z/OS

JES2 Introduction

SA22-7535-06

z/OS

JES2 Introduction

SA22-7535-06

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 31.

Seventh Edition, September 2007

This is a major revision of SA22-7535-05.

This edition applies to Version 1 Release 9 of z/OS (5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this document, or you may address your comments to the following address:

International Business Machines Corporation
MHVRCFS, Mail Station P181
2455 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+845+432-9405
FAX (Other Countries):
Your International Access Code +1+845+432-9405

IBMLink™ (United States customers only): IBMUSM10(MHVRCFS)
Internet e-mail: mhvrcfs@us.ibm.com
World Wide Web: http://www.ibm.com/servers/eserver/zseries/zos/webqs.html

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this document
- Page number or topic related to your comment

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

Contents

Figures

About this document

This document provides an introduction to the job entry subsystem 2 (JES2). It is meant to provide an overview of JES2; it is not meant to be an instructional manual. This document is specifically designed for installations running z/OS. This document presents:

- JES2 as a unified set of related functions whose purpose is to accomplish certain data processing goals for MVS
- JES2 and its relationship to MVS, Time Sharing Option Extensions (TSO/E), and components of the MVS operating system
- An overview of JES2 functions, device control, network job entry (NJE), remote job entry (RJE), initialization, operations, customization techniques, and diagnostic tools.

Who should read this document

This document is intended for the MVS system programmer, operator, data processing manager, or student who is unfamiliar with JES2 and its functional relationship to the base control program of MVS.

This document is specifically designed for installations that are evaluating or running z/OS JES2. However, much of the function described is available in previous versions of the JES2 component in MVS/System Product Version 1, Version 2, Version 3, Version 4, and Version 5. Therefore, most of the material presented here applies to JES2 at those version/release levels because of the overview-level of detail. The document, however, makes no distinction between functions and in what version/release they were added to JES2.

How to use this document

To obtain the most from this document, you should read it sequentially. The information presented in each chapter and topic is built upon previous information. If you are unfamiliar with JES2 concepts and terminology, you will find that reading the document in this manner is most helpful.

Related information

When this document references information in other documents, the shortened version of the document title is used. See *z/OS Information Roadmap* for their full titles and order numbers.

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most of the IBM® messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can use LookAt from these locations to find IBM message explanations for z/OS® elements and features, z/VM®, z/VSE™, and Clusters for AIX® and Linux™:

- The Internet. You can access IBM message explanations directly from the LookAt Web site at www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/.
- Your z/OS TSO/E host system. You can install code on your z/OS systems to access IBM message explanations using LookAt from a TSO/E command line (for example: TSO/E prompt, ISPF, or z/OS UNIX® System Services).
- Your Microsoft® Windows® workstation. You can install LookAt directly from the *z/OS Collection* (SK3T-4269) or the *z/OS and Software Products DVD Collection* (SK3T-4271) and use it from the resulting Windows graphical user interface (GUI). The command prompt (also known as the DOS > command line) version can still be used from the directory in which you install the Windows version of LookAt.

- Your wireless handheld device. You can use the LookAt Mobile Edition from www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookatm.html with a handheld device that has wireless access and an Internet browser (for example: Internet Explorer for Pocket PCs, Blazer or Eudora for Palm OS, or Opera for Linux handheld devices).

You can obtain code to install LookAt on your host system or Microsoft Windows workstation from:

- A CD-ROM in the *z/OS Collection* (SK3T-4269).
- The *z/OS and Software Products DVD Collection* (SK3T-4271).
- The LookAt Web site (click **Download** and then select the platform, release, collection, and location that suit your needs). More information is available in the LOOKAT.ME files available during the download process.

Using IBM Health Checker for z/OS

IBM Health Checker for z/OS is a z/OS component that installations can use to gather information about their system environment and system parameters to help identify potential configuration problems before they impact availability or cause outages. Individual products, z/OS components, or ISV software can provide checks that take advantage of the IBM Health Checker for z/OS framework. This book might refer to checks or messages associated with this component.

For additional information about checks and about IBM Health Checker for z/OS, see *IBM Health Checker for z/OS: User's Guide*.

SDSF also provides functions to simplify the management of checks. See *z/OS SDSF Operation and Customization* for additional information.

Additional Information

Additional information about z/OS elements can be found in the following documents.

Title	Order Number	Description
z/OS Introduction and Release Guide	GA22-7502	Describes the contents and benefits of z/OS as well as the planned packaging and delivery of this new product.
z/OS Planning for Installation	GA22-7504	Contains information that lets users: • Understand the content of z/OS • Plan to get z/OS up and running • Install the code • Take the appropriate migration actions • Test the z/OS system
z/OS Information Roadmap	SA22-7500	Describes the information associated with z/OS including z/OS documents and documents for the participating elements.
z/OS Summary of Message and Interface Changes	SA22-7505	Describes the changes to messages for individual elements of z/OS. **Note:** This document is provided in softcopy only on the message bookshelf of the z/OS collection kit.

Summary of changes

Summary of changes
for SA22-7535-06
z/OS Version 1 Release 9

This document contains information previously presented in *z/OS JES2 Introduction*, SA22-7535-05, which supports z/OS Version 1 Release 8.

New information

- None

Changed information

- None.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

You may notice changes in the style and structure of some content in this document—for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our documents.

Summary of changes
for SA22-7535-05
z/OS Version 1 Release 8

This document contains information previously presented in *z/OS JES2 Introduction*, SA22-7535-04, which supports z/OS Version 1 Release 7.

New information

- None

Changed information

- None.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Summary of changes
for SA22-7535-04
z/OS Version 1 Release 7

This document contains information previously presented in *z/OS JES2 Introduction*, SA22-7535-03, which supports z/OS Version 1 Release 5.

New information

- Added TCP/IP protocol information to JES2 remote job entry (RJE) and JES2 network job entry (NJE) in Chapter 2, Scope of Control and Configurations. See “JES2 remote job entry (RJE)” on page 7 and “JES2 network job entry (NJE)” on page 10.

Changed information

- Modified a figure about phases of job processing. See Figure 6 on page 13.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Chapter 1. Introduction to JES2

This chapter answers the following questions:

- What is a JES?
- Why is JES required in an MVS system?
- What is the relationship between JES and the system users?
- Is there really any difference between JES2 and JES3?
- What are some of JES2's key functions?

By the time you pick up this manual , you are probably aware that one of your initial concerns when installing MVS is whether to select an MVS system with the JES2 or the JES3 component. This chapter provides some of the background you will need if you are new to MVS, and the remainder of the book provides a somewhat more detailed discussion of JES2 function. If you require further insight into the functional differences between JES2 and JES3, or are deciding which to install, you should also read *z/OS JES3 Introduction* which provides a similar overview of JES3.

What is a JES?

MVS uses a job entry subsystem (JES) to receive jobs into the operating system, schedule them for processing by MVS, and to control their output processing. JES2 is descended from HASP (Houston automatic spooling priority). HASP is defined as: *a computer program that provides supplementary job management, data management, and task management functions such as: scheduling, control of job flow, and spooling*. HASP remains within JES2 as the prefix of most module names and the prefix of all messages sent by JES2 to the operator.

JES2 (job entry subsystem 2) is *a functional extension of the HASP II program that receives jobs into the system and processes all output data produced by the job*. So, what does all that mean? Simply stated, JES2 is that component of MVS that provides the necessary functions to get jobs into, and output out of, the MVS system. It is designed to provide efficient spooling, scheduling, and management facilities for the MVS operating system. (See "Spool data sets and spooling" on page 5 for a full definition of Spooling.)

But, none of this explains why MVS needs a JES. Basically, by separating job processing into a number of tasks, MVS operates more efficiently. At any point in time, the computer system resources are busy processing the tasks for individual jobs, while other tasks are waiting for those resources to become available. In its most simple view, MVS divides the management of jobs and resources between the JES and the base control program of MVS. In this manner, JES2 manages jobs before and after running the program; the base control program manages them during processing. [1] Figure 1 on page 2 presents a diagram of the relationship between JES and MVS.

1. JES3, in contrast, manages jobs throughout the entire process cycle (before, during, and after running the programs).

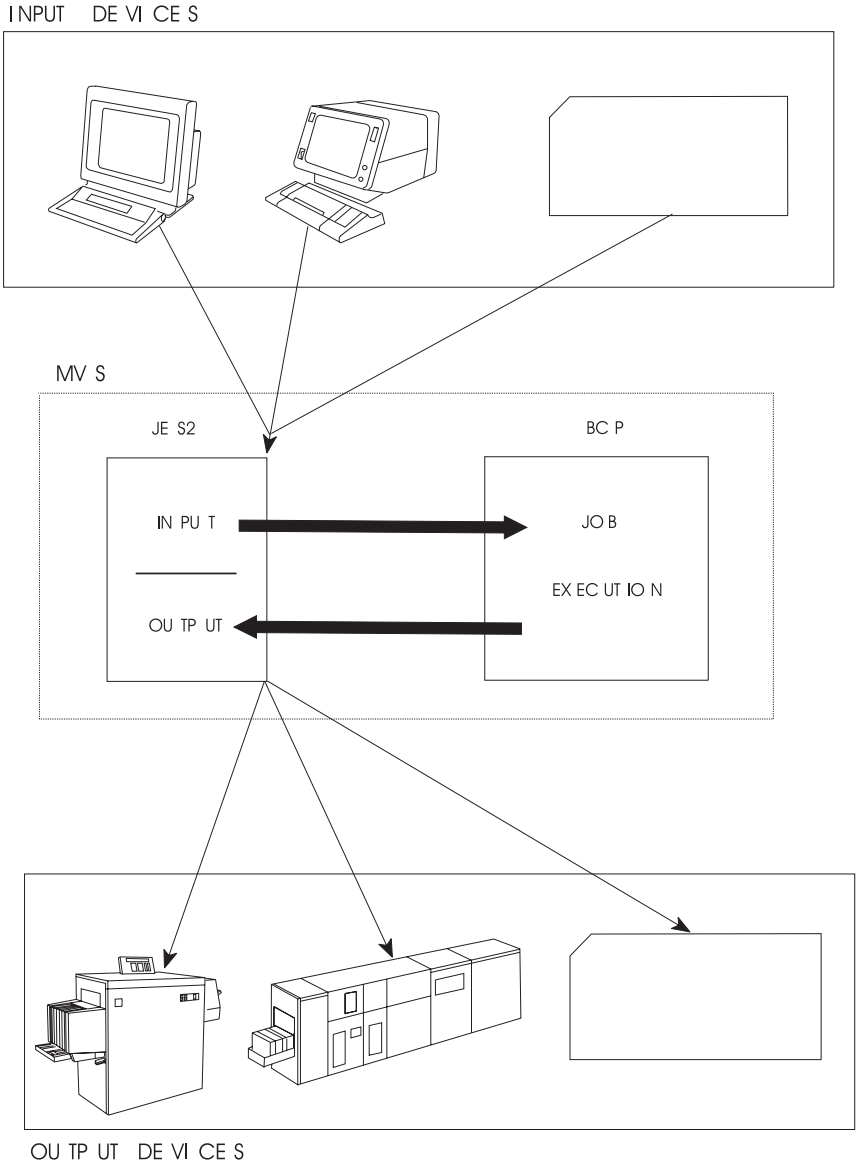


Figure 1. Relationship of JES2 to the Base Control Program

How JES2 fits into the z/OS System

During the life of a job, both JES2 and the base control program of z/OS control different phases of the overall processing. Generally speaking, a job goes through the following phases:

- Input
- Conversion
- Processing
- Output
- Print/punch (hard copy)
- Purge

Except for processing, all the job phases are controlled by JES2.

These phases are explained in more detail in Chapter 3, Chapter 3, “JES2 job processing and functions,” on page 13.

Relationship of JES2 to JCL and submitted jobs

JES2 initialization statements give the system programmer a single point of control to define an installation’s policies regarding the degree of control users have over their jobs. For example, consider the confusion an installation would experience if users were allowed to define their own job classes and priorities. Very likely, all users would define all their jobs as top priority, resulting in no effective priority classifications at all. So, although a user can use job control language (JCL) options to define a priority, a JES2 initialization statement (defined by the system programmer) determines whether JES2 acknowledges that priority.

This same type of control (validation of user specification and provision for a default) extends to many JCL definitions. For example, JES2 can specify the maximum time a particular job is allowed to run, the storage a job may consume, the number of copies of output a job can print, and the type of paper (form) on which the output prints.

