES3 facilities.

System programmers must define the global processor and each local processor in the JES3 complex. Additionally, they define each processor's I/O configuration.

Besides defining resources, the system programmer can turn on or turn off specific JES3 facilities or features. For example, the writer output multitasking facility enables JES3 output writers to do more work in parallel with other JES3 functions. The system programmer (or the operator) must turn on this facility before JES3 can use it.

When defining JES3, system programmers can distribute the workload between the global processor and the local processors. Balancing the workload can relieve storage constraints on the global processor and result in a better running system.

An example of distributing work from the global processor to a local processor is to offload part of C/I phase of JES3 processing onto a local processor. The C/I portion of JES3 analyzes JCL statements on incoming jobs and converts them into internal control blocks usable by the operating system. The process of analyzing JCL frequently uses large amounts of storage, especially in installations that process large numbers of jobs.

If all of this processing is done on the global processor, your system might not run efficiently. By initializing JES3 so that part of the C/I processing occurs on local processors, you can reduce the amount of work to be done on the global processor, which can improve overall system performance.

JES3 system programmers can initialize JES3 in ways that efficiently make use of system-wide workload management functions in MVS. This can maximize JES3's efficiency at processing jobs. The system programmer can dynamically initialize JES3 with minimal disruption to normal operations.

Dynamic system interchange (DSI) is a JES3 feature that enables an installation to make a local processor become the global processor without the loss of work. During a planned or unplanned outage of the global processor, DSI provides a way for the installation to continue doing its work. To enable the installation to use DSI, the system programmer must define one or more local processors as alternate global processors.

Workflow Management for System Programmers

One important feature of JES3 is the amount of control users have over the way JES3 processes jobs. Initialization statements do not 'freeze' JES3 into a framework within which all jobs are similarly handled. Job processing flexibility results not only from:

- The variety of options available.
- The design of the statements and commands that identify required options.
- The flexible and non-disruptive ways you can change how JES3 processes jobs.

Application programmers and operators can override many of the options specified on JES3 initialization statements by using JES3 control statements or commands. Also, named subsets of options identified on initialization statements can be used in control statements to apply the subsets to individual jobs. There is even an initialization statement to define default options to be used by JES3 when specific options are omitted from control statements.

To provide more flexibility in the way your system runs jobs, JES3 allows you to insert your own programs at certain pre-defined locations in JES3 processing. This feature is called *installation exits*. The programs you write are called installation exit routines.

Each installation exit has a name, and at each place where a installation exit is defined, there is a sample JES3 routine. You can, if you want, write another routine, give it the name of the exit, and place it in the JES3 library that contains exit routines. The new routine will replace the JES3 routine with the same name and will be processed in its place.

JES3 has many ways to identify and define job workload so that the maximum use of system-wide workload management mechanisms are used. Through initialization statements, operator commands, and installation procedures, your installation has the flexibility to adjust its processing of jobs to meet its own unique criteria.

If more flexibility is requiredd, system programmers can alter the way JES3 does its work. JES3 is composed of routines called dynamic support programs, or DSPs. With the same mechanisms it uses for processing jobs, JES3 schedules and processes DSPs. JES3 users can write their own dynamic support programs to customize JES3 processing in their system.

Job Classes and Job Class Groups

Job classes and job class groups give installations a disciplined approach to specifying rules for JES3 to follow in processing jobs, scheduling jobs, and assigning resources to jobs. Instead of each application programmer specifying rules for each job, system programmers define sets of rules, and application programmers choose which sets apply to their jobs. Generally speaking, job classes contain rules for processing and scheduling jobs, while job class groups contain rules for assigning resources to jobs.

Job Classes

A job class is a named set of job processing and scheduling rules. System programmers define job classes using JES3 initialization statements. On these, they give classes unique names and specify the rules they want JES3 to apply to jobs. The link between a job and a class is established when an application programmer uses a class name on a JES3 control statement accompanying a job. JES3 will apply the rules represented by the class name to the job.

The job processing and scheduling rules that can be grouped by job class name on initialization statements are the following:

- The priority JES3 should use while processing the job (this is a special JES3 priority that can be overridden by a priority specified in a JCL statement)
- The estimated I/O rate (high, medium, or low) for the job (this is a default I/O rate, to be used unless overridden by an I/O rate on a JES3 control statement)
- The maximum number of jobs of the class to be run on a particular processor
- The maximum number of jobs of the class that can be running in the JES3 complex at one time
- The maximum number of jobs of other classes that can run on a particular processor and still allow jobs of this class to be scheduled
- The maximum number of jobs of other classes that can run in the JES3 complex and still allow other jobs of this class to be scheduled
- The maximum number of jobs of the class requiring volume mounting that can be set up at one time in the complex
- Whether jobs of the class must run only on specific processor(s), or must not run on specific processor(s)

- The actions JES3 will take if a job fails (JES3 can cancel the job, hold the job for later restart, or restart the job)
- Whether JES3 should keep a journal for a job

Job Class Groups

A job class group is a named set of resource assignment rules to be applied to a group of job classes. System programmers define job class groups on JES3 initialization statements. They establish a link between a job class group and a job class by specifying a job class group name when they define the job classes.

