ABCs of z/OS System Programming Volume 13

- JES3 internals and externals
- JES3 initialization statements, JES3 SDSF
- JES3 operator commands

Paul Rogers
Juha Vainikainen

ibm.com/redbooks

International Technical Support Organization

ABCs of z/OS System Programming Volume 13

January 2012

SG24-7717-01

Note: Before using this information and the product it supports, read the information in “Notices” on page xv.

Second Edition (January 2012)

This edition applies to version 1 release 13 modification 0 of IBM z/OS (product number 5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites 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.

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

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

Preface

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

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

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

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

This IBM® Redbooks® publication describes a JES3 environment that includes the following:

- Job entry subsystem (JES3)
- Spool data sets and checkpoint
- JES3 job flow and scheduling
- JES3 spool data management
- JES3 initialization
- JES3 input service
- Converter/interpreter processing
- Main device scheduling (MDS)
- JES3 job scheduling - GMS
- WLM batch initiator management
- JES3 output processing
- JES3 and multisystem consoles
- MVS™ System Logger/JES3 DLOG
- RJP and NJE
- JES3 dynamic support programs
- Spool partitioning and spool recovery
- JES3 Monitoring Facility (JMF)

This book will help you install, tailor and configure a JES3 system.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Paul Rogers is a Consulting IT Specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on various aspects of z/OS®, z/OS UNIX, JES3, and Infoprint Server. Before joining the ITSO 20 years ago, Paul worked in the IBM Installation Support Center (ISC) in Greenford, England for eight years providing OS/390® and JES support for IBM EMEA and also in the Washington Systems Center for three years. He has worked for IBM for 44 1/2 years.

Juha VainikainenJuha Vainkainen is a Senior I/T Specialist in IBM Finland. He has extensive knowledge of all aspects of OS/390 and z/OS and all related subsystems. He has worked with Infoprint Server extensively since OS/390 Release 5. Juha is now retired from IBM and was temporarily rehired to work on this book.

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTF Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

Job entry subsystem (JES3)

IBM's z/OS operating systems use a job entry subsystem (JES) to receive jobs into the operating system, schedule them for processing by MVS or z/OS, and to control their output processing.

There are two versions of the job entry subsystem concept, JES3 and JES2. Some principle differences between the two JES systems include:

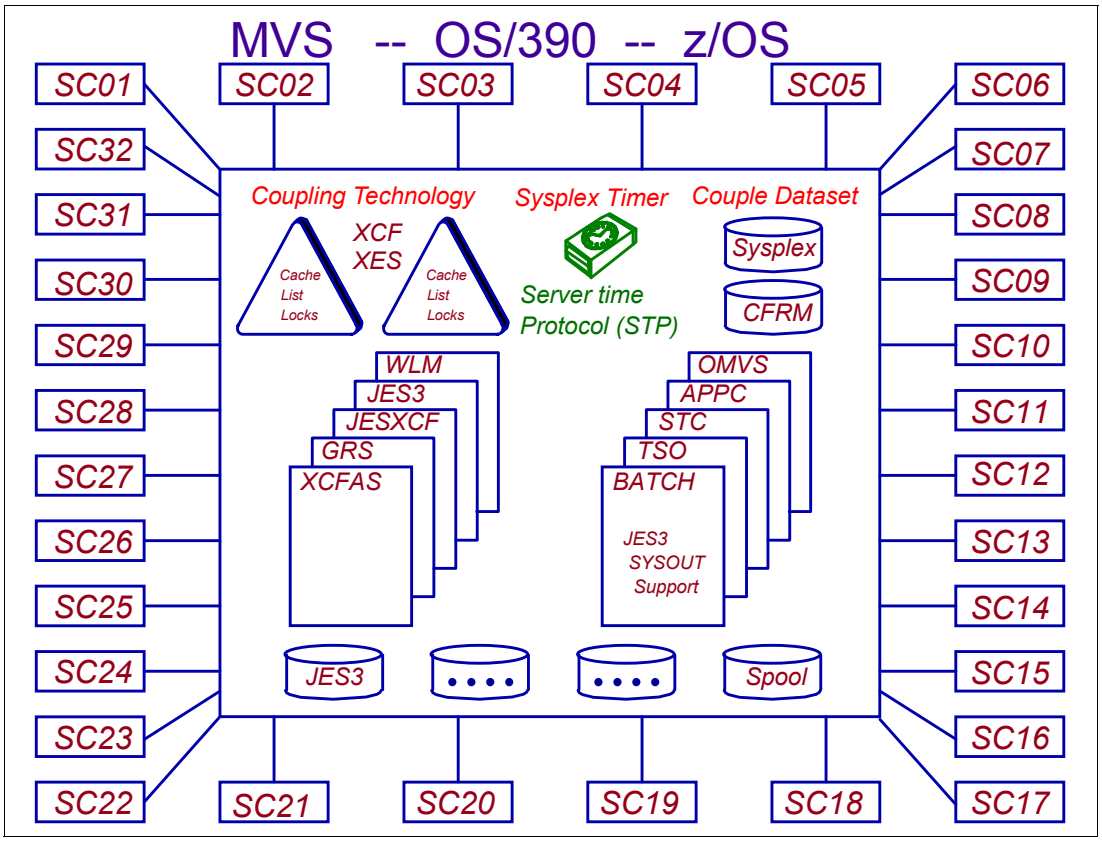
- JES3 provides resource management, dependent job control, and deadline scheduling for users of the system, while JES2 in the same system would require its users to manage these activities through other means.
- In cases where multiple z/OS systems are clustered (a sysplex), JES3 exercises centralized control over its processing functions through a single global JES3 processor. The global processor provides all job selection, scheduling, and device allocation functions for all of the other JES3 systems in the sysplex. JES2 is that component of MVS that only provides the necessary functions to get jobs into, and output out of, the MVS system. JES2 is designed to provide spooling, scheduling, and management facilities for the z/OS systems in a sysplex.

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

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

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

1.1 What is a sysplex



What is a sysplex

A sysplex is a collection of MVS systems that cooperate, using certain hardware and software products, to process work. The term MVS - Multiple Virtual Storage - refers to the IBM operating system that manages resources and work flow for multiple jobs executing concurrently. A sysplex shares the processing of work across multiple MVS systems and lays the groundwork for simplified multisystem management through the cross-system coupling facility (XCF) component. XCF services allow authorized applications on one system to communicate with applications on the same system or on other systems.

The innate robustness and reliability of the MVS operating system and z/Architecture® processors are the foundation of a sysplex. That robustness and reliability are extended to all systems in a sysplex through cross system workload balancing and data sharing using the coupling technologies.

The sysplex increases the number of processing units and MVS operating systems that can cooperate, which in turn increases the amount of work that can be processed.

A major goal of operating systems is to process user work while making the best use of system resources. Thus, one way of viewing operating systems is as resource managers. Workload Management (WLM) - component of MVS - manages resources throughout a sysplex, WLM cooperates with resource managers that span systems. WLM uses customer-defined policies for performance objectives to help balance workloads in the sysplex.

What is a job

Job control language (JCL) is a set of statements that tell the z/OS operating system about the work (jobs) that is to be performed. Although JCL set of statements is quite large, most jobs can be run using a very small subset.

JCL statements tell z/OS where to find the appropriate input, how to process that input (that is, what program or programs to run), and what to do with the resulting output. All jobs use three main types of JCL statements:

- JOB statement to identify the unit of work you want the operating system to perform
- One or more EXEC statements, depending on the number of job steps within the job
- DD statements to identify the input and output data sets

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.

Simply stated, JES 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.

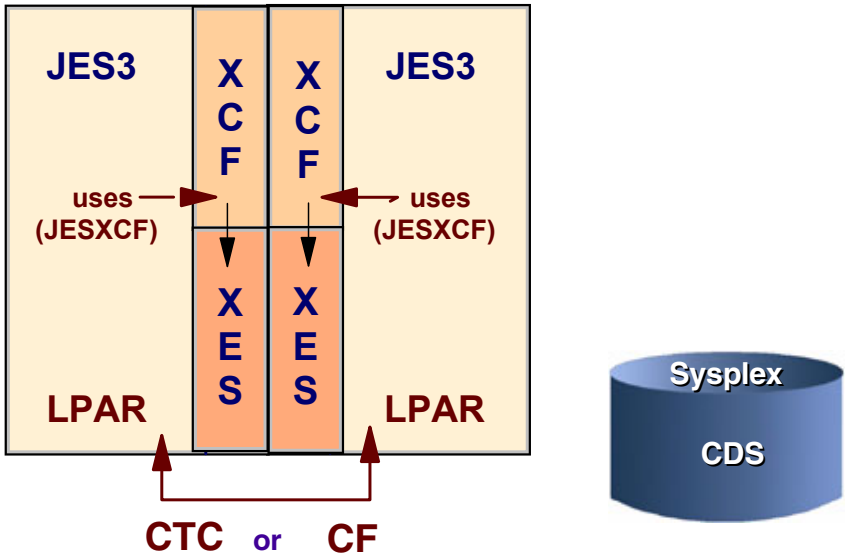
The resource management and workflow management are shared between MVS and its JES component. Generally speaking, JES does resource management and workflow management before and after job execution, while MVS does resource and workflow management during job execution.

IBM provides two JESs from which to choose: JES3 and JES2. From job flow point of view JES3 and JES2 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.

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.

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.

1.2 JES3 LPAR sysplex - MULTISYSTEM



PLEXCFG=MULTISYSTEM in IEASYSxx parmlib member

PR/SM - Provides common time reference

Multisystem sysplex

PLEXCFG=MULTISYSTEM indicates that the system is to be part of a multisystem sysplex consisting of one or more MVS systems that are running on one or more images. The same sysplex couple data sets must be shared by all systems in the sysplex

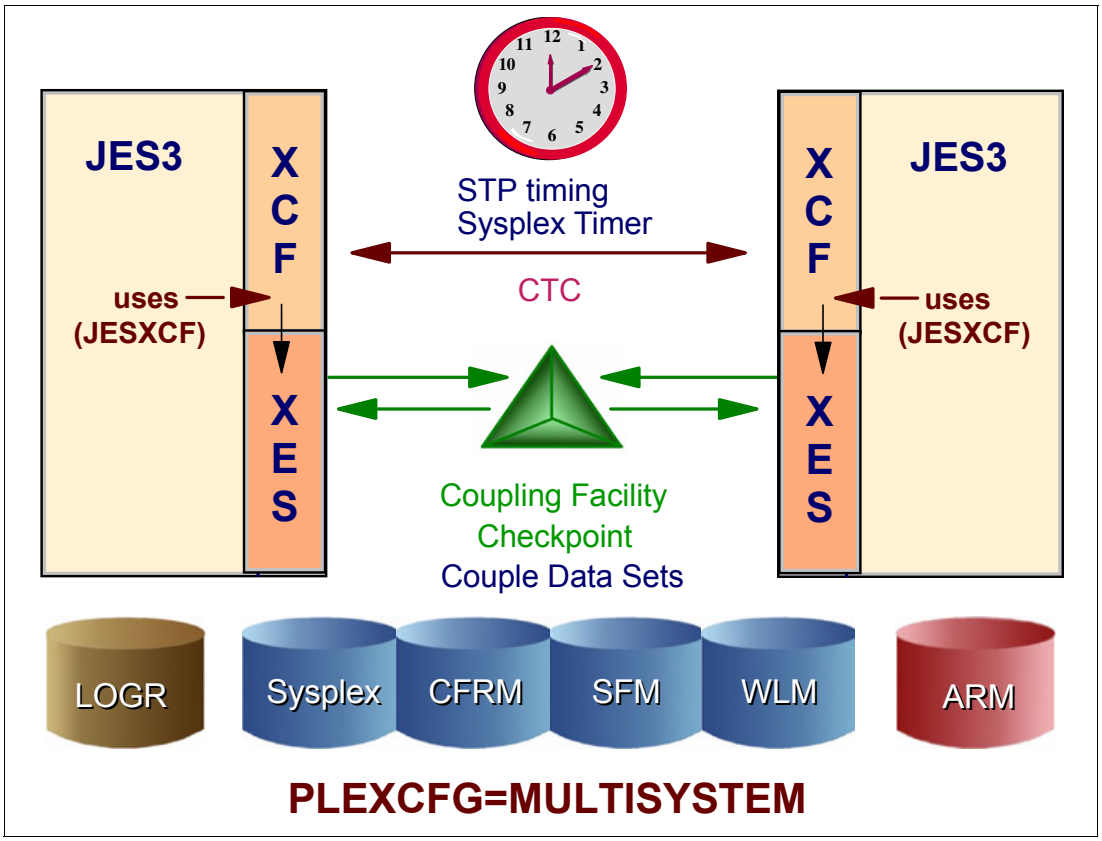
You must also specify a COUPLExx parmlib member through COUPLE=xx parameter that identifies the same sysplex couple data sets for all systems in the sysplex.

Use MULTISYSTEM when you plan to initialize one or more MVS systems into a multisystem sysplex and exploit full XCF coupling services. GRS=NONE is not valid with PLEXCFG=MULTISYSTEM.

With multiple MVS images in a single CPC environment using PR/SM™, PLEXCFG=MULTISYSTEM is the XCF mode required. A Sysplex Timer® is not required as PR/SM provides a common time reference, as shown in the figure. The JES3 systems communicate using XCF services, and the signalling services can be through either CTCs or the coupling facility.

A single system JES3 complex can be implemented in PLEXCFG=MONOPLEX sysplex. PLEXCFG=MONOPLEX indicates that the system is to be a single-system sysplex that must use a sysplex couple data set. Additional couple data sets, such as those that contain policy information, can also be used. XCF coupling services are available on the system, and multisystem applications can create groups and members. Messages can flow between members on this system (but not between this system and other MVS systems) via XCF signaling services. If signaling paths are specified, they are not used.

1.3 Multiprocessor JES3 sysplex



Multisystem sysplex

PLEXCFG=MULTISYSTEM indicates that the system is to be part of a multi-system sysplex consisting of one or more MVS systems that are running on one or more images. The same sysplex couple data sets must be shared by all systems in the sysplex

You must also specify a COUPLExx parmlib member through COUPLE=xx parameter in the IEASYSxx parmlib member that identifies the same sysplex couple data sets for all systems in the sysplex.