An installation has the ability to allow its users to override system and JES2 specifications at three distinct levels (user-specified JCL, installation-specified JCL, and JCL defaults). The following list shows the hierarchy of control, from highest priority to lowest priority, of user JCL, JCL defaults, and JES2 defaults:

1. JCL specification on a user job.

   The user JCL overrides:
2. JCL default, which z/OS uses if there is no user definition or the user specification is disallowed.

   The JCL default overrides:
3. JES2 default, which z/OS uses if:
   - No JCL default exists
   - The user specification is disallowed or undefined
   - The JCL default definition is not supported.

Note that the user can override the JCL defaults and the JCL defaults can override the JES2 specifications, but the ability to override is either permitted or disallowed by specifications that only the system programmer can control. This structure thereby puts the system control in the hands of the system programmer, not the individual user who is submitting the job. JES2 becomes the base for input and output specifications that can then be overridden, as allowed by your installation, through the JCL and job submittor.

JES2 compared to JES3

IBM provides two JESs from which to choose: JES2 and JES3. In an installation that has only one processor (computer), JES2 and JES3 perform similar functions. That is, they read jobs into the system, convert them to internal machine-readable form, select them for processing, process their output, and purge them from the system. However, for an installation that has more than one processor in a configuration, there are noticeable differences in how JES2 exercises **independent control** over its job processing functions. That is, within the configuration, each JES2 processor controls its own job input, job scheduling, and job output processing. In contrast, JES3 exercises **centralized control** over its processing functions through a single **global** JES3 processor. This global processor provides all job selection, scheduling, and device allocation functions for all the other JES3 systems. The centralized control that JES3 exercises provides: increased job scheduling control, deadline scheduling capabilities, and increased control by providing its own device allocation. To gain a more complete understanding of the functional differences between JES2 and JES3, see *z/OS JES2 Initialization and Tuning Guide* and *z/OS JES3 Initialization and Tuning Guide*.

Overview of JES2

The remainder of this book provides an overview of the processing configurations that JES2 supports, the functional capabilities of JES2, the ability to customize JES2 to meet your specific processing needs, and the means by which JES2 communicates its status through messages and diagnostic information. Each topic is discussed briefly here.

JES2 configurations

You can run z/OS in your installation in many processing configurations that range from a single base control program with a single JES2 that is completely isolated from other processing systems to one that is a part of a worldwide processing network. The choice of configuration complexity is yours and generally is a dynamic one that grows as your business needs grow. Basic configurations are:

- Single-processor
- Multiple-system (multi-access spool)
- Poly-JES
- Remote job entry (RJE)
- Network job entry (NJE)

These are discussed in greater detail, with examples, in Chapter 2, Chapter 2, “Scope of control and configurations,” on page 5.

JES2 functions

To manage job input/output responsibilities for z/OS, JES2 controls a number of functional areas, all of which you can customize to your installation’s need. Some of the major functional areas and processing capabilities are:

- Getting work into and out of MVS (input/output control)
- Maximizing efficiency through job selection and scheduling
- Offloading work and backing up system workload
- Supporting advanced function printers (AFPs)
- Running multiple copies of JES2
- System security.

Each of these is discussed in Chapter 3, Chapter 3, “JES2 job processing and functions,” on page 13.

Customizing JES2

JES2 has the flexibility to fulfill an installation’s unique processing needs. An installation can virtually control every JES2 function. You can perform many customization tasks when JES2 is installed, and you can dynamically customize JES2 whenever your processing needs change. JES2 provides initialization statements, commands, IBM-supplied exit points, the ability to add installation-defined exit points that require minimal source code modification, and the ability to change many commands and system messages, all without the need to modify the IBM-supplied code. See Chapter 4, Chapter 4, “Tailoring your JES2 system,” on page 19, for a more complete description of these customization facilities.

Interacting with JES2

No large data processing system or subsystem can continually operate independently of system programmer or operator intervention. A two-way communication mechanism between the operator and JES2 is required. Based on system workload and device requirements, JES2 must communicate its status to the operator, system programmer, or both. JES2 issues messages to communicate job and device status, problem situations, system resource constraint situations, and performance status. Through commands, you can request current status and through the use of various tools you can obtain further information to diagnose and correct problem and system failure situations. Chapter 5, Chapter 5, “Interacting with JES2,” on page 23, provides an overview of these topics.

Chapter 2. Scope of control and configurations

This chapter answers the following questions:

- How does JES2 manage its work?
- Where does JES2 store its data, and how does JES2 provide data integrity?
  - What is spooling?
  - What is checkpointing?
- What processing configurations does JES2 support?
  - What is a multi-access spool complex?
  - What is poly-JES?
  - What is remote job entry?
  - What is network job entry?

The control that JES2 has over certain data sets and devices allows z/OS to offload work to JES2. The means by which JES2 maintains these data sets is unique to JES2. Furthermore, the configurations in which JES2 operates range from simple to rather complex. This chapter provides a basic understanding of the JES2 operating configurations; it is important to your complete understanding of the following chapters .

JES2 data set control

JES2 maintains copies of its data sets that contain job and output queues (that is, lists of jobs to be processed by z/OS) on direct access storage devices (DASD). Future work is added to these queues and JES2 selects work for processing from them. These data sets and queues must remain current and accurate to maintain system integrity and performance. The following is a discussion of the JES2 spool and checkpoint data sets and the processing JES2 uses to maintain them.

Spool data sets and spooling

Simultaneous peripheral operations online (spool) has several meanings as used in this book and throughout JES2 documentation. **Spooling** is a process by which the system manipulates its work. This includes:

- Using storage on direct access storage devices (DASD) as a buffer storage to reduce processing delays when transferring data between peripheral equipment and a program to be run.
- Reading and writing input and output streams on an intermediate device for later processing or output.
- Performing an operation such as printing while the computer is busy with other work.

Spool also refers to the direct access device that contains the spool data sets. This definition is generally apparent from the context of its use and should not cause any misunderstanding in the following sections of this book or other JES2 documentation.

Spooling is critical to maintain performance in the z/OS-JES2 environment. JES2 attempts to maximize the performance of spooling operations, which ultimately benefits the throughput of work through the JES2 installation.

As noted previously, spooling provides simultaneous processing and a temporary storage area for work that is not yet completed. When JES2 reads a job into the system, JES2 writes the job, its JCL, its control statements, and its data to a spool data set until further processing can occur.

Maintaining integrity

Errors can occur while processing jobs and data. Some of these errors can result in the halting of all system activity. Other errors can result in the loss of jobs or the invalidation of jobs and data. If such errors

occur, it is preferable to stop JES2 in such a way that allows processing to be restarted with minimal loss of jobs and data. The **checkpoint data set**, **checkpointing**, and the **checkpoint reconfiguration dialog** all help to make this possible.

Checkpoint data set is the general term used to describe the checkpoint data set that JES2 maintains on either DASD or a coupling facility. The checkpoint data set (regardless of whether it resides on a coupling facility structure or a DASD volume) contains a backup copy of the job and output queues, which contain information about what work is yet to be processed and how far along that work has progressed. Similar to the spool data sets, the checkpoint data set is commonly accessible by all members of the multiple-processor (multi-access) spool complex, but only one member will have control (access) of the checkpoint data set at any one time. Furthermore, the checkpoint data set provides communication among all members of the configuration about jobs and the output from those jobs. JES2 periodically updates the checkpoint data set by copying the changed information from the in-storage copy to the checkpoint data set copy that resides on either a coupling facility structure or on DASD. Information in the checkpoint data set is critical to JES2 for normal processing and in the event that a JES2 or system failure occurs.

Checkpointing is the concept of keeping a copy of the checkpoint data set that contains essential workload information on either a coupling facility structure or a DASD volume. This copy is updated from the in-storage copy of the checkpoint as each JES2 member of a multi-access spool updates the in-storage checkpoint data set. **Checkpointing** allows JES2 to be stopped and then restarted with little or no loss of job or data integrity.

The **checkpoint reconfiguration dialog** is a dynamic means by which the current checkpoint configuration can be changed (for example, adding a checkpoint device or moving the checkpoint data set to a different device). It is possible to enter the checkpoint reconfiguration dialog to continue processing without necessitating a JES2 restart. This process increases system availability.

JES2 configurations

The size and complexity of your data processing configuration is based on your business needs. Many factors contribute to the configuration your installation requires. For example, you need to consider the:

- Number of concurrent interactive users
- Size of your data base(s)
- Number of customers
- Geographic location of users and resources
- Number of users that need to run jobs, access data bases, and use programs.

JES2 provides two distinct facilities for the expansion of your JES2 processing configuration: remote job entry and network job entry. **Remote job entry** allows for the extension of your local processing configuration by defining remote locations that can consist of job input terminals and output devices at a different physical site connected to the z/OS–JES2 processor through telecommunication links such as telephone lines and satellites. In this manner, an installation can provide input and output support throughout a large office building, to locations across town, or even in another city. But all the remote sites and their attached devices are defined to a single z/OS–JES2 configuration. **Network job entry** takes this concept further by allowing individual z/OS–JES2 processors, that are geographically dispersed, to be connected in a network of JES2 and non-JES2 systems that can communicate, pass jobs, and route output to any attached output devices.

JES2 allows you to take advantage of the power of z/OS in:

- A single-system complex (one processor and one JES2)
- A multiple-system complex (up to 32 processors, each with its own JES2)
- A poly-JES configuration (more than one JES2 operating concurrently within a single z/OS)
- Remote job entry (RJE) workstations (remotely attached to a processor in the configuration)
- A network job entry complex (the linking of two or more single-system or multi-system configurations).

The remainder of this chapter addresses each of these processing configurations.

Single-system configuration

A JES2 **configuration** can contain as few as one processor (one z/OS and JES2 system) or as many as 32 processors linked together. A single processor is referred to as a **single-system configuration**. Such a system is suitable for an installation that has a relatively small work load, or possibly an installation that requires a single processor to be isolated from the remainder of the data processing complex, possibly to maintain a high degree of security.

Multiple-system configuration

Many installations take advantage of JES2's ability to link processors together to form a multiple-processor complex, which is generally referred to as a multi-access spool (MAS) configuration. A **multi-access spool configuration** consists of two or more JES2 (MAS) processors at the same physical location, all sharing the same spool and checkpoint data sets. There is no direct connection between the processors; the shared direct access data sets provide the communication link. Each JES2 processor can read jobs from local and remote card readers, select jobs for processing, print and punch results on local and remote output devices, and communicate with the operator. Each JES2 processor in a multiple processor complex operates independently of the other JES2 processors in the configuration.

The JES2 processors share a common job queue and a common output queue, which reside on the checkpoint data sets. These common queues enable each JES2 processor to share in processing the installation's workload; jobs can run on whatever processor is available and print or punch output on whatever processor has an available device with the proper requirements. Users can also specifically request jobs to run on a particular processor and output to print or punch on a specific device. If one processor in the configuration fails, the others can continue processing by selecting work from the shared queues and optionally take over for the failed processor. Only work in process on the failed processor is interrupted; the other JES2 systems continue processing.

Running multiple copies of JES2 (poly-JES)

z/OS allows more than one JES2 subsystem to operate concurrently, if one subsystem is designated as the primary subsystem and all others are defined as secondary subsystems. A secondary JES does not have the same capabilities as the primary JES2 and some restrictions apply to its use. Most notably, TSO/E users can only access the primary subsystem. The restrictions are necessary to maintain the isolation from the z/OS-JES2 production system, but the convenience for testing is a valuable function. Operation of multiple copies of JES2 is referred to as **poly-JES**. Secondary JES2s can be useful in testing a new release or installation-written exit routines. This isolation prevents potential disruption to the primary JES2 and base control program necessary for normal installation production work.

JES2 remote job entry (RJE)

The **remote job entry** (RJE) facility allows JES2 to define and use RJE workstations. An **RJE workstation** is a workstation that is connected to a member by means of data transmission facilities. The workstation can be a single I/O device or group of I/O devices or can include a processor such as a System/36, or System/390. Generally, RJE workstations either include a programmable workstation (such as a personal computer) or a communication terminal (such as a 3770, 2780, or S/360) connected to the z/OS system through a telecommunication link. Such a link utilizes synchronous data link control (SDLC), or binary synchronous communication (BSC) for communicating between JES2 and remote devices. The remote device will be either a system network architecture (SNA) remote, that uses SDLC, or a BSC remote, that uses BSC. Figure 2 on page 8 shows a simple RJE configuration.