The resource assignment rules that apply to job classes are:

- The maximum number of jobs of the job class group that can run concurrently on one processor
- Whether JES3 alone or JES3 in conjunction with the workload management (WLM) component of MVS assigns initiators to run jobs.
- The number of MVS initiators to be dedicated to jobs of the job class group, and when the MVS initiators are to be assigned. (Initiators can be assigned as needed, all can be assigned when the first job is ready)
- When MVS initiators are to be released (initiators can be released when there is no more work, when there are fewer executable jobs than free initiators, or when there are no more executable jobs for the job class group). This applies when JES3 is responsible for initiator management.
- Which devices requiring volume mounting are to be dedicated for use by jobs in the job class group
- Whether non-dedicated devices can be used in addition to dedicated devices
- How many of the named devices are necessary
- How many jobs JES3 should examine when selecting a job for processing
- A job priority limit (JES3 will schedule all jobs in the group with a priority at or above the limit before attempting to schedule jobs below the limit)

Job Output

Using JES3, a system programmer can set limits involving the output produced by the jobs in your installation. For example, a programmer could define a limit for the number of pages of printed output produced by any job. If a job's output exceeds the limit, JES3 might follow a predetermined action, such as notifying the operator, who can cancel the job or let it complete.

Also, a system programmer can set a limit on the amount of spool space that can be used by a job. If a particular job's output uses enough spool space to reach the limit, JES3 notifies the operator, who can take a predefined action. This limit on spool space usage helps control and make better use of system resources.

Device Allocation Options

System programmers can choose from three types of device allocation:

- *Job setup*, where jobs will not be sent to MVS until all devices needed by the jobs are available and all mountable volumes (in the case of multivolume files, the first volume) have been mounted.
- *High-watermark setup*, where JES3 finds the job steps requiring the most devices of each type, allocates that number of devices, and mounts the first volumes to be used on those devices.
- *Explicit setup*, which has the processing advantages of job setup and the device allocation advantages of high-watermark setup. JES3 allocates devices for all

data sets; however, the user can request that devices be scheduled based on a job's processing phase. For example, a particular data set might request a tape volume and that volume is not to be mounted until JES3 has scheduled the job for execution.

Job setup and high-watermark setup are specified on JES3 initialization statements. If installation practices allow, JES3 control statements may be used to override, on a job-by-job basis, what was specified on JES3 initialization statements.

Installation Exits

Installation exits are places in JES3 code where exit routines can receive control. Optionally, system programmers can write their own exit routines and replace the IBM-supplied exit routines. Installation exit routines are one way for an installation to customize JES3. For example, installation exit routines can:

- Change JCL statements
- Change JES3 initialization statements
- Take corrective action when needed data cannot be found
- Enforce installation standards
- Cancel jobs
- Put text into header or trailer pages of printed output
- Align paper for printed output
- Verify nonstandard tape labels
- Monitor, accept, or reject information sent to or from a network node

Installation-Written Dynamic Support Programs

Most of the JES3 program in the global processor is divided into parts called dynamic support programs, or DSPs. There are DSPs for reading job input, for processing jobs, and for writing job output. What distinguishes DSPs from ordinary routines or subroutines is that DSPs are schedulable units. Before a DSP is executed, it must be scheduled by JES3. (DSPs have priorities that govern their position in a JES3 dispatching queue.)

System programmers can alter what DSPs do (with installation exit routines), or they can write new DSPs to supplement or replace the DSPs shipped with JES3.

Service Aids for System Programmers

JES3 has many service aids to help system programmers test for, diagnose, and correct problems.

JES3 Monitoring Facility (JMF)

The JES3 monitoring facility collects data from your system to see how your installation uses its resources. This information can help detect any performance problems which help you tune your installation. A JES3 command invokes the facility. You have the choice of producing either a hard-copy report or SMF records.

Control Block Print Programs

Control block print programs print selected JES3 and MVS control blocks. Special JES3 control statements containing control block names will cause the printing to take place.

Display Dependent Job Control Network Program

The display dependent job control (DJC) network (this is not related to a teleprocessing network) program prints a report describing a DJC network. Included are the number of jobs in the network, the number of completed jobs, the names and status of each job in the network, and dependencies of jobs on other jobs. A JES3 command causes printing of the report.

Display JES3 Job Queue Program

The display JES3 job queue program prints information about jobs in the JES3 job queue. (JES3 uses the job queue to track jobs being readied for processing). A JES3 command will cause printing of the data. Users can print information about a specific job or about all jobs of a specific priority.