Use PLEXCFG=MULTISYSTEM when you plan to initialize one or more MVS systems into a multi-system sysplex and exploit full XCF coupling services. GRS=NONE is not valid with PLEXCFG=MULTISYSTEM.

When a multi-system sysplex is implemented in a single central processor complex (CPC) PLEXCFG=MULTISYSTEM is the XCF mode. A Sysplex Timer is not required. SIMETRID nn parameter on the CLOCKxx parmlib member specifies the simulated Sysplex Timer identifier. SIMETRID allows MVS images running on the same CEC, in native mode in LPARs, as VM guests, or combinations of these, to participate in a multi-system sysplex when no real sysplex timer is available. In these environments, the MVS TOD clocks are synchronized by PR/SM or the VM host. If a real sysplex timer is available, IBM recommends that you use it instead of SIMETRID.

The JES3 systems in a sysplex communicate using JESXCF services, and the XCF signalling services can be through either CTCs or the coupling facility.

With multiple MVS images in multiple CPCs, PLEXCFG=MULTISYSTEM is the XCF mode. A Sysplex Timer or IBM Server Timing Protocol (STP timing) provides the common time reference for the sysplex systems. The JES3 systems communicate using JESXCF, and the XCF signalling services can be through either CTCs or the coupling facility.

Couple data sets

A sysplex require a sysplex couple data set to store information about its systems, the XCF groups and members running in the sysplex, and general status information.

Depending on the policies you define to help manage resources and workload for the sysplex, you might need to define additional couple data sets to store policy-related information.

The coupling facility resource management (CFRM) couple data set holds the CFRM policy, which allows you to define how MVS is to manage your coupling facility resources.

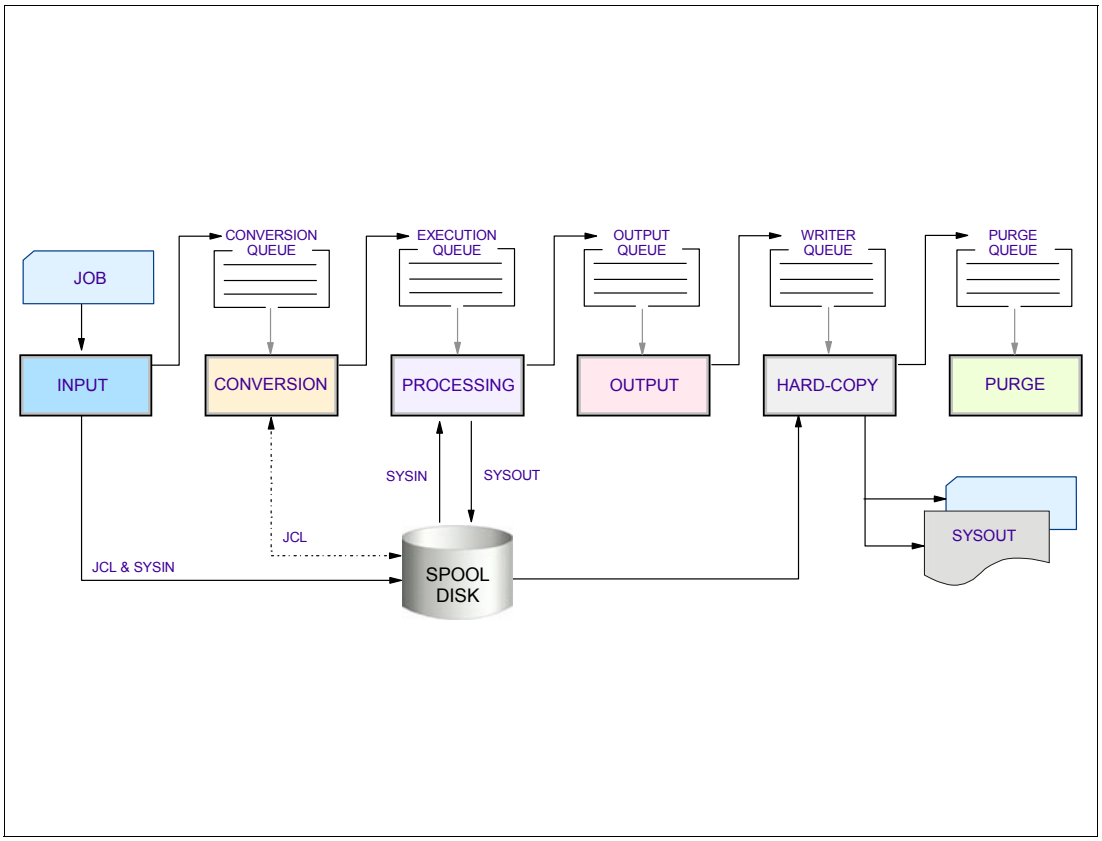
The sysplex failure management (SFM) couple data set holds the SFM policy, which allows you to define how system failures, signaling connectivity failures, and PR/SM reconfiguration actions are to be managed.

The workload management (WLM) couple data set holds the WLM policy, which allows you to define service goals for workloads.

The automatic restart management (ARM) couple data set holds the policy that defines how MVS is to manage restarts for specific batch jobs and started tasks that are registered as elements of automatic restart management.

The system logger (LOGR) couple data set holds the policy that allows you to define log stream or structure definitions.

1.4 How JES fits into the MVS system



How JES fits into the MVS System

During the life of a job, both JES and the base control program of MVS control different phases of the overall processing. Generally speaking, job goes through the following phases:

- Input
- Conversion
- Execution
- Output
- Writer Processing (hard copy)
- Purge

Except for execution, all the job phases are controlled by JES.

Input phase

JES accepts jobs (in the form of an JCL 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 JES.

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

As JES reads the input stream, it assigns a job identifier to each job and places each job's JCL, optional JES control statements, and SYSIN data onto DASD data sets called spool

data sets. JES then selects jobs from the spool for subsequent JES processing phases and MVS execution.

Conversion phase

JES uses the MVS converter/interpreter programs to analyze each job's JCL statements. The conversion takes the job's JCL, merges it with JCL from a procedure library (such as SYS1.PROCLIB), and converts the composite JCL into internal format (scheduler work area (SWA) control blocks) that both JES and the job scheduler functions of MVS can recognize. JES then stores the output of the C/I processing on the spool data set. If any JCL errors are detected during the C/I processing JES issues messages, and the job is queued directly for output processing. If there are no errors, JES queues the job for the next JES processing phase. JES supports multiple converters; therefore, jobs may not always be processed in a first-in-first-out (FIFO) order. When MVS work load management (WLM) batch management is in use, JES queues jobs according to their arrival time.

Execution phase

In the execution phase, JES responds to requests for jobs from the MVS initiators. JES selects jobs that are waiting to execute and have required resources available.

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. The initiator JES communication is implemented through subsystem interface communication (SSI). Initiators are started and controlled by JES or by MVS workload management (WLM).

When an initiator requests work, JES selects a job and passes its SWA control blocks to the initiator. The job and its resource requirements are defined to the initiator by the SWA control blocks. A SWA is the internal representation of a job's JCL.

The initiator then allocates the resources specified in the JCL for the first step of the job and starts the program requested in the JCL EXEC statement for the first step. After the step ends execution, the initiator unallocates the step's resources and allocates the resources for the next step of the job. This process repeated for each step of the job.

Output phase

JES 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, as well as data sets created by the job that must be printed or punched. After a job finishes, JES analyzes the characteristics of the job's output in terms of its output class and printer device setup requirements; then JES groups data sets with similar characteristics. JES queues the output for writer (print or punch) processing.

Writer processing phase

JES writer processing selects output for processing from the output queues by output class, route code, priority, and other criteria. An 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 NJE node). JES handles each of these situations in different ways. After processing all the output for a particular job, JES puts the job on the purge queue.

Purge phase

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

1.5 Subsystems

- Subsystems - service providers
  - Perform defined functions on SSI requests
  - Functions requested by system or user routines
- MVS subsystems
  - Master subsystem - MSTR
  - Job entry subsystem - IBM JES2 and JES3
  - Other subsystems
- JES Functional subsystems
  - Provide JES-related functions on FSI requests

What is a subsystem

A subsystem is a service provider that performs defined functions when requested through the MVS subsystem interface (SSI). The SSI allow routines to request services of, or to pass information to subsystems. The SSI acts only as a mechanism for transferring control and data between a requestor and the subsystem; it does not perform any subsystem service functions itself. If the requestor of subsystem service does not identify the subsystem from which the service is wanted, the request is routed to the subsystem that initiated the job.

Service requests from IBM subsystems are identified by function codes. For example MVS initiators request JES job selection service with function code 5 and can expect the response format to be the same from JES2 and JES3. However, the internal job selection processing is implemented differently in JES2 and JES3. Thus, the SSI service requestor does not have to make any program changes when the service provider is changed.

Subsystems are defined to MVS in one of the following ways:

- In the IEFSSNxx parmlib member - Static definition
- Issuing the IEFSSI macro - Dynamic definition
- Issuing the SETSSI operator command - Dynamic definition

Note: The master subsystem (MSTR) is a part of MVS and is defined automatically during the MVS operating system initialization.

IBM subsystems

Some examples of IBM-supplied subsystems that provide services through the SSI:

- MSTR
- JES2
- JES3
- IMS™
- NetView®
- Tivoli® Workload Scheduler

Types of subsystems

There are four types of subsystems as follows:

- Master subsystem - One of the functions provided by the MSTR is the subsystem initiation service for the MVS initiator's job select request during a subsystem start processing.
- Job entry subsystem - The job entry subsystem that MVS uses as default to select work from is defined also as a primary subsystem. The primary subsystem can be either JES2 or JES3.
- Secondary job entry subsystems - MVS allows more than one JES2 subsystems to operate at a time, as long as one subsystem is designated as the primary. The other JES2 subsystems are secondaries.

  In JES3 complex secondary job entry subsystems are not supported.
- Other subsystems - Provide functions as needed by IBM products, vendor products, or the installation.

What is a functional subsystem?

A functional subsystem (FSS) is a collection of programs residing in an address space separate from JES that communicates with JES to provide a JES-related function, such as print processing and converter/interpreter (C/I) processing for JES3. An FSS extends the scope of JES processing. Because an FSS operates in its own address space, it functions independently of JES in several areas.

An FSS is responsible for:

- The management of storage resources that it needs during data set processing including print buffers.
- Its own recovery and serviceability.
- Its performance and accounting measurements.
- The security of its own resources.

JES and the FSS communicate through the functional subsystem interface (FSI) and SSI. The FSI is a one-level interface which provides two way communication. The FSI consists of a set of macro-invoked service routines provided by both JES and the FSS/FSA.

1.6 JES3 as the primary subsystem

- Primary subsystem - JES3
  - System input and output - (JES3 Global)
  - Manage jobs and spool - (JES3 Global)
- Functional subsystems - FSS
  - Offload functions to locals and global address spaces
    - Converter/Interpreter and printers
  - JES3 global controls
    - FSS start and stop
    - Scheduling and selection of work for MVS
    - Operator communication

JES3 as the primary subsystem

As the primary subsystem, the global JES3 plays an important role and the following functions provided by JES3 indicates why communication is needed between MVS and JES3. The global JES3:

- Introduces all jobs into the system, no matter what the source.
- Handles scheduling of conversion and interpretation of JCL.
- Performs pre-execution setup of devices.
- Schedules MVS jobs to all main processors.
- Maintains awareness of all jobs in execution.
- Handles the scheduling of all SYSOUT data sets for writer processing.
- Manages the allocation and deallocation of space on the shared-spool devices.

When carrying out some of these responsibilities, global JES3 needs the assistance of local JES3.

Functional subsystems

JES3 allows certain functions to operate outside the JES3 address space. JES3 does this using:

- The functional subsystem address space (FSS)
- The functional subsystem interface (FSI)

- The functional subsystem application (FSA)

WTR FSS

The JES3 FSS that deals with output services is one type of FSS (TYPE=WTR). This particular FSS address space may be created automatically or established by a CALL command for a printer device which is capable of running under the control of an FSS address space. The operator CALL command (*X) designates a printer as a "hot writer" while a writer invoked automatically when output is queued is called a "dynamic writer". A “hot writer” stays active until the operator stops it. A “dynamic writer” stops automatically after the specified time-out elapses if there is no more output to process.

CI FSS

Another FSS (TYPE=CI) deals with converter/interpreter services similar to those that occur in the JES3 global

The functional subsystem address space can be on any processor in the JES3 complex and is started by JES3 issuing the MVS START command with a token. When the first writer defined to a FSS is started, a functional subsystem address space is created. The operator can use the **MODIFY** command (***F**) to change the processor where the FSS runs. When the operator stops a writer, JES3 stops the writer and when the last writer in the FSS address space is stopped the address space is terminated.

1.7 Defining subsystems

- IEFSSNxx SYS1.PARMLIB definition
  - SETSSI {ADD,{SUBNAME|SUB|S}=subname [,{CONSNAME|C}=consname][INITRTN(initrtn) [INITPARM(initparm)]] [PRIMARY({NO|YES})[START({YES|NO})]]
  - ssname,[initrtn[,initparm]][,PRIMARY[,NOSTART]] comments (old form)
- SETSSI operator command
  - SETSSI {ADD,{SUBNAME|SUB|S}=subname [,{CONSNAME|C}=consname]
    ,{INITRTN|I}=initrtn[,{INITPARM|P}=initparm]] }
- IEFSSI authorized macro service
  - IEFSSI REQUEST=ADD,SUBNAME=subname,INITRTN=initrtn ,INITPARM=initparm,INITPLEN=initplen,CONSNAME=consname ,RETCODE=retcode,RSNCODE=rsncode

IEFSSNxx definition

The IEFSSNxx parmlib member contains parameters that define the primary and the various other subsystems that are to be initialized during the MVS system initialization.

IEFSSNxx allows:

- Name the subsystem initialization routine to be given control during MVS initialization.
- Specify the input parameter string to be passed to the subsystem initialization routine.
- Specify the primary subsystem name and whether it is started automatically.