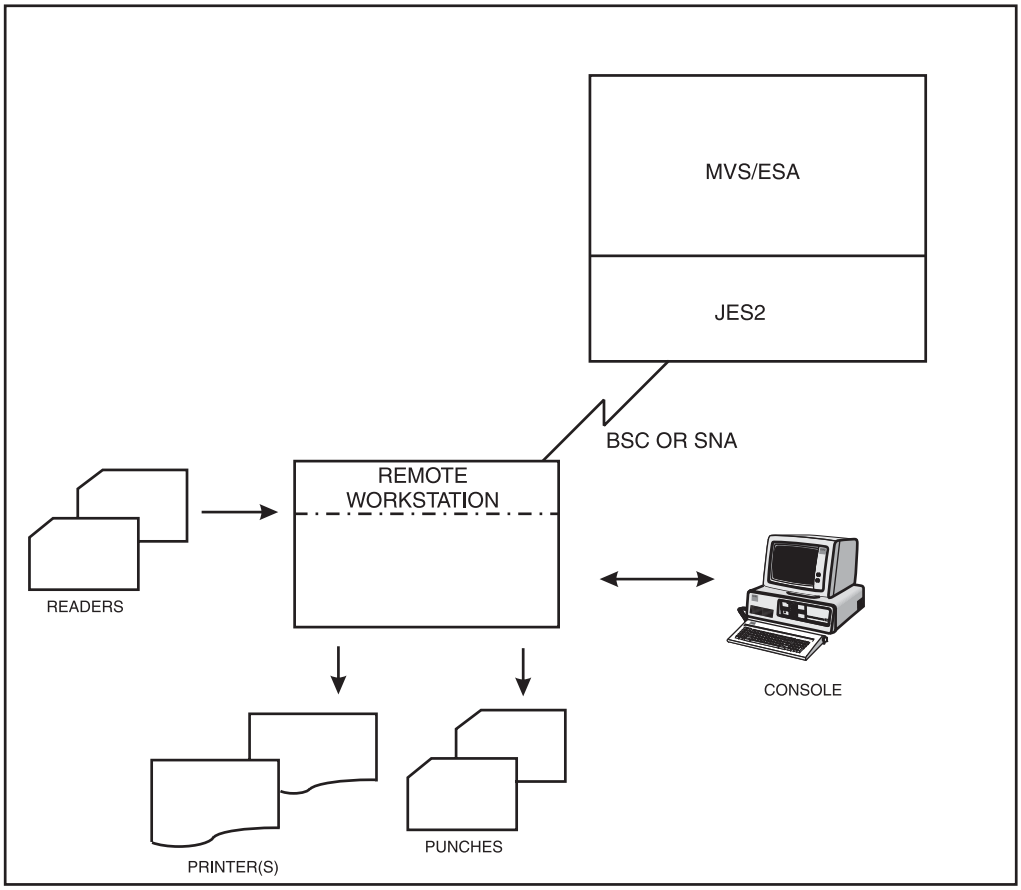


Figure 2. Remote Job Entry Configuration

An RJE workstation is an extension of the local processing facility, except that work is sent across teleprocessing lines. Sending work across teleprocessing lines is convenient for an installation that needs to provide many data entry points at remote locations or route output to many diverse locations. The following illustrates the use of RJE in two familiar examples of daily business:

- In a large department store, many sales clerks need to print output on printers located at the many customer check-out desks throughout the store. Because the location of the building prevents direct connection to the z/OS system located in the central office located several hundred miles away, each printer can be defined to JES2 as a remote printer.
- Consider a clothing store chain in which the store managers at seven different stores all need to place orders, access inventory, and provide billing information, all the information for which is maintained on storage devices located in the computer center at the main office. An individual store's workstation is defined to JES2 as an SNA-attached remote to which the individual terminals are defined and as such become an extension of the JES2 configuration located at the main office.

Figure 3 on page 9 presents a diagrammatic view of the RJE configuration as described in the preceding clothing store example.

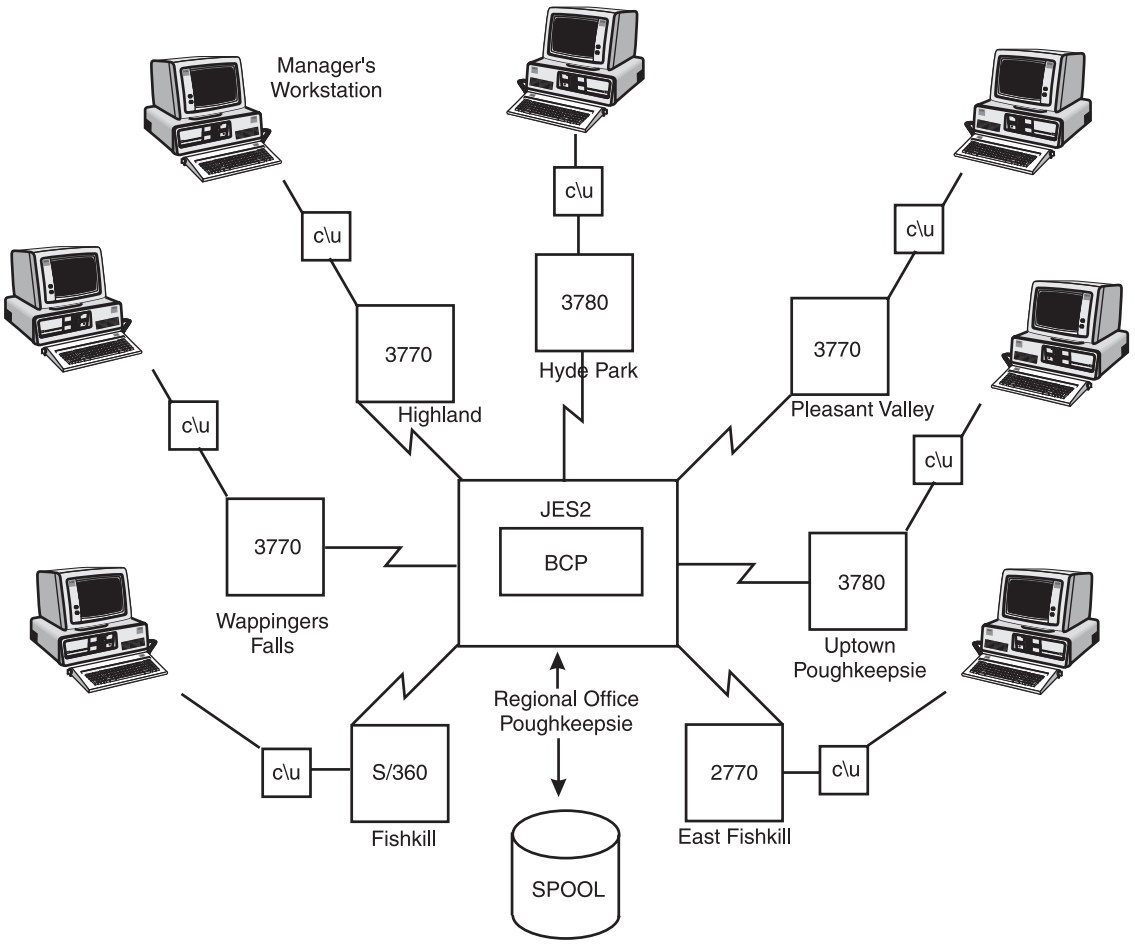


Figure 3. Example of a Remote Job Entry (RJE) Configuration

In Figure 3 each of the clothing stores within a localized region of New York, the Hudson Valley (Highland, Hyde Park, Pleasant Valley, and so forth), is connected to the single processor located at the regional office in Poughkeepsie. One z/OS–JES2 system conducts the business of inventory control, shipping, and billing for all of the stores in the region.

JES2 processes remote jobs in the same manner as those received from a local reader. (**Local devices** are printers, punches, card readers, and lines directly attached to the system without the need of transmission facilities.) The terminals and printers located in the Poughkeepsie office are **locally attached**; all other I/O devices (terminals and printers) in all the other branch stores are defined to JES2 as **remote** terminals and printers.

To provide RJE processing, the RJE workstation must be defined to the local processor. There are two protocols available by which JES2 can communicate with the RJE workstations: **synchronous data link control** (SDLC), and **binary synchronous communication** (BSC).

An RJE workstation can have a processor, like the System/370, that runs a JES2-generated program. The JES2-generated program allows the processor to send jobs to, and receive data from, JES2. Such RJE workstations have generally been replaced by either a programmable workstation, such as a personal computer or a network job entry configuration, and they are rarely used in today's processing environment. Some RJE workstations do not have a processor. These workstations use a remote terminal, for example, a 2780 or 2770, to enter jobs into, and receive data from, JES2.

See *z/OS JES2 Initialization and Tuning Guide* for a more complete discussion of RJE concepts.

JES2 network job entry (NJE)

The JES2 **network job entry** (NJE) facility is similar to remote job entry (RJE) in that they both provide extensions to a computer system. In its simplest terms, NJE is ″networking″ between systems that interact as peers, whereas RJE is networking between JES2 and workstations. The main difference between them is one of overall compute power and processor location. Remember, RJE is an extension of a *single* computer system (that is, either a single-processor or multi-access spool complex) that allows jobs to be submitted from, and output routed back to, sites that are remote to the location of that system. NJE provides a capability to link many such single-processor systems or multi-access spool complexes into a processing network. Each system can be located on the same physical processor, side-by-side in a single room, or across the world in a network of thousands of nodes. The important difference is that a processor and its local and remote devices make up a **node**. Three or more attached nodes make up an NJE network.

Furthermore, as discussed in the section , “Multi-System Complex”, a node can include as few as one processor with z/OS or as many as 32, all sharing the same spool and checkpoint data sets. NJE network nodes communicate (send and receive data) using various teleprocessing facilities. Nodes on the same physical processor use the Virtual Telecommunications Access Method (ACF/VTAM) program product to communicate (using SNA or TCP/IP);... ; no hardware is required. Nodes located in the same room or building can use channel-to-channel adapters (CTCA) or telecommunication links. Nodes that are geographically dispersed use SNA, BSC, or TCP/IP telecommunication links. The following example further explains this concept.

If we return to the previous example of the clothing store chain, the Poughkeepsie regional office is the location of the processor (z/OS and JES2); each of the sites (Highland, Hyde Park, Pleasant Valley, Fishkill, and so forth) are attached to the Poughkeepsie office as remote sites. Together, all the locations in Figure 3 on page 9 comprise the Poughkeepsie node. For a locally owned, relatively small clothing store chain such as the one depicted, this complex may suffice. However, for a national clothier or one dealing in the import/export business, one processor would likely prove to be inadequate. Such a company would likely elect to establish many nodes throughout the world, each connected to all others in an NJE complex. See Figure 4 on page 11 for a diagrammatic view of such a network. Note that the different groups of the ordering/billing department and the business administration/payroll department can be separate members of a MAS making up the New York and London nodes.