Display Selected Areas of Storage Program

This utility displays storage, alters data in storage, produces dumps, causes a wait (a suspension of execution) at a specific address, displays register contents at a wait, and displays JES3 control blocks. A JES3 command causes the requiredd activity to take place. The readable text is displayed as well as the hexadecimal values.

Dump Job Queue to Tape Program

The dump job queue to tape program transfers the contents of the JES3 job queue to tape. This program also returns the JES3 job queue to storage, so that JES3 can resume processing jobs where processing stopped when the job queue was dumped. A JES3 command causes dumping or restoration of the JES3 job queue.

Event Tracing Facilities

Event tracing is the collection and recording of system data at various times during the processing of jobs. The reports produced are records of key system events and show data collected during the events.

One advantage to event tracing is that a sequence of events is recorded, so the data will reflect a continuous picture of JES3 processing. Also, tracing can be started whether or not a failure has occurred, and reports can be a valuable source of tuning information.

Generally, a trace report identifies the JES3 module in which the data was collected, the contents of registers, addresses of key tables, and other data related to the events. Some of the events traced are:

- Activity on communication lines (used for remote job processing and network configurations)
- Job processing activity, including both normal and error job processing conditions.

Initialization Debugging Aid Program

The initialization debugging aid program produces storage dumps during JES3 initialization. Error messages trigger the dumps. The text of any initialization message can be put into a special JES3 initialization statement. Then, whenever JES3 writes the message, the initialization debugging aid will produce a storage dump.

JCL Test Program

The JCL test program checks JCL statements for errors. Basically, a job is handled normally by JES3 until after the job's JCL is interpreted and internal tables are constructed. JES3 does not process the job any further. JES3 does not set up any devices and will not schedule the job for processing. A special parameter on the job's EXEC statement causes testing of JCL in place of normal job processing.

Initialization Stream Checker Utility

To help avoid errors during JES3 initialization, JES3 provides a utility called the initialization stream checker. This utility simulates the JES3 initialization process and enables you to verify your initialization stream before actually initializing JES3. This utility scans the entire initialization stream for syntax errors and certain inconsistencies in the statements which would cause JES3 to either fail to initialize or to initialize with errors.

A sample JES3 initialization stream is shipped with JES3. The sample stream can be used to help a new JES3 user get started on creating an initialization stream.

Interactive Problem Control System (IPCS)

IPCS allows installations to format and display JES3 control blocks. System programmers can use IPCS to:

- Locate and view JES3 control blocks
- View JES3 formatted dumps

IPCS can display trace data for all active trace entries and functions for any JES3 trace table. See *z/OS JES3 Diagnosis* for more information about using IPCS.

Providing Security

Security in a data processing environment involves controlling and auditing access to resources that are important to your installation. In the JES3 environment, these resources include:

- JES3-owned data sets
- Input (from nodes, remote workstations, readers, offload devices, and commands)
- Job names
- System input/output residing on spool (SYSIN/SYSOUT)
- Output devices (nodes, printers, punches, remote workstations, and offload devices).

JES3 provides security for some of these resources through initialization statements. For example, each node in a network can be defined as having a certain level of control over work at each of the other nodes in the system, which can gives one operator limited control over each of the other nodes. This level of control is based on mutual agreement between the nodes and the degree of “trust” one node has for the other.

The control available through initialization statements can be broadened by implementing several JES3 exits available for this purpose. You can implement a more complete security policy by using the security authorization facility (SAF) component of the BCP and a security product such as Resource Access Control Facility (RACF®). SAF provides an interface to the security product to define additional control and whatever additional security controls your installation might require.

JES3 passes information to SAF to perform password validation, to request authority to access a resource, and to determine security information in various environments. When SAF and the security product indicate a decision on a security request, JES3 bypasses its own security processing.

Appendix. Accessibility

Accessible publications for this product are offered through the z/OS Information Center, which is available at www.ibm.com/systems/z/os/zos/bkserv/.

If you experience difficulty with the accessibility of any z/OS information, please send a detailed message to mhvrcfs@us.ibm.com or to the following mailing address:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Accessibility features

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size.

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing the z/OS Information Center using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually

exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.
- ! means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1!

(KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

- * means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

  Note:

  1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
  2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
  3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

- + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

Notices

This information was developed for products and services offered in the U.S.A. or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of 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.

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

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

COPYRIGHT LICENSE:

This information might contain sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted

for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: IBM Lifecycle Support for z/OS (http://www.ibm.com/software/support/systemsz/lifecycle/)
- For information about currently-supported IBM hardware, contact your IBM representative.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml (http://www.ibm.com/legal/copytrade.shtml).

Index

K

L

M

N

O

P

R

S

T

U

V

W

Product Number: 5650-ZOS

Printed in USA

SA32-1004-00