During system initialization, the control structures that define subsystems to MVS are initialized. The recommendation is that you place the Data Facility Storage Management Subsystem (SMS) record before the primary subsystem's (JES) record in the IEFSSNxx to start SMS before starting the primary subsystem. After the primary subsystem definition the other subsystems should be defined, because the other subsystems such as DB2®, require the services of the primary subsystem in their initialization routines. Problems can occur if subsystems that use the subsystem affinity service in their initialization routines are initialized before the primary subsystem. After the primary JES is initialized, then the other subsystems are initialized. in the order in which the IEFSSNxx parmlib members are specified by the SSN parameter in the IEASYSxx parmlib member.

SSN parameter

The SSN parameter in the IEASYSxx parmlib member identifies the IEFSSNxx parmlib members that the system is to use to initialize subsystems, as follows:

```
SSN=aa|(aa,bb,.......)
```

For example, for SSN=(aa,bb) parmlib member IEFSSNaa would be processed before IEFSSNbb.

IBM recommends that you use the keyword parameter form of the IEFSSNxx parmlib member. However, also the old positional parameter form of the IEFSSNxx parmlib member is supported.

SETSSI operator command

The **SETSSI** operator command can be used to add, activate, or deactivate subsystems dynamically after MVS system initialization.

IEFSSI macro service

An authorized program can dynamically control a subsystem with the IEFSSI macro service in any of the following ways:

- Add and define a subsystem to the system
- Activate a subsystem so that its function routines can process function requests
- Define a set of optional subsystem characteristics
- Deactivate a subsystem
- Swap the current SSVT with a new SSVT
- Store subsystem-defined data for a subsystem
- Retrieve subsystem-defined data for a subsystem that was previously stored with the put request
- Query information for all subsystems defined to the SSI

Subsystems have the choice of being dynamic. Subsystems that are not dynamic can be defined only at IPL using the *positional* form of the IEFSSNxx parmlib member and they cannot use dynamic SSI services.

Starting subsystems

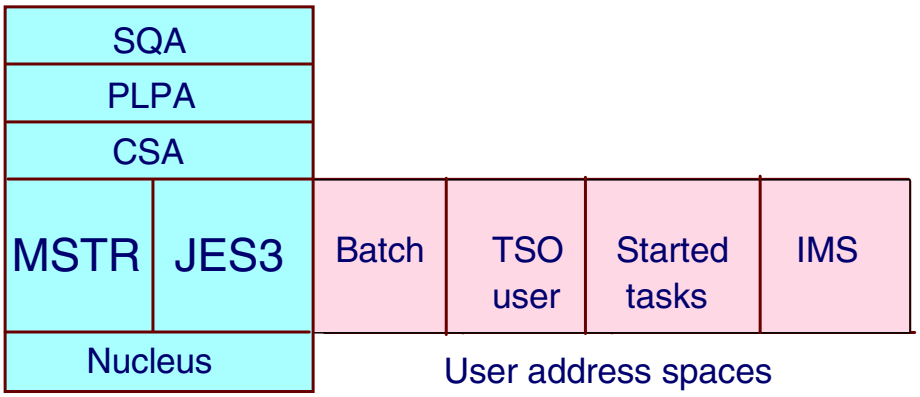
Once a subsystem name is defined to the system, any attempt to start that subsystem (or any started task with the same name as that subsystem) with a **START** command which does not explicitly specify SUB=JESn will result in that subsystem or started task being started under the Master subsystem rather than under the job entry subsystem. Because the only procedure libraries available to the Master subsystem are those specified in the MSTJCLxx IEFPDSI data set, any procedures being started that are defined in the job entry subsystem's procedure library concatenation, but not in the MSTJCLxx IEFPDSI data set, will be unavailable. Therefore they will not be found and the **START** command fails.

Before a subsystem can provide services to the SSI calls it has to create its own subsystem vector table (SSVT) using the IEFJSVEC or IEFSSVT macro services. The SSVT defines the SSI functions supported by the subsystem.

The **DISPLAY SSI** operator command displays information about subsystem, dynamic or not, or about all subsystems.

1.8 Subsystem interface - SSI

- Means of communication between:
  - Address spaces routines and subsystem
    - Request services of or pass information from
- Consists of:
  - Control blocks - macros - routines
    - SSCVT - SSVT - SSOB - SSIB - (IEFSSREQ)



Subsystem interface

The subsystem interface (SSI) is an interface that provides intraprocessor communication between MVS routines and JES through the IEFSSREQ macro. The SSI is the interface used by routines (IBM, vendor, or installation-written) to request services of, or to pass information to, subsystems. An installation can design its own subsystem and use the SSI to request subsystem services.

Note: The SSI acts only as a mechanism for transferring control and data between a requestor and the subsystem; it does not perform any subsystem functions itself.

IEFSSREQ macro

MVS functions running in address spaces issue the IEFSSREQ macro to invoke subsystem functions. The calling routine uses the subsystem option block (SSOB), the SSOB function dependent area, and an optional subsystem identification block (SSIB) to identify the subsystem and requested processing. If the SSIB is not is not part of the IEFSSREQ parameters, the request is routed using the life-of-job SSIB. The life-of-job SSIB is created during the job initialization for the address space and contains a pointer to the JES control blocks for the job.

SSI control blocks

The major SSI control blocks are:

SSCVT Subsystem Communications Vector Table (SSCVT) maps information for defined subsystems

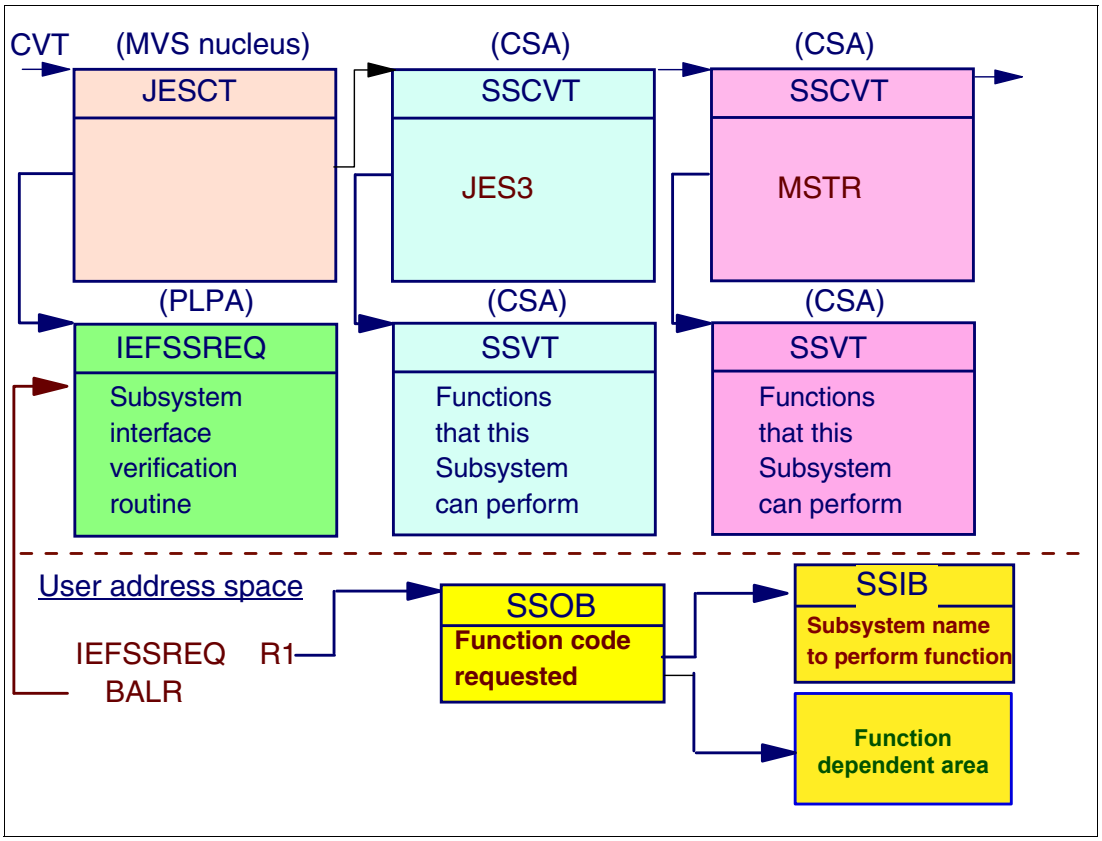
SSVT Subsystem Vector Table (SSVT) indicates the SSI functions supported by the associated subsystem

SSOB Subsystem Options Block (SSOB) identifies the requested SSI function. Passed as IEFSSREQ parameter to SSI

SSOB Function Dependent Area contains additional information, which can be passed to the subsystem along with the SSOB.

SSIB Identification Block (SSIB) identifies the particular subsystem to which a request is being directed. If a program does not provide an SSIB, the system uses the life-of-job SSIB.

1.9 SSI control blocks and routines



SSI control blocks and routines

The MVS nucleus contains control block named the JES control table (JESCT). This block contains pointer to the primary subsystem (JES) communication vector table (SSCVT), which is located in CSA.

IEFSSREQ macro

The IEFSSREQ macro provides communication between MVS and the target subsystem.

Before issuing the IEFSSREQ macro, the caller must ensure that register 1 points to the SSOB.

The calling routine uses the subsystem option block (SSOB), the SSOB function dependent area, and optionally the subsystem identification block (SSIB) to identify the required processing. The calling routine uses the IEFSSREQ macro to pass the SSIB and SSOB to the MVS SSI directed request router. The router locates the target subsystem's SSCVT/SSVT pair and passes control to the requested subsystem function routine pointed from the SSVT.

JESCT JES control table (JESCT). A control block in the MVS nucleus that contains information used by subsystem interface routines.

SSCVT Subsystem communication vector table (SSCVT). These control blocks is the common storage area (CSA) and contain the information from subsystem definitions.

SSVT The subsystem vector table (SSVT). The SSVT resides in the common service area (CSA), is created by the subsystem, and contains the function matrix and function routine pointers.

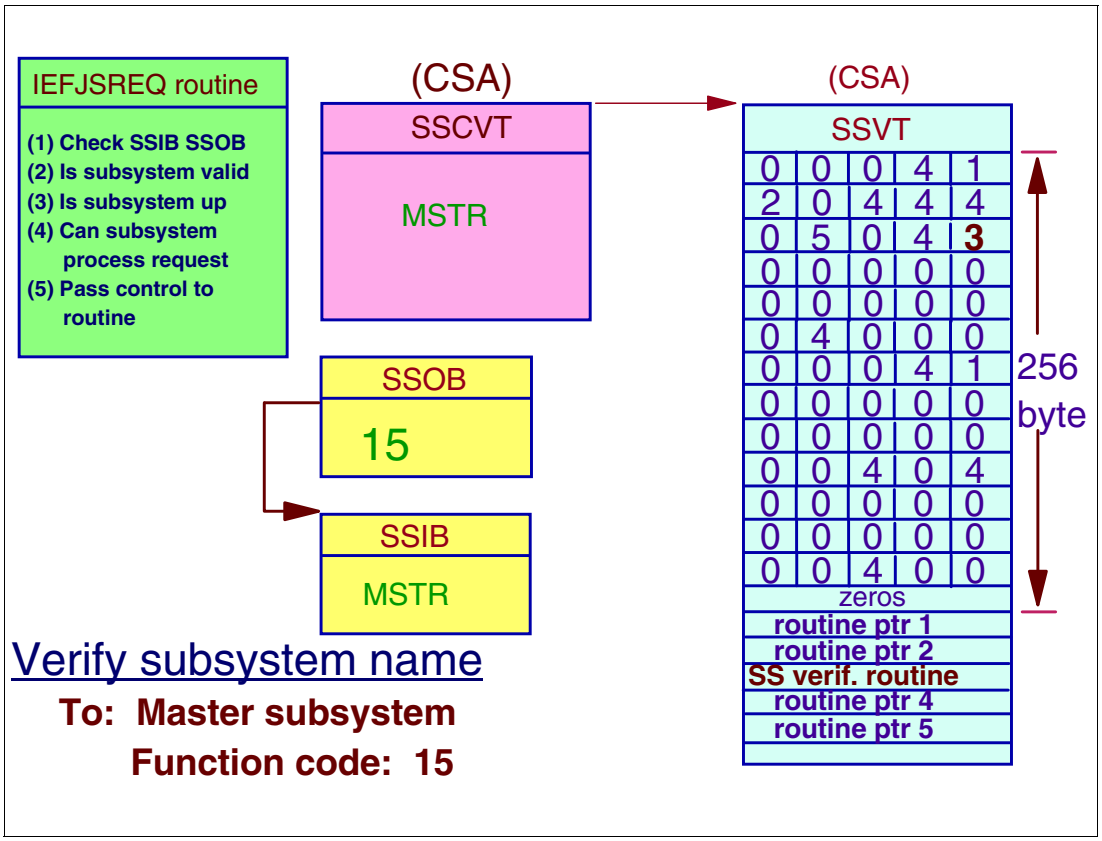
SSOB Subsystem options block (SSOB). The control block which contains the SSI function code. The function code is used by the MVS SSI router to index into the SSVT matrix, The matrix entry contains an index to the subsystem function routine which is to get control as the result of the execution of the IEFSSREQ macro.

SSIB Subsystem identification block (SSIB). The control block into which the requestor of a subsystem function places the name of the subsystem from which service is requested.

Subsystem verification routine

This routine checks the validity of the SSOB and SSIB; the request routine determines that the target subsystem exists and is started. It next uses the function code to determine if the subsystem performs the requested function and to derive the address of a routine to which the request is to be passed.