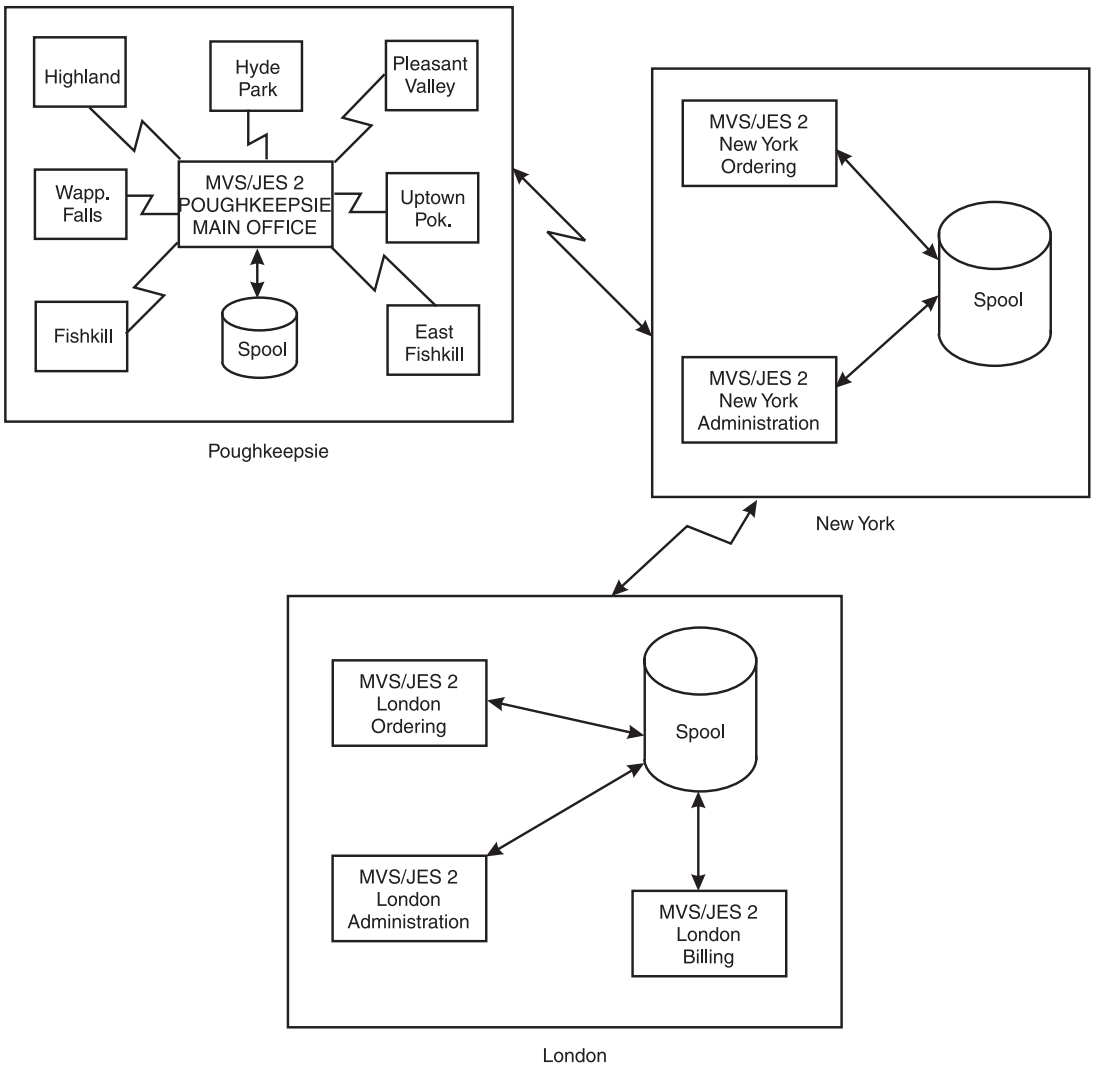


Figure 4. Example of a Network Job Entry (NJE) Configuration

Note that the New York and London nodes are multi-access spool configurations, Poughkeepsie is the configuration as described in Figure 3 on page 9, and all the other nodes located around the world (which are not depicted) are simple single-system configurations. In the network, a store manager in Pleasant Valley can order an item, the Poughkeepsie office (the main office for the Hudson Valley region) tallies the orders from this store and its other six stores and submits its order to the clothier chain headquarters located in New York City. The order is then forwarded to the London export office, where the item is procured and shipped. The request can be routed through Poughkeepsie and New York to London for accounting purposes or it could have been sent directly to London if the business is organized in that manner. The request and subsequent confirmation of the order is not instantaneous because of the distance and the **traffic** in the system but is faster and a more efficient method of doing business than a telephone conversation, particularly if you further consider time zone differences. Traffic refers to the number of users, requests, jobs, and data currently being routed across available teleprocessing lines.

Individual nodes in a network need not all be z/OS–JES2 nodes. As Figure 5 on page 12 illustrates, each node can contain different processing systems; nodes with various levels of z/OS–JES2, z/OS–JES3, VM/RSCS, and VSE/POWER can all be linked in a network. This is true because the NJE formats and protocols used by each of these systems are, by design, release and product independent.

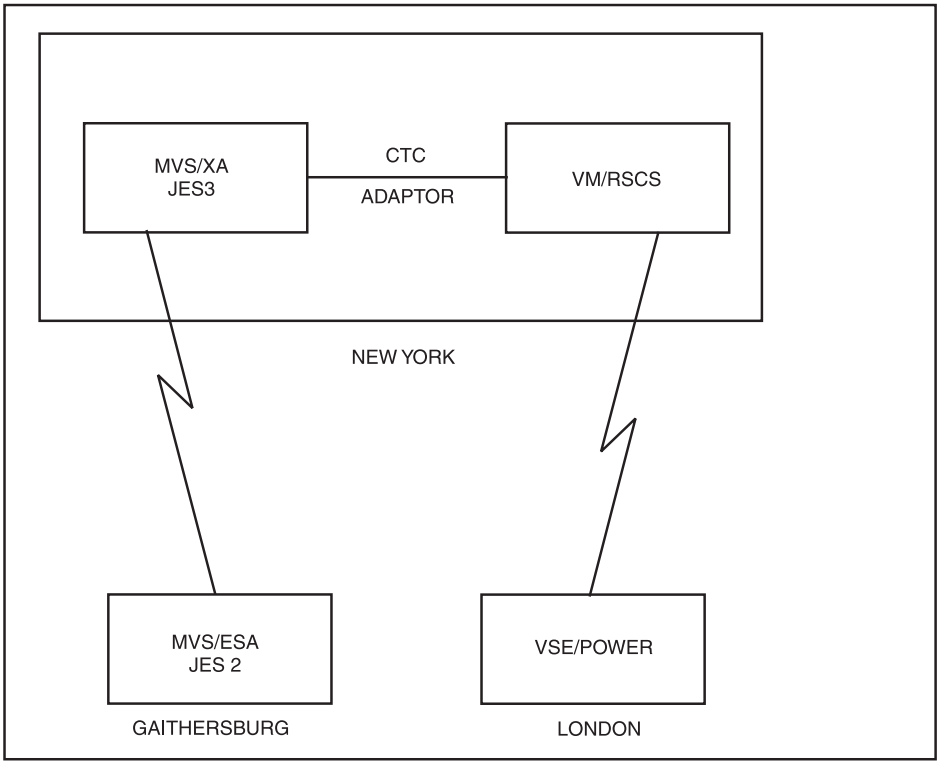


Figure 5. A Network of Various Processing Systems

Chapter 3. JES2 job processing and functions

This chapter answers the following questions:

- During the life of a job, what processing does JES2 do for z/OS?
- What are some of the major JES2 functions that make it special?

JES2 is responsible for all phases of job processing and monitors the processing phase. This chapter outlines the six phases and presents an overview of some of the major JES2 functions.

Phases of job processing

The z/OS (the base control program) and JES2 share responsibility in a zSeries system. JES2 is responsible for job entry (input), the base control program for device allocation and actual job running, and finally JES2 for job exit (output).

Figure 6 presents a view of the six phases, followed by a description of each.

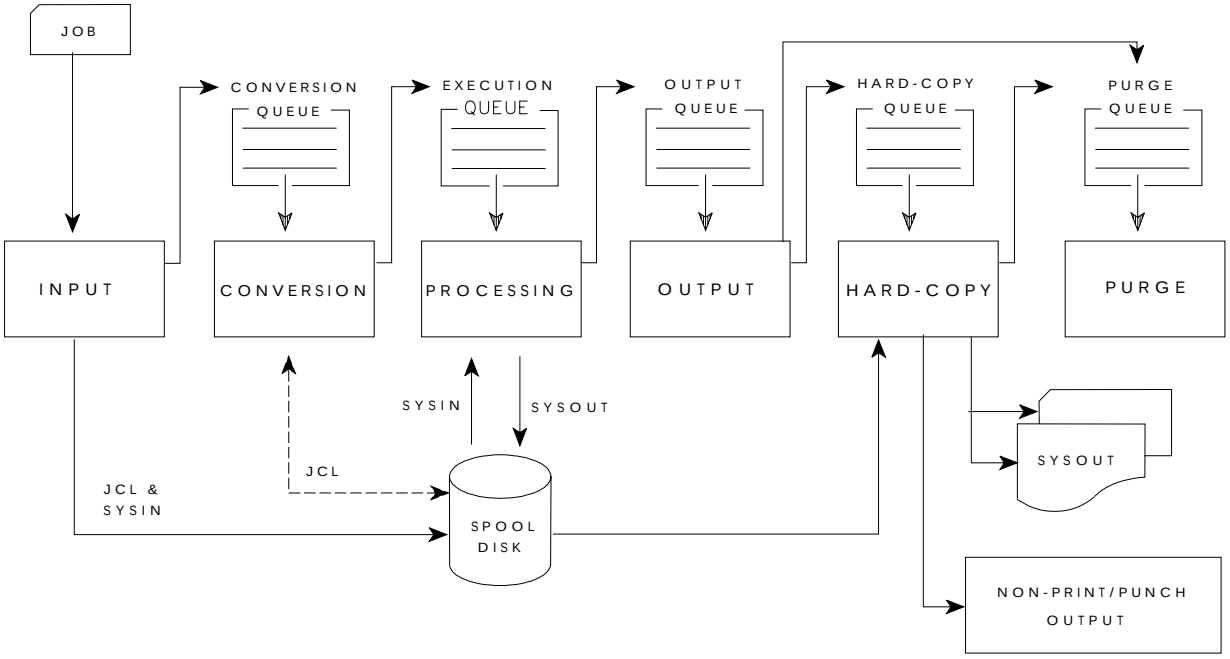


Figure 6. Job Processing Phases

The job queues contain jobs:

- Waiting to run - conversion queue
- Currently running - execution queue
- Waiting for their output to be produced - output queue
- Having their output produced - hard-copy (print/punch) queue
- Waiting to be purged from the system (following completion of all processing) - purge queue.

Input phase

JES2 accepts jobs (in the form of an input stream) from input devices such as card readers, remote terminals, or other programs. Input streams can also come from other nodes in a job entry network and from internal readers. An **internal reader** is a program that other programs can use to submit jobs, control

statements, and commands to JES2. Any job running in z/OS can use an internal reader to pass an input stream to JES2, and JES2 can receive multiple jobs simultaneously through multiple internal readers.

z/OS uses internal readers, allocated during system initialization, to pass to JES2 the job control language (JCL) for started tasks, START and MOUNT commands, and TSO LOGON requests.

The system programmer defines internal readers used to process all batch jobs other than STCs and TSO requests. JES2 initialization statements define these internal readers which JES2 also allocates during its initialization processing. The internal readers for batch jobs can be used for STCs and TSO requests, if not processing jobs.

As JES2 reads the input stream, it assigns a job identifier to each job and places each job's JCL, optional JES2 control statements, and SYSIN data onto DASD data sets called spool data sets. JES2 then selects jobs from the spool data sets for processing and subsequent running. See "Spooling" in Chapter 2, Chapter 2, "Scope of control and configurations," on page 5 for an explanation of spool data sets and the concept of spooling.

Conversion phase

JES2 uses a converter program to analyze each job's JCL statements. The converter takes the job's JCL, merges it with JCL from a procedure library (such as SYS1.PROCLIB), and converts the composite JCL into converter/interpreter text that both JES2 and the job scheduler functions of z/OS can recognize. JES2 then stores the converter/interpreter text on the spool data set. If JES2 detects any JCL errors, JES2 issues messages, and the job is queued for output processing rather than run. If there are no errors, JES2 queues the job for execution. JES2 supports multiple converters; therefore, jobs may not always be processed in a first-in-first-out (FIFO) order. When work load manager (WLM) batch management is in use, JES2 queues the job according to its arrival time.

Processing phase

In the processing phase, JES2 responds to requests for jobs from the z/OS initiators. JES2 selects from a job queue, jobs that are waiting to run and sends them to z/OS.

By recognizing the current processing phase of all jobs on the job queue, JES2 can manage the flow of jobs through the system.

JES2 Job Scheduling: To process the jobs on the job queue, JES2 communicates with an initiator. An **initiator** is a system program that starts a job to allow it to compete for system resources with other jobs that are already running.

Initiators are controlled by JES2 or by z/OS workload management (WLM).

- JES2 initiators are started by the operator or by JES2 automatically when the system initializes. The initiators select jobs based on the job class(es) that are assigned to the initiator and the priority of the queued jobs. The installation associates each initiator with one or more job classes in a way to encourage the efficient use of available system resources.
- WLM initiators are started by the system automatically based on the performance goals, relative importance of the batch workload, and the capacity of the system to do more work. The initiators select jobs based on their service class and the order they were made available for execution.

After JES2 selects a job and passes it to the initiator, the initiator invokes the interpreter to build control blocks from the converter/interpreter text that the converter created for the job.