1.10 Example: SSI request to master subsystem



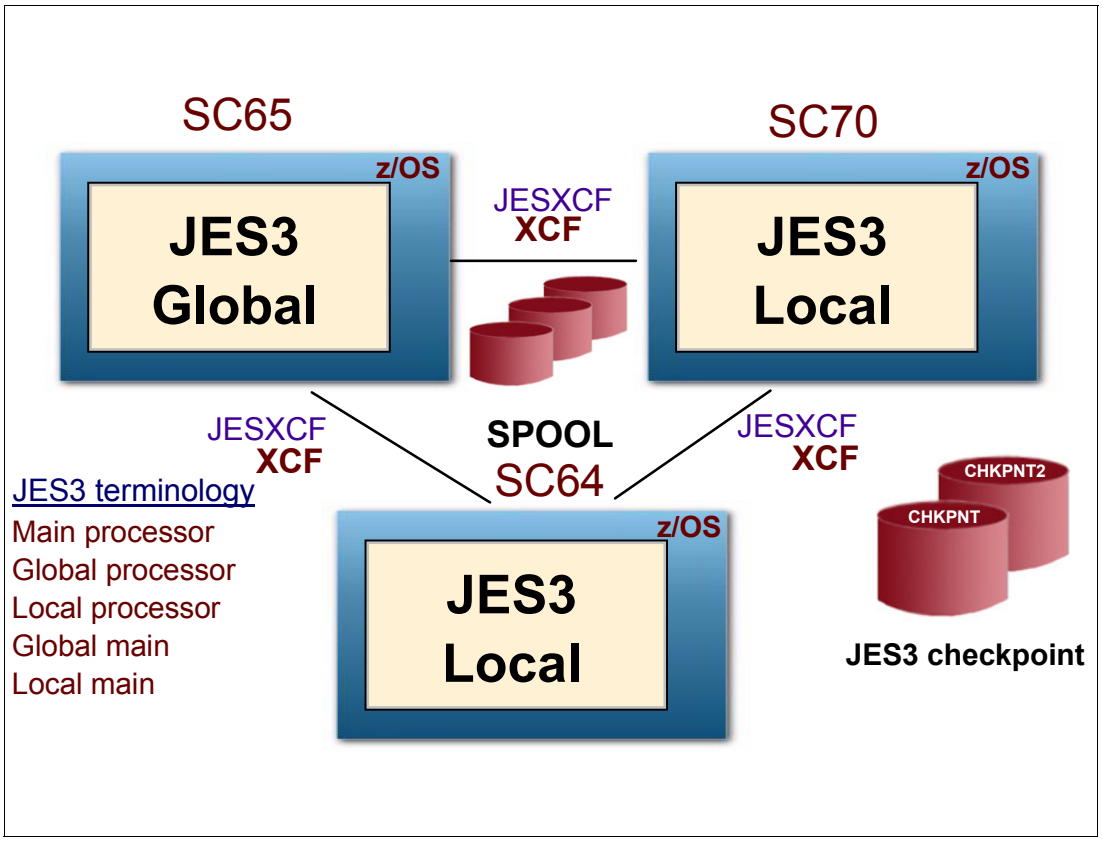
SSI request to master subsystem

The verify subsystem function call (SSI function code 15) allows a program to:

- Verify the existence of a specific subsystem
- Obtain the address of the SSCVT that corresponds to a specific subsystem
- Obtain the subsystem affinity index value used when making subsystem affinity requests.

The MVS SSI directed request router (IEFJSREQ) routine checks the validity of the SSOB and SSIB; the request routine determines that the target subsystem exists and is started. It next uses the function code to determine if the subsystem performs the requested function and to derive the address of a routine to which the request is passed.

1.11 JES3 complex - sysplex



JES3 complex in a sysplex

A sysplex is a set of MVS systems communicating and cooperating with each other through certain multisystem hardware components and software services to process customer workloads. A sysplex can have up to 32 MVS systems.

The cross-system coupling facility (XCF) provides the MVS coupling services that allow authorized programs on MVS systems in a multisystem environment to communicate (send and receive data) with programs on the same MVS system or other MVS systems. The cross-system extended services (XES) allow authorized programs on MVS to use a coupling facility to share data with other programs on the same MVS system or other MVS systems. The eventual goal for the coupling services provided by XCF and XES is to allow an installation to view multiple MVS systems as one MVS system.

Each sysplex is given an XCF sysplex name (in the visual **SANDBOX**) and each system in the sysplex has a unique name as well. The COUPLExx parmlib member for each system in the sysplex is used to specify the installation values for sysplex systems.

A JES3 complex must be fully contained within a sysplex and may or may not be the primary subsystem for all the MVS systems in the sysplex.

Note: Each JES3 complex in a sysplex is required to use a unique JES XCF groupname (in the visual **WTSCPLX4)**. The NAME parameter on the NJERMT statement that defines HOME=YES is the default for the XCFGRPNM parameter on the OPTIONS statement.

XCF and JESXCF

In a JES3 complex one of the JES3 processors is designated as the focal point for the entry and distribution of jobs and for the control of resources needed by the jobs. That processor, called the *global* processor, distributes work also to all other processors, called *local* processors. The global processor and the local processors communicate using the MVS cross-system coupling facility (XCF) through the JES common coupling services (JESXCF) component. Among other things, the JES3 inter-processor communication is used to extend the intra-processor SSI communication from local processors to the global which provides the services for most of the SSI calls.

The JES3 global processor manages jobs and resources for the entire complex and selects jobs with readily available resources for MVS execution. The resources that JES3 manages include processors, I/O devices, spool volumes, the job queue, the checkpoint data set, and sysin/sysout data. To avoid job execution delays that result when resources are not available, JES3 ensures that they are available before selecting the job for processing.

JES3 terminology

In the JES3 terminology a processor is defined as a hardware unit that contains software to interpret and process instructions.

Global processor The processor that controls job scheduling and device allocation for a complex of processors. See also local processor.

Local processor In a complex of processors under control of JES3, a processor connected to the global main by a JESXCF services, for which JES3 performs centralized job input, job scheduling and job output services by the global main.

Main A processor named by a JES3 MAINPROC initialization statement, on which jobs can execute; represents a single instance of MVS. The two types of mains are global main, and local main.

Global main The global main controls job scheduling and device allocation for a complex of JES3 processors. Each local main in the complex exists under control of the JES3 global main and is connected to the global main by JESXCF services. The JES3 on the global main can perform centralized job input, job scheduling, and job output services. Only the global main performs scheduling functions, although scheduled work executes on the local mains.

Local main In a complex of processors under control of JES3, a processor connected to the global main by a JESXCF services, for which JES3 performs centralized job input, job scheduling and job output services by the global main.

1.12 JES3 global benefits

- Global processor manages jobs and resources
  - For the entire complex
  - Matches jobs with available resources
- JES3 global manages
  - Processors - I/O devices - Volumes - Data sets
- When resources are not available
  - JES3 ensures that they are before selecting the job for processing to an initiator
- Operations aid
  - Operator commands
  - Diagnostic tools

JES3 global processor

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

JES3 provides installation benefits from the distribution of work among processors as a workflow manager. The entry of all jobs through a central point means that control of the actions needed to prepare jobs for execution can be centralized. The distribution and duplication of job management function becomes unnecessary, and an awareness of the status of all jobs entering or leaving the system can be easily maintained. This awareness is particularly useful in recovery situations, where the scope of recovery is largely a function of the quality of the tracking performed prior to the failure.

Resource manager

Another benefit is resource management. All jobs, all input required for the jobs, and all output produced by the jobs enters or leaves the system at a single point. This single point, JES3, can coordinate the subsystem, the allocation of devices, volumes, and data sets. Centralized resource control expands the opportunity for full resource utilization. If you are using the storage management subsystem (SMS), you can allow SMS to coordinate the allocation of permanently resident catalog volumes and cataloged data sets. When SMS is activated, JES3 will not manage devices and volumes for SMS-managed data sets.

JES3 keeps track of I/O resources, and manages workflow in conjunction with the workload management component of MVS by scheduling jobs for processing on the processors where

the jobs can run most efficiently. At the same time, JES3 maintains data integrity. JES3 will not schedule two jobs to run simultaneously anywhere in the complex if they are going to use the same serially reusable resources. If you want to share input/output (I/O) devices among processors, JES3 manages the sharing. Operators do not have to manually control the online/offline status of devices to keep up with changing processor needs for the devices.

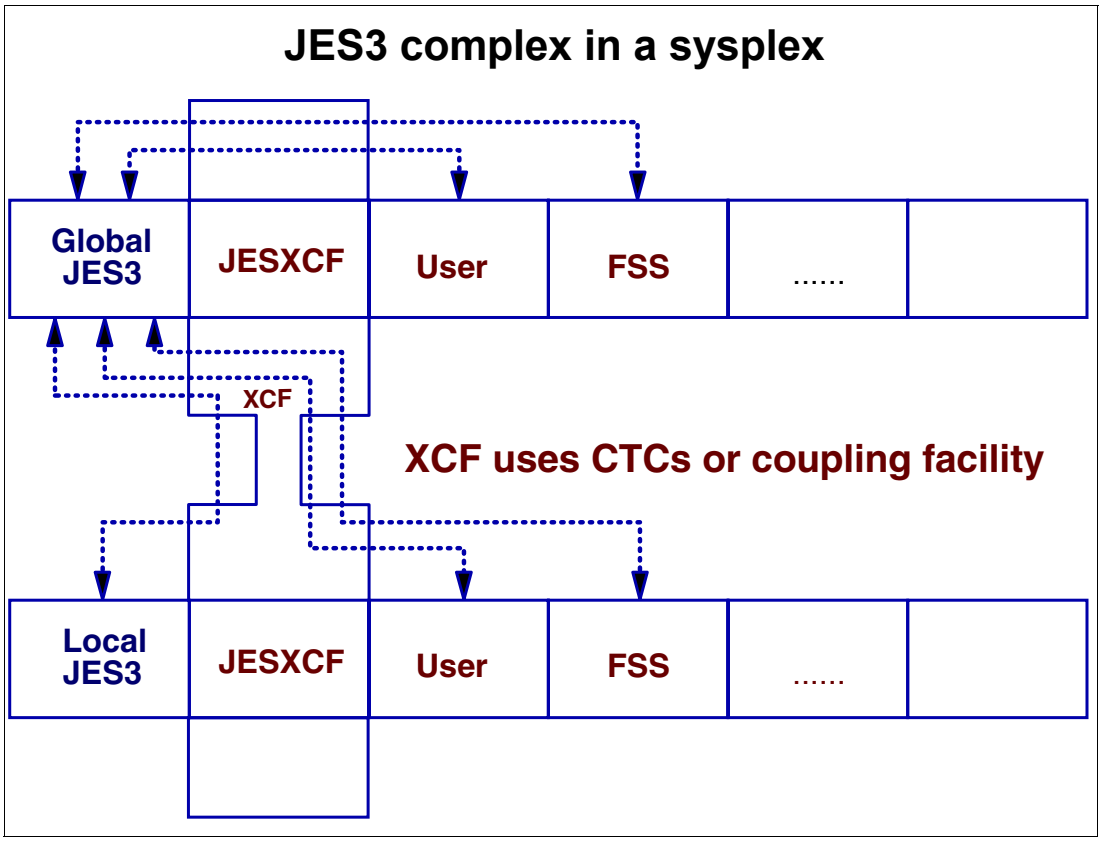
Operations aid

Operator control also benefits from improved resource utilization and centralized job management. With all system resources known to JES3 and with one job management mechanism, it is relatively simple to provide control over the entire system. And yet, the need for operator control can be minimized because JES3 is aware of job mix and resource availability and can coordinate them with less need for operator intervention and decision-making.

Operators have special JES3 commands. Some commands activate programs to handle I/O devices, while others obtain or change the status of jobs being processed. With multiple processing-unit systems, JES3 operators have less to do than for an equal number of individual systems, because they can control the entire complex from a central point, and because JES3 decides where and when jobs will be processed.

For diagnostic purposes, JES3 provides traces and dumps. Traces are copies of data made during normal processing to help monitor activity in the system, and dumps are copies of data made at times of failure.

1.13 JES and JESXCF communication



JES and JES XCF processing overview

JESXCF is an XCF multisystem application and provides, based on XCF coupling services, common inter-processor and intra-processor communication services for both JES3 and JES2 subsystems. The JESXCF services are tailored to meet JES specific requirements and are provided through macros that enable communication among JES members in a sysplex. The macros are available to either JES3 or JES2 and can be used *only in JES environments*, except for the send message service, which can be used in all environments.

JESXCF address space

The JESXCF address space is created during the first JES initialization on that system. As each JES is initialized, it joins the JES XCF group for the JES complex in which the newly initialized JES will be a member. After all JES members have successfully joined an XCF group, JES routines running on that JES member can use the JES common coupling services to communicate with the other JES members in the same JES XCF group. There are a number of restrictions that apply to JES XCF groups and their use. These include:

- The JES common coupling services component and its exits and macros are not available outside either a JES2 or JES3 environment.
- The JES systems in the JES XCF group must be either all JES2 or all JES3
- All JES2 multi-access spool (MAS) members or JES3 complex mains must join JES XCF if they are to communicate.
- All JES2 MAS members or JES3 mains must be members of the same sysplex.

There is one JESXCF address space in each MVS image and it acts as a server address for all JESes in a given MVS system.

JES3 initialization processing

During JES3 initialization, the JES XCF attach/detach exit receives control under the JES main task. At this time, the JES member is attached automatically to the JES XCF group. Each member has a different name, but each is attached to the same group to form the JES XCF group, thereby allowing each to communicate (send messages, obtain member and JES XCF group status, and so on).

JES3 message processing

Each JES3 member is in constant communication with every other JES3 member of its JES3 complex. In JES3, there is a continuous flow of SSI requests -- allocation requests, system status, job status, work requests, and so forth -- that is needed to properly coordinate the work JES3 is putting into and taking out of the MVS system. These pieces of communicated data are considered messages, and each causes the invocation of two JES XCF exits: the transport exit and the receive exit.

Exit IXZXIT01, called the transport exit, gets control whenever JES or installation code sends a message. JES XCF packages the message data in an envelope that contains the address information and other related data, and sends it to a specified mailbox.

Once a message is sent to a mailbox (on the same member or another), the receiving member is informed that mail is waiting. After the receiving member issues the receive macro, IXZXIXRM, the receive message exit IXZXIT02 gets control. This exit allows the receiving member to view or modify the message and to receive any extents that have been added to it by the transport exit. Once the message is received, the receiving member acknowledges the mail, informs JES XCF of its receipt and, if requested by the sending member, also returns an acknowledgement to the sender.

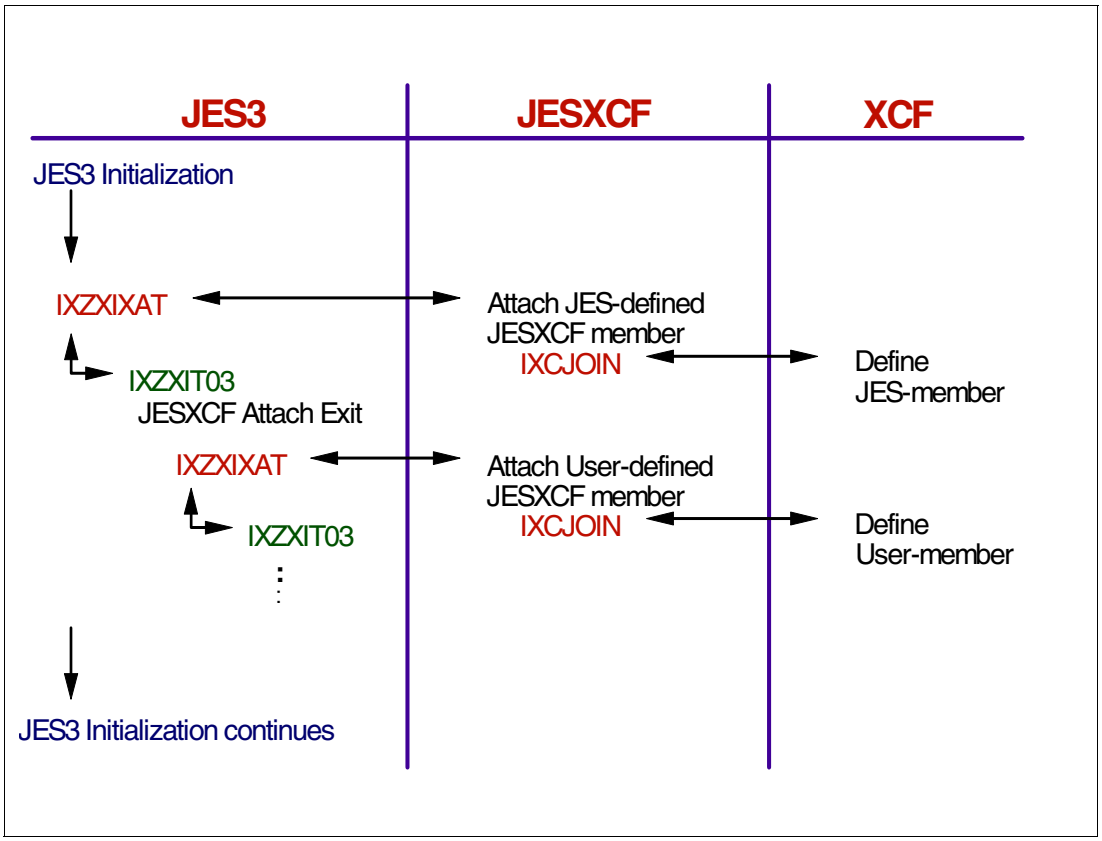
JES3 termination processing

During JES3 termination processing only JES3 FSS provide clean up when terminating normally. The JES3 termination (*RETURN) does not invokes the detach macro.

Therefore, JES3 FSS normal termination only issues the detach macro, IXZXIXDT. IXZXIT03 receives control as a result of the IXZXIXDT macro call and does clean up.

Note: In a JES3 environment to restart the JESXCF address space a MVS IPL is required to recover critical resources.

1.14 JES3 and JESXCF initialization



JES3 and JESXCF initialization

During the JES3 initialization or the JES3 FSS connect processing, the JESXCF attach service is invoked through the IXZXIXAT macro. The JESXCF attach service creates an address space for JESXCF if it does not already exist and attaches the JES3 member to a JESXCF group. During the JESXCF attach processing, the MVS XCF group services are invoked (using IXCJOIN macro) to create an XCF member for the JES3 member. JES3 uses as the XCF group name the XCFGRPNM= specification from the OPTIONS initialization statement when present. Otherwise the NJE home node name is used. The default name for the home node is N1.

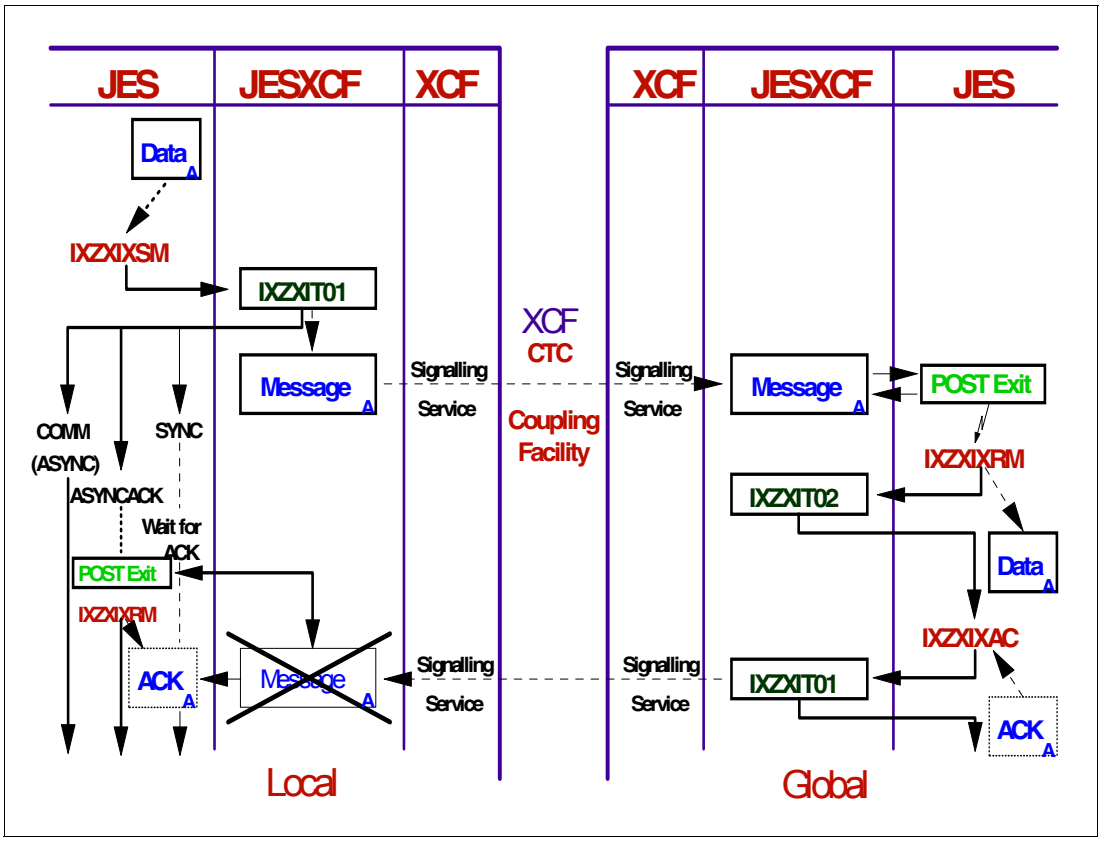
JES3 uses as the XCF member name the name defined on the NAME= parameter on the MAINPROC initialization statement. The JES3 FSSs use as the XCF member name the FSSNAME= specification on the FSSDEF statement.

Note: Ensure that the name you specify on the MAINPROC initialization statement matches the name specified on the SYSNAME parameter in the IEASYSxx parmlib member used for MVS IPL. Otherwise JES3 initialization will fail.

Installations may create their own independent attachments to JESXCF. If an installation-defined JESXCF group is required, it should be created in the JESXCF attach/detach exit IXZXIT03 during the JESXCF attach/detach processing.

A successful attach JESXCF group request returns a JESXCF group token. All subsequent JESXCF service requests require this group token as input.

1.15 JESXCF communication flow



JESXCF communication flow

The JESXCF communication is in the form of messages. In the JESXCF context, a message is any data, including XCF events, that comprises a data packet. A data packet is the data that one JES3 member sends to itself or another member within the JESXCF group. The maximum size of the data is 60KB. If the quantity of data being sent is larger than 60KB, it can be broken into parts and transported as a multi-segment message.

JESXCF messages are received through mailboxes. A mailbox is a logical queue of ordered messages and is maintained by JESXCF in a location associated with a JES3 member. Messages are held in mailboxes until the receiving JES3 receives them or clears the mailbox. When a message is received, it is preceded by a message envelope. The message envelope is the header for the message and contains information that includes:

- The addresses of the sender and receiver of the message
- An identifier of the message type
- An offset to the actual message

Envelopes, mapped by IXZYIXEN macro, are for the following types of messages in a mailbox:

- System events (SYSEVENT)
- Acknowledgements (ACKS)
- Messages (MESSAGES)

The message envelope includes information such as:

- Status of the message in the mailbox:

  - Message has been resent to the receiving system because the system was re-IPLed.
  - Message has been rerouted.
  - Message was present in the mailbox when the attacher disconnected.
  - Message has been received.
  - Message has been checkpointed.
- Maintenance level of the JESXCF component
- Message sequence number
- Address information of the receiver of the message (group name, member name, and mailbox name)
- Address information of the sender of the message (group name, member name, and mailbox name)
- Type of the send message request:
  - Synchronous message
  - Asynchronous message that does not return an acknowledgement to the sender
  - Asynchronous message that returns an acknowledgement message to the sender
  - Asynchronous message that will not be resent to the receiver if the receiving system re-IPLs. No acknowledgement will be sent to the sender of the message
  - Acknowledgement message
  - Acknowledgement message sent because time specified on the on the IXZXIXSM macro has expired
- Single segment or multi-segment message indicators.
- Content of the message:
  - A system event
  - An acknowledgement message
  - Application message
- Length of the message not including the envelope

Acknowledgement (ACK)

An acknowledgement provides notification information on delivery of messages issued through the IXZXIXSM macro service. The acknowledgement data includes:

- Request token for the message that this acknowledgement is for
- Return code information returned by the receiving routine
- Length of the data returned to the sender through the IXZXIXAC macro service

User data (MESSAGE)

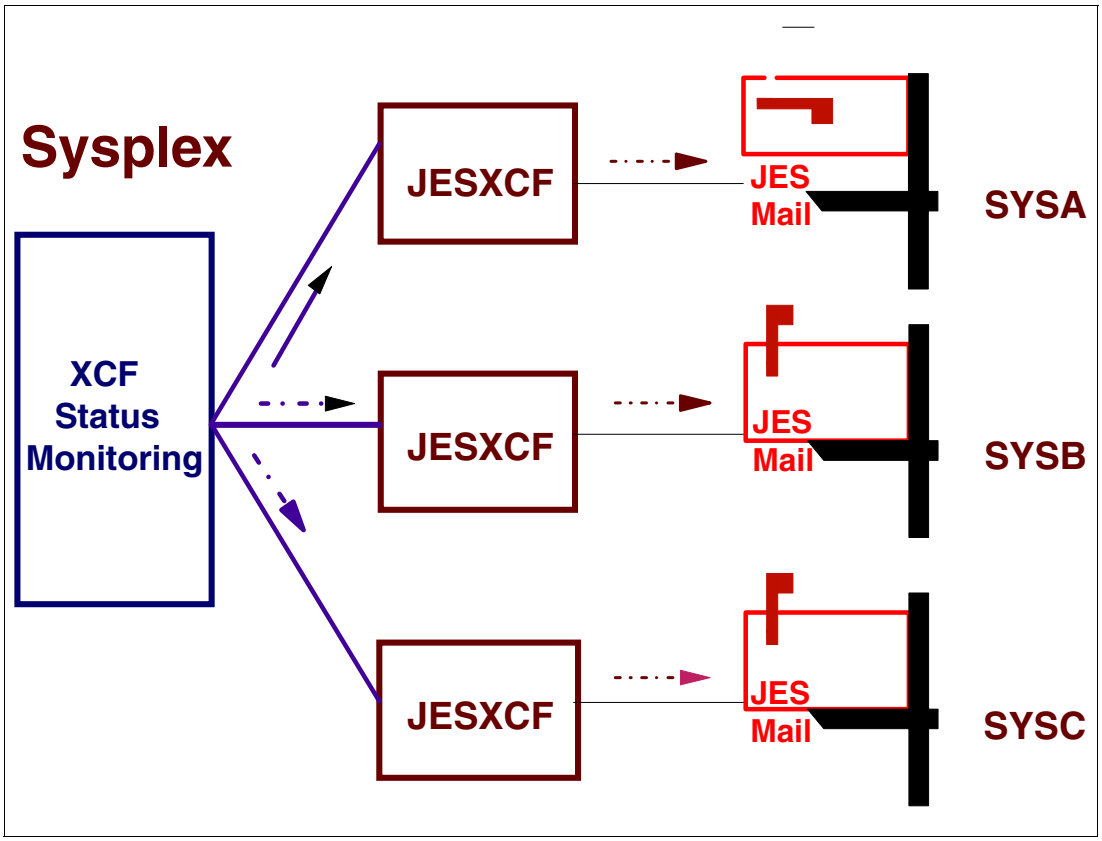
JESXCF does not impose any format requirements on user data.

- System Event (SYSEVENT)

The system event data includes three types of system event data:

- **JESXCF Data (IXZYIXIF)** - JESXCF data provides information about the JES and XCF connections, such as notification of an event detected by the JESXCF address space or such as termination of the connection between JESXCF and the JES address space.
- **JES Data (IXZYIXJE) -** JES data provides a notification of events that the JESXCF address space has detected, such as termination of the connection between JESXCF and the JES address space. The JESXCF event data:
  - Connection between JESXCF and specified JES terminated
    - Group name of the member whose connection terminated
    - Member name of the member whose connection terminated
    - The request token for the message that timed out
- **XCF Data (IXCYGEPL) -** The XCF data is the same that is passed to the XCF group user exit of an active member. XCF data notifies about changes that occur to the XCF members of a group, or changes to the systems in the sysplex.

1.16 JESXCF status monitoring



JESXCF status monitoring

Status monitoring services - Status monitoring services are authorized services that notify member exit routines (the group exit routine) of changes to the status of other members of their group and the systems in the sysplex. Members can request that XCF monitor their activity (the status exit routine).

There are two kinds of monitoring services in XCF:

- System status monitoring

  System status monitoring services monitor systems in the sysplex. The monitoring function uses the system status fields, which are periodically updated. All active application group exits receive control if a system fails to update its status field within a defined interval.