The initiator then allocates the resources specified in the JCL for the first step of the job. This allocation ensures that the devices are available before the job step starts running. The initiator then starts the program requested in the JCL EXEC statement.

Priority Aging: When all initiators are busy, throughput of certain jobs might fall below normal expectations. To help in these situations, JES2 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.

JES2-Base Control Program Interaction: JES2 and the base control program communicate constantly to control system processing. The communication mechanism, known as the subsystem interface, allows z/OS to request services of JES2. For example, a requester can ask JES2 to find a job, do message or command processing, or open (access) a SYSIN or SYSOUT data set. Further, the base control program notifies JES2 of events such as messages, operator commands, the end of a job, or the end of a task.

Output phase

JES2 controls all SYSOUT processing. **SYSOUT** is system-produced output; that is, all output produced by, or for, a job. This output includes system messages that must be printed and data sets requested by the user that must be printed or punched. After a job finishes, JES2 analyzes the characteristics of the job's output in terms of its output class and device setup requirements; then JES2 groups data sets with similar characteristics. JES2 queues the output for print or punch processing.

Hard-copy phase

JES2 selects output for processing from the output queues by output class, route code, priority, and other criteria. The output queue can have output that is to be processed locally or output to be processed at a remote location (either an RJE workstation or another node). JES2 handles each of these situations in different ways.

- Local Output:

  When output is to be processed at a local or remotely-attached output device, JES2 uses these local and remotely-attached output devices to produce a job's output. JES2 queues a job's print and punch data sets on the output queue for the local and remote output devices. The active devices, that are attached locally or through RJE connections, select the output data sets with characteristics that best match their selection criteria.
- Network Job Entry Output:

  Job output passing through to another JES2 node resides on the network output queue. JES2 selects a job's output from the network output queue for transmission to another node based upon the priority and the desirability of reaching the output-processing node over the available transmission line. After the receiving node signals that it has accepted total responsibility for the output, the transmitting JES2 node releases the resources used to represent the output.

After processing all the output for a particular job, JES2 puts the job on the purge queue.

Purge phase

When all processing for a job completes, JES2 releases the spool space assigned to the job, making the space available for allocation to subsequent jobs. JES2 then issues a message to the operator indicating that the job has been purged from the system.

JES2 capabilities and functions

JES2 (in conjunction with VTAM) is the link between the Time Sharing Options/Extensions (TSO/E) user and z/OS. As such, it is very externally oriented. That is, it is visible to the data processing personnel and provides the ability to specify and tailor many installation-specific functions through JES2 initialization statements and JES2 commands. These statements and commands are analogous to the knobs, buttons, and handles with which you control a machine. You set them, knowing what response the machine will provide, but you need not be concerned with how the button sets the gears and belts in motion.

This section provides an overview of the following major functions JES2 provides to manage its job input/output responsibilities for z/OS; all of which are under system programmer control.

- Getting work out of z/OS
- Selecting work to maximize efficiency
- Offloading work and backing up the system
- Supporting advanced function printers
- Providing security.

All are discussed in greater depth in *z/OS JES2 Initialization and Tuning Guide*.

Getting work out of z/OS

As the central point of control over the job output (or exit) function, JES2 controls output devices: local and remote printers, punches, and card readers. You can use JES2 initialization statements to define each device. All are directly under JES2's control with the exception of those printers that operate under the Print Services Facility (PSF). These printers provide all-points-addressable capabilities, and although defined by JES2, are driven by PSF. PSF thereby assumes the processing overhead that JES2 typically performs to support printer operation. (See "Supporting Advanced Function Printers", for an introduction to PSF.)

Printed and punched output can be routed to a variety of devices in multiple locations. The control JES2 exercises over its printers ranges from the job output classes and job names from which the printer can select work to such specifications such as the forms on which the output is printed. This control allows the system programmer to establish the job output environment most efficiently without causing unnecessary printer backlog or operator intervention.

Through JES2, the installation defines the job input classes, reader specifications, and output device specifications. As a result, JES2 is the central point of control over both the *job entry* and *job exit* phases of data processing.

Selecting work to maximize efficiency

To minimize contention for output devices, JES2 allows the installation to define work selection criteria that can be specific for each output device (local and remote printers and punches and offload devices). The work selection criteria on the respective device initialization statements define the:

- Specification of job and output characteristics that JES2 considers when selecting work for an output device
- Order of importance (priority) of the selection characteristics
- Characteristics of the printer and job that must match exactly.

JES2 initializes the setup characteristics of a device based on the specifications supplied on a device (printer/punch) initialization statement.

A job's output is grouped based on the data sets' output requirements. These requirements are defined by the job submitter with the job's job control language (JCL) or by JES2-supplied defaults.

When selecting work to be processed on a device, JES2 compares the device's characteristics to the output requirements associated with the pieces of work awaiting processing. In addition, JES2 compares each piece of work against all others to determine the best match for the device. If work output requirements are found that match a device's characteristics, JES2 sends the output associated with that piece of work to that output device. If a piece of work and an output device cannot be paired, the work will not be selected for output until the operator changes the output device specifications or the output requirements for the piece of work.

Work selection control provides efficient use of output devices by allowing them to print or punch specified output classes without requiring continual operator intervention of setup characteristics such as forms or

print trains (the piece of hardware that carries the print type). Also, when too much work is adversely affecting system performance, the system programmer or operator can specify work to be **offloaded** (that is, moved off the work queues and temporarily out of contention for system resources. See the topic, “Offloading work and backing up the system” for an explanation of offloading.)

Offloading work and backing up the system

All input jobs and system output are stored on spool. JES2 gives your installation the capability to offload data from and later reload data to the spool. This is useful if you need to:

- Preserve jobs and SYSOUT across a cold start, which entails the total rebuilding of work and output queues.
- Migrate your installation to another release of JES2. (You can use spool offload to reload the spool to the new or previous release.)
- Converting to another DASD type for spool.
- Archive system jobs and SYSOUT.
- Relieve a full-spool condition during high-use periods. (You can reload at a later time.)
- Provide a back-up for spool data sets.
- Back-up network connections.

Many selection criteria can be used to limit the scope of the spool offload operation. For example, work can be selected based on: a system on which the job is to run, job number, job disposition (for example, only held data sets), destination, class, or many other criteria. These selection criteria can be changed by operator command following your initial specification with JES2 initialization statements.

Supporting advanced function presentation (AFP) printers

Advanced function presentation (AFPs) printers, such as the 3800-3 and 3820 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) and used by JES2. JES2 and PSF work together by communicating through the functional subsystem interface (an extension of JES2 device control that removes the need for JES2 to be dependent on specific characteristics of the printing device. This independence allows JES2 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. [2]

Providing security

Security in a data processing environment involves controlling and auditing access to resources that are important to your installation. In the JES2 environment, these resources include:

- JES2-owned data sets
- Input (from nodes, remote workstations, readers, offload devices, and commands)
- Job names
- System input/output residing on spool (SYSIN/SYSOUT)
- Output (to nodes, printers, punches, remote workstations, and offload devices).

---

2. 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.

JES2 provides a basic level of 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 give 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's security procedures.

The control available through initialization statements can be broadened by implementing several JES2 exits available for this purpose. You can implement a more complete security policy by using the system authorization facility (SAF) component of the base control program and a security product such as Resource Access Control Facility (RACF). SAF provides a link to the security product to define any additional security controls your installation may require.

JES2 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, JES2 bypasses its own security processing.

Supporting APPC

JES2 processes SYSOUT data sets for **APPC transaction programs**, which are application programs that communicate with other APPC transaction programs through the Advanced Program-to-Program Communication/MVS (APPC/MVS). For details about APPC/MVS, see *z/OS MVS Planning: APPC/MVS Management*. SYSOUT processing is the only function JES2 provides for APPC transaction programs.

Chapter 4. Tailoring your JES2 system

This chapter answers the following questions:

- How can JES2 be tailored at initialization to meet my installation's needs?
- Do I need to get so involved in customizing JES2 that I need to write 'operating system' code?
- What other means can I use to enhance JES2 if it does not meet all my specific processing needs?
- What are JES2 'table pairs'?
- What's the difference between an IBM-defined exit and an installation-defined exit?

JES2 is designed to be tailored to meet an installation's particular processing needs. You can address your basic customization needs when you create the JES2 initialization data set. If you find that this control over JES2 processing to be insufficient, JES2 also provides exits and table pairs to change many JES2 functions or processes.

If you need to modify JES2 processing above the capability provided by initialization statements, and elect to do so through installation-written code, such code should be isolated from the IBM-shipped source code. Changes to JES2 processing implemented through direct source code modification are error prone, counter-productive during migration to future releases, and can prove to be very time consuming when debugging, diagnosing, and applying IBM-written fixes (program temporary fixes (PTFs) and authorized program analysis report (APARs) fixes) to code. Further, alteration of JES2 processing in this manner complicates IBM service assistance. Therefore, JES2 provides several means of allowing you to tailor its processing without direct source code modification.

Recommended methods for tailoring JES2 processing include JES2 table pairs, IBM-defined exits, and installation-defined exits. A general discussion of each is provided later in this chapter . See *z/OS JES2 Installation Exits*for a complete description of each IBM-defined exit.

JES2 initialization data set

Because every installation that uses JES2 to manage its work input and output is unique, so too are the requirements each installation has on JES2. To meet these needs, JES2 is highly tailorable, and with minimal effort, a system programmer can customize most JES2 functions by changing and using the **JES2 initialization data set** that is provided with the product in the HASIPARM member of the SYS1.SHASSAMP data set. Although this data set will not run as shipped without some installation additions, it is a valuable model that can save hours of system programmer input time.

With a set of approximately 70 initialization statements, you can control all JES2 functions. The JES2 initialization data set provides all the specifications that are required to define output devices (printers and punches), job classes, the JES2 spool environment, the checkpoint data sets, the trace facility, and virtually every other JES2 facility and function.

Each initialization statement groups **initialization parameters** that define a function. The use of most JES2 initialization statements is optional. That is, you need not define them unless you need to implement or tailor a particular function. Further, many of the parameters provide default specifications that allow your installation to perform basic JES2 processing with no explicit definition on your part. JES2 requires only a minimal set of initialization statements (or parameters) that you define when first installing JES2.

Minimum required statements

When you are first getting started, you need not define or even be concerned about some of the more sophisticated processing environments such as a multi-access spool complex, nodes, or remote workstations. You are building a base on which your installation can grow. There is no need to be

overwhelmed with the potential complexity of your JES2 system. As your installation grows, you will likely use more and more of JES2's available function. To assist the task of JES2 initialization, there is a sample initialization data set shipped with the product.

The sample data set shipped in SYS1.PARMLIB requires only the addition of installation-defined devices and installation-specific values. It contains all the JES2 initialization statements and the defaults for all parameters for which they are provided. A new installation would initially want to delete many of these definitions. A more realistic initialization "starter set" is shipped by IBM if you request Custom Built Product Delivery Offering (CBPDO) [3] to build your system.

JES2 table pairs

Table pairs provide a facility to change, delete, or add to JES2 processing, function, or both. Changes made to JES2 processing using table pairs are generally less prone to error than are changes made through installation exits. This is true because JES2 macros generate the tables and generally require you to write less code to be run.

A number of JES2 functions (such as initialization statement processing, command processing, and message building) use tables. You can customize these JES2 functions, and others, by extending their associated tables. JES2 examines two tables, known as a **table pair**. The first table (the **JES2 table**) provides the default processing specifications; the second table (the **user table**) is used to extend, change, or delete the default processing specifications. For example, you can add your own JES2 commands and messages, add a new initialization statement or parameter, abbreviate the length of a JES2 command, or delete an unnecessary command to prevent its accidental misuse.

To simplify the use of this facility you can use the JES2 default tables as templates for construction of installation-written tables. Depending on the table(s) from which you choose to add, change, or delete, using table pairs generally takes less detailed knowledge of JES2 internal structure than does writing an exit.

Table pairs do not replace the need for exits. Table pairs and exit points can provide added capability either independently or in conjunction with one another.

JES2 IBM-defined exits

JES2 exits provide a clean, convenient, relatively stable interface between JES2 and your installation-written code. Installation-written exit routines are invoked from standard JES2 processing at various strategic locations in JES2 source code. These strategic locations in JES2 source code are called **exit points**. A JES2 exit is established by one or more exit points.

JES2 can have up to 256 exits; IBM has defined only a number of these to allow customization of the most commonly modified functions. New exits are often added when new function is added with each new release of the product. These new exits provide a facility for you to alter the new processing as appropriate to meet your installation's needs.