- Member status monitoring

  Member status monitoring lets a member actively monitor its status. The services are optional, but if a member does use the services, then it has to update its status field periodically. The member status exit gets control if the member fails to update its status field within a specified interval. When the member status exit confirms that the member's state changed, XCF notifies other members about the change through the group exit routines.

1.17 JESXCF macro services

- IXZXIXAT - Attach to server address space
- IXZXIXMB - Create a mailbox
  - IXZXIXMC - Clear a mailbox
  - IXZXIXMD - Delete a mailbox
- IXZXIXSM - Send a message
- IXZXIXRM - Receive a message
- IXZXIXAC - Acknowledge a message
- IXZXIXIF - Information retrieval
- IXZXIXDT - Detach from server address space

JESXCF macro services

The following JESXCF macro services are available:

IXZXIXAT During JES3 initialization, each member attaches to a JESXCF group and identifies itself by a unique member name and a JESXCF group name representing the whole JES3 complex.

IXZXIXMB JES3 functions that intend to receive messages must build a mailbox (a logical queue of ordered messages that is maintained by JESXCF). This also implies that all JES3 functions that are to receive the JES3 global initiated communication (for example FSS order processing) must create a mailbox. When a JES3 function retrieves and acknowledges a message, JESXCF removes that message from the mailbox. A mailbox may also be cleared (IXZXIXMC) without receiving queued messages. When a mailbox is cleared, JESXCF acknowledges those messages it clears.

When JESXCF returns control after a successful attach JESXCF group request, a default mailbox (named SYSJES$DEFAULT) is supplied. This mailbox collects system event data. System event data includes any XCF events on any JES3 member of the JESXCF group and events on the MVS system under which it runs. JES3 does not use the default mailbox and deletes it.

IXZXIXSM The JESXCF send a message service sends a data packet to the same or another JES3 function in the same JESXCF group. A send message request can be issued from a function running under a JES3 main task, a JES3 subtask,

JES3 FSS address space, or a user address space. Once a message is sent, JESXCF delivers the message to the requested receiver.

JESXCF maintains the order of the messages sent. All messages are received in the order in which they are sent. Multi-segment messages are not sent until the entire message (all segments) is available.

The message sender identifies the destination of the message by supplying the member name and the mailbox name of the receiver. This receiving member can be any member of the JESXCF group, including the sending member.

IXZXIXRM Once a message is sent to a mailbox, the receiving member is informed through the POST exit routine. This posting action notifies the mailbox owner of a message to receive.

IXZXIXAC All received messages *must* be acknowledged to JESXCF independent of whether the sender requested an acknowledgement or not. After a message is acknowledged, JESXCF releases all resources held by the original message.

If the sender of the message requests an acknowledgement, the JESXCF returns an acknowledgement message as follows:

- To the mailbox specified on the ASYNCACK type send request as the mailbox into which the acknowledgement message is to be returned
- To the area specified on the SYNC type send request

IXZXIXIF The request JES3 member information service (IXZXIXIF) returns a record of data about the specific group member, and also information about the MVS system on which the member is running. The data is returned either through a mailbox or is placed directly into the requestor's data area in an array format. If the data is returned to the mailbox, JESXCF treats it as system event data.

The returned member information includes:

- Request token that was returned from IXZXIXIF service
- The release level of the JES product and JESXCF maintenance level
- XCF group name and member name
- MVS system name that the JES is running on
- User state information as set by IXZXIXUS macro service
- XCF member token and XCF sysplex token

Member status:

- Member is active, connection between the JESXCF address space and the JES address space is functioning.
- MVS XCF state of the member is active but the connection between the JESXCF address space and the JES3 address space is not functioning. The probable cause is a JES3 abend.
- Both MVS XCF status and JESXCF connection status indicate that the member is not active.

IXZXIXDT The detach from a JESXCF group service (IXZXIXDT) deactivates and removes a JES from an XCF group.

The JES3 FSS/FSA Connect/Disconnect SSI routine invokes IXZXIXDT during FSS disconnect processing. JES3 mains *do not* detach from the JESXCF group during JES3 termination.

1.18 JESXCF user exits

- IXZXIT03 - Attach/Detach exit
- IXZXIT01 - Send message exit
  - Change destination, source, data, (add data)
- IXZXIT02 - Receive message exit
  - Change source, data, (remove data)
- Sample exits in SYS1.SAMPLIB (5.2)
  - IXZEX01A - IXZEX02A - IXZEX03A

JESXCF user exits

JESXCF provides the following user exits:

- **IXZXIT03 -** If an installation-defined JESXCF group is required, it should be created in the JESXCF attach/detach exit IXZXIT03. Exit IXZXIT03 receives control during the JESXCF attach/detach processing. Care should be taken not to inadvertently cause JESXCF to enter a recursive loop caused by the attach service calling the attach exit.

  Exit IXZXIT03 receives control during the IXZXIXAT JESXCF processing, and it may obtain a data area for later IXZXIT01 and IXZXIT02 processing, or it may provide an additional attachment to JESXCF. The IXZXIT03 exit receives control after the JES3 member has joined an XCF group but before either IXZXIT01 or IXZXIT02 receives control.

  During detach processing, IXZXIT03 receives control to free the data area obtained during the attach processing or to drop the attachment from JESXCF. After IXZXIT03 receives control, neither IXZXIT01 nor IXZXIT02 receive control for the XCF group from which JES3 is detaching.

  Exit IXZXIT03 communicates with JESXCF by setting flag bits and changing data fields in the parameter list passed to the exit.

- **IXZXIT01 -** Exit IXZXIT01 enables you to view, modify, add to, or reroute a message prior to the message being delivered to the receiving member. This exit gets control for the send message and the acknowledge message macro invocations.

  Exit IXZXIT01 receives control under the same task that issues the IXZXIXSM or IXZXIXAC macros. JESXCF invokes the exit after a message packet has been created but

before that packet is sent to its target member, and when the acknowledging member replies to the originating system for all acknowledgements except for the request COMM type message acknowledgements.

The data passed to this exit is in a data space; therefore, the data can be accessed through an access register. The exit receives control in access register address space control (AR ASC) mode. AR1 contains the ALET of the data space containing the exit parameters.

- **IXZXIT02 -** When the receiving function issues the receive macro, IXZXIXRM, the receive message exit (IXZXIT02) gets control first. This exit allows the receiving member to view or modify the message and to receive any extents that have been added to it by the transport exit. Once the message is received, the receiving function acknowledges the mail, informs JESXCF of its receipt, and if requested by the sending member, also returns acknowledgement data to the sender.

  The receiver of a message must:

  - Provide the group token that represents the JESXCF member and group
  - Identify the mailbox from which to retrieve the message
  - Indicate the type of message to be received:
    - Only system event messages
    - Only acknowledgement messages
    - Only JES- or installation-created messages

  Each segment of a multi-segment message is presented as a single message, and all segments are ordered as originally sent. However, individual messages from multiple members might not be ordered in the mailbox in a timestamp order.

Samples of these exits can be found in SYS1.SAMPLIB.

1.19 XCF signalling paths

- CTC device or coupling facility list structure
- Defined in COUPLExx parmlib member
- JES3 signalling paths can be:
  - Default paths - for use by all groups
  - Dedicated paths - for a specific group
- CTC paths are one directional
  - Requires a PATHIN and a PATHOUT

XCF signalling

XCF signaling services are the primary means of communication between members of an XCF group. XCF provides the following macros for sending and receiving messages:

- **IXCMSGO** - Allows members to send messages to other members in their group
- **IXCMSGI** - Allows the message user routine to receive messages sent from a member. When a member joins an XCF group, it must specify the address of a message user routine to be given control when another member sends you a message.
- **IXCMSGC** - Allows members to interact with the signalling services for enhanced signaling functions.

XCF groups

A group in XCF is defined as the set of related members defined to XCF by the multisystem application (for example, JES3). A group can span one or more of the systems in a sysplex and represents a complete logical entity to XCF.

JES3 mains join the sysplex using JESXCF services. Communications between JES3 mains are through JESXCF signalling services. JESXCF uses XCF signaling services for communications, XCF uses whatever connections are configured to z/OS, which could be the coupling facility or ESCON® CTCs.

XCF signalling paths

An XCF signalling connection consists of at least two operational signalling paths (one inbound and one outbound); a signalling path can be any combination of a coupling facility list structure, an ESCON CTC channel (SCTC control unit), or a CTC through a Multisystem Channel Communication Unit.

Implementing XCF signaling through coupling facility list structures provides significant advantages in the areas of systems management and recovery and, thus, provides enhanced availability for sysplex systems.

While a signaling path defined through CTC connections must be exclusively defined as either outbound or inbound, you can define a coupling facility list structure so that it is used for both outbound and inbound signaling paths, and MVS automatically establishes the paths. For example, if you define a list structure for outbound message traffic, MVS automatically establishes a signaling path with every other system that has the structure defined for inbound message traffic. Similarly, if you define a list structure for inbound traffic, MVS automatically establishes a signaling path with every other system that has the structure defined for outbound traffic.

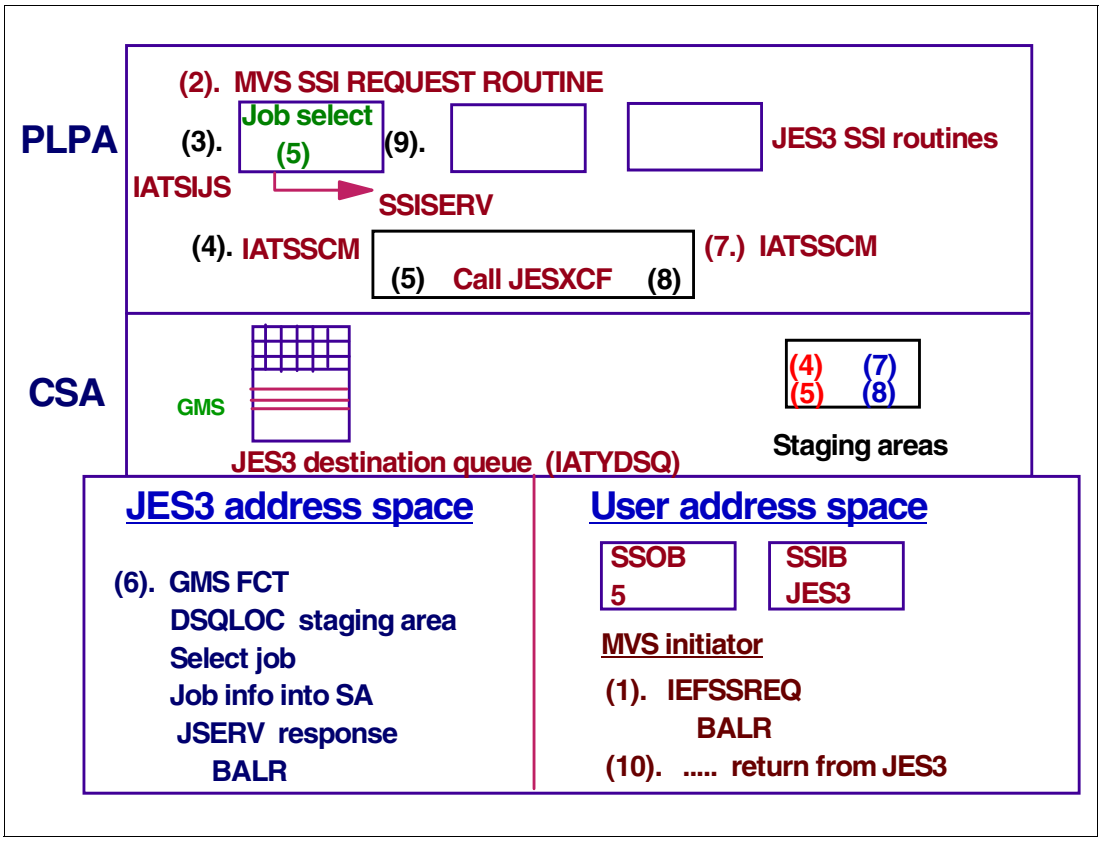
Signaling paths implemented through CTC connections are one directional. That is, on each system, messages sent to other systems require an outbound path, and messages received from other systems require a separate inbound path.

Communication paths

An XCF communication path is determined by the PATHIN and PATHOUT definitions initially defined in the COUPLExx parmlib member. The definitions can be modified with the SETXCF operator command.

Note: Two paths are recommended between systems.

1.20 JES3 SSI communication implementation - user to JES3



SSI communication - user to JES3

The subsystem interface (SSI) function routines and the JES3 subtasks use the SSISERV macro service to communicate with the global JES3. There are several types of SSISERV requests:

- **COMM -** Specifies that the request is for communication only. The data area address must be contained in a data area called JES3 subsystem communications service entrance list (SEL). The SEL is the parameter list for the SSISERV service.
- **WAIT** - Specifies that a response is required and that the routine is to wait until the response is received. The data area, ECB, and buffer addresses must be in the SEL.
- **REPLY** - Specifies that a response is required but the routine is not to wait for the response. The data area address must be in the SEL.
- **ACK -** Specifies that the request is for communication and that the request requires an acknowledgment. The RESP subparameter is used to make the acknowledgment.
- **RESP** - Specifies the answer to a WAIT or REPLY request, or the acknowledgment to an ACK request. The staging area address is required. If you specify TYPE=RESP, do not specify the FUNC parameter.
- **PURGE -** Specifies that a staging area that had been routed to the requester is to be deleted. The staging area address in the SEL is required. If you specify TYPE=PURGE, do not specify the FUNC parameter.
- **EOMT** - Specifies the special interface from EOM/EOT SSI for staging area cleanup. The data area address in the SEL must contain the address of the SSIB/SSOB control block pair.

The service routine for the SSISERV macro requests is IATSSCM. It builds a staging area from the information provided in the SEL and calls JESXCF to send the request JES3. The JSERV macro provides communication between JES3 and a user or functional subsystem (FSS) address space, between JES3 on one processor and JES3 on another processor, and between FCTs in the same JES3 address space. The IATSSJS service routine converts parameters from a JSERV macro to those required by SSISERV and then issues the SSISERV request.

Staging areas

The data area JES3 uses in the SSI communication is called a staging area (IATYSTA). It contains the data describing requests for JES3 services, for transport to and from JES3 and related address spaces. When JES3 passes a staging area to JESXCF for transportation, JESXCF wraps an message envelope around the staging area. The envelope (IXZYIXEN) provides header and control information about messages being sent between JES software components.

Steps shown in the visual

The following steps are shown in the visual.

Step 1 An MVS initiator issued a function code 5 job select request. A subsystem options block (SSOB), contains a function code that defines the request being made. The SSOB also contains the address of the subsystem identification block (SSIB), which identifies the subsystem to which the request is to be sent (in this case JES3). The expansion of the IEFSSREQ macro contains instructions to cause entry to the MVS request router routine IEFJSREQ.

Step 2 The MVS SSI request router routine get control, verifies the validity of the request, and for a valid request passes control to the subsystem's (JES3) job select SSI routine.

Step 3 The function code 5 (Job Select) routine, IATSIJS, gets control. IATSIJS builds a SEL for job select queue element (JSQ) to be passed to generalized main scheduling (GMS - active on the global processor). IATSIJS obtains a storage area to be used for the response buffer for the SSISERV TYPE=REPLY macro request that will pass control to the JES3 subsystem communication services (IATSSCM).

Step 4 Module IATSSCM validates the SSISERV request, determines the JESXCF member (the global) and the mailbox to receive the request. The request is sent to JESXCF with IXZXIXSM macro service specifying the requester's response buffer as the area to be used to receive the response data. After passing the request to JESXCF IATSSCM waits for the response.

Step 5 JESXCF sends the message to the destination member. Once the message arrives at the destination, JESXCF on that member invokes the JES3 main processor mail box post exit routine in JES3 subsystem communications read end module (IATSSRE). The post exit routine posts the JES3 job selection function (GMS) associated with the JES3 main that issued the job select SSI request. The JES3 nucleus task is also posted for work.

Step 6 The GMS function (IATMSMS), posted (in step 5) to process a job select request, retrieves the incoming requests with the JES3 DSQLOC macro service. The DSQLOC macro invokes JES3 subsystem communication destination queue services module (IATSSDQ). IATSSDQ makes calls to the JESXCF receive message service (IXZXIXRM) to receive each message that has been sent to the mailbox. Messages are unwrapped and all information about the message that JES3 needs to know (such as if the message was resent, or if the sending member was restarted) will be propagated into the staging area. The staging area will be queued to the end of the target destination queue's staging area queue. IATSSDS

routine then returns to the caller, passing back the address of the first staging area. Once GMS finds a job for the job select SSI request, a reply is send back to the requestor with the JSERV macro service implemented in JSERV conversion routine (IATSSJS). IATSSJS invokes the SSISERV service which passes the response back to the requesting main processor.

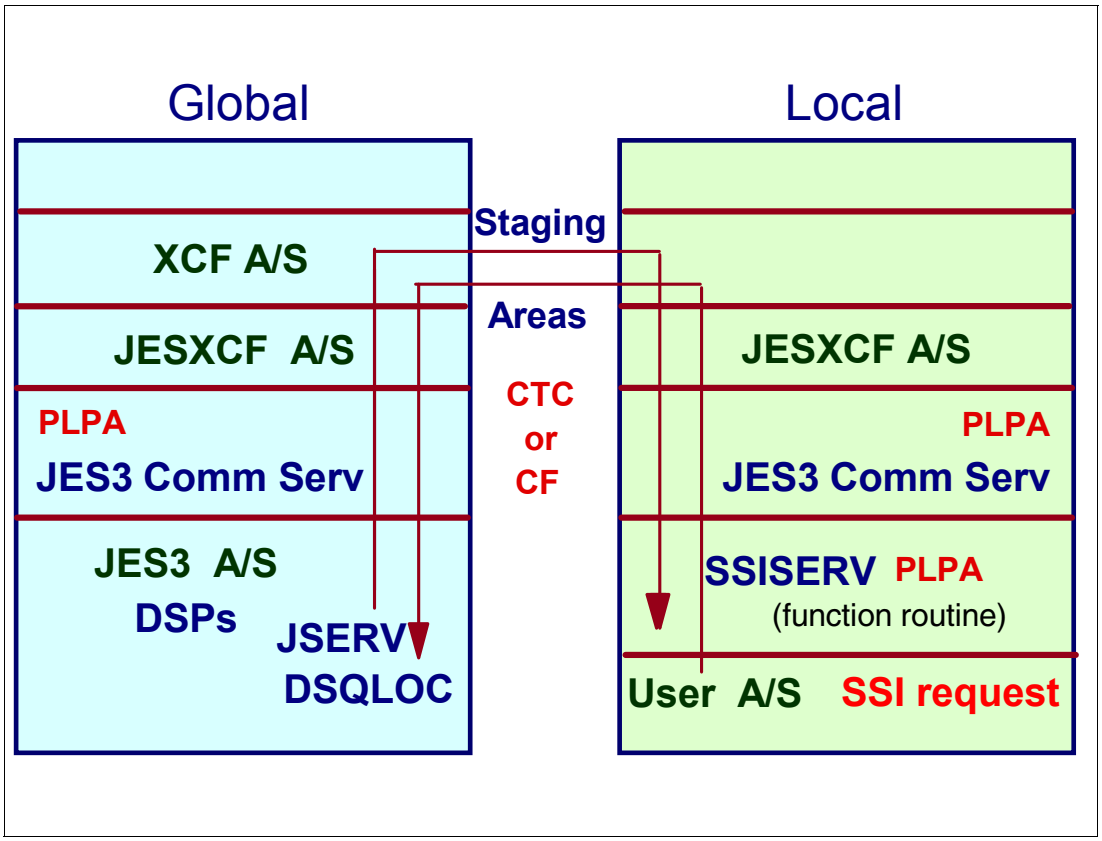
Step 7 On the requesting main the IATSSRE routine gets control, it copies the response data to the response area obtained by the ITASIJS function and posts the waiting function IATSSCM routine.

Step 8 The IATSSCM routine resumes processing and returns to the job select SSI routine IATSIJS.

Step 9 IATSIJS updates the SSOB and the SSOB's function dependent area (SSJS) with the job select information returned from the global JES3.

Step 10 IATSIJS returns control to the initiator.

1.21 SSI communication implementation - Global / Local



Global - Local communication

When a SSI request comes from an address space on a local main, the flow is the same as in the last figure until step 6. Since the GMS function is on the global processor, it is necessary to send the staging area to the global. JESXCF uses XCF to move the staging area across the CTC to the global. XCF on the global receives the request and uses JESXCF to pass the staging area to IATSSCM. The staging is then placed on the destination queue of GMS.

GMS selects a job and uses the JSERV macro to pass the data in a staging back to the address space on the local. After a function has been posted that an SSISERV request has arrived for it to process, the function uses the DSQLOC service to get the request. DSQLOC adds any new requests to the existing DSQ staging area queue, and returns the first request on the queue to the caller. The caller can then process the first request, or run the queue to determine which request it wants to process. When the request is processed, the JSERV service is then used to purge it, or to send a response back to the original requester.

When JSERV or SSISERV is called, JESXCF takes the information provided by the requester and builds a staging area. Then, based on the request type, it calls a JESXCF service to build a JESXCF message out of the staging area and then transport the message to its destination.

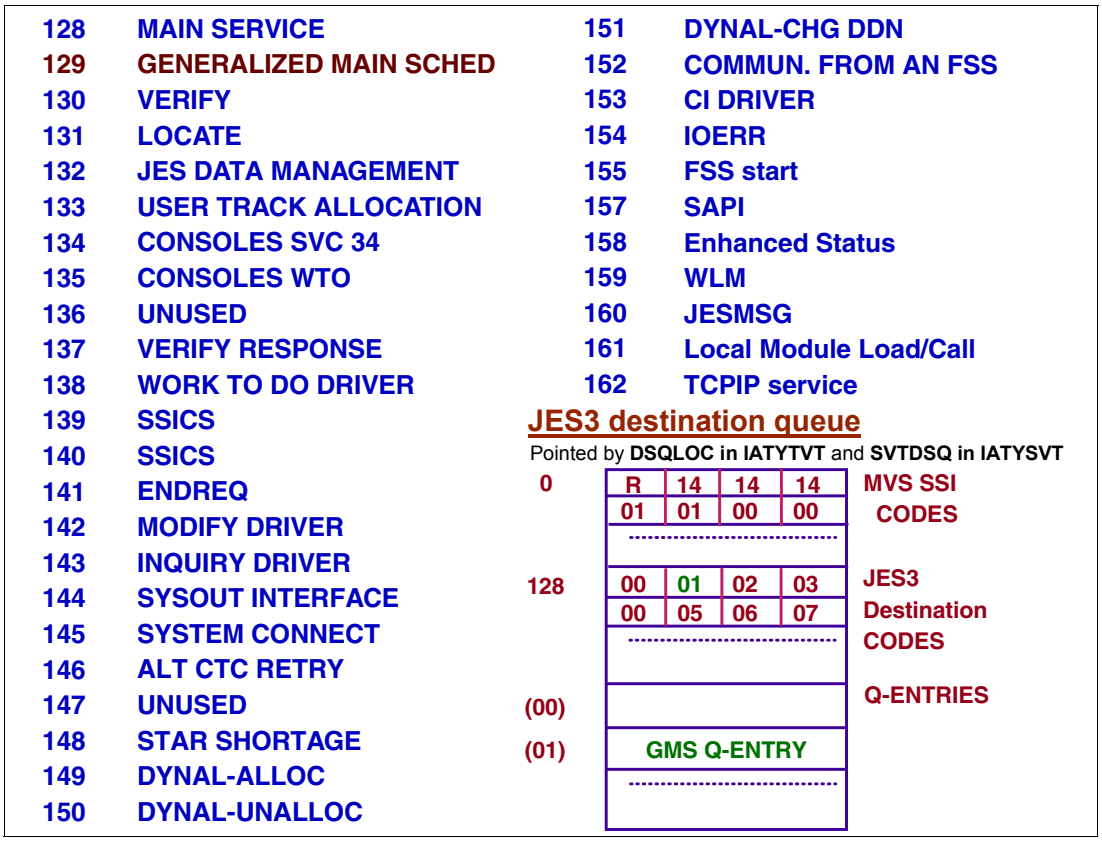
1.22 JES3 SSI functions

1	SYSOUT	24	COMMON ALLOCATION
2	CANCEL	25	COMMON UNALLOCATION
3	JOB STATUS	26	CHANGE DDNAME
4	EOT	27	CHANGE ENQ USE ATTRIBUTE
5	JOB SELECTION	28	DDR CANDIDATE SELECT
6	ALLOCATION	29	DDR CANDIDATE VERIFY
7	UNALLOCATION	30	DDR SWAP NOTIFICATION
8	EOM	31	DDR SWAP COMPLETE
9	WTO/WTOR	32	SVC34 COMMAND FAIL
10	CMD PROCESSING (SVC34)	34	WRITE TO LOG
11	REMOT DEST VALIDITY CK	40	EARLY VOLUME RELEASE
12	JOB DELETION	53	FSS/FSA CONNECT/DISCON
13	JOB RE-ENQUEUE	54	SUBSYSTEM VERSION
16	OPEN	56	SMS TO JES3 COMMUNICATION
17	CLOSE	62	BDT SUBSYSTEM
18	CHECKPOINT	64	TRANSACTION PROCESSING
19	RESTART	72	VARY PATH CALL
20	REQUEST JOB ID	75	NOTIFY-USER MSG ROUTER
21	RETURN JOB ID	77	PERSISTENT JCL
22	STEP INITIATION	79	SYSOUT API
23	DYNAMIC ALLOCATION	80	ENHANCED STATUS
		81	JES3 SSI Authorization PC

JES3 SSI function

The JES3 SSVT mapping macro IATYSVT contains a list of the JES3 supported SSI function codes. In the SSVT area, followed by the MVS function code matrix and address list, JES3 maintains its own data (queue anchors and address constants). The “extension data” is mainly used by the JES3 routines that execute outside the JES3 address space. For example the JES3 destination queue (DSQ), a control block used by subsystem interface routines to route requests to the JES3 functions responsible for servicing the requests, is pointed from the JES3 SSVT extension.

1.23 SSISERV function codes and JES3 destination queue



SSISERV function codes and JES3 destination queue

The subsystem interface (SSI) function routines and the JES3 subtasks use the SSISERV macro to communicate with JES3. The FUNC= keyword parameter specifies either the subsystems options block (SSOB) function code of the requester or, if the originator is JES3, a JES3 destination code. The actual data transportation requested by the SSISERV macro is done by JESXCF. Once the JESXCF message arrives at its destination, the staging area in the message envelope is queued into the JES3 destination queue.

Destination queue

The JES3 destination queue (DSQ) is a control block used by subsystem interface routines to route requests to the JES3 functions responsible for servicing the requests. The DSQ control block is made up of a queue index section and followed by the queue entry section. In the queue index section offsets 00 - 7F are SSOB function codes and offsets 80 - FF are JES3 destination codes. An index entry byte used to calculate the offset into the corresponding queue entry. The queue entries include: event completion flag (ECF) and mask, entry status flags, pointers to the first and last staging are on the queue, and the JESXCF mailbox name.

SSISERV service macro

When SSISERV service routine (IATSSCM) calls JESXCF to transport a staging area, it provides JESXCF with the name of the mailbox to receive the message envelope wrapping the staging area. Later, when the target JES3 function invokes the DSQLOC service (IATSSDS) to receive the staging area, the JESXCF receive message service (IXZXIXRM) is invoked. All messages that have been sent to the mailbox corresponding to the destination queue entry are received and unwrapped.