The exits allow a wide range of JES2 customization. For example, you can add code to:

- Design your own print job separator page
- Verify or change jobs submitted by TSO/E users
- Change or disallow selected commands
- Define alternate processing for a job that uses too many resources

3. CBPDO is an IBM offering which builds, tailors, and delivers a full MVS operating system, such that all you need do is install and use the system. CBPDO is useful for all new MVS installations to ease the entire migration effort from the previous operating system. This option is particularly useful to new MVS installations that may lack the expertise to perform this task.

- Provide increased security and password checking for remote terminals and system data sets.

For the **IBM-defined exits** you need only write your own exit routines and incorporate them through use of two initialization statements. IBM has already placed these JES2 exit points in the code. To ensure a proper implementation you should thoroughly understand the IBM-defined exit and its JES2 operating environment. A comprehensive description of each exit is presented in *z/OS JES2 Installation Exits*.

If you find that none of these exits meet your installation's needs you can establish your own exit point and provide your own exit routine. This, of course, requires a more thorough knowledge of JES2 processing than the use of an IBM-defined exit point. But remember, the use of exits is still far superior to adding in-line source code modification that may necessitate the addition of many lines of code; code that cannot be dynamically disabled as can an exit.

JES2 installation-defined exits

The JES2 exit facility allows you to establish your own exits, should the IBM-defined exits not suffice. Exits established by you are modifications to JES2 and are called **installation-defined** exits. You define them by placing an exit point at appropriate points in the JES2 code (or in your own exit routine code). (IBM-defined exit points are established in the same manner, but the exit point is placed appropriately for the specific function for which it is intended.) Note, however, that implementing your own exit can be considerably more difficult than writing an exit routine for an IBM-defined exit. See *z/OS JES2 Installation Exits*for explanations of JES2 coding conventions, restrictions, how JES2 exits are packaged, and to *z/OS JES2 Macros*for the JES2 macros that are available and intended for your use.

Chapter 5. Interacting with JES2

This chapter answers the following questions:

- How does the data processing staff communicate with JES2?
- How much control do operators have over JES2 when it is running?
- Are there ways to automate some of the routine processing needs?
- If JES2 experiences problems, how will it inform the operator?
- Are there any tools available to assist diagnosis and recovery?

JES2 operations

To help you maintain your overall work environment, JES2 provides an interactive means to control much of its function and the devices under its control. Although the JES2 environment is initially determined through initialization statements, you can alter many of those definitions as the system's workload changes or your installation adds new devices or needs to redefine its overall configuration. JES2 provides commands to request current status of devices and functions and JES2 responds with informational messages. Based on those messages, the operator, system programmer, or automated operations program, such as NetView or Automated Operations Control/MVS (AOC/MVS), can issue further commands to change processing (such as, implement a newly written exit routine), start or stop a printer, or start diagnostic functions (such as the JES2 trace facility). Much of the processing can be changed without disrupting the remainder of the system. The following sections provide a brief overview of the control you have over JES2. (See *z/OS JES2 Commands* for an explanation of JES2 commands, and *z/OS JES2 Messages* for the text and explanation of JES2 messages.)

Operator control

Almost all JES2 initialization statement definitions can be changed by operator commands. These commands are available to both operators and system programmers to allow them to change current definitions. The system programmer can implement certain security features or customization techniques to limit the degree of control an individual or group can have over the operating system.

As your JES2 complex becomes more sophisticated, you might connect your system to others to form a network of systems. You can use operator commands to control the lines that connect the separate systems and define the separate systems to yours. This is typically a very dynamic environment, as different systems are added or deleted from the network because of maintenance, hardware reconfiguration requirements, workload balancing, or the need to access a data base at a different location. JES2 permits you to use commands to alter most of your original network definition, as required.

If this dynamic control were not available, the operator or system programmer would need to change the initialization data set definition, stop JES2 processing, and then restart the system so those changes can take effect. This is, of course, required when redefining some areas of processing, but it denies system users valuable time.

Stopping and restarting JES2

However, there are instances when JES2 must be stopped and restarted either by a warm or cold start. For example, redefining the number of systems in a network job environment requires a warm start. A **warm start** is a restart of JES2 that does not cause the current work and output queues to be rebuilt, but does provide the required changed information to be propagated and used by all necessary components. A warm start is far superior to a cold start.

The definition (or redefinition) of some JES2 facilities and resources require that the JES2 system be totally shut down. JES2 must be restarted with a cold start to allow all component systems to be aware of the changed facilities and resources. A **cold start** is a JES2 restart that causes all current work and output

queues to be lost and rebuilt with new data. The time to restart JES2 in this manner is based on the work in the system and, if not scheduled, causes a disruption in data processing services.

JES2 commands

JES2 processes its initialization statements and commands in a way that allows most initialization statements to be changed by operator command. The following is a partial list of the control that JES2 commands have over JES2 processing. Operator commands can be used to:

- Add function and functional subsystems
- Modify previously defined processing, such as: output definition, the dynamic alteration of the checkpoint definition, enabling installation-defined exits, offload devices, printer and punch characteristics, and job characteristics
- Delete function, and network systems, exits, and diagnostic traces
- Start, stop, and halt devices under JES2's control
- Assign units to local printers, punches, card readers, and lines or reassign units to these devices
- Display current facility and device definition.

All JES2 commands are entered using standard MVS command interfaces (such as through an MVS console or within the JES2 initialization data set). The command prefix character, which defaults to a dollar sign ($), distinguishes JES2 commands and messages from other components of the operating system. For commands, the prefix character defines the scope of the command as being JES2 only; for messages, the prefix character is informational in that it designates that the message was issued by the JES2 component.

Automatic commands

The operator can specify that certain commands or strings of commands take effect automatically at specific times or at regular intervals. Automatic command processing can be used to provide status displays and to lessen the operator's work for common, preset routines or schedules. For example, if an installation normally does one specific kind of work at 8 AM and another at 9 AM every day, it is possible to preset automatic command processing to issue the operator commands that would ordinarily be necessary at those times. The set of commands can be entered directly into the initialization data set, and the command processor will then issue them daily. Such commands can be a single command, a rather complicated set of commands, or just commands that are tedious (and error prone) when entered manually.

The use of the automatic command facility is, of course, optional, and provided to ease the operator workload. A second method of automating the operating system is to prompt system response based on JES2 messages through the use of automation products such as Automated Operations Control/MVS (AOC/MVS).

Automating JES2 operations

JES2 messages (similar to all MVS messages) contain various elements. All messages contain a unique identifying number, some messages contain non-variable text, others contain variable text (the text is based on the specific error condition or status), others contain specific reason codes (a numeric value - the explanation of which is documented in *z/OS JES2 Messages* and some contain combinations of the preceding elements. Based on the message number, reason code value or variable text, an installation can interpret the system status and have a programmable console automatically issue the required commands to respond to the situation. Also, there are other IBM products, NetView and the application product Automated Operations Control/MVS (AOC/MVS), that can be installed to help you automate your operation based on your installation's policies regarding the handling of particular message or error conditions.

For example, AOC/MVS can be used to issue the appropriate JES2 commands to relieve a spool shortage condition or issue the appropriate MVS command(s) to complete the shutdown of JES2 after a serious error condition.

JES2 communication mechanisms

JES2 Tracing Facility

To record register contents and data at specific points in JES2 processing, you can use the JES2 tracing facility. JES2 has defined a number of the 255 trace types that can be defined; new ones are occasionally added to new product releases as their need arises. Here, again, is another JES2 facility that your installation can augment. (Your installation can add installation-specific trace points if the available trace functions do not meet your requirements.)

The use of the facility is optional. Tracing is initially activated under system programmer control (by specifying initialization statements) and subsequently controlled through several operator commands. Generally, trace points are only activated as a diagnostic tool for short periods of time.

Traced data can be accessed either through a dump of unformatted trace tables or in formatted system output. (See *z/OS JES2 Initialization and Tuning Guide* and *z/OS JES2 Diagnosis* for further information regarding the initialization and use of the JES2 trace facility.)

JES2-IPCS formatting

To facilitate diagnosis of errors that require the viewing of storage dumps or control blocks, both the base control program and JES2 components exploit the interactive problem control system (IPCS). When diagnosing a problem, IPCS panels provide you the opportunity to continue the diagnostic process through the base control program and into selected JES2 data areas.

IPCS is a menu-driven facility that lets you interactively select control blocks that you need to examine. Selected control blocks are formatted, displayed, and available for printing. (See *z/OS JES2 Diagnosis* for a discussion of IPCS-JES2 support.)

Appendix A. JES2 Publications

The following list of JES2 publications includes the basic set of those publications provided with the product.

z/OS JES2 Initialization and Tuning Guide, SA22-7532
Contains descriptions of JES2 initialization process and the JES2 facilities and task descriptions for the system programmer to use the JES2 initialization statements and their parameters to use and tune those facilities.

z/OS JES2 Initialization and Tuning Reference, SA22-7533
Contains complete descriptions of the JES2 initialization statements and their parameters to include syntax and defaults. References tables provide lists of hardware supported by JES2, JES2 values for optimizing hardware usage, and a copy of the sample initialization data set shipped in SYS1.VnRnMn.SHASSAMP.

z/OS JES2 Commands, SA22-7526
Describes how to operate JES2. The book explains the use of each JES2 operator command in detail.

z/OS JES2 Messages, SA22-7537
Describes all JES2 messages and suggests appropriate operator and system programmer responses.

z/OS JES2 Installation Exits, SA22-7534
Provides information to customize JES2 through JES2 installation exits.and describes how to use JES2 programmer macro instructions.

z/OS JES2 Macros, SA22-7536
Describes how to use JES2 programmer macro instructions.

z/OS JES2 Diagnosis, GA22-7531
Describes procedures to use and interpret the output of diagnostic aids/tools and their output. The book assists the system programmer to: identify a problem, collect information about the problem, report the problem to IBM, and fix the problem.

Appendix B. Accessibility

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.

z/OS information

z/OS information is accessible using screen readers with the BookServer/Library Server versions of z/OS books in the Internet library at:

`http://www.ibm.com/servers/eserver/zseries/zos/bkserv/`

Notices

This information was developed for products and services offered in the USA. 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
USA

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

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, 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:

IBM Corporation
Mail Station P300

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.

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

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

- AFP/VTAM
- Advanced Function Printing (AFP)
- AnyNet
- BookManager
- CBPDO
- CICS
- DFSMS/MVS
- DFSMSdfp
- DFSMSdss
- DFSMShsm
- DFSMSrmm
- ESCON
- ES/3090
- ES/9000
- GDDM
- IBM
- IBMLink
- IMS
- MVS/DFP
- MVS/ESA
- NetView
- OS/2
- OS/390
- PR/SM
- Print Services Facility (PSF)
- RACF
- Resource Link
- RMF
- SOMobjects
- SP
- SP1
- SP2
- SystemView
- System/36
- System/360
- System/370
- System/390
- VTAM
- z/OS

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Glossary

This glossary defines technical terms and abbreviations used in JES2 documentation. If you do not find the term you are looking for, refer to the IBM Terminology Web site at: http://www.ibm.com/terminology.

ACF. See Advanced Communications Function.

address space. The range of addresses available to a computer program or process. Address space can refer to physical storage, virtual storage, or both. See also virtual address space.

Advanced Communications Function (ACF). A group of IBM licensed programs that use the concepts of Systems Network Architecture (SNA), including distribution of function and resource sharing.

Advanced Function Presentation (AFP). A set of licensed programs, together with user applications, that use the all-points-addressable concept to print data on a wide variety of printers or to display data on a variety of display devices. AFP includes creating, formatting, archiving, retrieving, viewing, distributing, and printing information.

Advanced Program-to-Program Communication (APPC). An implementation of the SNA LU 6.2 protocol that allows interconnected systems to communicate and share the processing of programs.

affinity. An association between objects that have some relationship or dependency upon each other.

AFP. See Advanced Function Presentation.

aging. The process of increasing a job's priority.

all-member warm start. A warm start that is specified by the operator that occurs when no other no other members of the configuration are active or there is only one member in the configuration.

allocate. To assign a resource to a specific task.

all-points addressability. The capability to address, reference, and position text, overlays, and images at any defined position or picture element (pel) on the printable area of a page. This capability depends on the ability of the hardware to address and to display each picture element.

APAR. See authorized program analysis report.

APPC. See Advanced Program-to-Program Communication.

artificial JQE (JQA). A control block containing a summary of information from a job control table (JCT) entry that consists of the base job queue element (JQE), the job queue element extension (JQX), and additional fields in the artificial JQE (JQA).

authorized program analysis report (APAR). A request for correction of a defect in a current release of an IBM-supplied program.

automatic restart. A restart that takes place during the current run, that is, without resubmitting the job. An automatic restart can occur within a job step or at the beginning of a job step. See also checkpoint restart, deferred restart.

automatic volume recognition (AVR). A feature that allows the operator to mount labeled volumes on available I/O devices before the volumes are needed by a job step.

AVR. See automatic volume recognition.

background. The conditions under which low-priority, noninteractive programs are run. See also foreground.

background job. A low-priority job, usually a batched or non-interactive job.

batch processing. A method of running a program or a series of programs in which one or more records (a batch) are processed with little or no action from the user or operator.

binary synchronous communication (BSC). A data-communication line protocol that uses a set of transmission control characters and control character sequences to send binary-coded data over a communication line.

binary synchronous communication adapter (BSCA). A synchronous communication adapter that supports point-to-point or multipoint operation.

binary synchronous transmission (bisync). Data transmission in which synchronization of characters is controlled by timing signals generated at the sending and receiving stations.

bind request. A request to establish a connection between systems or logical units.

bisync. See binary synchronous transmission.

broadcast data set. Under TSO, a system data set containing messages and notices from the system operator, administrators, and other users.

BSC. See binary synchronous communication.

BSCA. See binary synchronous communication adapter.

burst. To separate continuous-forms paper into individual sheets.

cataloged data set. A data set that is represented in an index or hierarchy of indexes that provide the means for locating it.

cataloged procedure. A set of job control language (JCL) statements that has been placed in a library and that is retrievable by name.

CCW. See channel command word.

centralized control. A type of control in which in which all the primary station functions of the data link are centralized in one data station. See also independent control.

central storage. Storage that is an integral part of the processor unit. Central storage includes both main storage and the hardware system area.
UNIX-experienced users refer to central storage as memory.

CES. See connection event sequence.

change log. The area of the checkpoint data set that contains the specific control blocks changed by the last member of the multi-access spool configuration to own the checkpoint data set.

channel command word (CCW). In zSeries systems, an 8-byte command issued to the channel subsystem by a central processor and operating asynchronously with the issuing processor.

checkpoint. A place in a program at which a check is made, or at which a recording of data is made to allow the program to be restarted in case of interruption.

checkpointing. The periodic copying of processing information to the checkpoint data set. Checkpointing ensures that information about in-storage job and output queues is not lost in the event of a hardware or software error.

checkpoint reconfiguration. A process that allows a user to dynamically redefine checkpoint data-set specification for the JES multi-access spool (MAS) configuration.

checkpoint reconfiguration dialog. An interactive form of a JES2 checkpoint reconfiguration that directs the reconfiguration process with replies to a series of WTOR messages.

checkpoint restart. The process of resuming a job at a checkpoint within the job step that caused abnormal termination. The restart can be automatic or deferred. A deferred restart requires that the job be resubmitted. See also automatic restart, deferred restart, step restart.

checkpoint/restart facility. (1) Under TCAM, a facility that records the status of the teleprocessing network at designated intervals or following certain events. Following system failure, the system can be restarted and continue without loss of messages. (2) A facility for restarting execution of a program at some point other than at the beginning, after the program was terminated due to a program or system failure. A restart can begin at a checkpoint or from the beginning of a job step, and uses checkpoint records to reinitialize the system.

checkpoint write. Any write to the checkpoint data set. A checkpoint write is a primary, intermediate, or final write that updates a checkpoint data set.

cold start. A process in which the system is initialized. All jobs that were active or in the job queue at the time of the cold start are removed from the system. See also warm start.

compatibility mode. A mode of operation in which a device can simulate the function of another device or model. The device will function like a different device of the same type, ignoring some or all of the additional features that the device might possess. Compatibility mode permits a migration between devices with minimal impact on programs that have device dependencies. See also page mode.

complex. The maximum set of hardware and software resources that support one or more images of a single operating system.

configuration. The manner in which the hardware and software of a system, subsystem, or network are organized and interconnected.

connection event sequence (CES). This value is copied to full name (NCC) records and used by the path manager to determine the most current record pertinent to the tracking of full name (NKJE) connections.

control statement. A statement placed into an input stream to identify special processing options for jobs

data set forwarding. The dynamic replacement of the specifications of the checkpoint data set with new specifications.

data set separator page. A page of printed output that delimits data sets.

deallocate. To release a resource that is assigned to a specific task.

deferred printing mode. A printing mode that spools output through JES to a data set instead of printing it immediately. Output is controlled using job control language (JCL) statements.

deferred restart. A restart performed by the system when a user resubmits a job. The operator submits the restart deck to the system through a system input reader. See also automatic restart, checkpoint restart.

dependent job control (DJC). A method of handling multiple jobs that must be run in a specific order because of job dependencies. DJC manages jobs that are dependent upon one another.

despooling. The process of reading records from the spool into central storage. During the despooling process, the physical track addresses of the spool records are determined.

DESTID. See destination identifier. See also explicit destination, symbolic destination.

destination identifier (DESTID). The 8-character subscript on the DESTID initialization statement or command that corresponds to a combination of a first-level destination and a second-level destination that determines where data should be sent in a JES2 installation. A DESTID can be either a symbolic destination or an explicit destination. See also explicit destination, symbolic destination.

destination node. The node to which a request or data is sent.

device partitioning. A pool of devices (called a fence) that is used exclusively by a set of jobs in a specific job class. Using device partitioning, the device usage of an installation can be tailored to its anticipated workload.

DJC. See dependent job control.

DUAL mode. A checkpointing mode that provides the alternate use of two primary checkpoint data sets (CKPT1 and CKPT2). The datasets are referred to as the to-be-read-from and to-be-written-to data sets.

dump. (1) To copy the contents of all or part of visual storage for the purpose of collecting error information. (2) To record or copy, at a particular instant, data from one storage device onto another storage device in order to protect the data and debug the program.

dynamic allocation. Assignment of system resources to a program when the program is executed rather than when it is loaded into main storage.

dynamic connection. A connection created at the time of sign-on or using the network connection control (NCC) record sent from another node.

dynamic table. An installation-defined table that is used to extend, modify, or delete the default processing specifications. See also table pair.

EBCDIC. See Extended Binary Coded Decimal Interchange Code.

ECSA. See extended common system area.

end of block (EOB). A code that marks the end of a block of data.

end-of-file (EOF). A code that signals that the last record of a file has been read.

EOB. See end of block.

EOF. See end-of-file.

execution node. The network job entry (NJE) node upon which a job is to be executed.

explicit destination. A destination identifier that refers to a specific route code. See also destination identifier, symbolic destination.

Extended Binary Coded Decimal Interchange Code (EBCDIC). A coded character set of 256 8-bit characters developed for the representation of textual data.

extended common system area (ECSA). A major MVS storage area above the 16 MB line, containing pageable system data areas addressable by all active virtual address spaces.

external writer. A program that performs output processing for data sets that are not eligible for processing by the primary job entry subsystem (JES). For example, an external writer might process a data set that has been directed to a printer and that will be stored on a device that is not supported by JES.

FCB. See forms control buffer.

final write. A write of the same information as the intermediate write that is performed at the end of the checkpoint cycle. See also intermediate write, primary write.

first-level destination. The part of a destination identifier that indicates a target node to which input is to be sent. See also second-level destination.

foreground. In TSO, the environment in which programs are swapped in and out of main storage to allow terminal users to share processing time.

forms control buffer (FCB). A buffer for controlling the vertical format of printed output. The FCB is a line-printer control that is similar to the punched-paper, carriage-control tape. For Advanced Function Presentation (AFP) printers, the forms control buffer is replaced by the page definition.

FSA. See functional subsystem application.

FSS. See functional subsystem.

full function mode. The state that permits a printer to produce page-mode output.

functional subsystem (FSS). An extension of JES that runs in an address space separate from the JES

address space. An FSS provides support for a function peripheral to JES processing, such as a peripheral device or other component.

functional subsystem application (FSA). An application that uses the support facilities of the functional subsystem (FSS) to communicate with JES.

global command. A command that is recognized and honored by any node in a JES2 network.

global processor. The processor that controls job scheduling and device allocation for a complex of processors.

HASP. See Houston Automatic Spooling Program.

host processor. A processor that controls a user application network.

host system. The data processing system to which a network is connected and with which the system can communicate.

hot start. A type of warm restart that is performed when JES terminates abnormally and an initial program load (IPL) has not yet occurred.

Houston Automatic Spooling Program (HASP). A mainframe spooling program that provides task management, job management, and data management functions.

IBM-defined exit. A location in source code at which IBM has added an exit point; an installation routine can receive control from the operating system at this IBM-defined exit. See also installation-defined exit.

impact printer. A printer in which printing is the result of mechanically striking the printing medium. (T) See also nonimpact printer.

independent control. The process by which each processor in a complex controls its own job input, scheduling, and job output. See also centralized control.

independent mode. A means of isolating a processor for testing purposes. A processor so designated will only process jobs that are both routed to it and are themselves designated to execute on a processor in independent mode.

initialization data set. A group of statements that are used when the system is initialized.

initialization parameter. An installation-specified parameter that is when the system is initialized.

initialization statement. An installation-specified statement that is used when the system is initialized.

initial program load. The process of loading the operating system and other basic software into main storage.

initiating task. The job management task that controls the selection of a job and the preparation of the steps of that job for execution.

initiator. The part of an operating system that reads and processes control statements from the system input device.

input service processing. The process of performing the following tasks for each job: reading the input data, building the system input data set, and building control table entries.

installation-defined exit. The location in source code at which an installation adds an exit point for an installation routine to receive control from the operating system. See also IBM-defined exit.

interface. A shared boundary between independent systems. An interface can be a hardware component used to link two devices, a convention that supports communication between software systems, or a method for a user to communicate with the operating system, such as a keyboard.

intermediate write. A type of update in which all changes made to checkpoint data are recorded. See also primary write, final write.

internal reader. A facility that enables jobs to be submitted to JES from time-sharing logons, started tasks, or other jobs.

interrupt. Suspension of a process, such as execution of a computer program, caused by an external event and performed in such a way that the process can be resumed.

JCL. See job control language.

JCS. See job control statement.

JES. See Job Entry Subsystem. See also JES2, JES3.

JES2. An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for execution, processes their output, and purges them from the system. In an installation with more than one processor, each JES2 processor independently controls its job input, scheduling, and output processing. See also Job Entry Subsystem, JES3.

JES2 table. A JES2-defined table that is used to specify the default characteristics of many of its initialization parameters, commands, and other externals. See also table pair.

JES3. An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for execution, processes their output, and purges them from the system. In complexes that have several loosely coupled processing units, the JES3 program manages processors so that the global processor exercises

centralized control over the local processors and distributes jobs to them using a common job queue. See also Job Entry Subsystem, JES2.

job. A separately executable unit of work.

job class. Any one of a number of job categories that can be defined.

job control language (JCL). A command language that identifies a job to an operating system and describes the job's requirements.

job control statement (JCS). A statement in a job that identifies the job or describes its requirements to the operating system.

Job Entry Subsystem (JES). An IBM licensed program that receives jobs into the system and processes all output data that is produced by jobs. See also JES2, JES3.

job output element (JOE). Information that describes a unit of work for the JES output processor and represents that unit of work for queueing purposes.

job priority. A value assigned to a job that, together with an assigned job class, determines the priority to be used in scheduling the job and allocating resources to it. (D)

job queue element (JQE). A control block containing a summary of information from a job control table (JCT) entry. JQEs move from queue to queue as work moves through each stage of processing. JQEs are used instead of JCT entries for the scheduling of work.

job separator page. A page of printed output that delimit jobs.

JOE. See job output element.

JQA. See artificial JQE.

JQE. See job queue element.

keyword. One of the predefined words of a programming language, artificial language, application, or command.

keyword parameter. A parameter that consists of a keyword followed by one or more values. See also positional parameter.

label. (1) One or more characters used to identify a statement or an item of data in a computer program. (2) An identification record for a tape or disk file.

line mode. An input-processing mode in which input is collected and processed one line at a time.

local device. A device physically attached to the local workstation, that is, the drives in the workstation and any machinery connected to its parts.

local system queue area (LSQA). In MVS, one or more segments associated with each virtual storage region that contain job-related system control blocks.

logical unit (LU). An access point through which a user or application program accesses the SNA network to communicate with another user or application program.

logoff. The process of disconnecting from a computer system or network.

logon. (1) The procedure by which a user begins a terminal session. (2) The process of connecting to a computer system or network.

loop. A situation in which an instruction or a group of instructions execute repeatedly.

LSQA. See local system queue area.

LU. See logical unit.

machine check interruption. An interruption that occurs as a result of an equipment malfunction or error.

MAS configuration. See multi-access spool configuration.

MCS. See multiple console support.

message. A communication sent from a person or program to another person or program.

MP. See multiprocessor.

multi-access spool complex. See multi-access spool configuration.

multi-access spool configuration (MAS configuration). A multiple-processor complex that consists of two or more processors at the same physical location, which share the same spool and checkpoint data sets.

multiple console support (MCS). A feature of MVS that permits selective message routing to multiple consoles.

Multiple Virtual Storage (MVS). An IBM operating system that accesses multiple address spaces in virtual storage.

multiprocessor (MP). A processor complex that has more than one central processor.

MVS. See Multiple Virtual Storage.

NAT. See nodes attached table.

network. In data communication, a configuration in which two or more locations are physically connected for the purpose of exchanging data.

network job entry (NJE). A facility for linking single-processor systems or multi-access spool complexes into a processing network

NIP. See nucleus initialization program.

NIT. See node information table.

NJE. See network job entry.

node. In communications, an end point of a communication link or a junction common to two or more links in a network. Nodes can be processors, communication controllers, cluster controllers, terminals, or workstations. Nodes can vary in routing and other functional capabilities.

node information table (NIT). An internal control block containing information about each network job entry (NJE) node.

node name. An 8-character alphanumeric name that represents a node to other parts of the network job entry (NJE) network.

nodes attached table (NAT). An internal control block containing information about each pair of nodes that is connected or recently disconnected.

nonimpact printer. A printer in which printing is not the result of mechanical impacts, for example, a thermal printer, an electrostatic printer, and a photographic printer. See also impact printer.

nonpageable dynamic area. In MVS, an area of virtual storage whose virtual addresses are identical to real addresses. It is used for programs or parts of programs that are not to be paged during execution.

nonpageable region. In MVS, a subdivision of the nonpageable dynamic area that is allocated to a job step or system task that is not to be paged during execution. In a nonpageable region, each virtual address is identical to its real address.

non-static connection. See dynamic connection.

nucleus. That portion of a control program that always remains in central storage.

nucleus initialization program (NIP). The MVS component that initializes the resident control program.

offload. To move jobs and work off work queues in order to remove them from contention for system resources or off spools to free system work space.

operand. An entity on which an operation is performed.

output group. A group of output data sets that share certain characteristics, such as class and destination.

output writer. A part of the Job Entry Subsystem (JES) that receives job output from the system spool.

overlay. A collection of predefined data, such as lines, shading, text, boxes, or logos, that can be merged with variable data on a page or form while printing.

page. A fixed-length block of instructions, data, or both instructions and data that can be transferred between active physical memory and external page storage.

pageable link pack area (PLPA). An area of virtual storage containing supervisor call (SVC) routines, access methods, and other read-only system and user programs that can be shared among users of the system. See also pageable region.

pageable region. In MVS, a subdivision of the pageable dynamic area that is allocated to a job step or a system task that can be paged during execution. See also pageable link pack area.

page data set. A data set in external page storage in which pages are stored.

page fault. In z/OS or System/390 virtual storage systems, a program interruption that occurs when a page that is marked not in central storage is referred to by an active page.

page mode. The mode of operation in which a page printer can accept an entire page of data at a time from a host processor to be printed on an all-points-addressable (APA) output medium. A page can consist of text, images, overlays, and page segments. See also compatibility mode.

page mode environment checkpointing. A process that preserves the information needed to resume page-mode printing.

page mode printer. An Advanced Function Presentation (AFP) printer that can print page-mode data.

parameter (parm). A value or reference passed to a function, command, or program that serves as input or controls actions. The value is supplied by a user or by another program or process.

parm. See parameter.

password. In computer and network security, a specific string of characters used by a program, computer operator, or user to access the system and the information stored within it.

path. In VTAM, the intervening nodes and lines connected a terminal and an application program in the host processor.

path manager. A function that controls network job entry (NJE) sign-on and sign-off, monitors all other

nodes and connections in the network, and determines the best path to reach those nodes.

PLPA. See pageable link pack area. See also pageable region.

positional parameter. A parameter that must appear in a specified location, relative to other parameters. See also keyword parameter.

primary write. A type of update that records changes to the checkpoint data set. See also intermediate write, final write.

Print Services Facility (PSF). An IBM licensed program that manages and controls the input data stream and output data stream required by supported IBM page printers.

priority aging. A scheduling function used to ensure that waiting jobs will be selected to run before those jobs that have just entered the system.

private connection. A connection known only to the two nodes making the connection.

process mode. The mode of operation in which data is processed by output devices.

processor storage. See central storage.

program temporary fix (PTF). For System i, System p, and System z products, a fix that is tested by IBM and is made available to all customers.

protocol. A set of rules controlling the communication and transfer of data between two or more devices or systems in a communication network.

PSF. See Print Services Facility.

PTF. See program temporary fix.

queue. A data structure for processing work in which the first element added to the queue is the first element processed. This order is referred to as first-in first-out (FIFO).

quick start. A type of warm start that can be performed in a multi-access spool (MAS) configuration.

quiescing. The process of bringing a device or a system to a halt by rejection of new requests for work.

remote job entry (RJE). The submission of a job through an input unit that has access to a computer by means of a data link.

remote terminal. A terminal attached to a system through a data link.

remote terminal access method (RTAM). A facility that controls operations between the job entry subsystem and remote terminals.

remote workstation. A workstation that is connected to the system by data communications.

RJE. See remote job entry.

routing. The assignment of the path by which a message is to reach its destination.

routing code. A code assigned to an operator message and used to route the message to the appropriate console.

RTAM. See remote terminal access method.

SDLC. See Synchronous Data Link Control.

SDSB. See spool data set browse.

secondary console. In a system with multiple consoles, any console other than the master console.

second-level destination. The part of a JES2 destination identifier that indicates a remote workstation, special local-route code, or user ID at the target node to which input is to be sent. See also first-level destination.

security classification. An installation-defined level of security printed on the separator pages of printed output.

segment. A collection of composed text and images, prepared before formatting and included in a document when it is printed.

session. A logical connection between two stations or SNA network addressable units (NAUs) that allows the two stations or NAUs to communicate.

setup. The preparation of a computing system to perform a job or job step. Setup is usually performed by an operator and often involves the performance of routine functions.

shared broadcasting. A condition in which the TSO data sets SYS1.UADS (TSO user definition) and SYS1.BRODCAST (TSO message transmission definition) are shared by all systems in the multi-access spool (MAS) complex.

single-member warm start. A type of warm start in which a member that ended abnormally joins an active configuration. This member can recover only work in process when it failed or stopped.

SMF. See System Management Facility.

SNA. See Systems Network Architecture.

spin data set. A data set that is available for printing, or deallocated, when it is closed.

spool data set. A data set written on an auxiliary storage device and managed by the Job Entry Subsystem (JES).

spool data set browse (SDSB). An application that allows a program to read spool data sets.

spooling. The sending of data to auxiliary storage for later processing. The most common spooling application is print spooling.

SQA. See system queue area.

SSCP. See system services control point.

static connection. A connection between two nodes created by either a JES2 initialization or an operator command.

step restart. A restart that occurs at the beginning of a job step. There are two types of step restart: automatic or deferred. See also checkpoint restart.

subnet. A network divided into smaller independent subgroups, which still are interconnected.

SVC interruption. An interruption caused by the execution of a supervisor call (SVC) instruction, causing control to be passed to the supervisor.

swap data set. A data set dedicated to the swapping operation.

swapping. A process that interchanges the contents of an area of real storage with the contents of an area in auxiliary storage.

symbol. In MVS, a group of 1 - 8 characters, including alphanumeric characters and the three characters: #, @, $. The symbol begins with either an alphabetic character or one of the three characters (#,@,$).

symbolic destination. A destination identifier specifying a symbolic name that represents a destination. See also destination identifier, explicit destination.

Synchronous Data Link Control (SDLC). A protocol for managing synchronous information transfer over a data link connection.

syntax. The rules for the construction of a command or statement.

sysplex. A set of z/OS systems that communicate with each other through certain multisystem hardware components and software services.

system affinity. See affinity.

System Management Facility (SMF). A z/OS facility that collects and records a variety of system and job-related information.

system output writer. A job scheduler function that transcribes specified output data sets onto a system output unit, independently of the program that produced the data sets.

system queue area (SQA). An area of virtual storage below the 16MB line reserved for system-related control blocks.

system services control point (SSCP). A focal point in an SNA network for managing configuration, coordinating network-operator and problem-determination requests, and providing directory support or other session services for network users.

Systems Network Architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information through and controlling the configuration and operation of networks.

table pair. A set of tables used for processing specifications: the JES2 table provides the default processing specifications and the user table provides updates or deletions to the default processing specifications. See also dynamic table, JES2 table, user table.

task control block (TCB). A z/OS control block that is used to communicate information about tasks within an address space that is connected to a subsystem.

TCAM. See Telecommunications Access Method.

TCB. See task control block.

Telecommunications Access Method (TCAM). An access method used to transfer data between main storage and remote or local storage.

teleprocessing. Processing data that is received from or transmitted to a remote location by way of communication channels.

terminal. In data communication, a device, usually equipped with a keyboard and display device, capable of sending and receiving information.

text transparency. In binary synchronous communication (BSC), a method of sending and receiving data containing any or all of the 256 character combinations in EBCDIC in specific bit patterns, including transmission control characters.

tightly coupled multiprocessing. A type of processing in which two computing systems operate simultaneously under one control program while sharing resources.

time-of-day clock (TOD clock). A timing device that counts units of time based on the starting point of 00 hours, 00 minutes, and 00 seconds on January 1, 1900. Time-of-day (TOD) information is used to monitor computer operations and events. See also time tolerance.

Time Sharing Option Extensions (TSO/E). A licensed program that is based on Time Sharing Option

(TSO). With TSO/E, MVS users can interactively share computer time and resources.

time tolerance. The difference between the TOD clocks on two adjacent nodes, beyond which the path manager will not allow a session to be established. See also time-of-day clock.

TOD clock. See time-of-day clock. See also time tolerance.

token. In checkpoint processing, an identifier that is used to determine checkpoint I/O status.

tracing routine. A routine that provides a historical record of specified events in the execution of a program.

traffic. In data communication, the quantity of data transmitted past a particular point in a path.

TSO/E. See Time Sharing Option Extensions.

type font. Type of a given size and style, for example, 10-point Latin1: Helvetica roman medium. (A)

UCS. See universal character set.

unit. A mechanical, electrical, or electronic piece of equipment for a special purpose.

unit address. The address of a particular device, specified at the time a system is installed.

universal character set (UCS). A printer feature that permits the use of a variety of character arrays.

user ID. See user identification.

user identification (user ID). The name used to associate the user profile with a user when a user signs on to a system.

user table. An installation-defined table that is used to extend, modify, or delete the default processing specifications. See also table pair.

virtual address space. In virtual storage systems, the virtual storage assigned to a job, terminal user, or system task. See also address space.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network.

VTAM. See Virtual Telecommunications Access Method.

warm start. A restart that allows reuse of previously initialized input and output work queues. See also cold start.

writer. In MVS, the part of the Job Entry Subsystem (JES) that controls the output of specified data sets.

z/OS. An operating system for the IBM eServer product line that uses 64-bit real storage.

Index

S

T

V

W

Readers' Comments — We'd Like to Hear from You

z/OS
JES2 Introduction

Publication No. SA22-7535-06

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.
- Send your comments via e-mail to: mhvrcfs@us.ibm.com

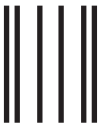
If you would like a response from IBM, please fill in the following information:

Name	Address
Company or Organization	
Phone No.	E-mail address

Readers' Comments — We'd Like to Hear from You
SA22-7535-06

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
MHVRCFS, Mail Station P181
2455 South Road
Poughkeepsie, NY
12601-5400

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Program Number: 5694-A01

Printed in USA

SA22-7535-06