Staging area

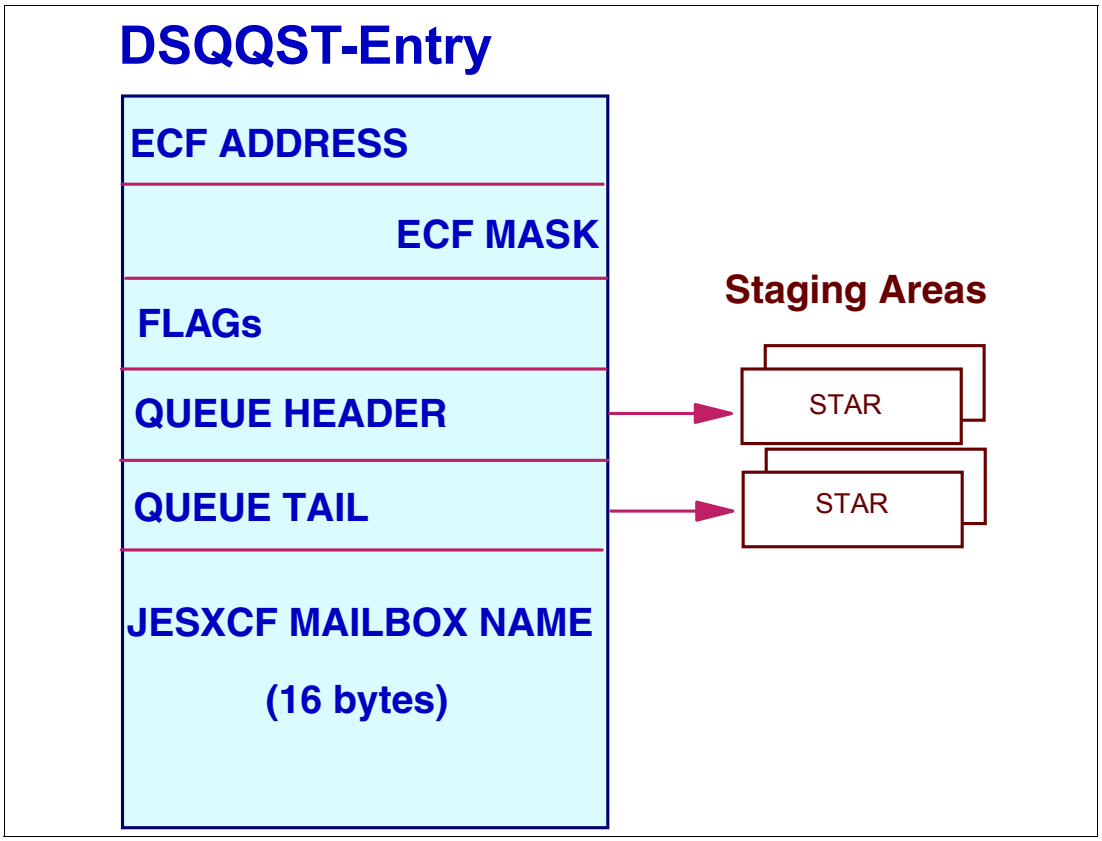
All information about the message that JES3 needs is propagated into the received staging area, and then the staging area is queued to the destination queue's staging area queue. The address of the first staging area on the queue is passed to the caller on return from the DSQLOC service.

The destination queue table (IATYDSQ) is built during JES3 initialization. The table can contain a maximum of 256 entries.

DLOCON macro

Issuing the DLOCON macro enables a function in the JES3 global or local address space to receive unsolicited communication (staging areas) from a user's address space, from JES3 on another processor in the processor complex, or from an FSS or FSA in another address space. The DLOCON service routine (IATSSDS) marks the requested destination queue entry as active, and save in it the input ECF address and mask. It also creates a JESXCF mailbox for the DSQ entry with the JESXCF IXZXIXMB macro. The mailbox name is also set into the destination queue entry. When a JESXCF mailbox is created, a subroutine in IATSSRE is specified as the JESXCF post exit routine for the mailbox.

1.24 JES3 destination queue - IATYDSQ



Destination queue entries

During JES3 initialization, the destination queue table (IATYDSQ) is built. The table can contain a maximum of 256 entries. Each entry in the destination queue table is a queue of staging areas that are waiting to processed. A single entry in the destination queue is used for all staging areas that were sent to be processed by that function for which the destination queue was built for.

The destination queue entry has the pointers to the staging areas waiting to be processed.

JESXCF mailboxes are used to manage SSISERV requests received by the JES3 address space. A JESXCF mailbox is created the first time DLOCON is called, and is used to receive requests associated with the DSQ entry indicated by the caller.

When SSISERV calls JESXCF to transport a staging area, it provides to JESXCF the name of the mailbox in which to place the message. When the DSQLOC service is called to process a DSQ, it receives all outstanding staging areas from the JESXCF mailbox associated with the DSQ. These staging areas are queued to the DSQ staging area queue before control is returned to the caller.

1.25 Function control table - FCT

- FCT - JES3 dispatching element
  - An FCT represents a dispatchable unit of work in the JES3 adrress space, or an FSS which supports JES3 nucleys services
- Mapped by IATYFCT macro
- Major FCTs part of JES3 nucleus - resident FCTs
  - Module IATGRPT contains the resident FCTs, DSP dictionary entries, and user exit address list.
- Other FCTs added to FCT chain when needed
  - IATXGFC macro initializes an FCT for a DSP
  - IATXATF macro attaches an FCT to the active FCT chain in priority sequence.

Function control table (FCT)

To decide which piece of work to do, JES3 maintains priority ordered chain of a function control table (FCT) entries. This chain serves as the JES3 “master dispatching queue”. The FCT is a chain of entries, each representing work to do in the form of a dynamic support program (DSP). What distinguishes DSPs from ordinary routines or subroutines is that DSPs are escheatable units. Before a DSP is executed, it must be scheduled by JES3. DSPs accomplish the pieces of work that JES3 performs.

The FCT chain elements are arranged according to the priority of the DSP, going from the highest to the lowest. Certain resident FCT entries that represent required DSPs have a higher priority than other FCT entries, however, priority is not related to residence.

The priority of an FCT entry is determined by the response requirements of the JES3 function involved. The last (lowest priority) FCT entry represents the WAIT DSP. When dispatched, the WAIT DSP issues an MVS WAIT macro indicating that JES3 has no more work to do.

Some FCT entries represent DSPs that perform required JES3 functions like operator communication and output services. These FCT entries are always present on the FCT chain. They are called resident FCT entries and they are not dynamically added or removed from the dispatching queue.

Non-resident FCT entries, on the other hand, are added (IATXATF) and deleted from the dispatching queue as needed. These non-resident FCT entries perform operator utility functions as well as other basic JES3 functions.

1.26 JES3 FCT dispatching

- JES3 tasks (TCB) dispatched by MVS - IATNUC
  - Multifunction monitor - MFM
    - Dispatcher of JES3 functions
  - Active JES3 functions - FCT
- JES3 functions - FCTs
  - Resident and transient
  - Dynamic support program - DSP
  - FCT function states:
    - Program loading
    - Executing
    - Waiting to execute
    - Function complete - remove FCT

JES3 dispatched by MVS

JES3 has its own address space in each of the mains in the complex. Each JES3 address space has identical functional capabilities, although each address space normally performs only a specific set of functions. The bulk of functions JES3 are performed under the JES3 nucleus task (IATNUC load module) in the global. The MVS dispatcher gives control to JES3 address space when it finds “ready” work (SRBs or TCBs) for the address space.

JES3 functions - FCTs

JES3 has a its own master dispatcher, called multifunction monitor (MFM), for JES3 functions under the nucleus task. The MFM, (module IATGRCT) controls which function of JES3 is to get control. The master dispatching queue for JES3 is called function control table (FCT). Entries in the FCT table are arranged in priority order and each represents one or more dynamic support program (DSP). A DSP is a JES3 program that implements a JES3 function. One or more DSPs may be executed for a single JES3 function.

The resident FCT table is build during JES3 initialization. The multifunction monitor selects the highest priority “ready” FCT entry from the table and transfer control to it.

Non-resident FCT entries, on the other hand, are added (IATXATF - attach FCT to active chain) and removed from the dispatching queue as they complete. These non-resident FCT entries perform operator utility functions as well as other basic JES3 functions. Each small piece of work that JES3 performs when processing a job is accomplished with a JES3 program called a dynamic support program, or DSPs. Each DSP is represented on the FCT chain by one or more FCT entries or elements. The elements on the FCT chain are executed

according to their priority, and are placed on the FCT chain with the high priority elements first. The higher priority elements are executed before the lower priority elements.

JES3 function synchronization - AWAIT

The AWAIT macro allows a JES3 function to wait for the occurrence of one or more events.

The AWAIT macro refers to an one byte event completion flag (ECF) to synchronize processing. The macro supplies an ECF mask, which is compared to the ECF to determine when one or more of the events have occurred.

For the AWAIT TYPE=ON macro, completion is indicated when any corresponding ECF mask bit is on in the ECF.

For the AWAIT TYPE=OFF macro, completion is indicated when all bits specified in the ECF mask are off in the ECF.

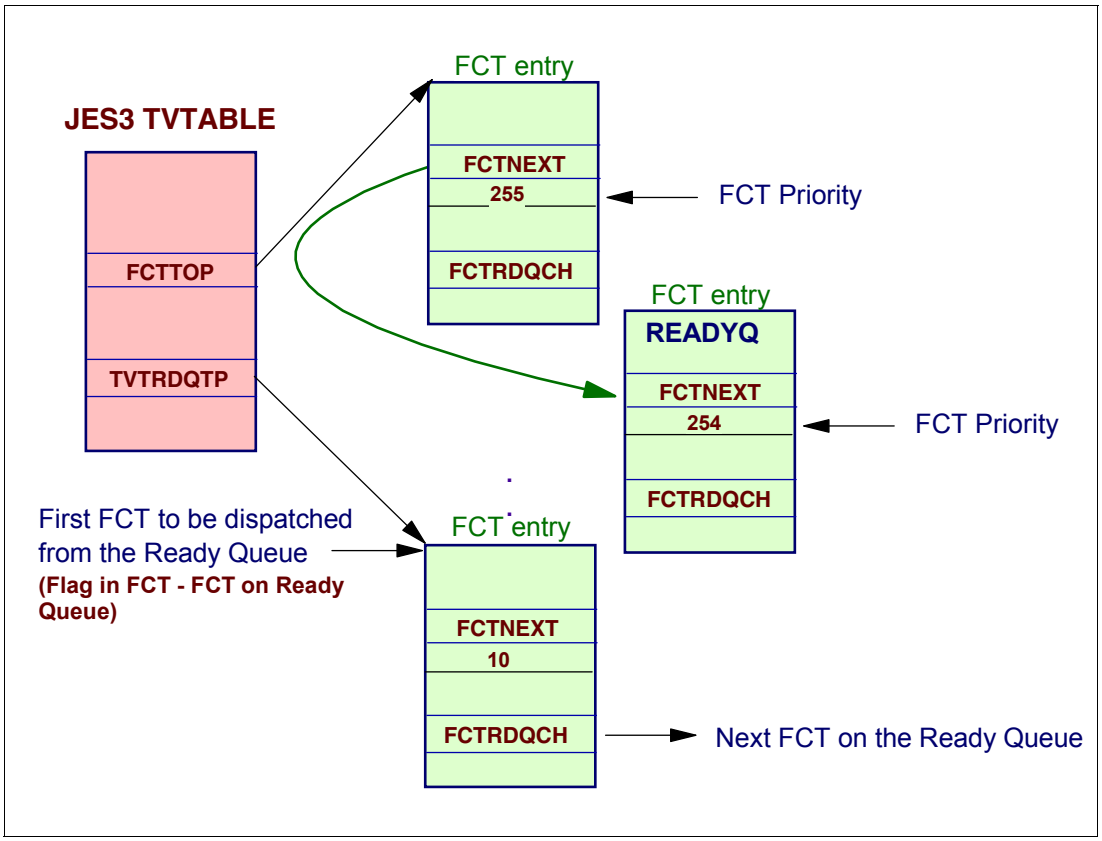
When the AWAIT condition is completed, the waiting FCT becomes “ready” for MFM dispatching.

Normal JES3 DSP processing consists of short execution intervals between waits. For most DSPs, these breaks occur naturally. However, if a DSP runs a long time without giving up control, it should issue the AWAIT TYPE=ON macro specifying a posted ECF at regular intervals.

The AWAIT TYPE=ON macro specifies that control is to be returned to the function issuing the macro only when at least one of the specified events have occurred. By providing dummy AWAIT macros in which the events being waited for are already posted as complete, the DSP gives the multifunction monitor (MFM) control to let higher priority DSPs execute.

JES3 does not provide a “POST” macro. ECFs are posted by using machine instructions that manipulate the ECF byte bits.

1.27 FCT chaining



FCT chaining

The FCT chain is expanded or contracted as functions become active or terminate; thus, the FCT reflects the active components of the system at any given time. The first FCT in the chain is pointed to from the JES3 TVTABLE entry at label FCTTOP.

JES3 dispatches an FCT entry the same way as MVS dispatches a task control block (TCB). The FCT entry representing a DSP is one element on the FCT chain. The FCT chain elements are arranged according to the priority of the DSP, going from the highest to the lowest. Certain resident FCT entries that represent required DSPs have a higher priority than other FCT entries, however, priority is not related to residence.

The priority of an FCT entry is determined by the response requirements of the JES3 function involved. The last (lowest priority) FCT entry represents the WAIT DSP. When dispatched, the WAIT DSP issues an MVS WAIT macro indicating that there are no more dispatchable FCT entries on the FCT chain. In other words, JES3 has no more work to do.

READYQ FCT

To handle the ready queue, JES3 uses the READYQ FCT entry on the normal FCT chain. The READYQ FCT is assigned a high priority of 254. Whenever an FCT is added to the ready queue, the READYQ FCT is posted. When it encounters the posted READYQ FCT, the multifunction monitor scans the ready queue before it scans the normal FCT chain. (However, before scanning the FCT ready queue, the multifunction monitor scans for FCTs with priority 255.) JES3 places FCTs on the ready queue in "last-in-first-out" order. The priority of the FCTs on the ready queue does not matter (as it does on the normal FCT chain).

1.28 Resident FCT chain

PRTY	FCTENTRY	DESCRIPTION	DRIVER MOD. - LOAD	
254	CONCMD	CONSOLE COMMAND SERVICE	IATCNCM	N
254	CONSERV	CONSOLE MESSAGES PROC.	IATCNSV	N
254	TIMER	TIMER ROUTINES	IATGRTM	N
254	READYQ	FCT READY QUEUE	IATRDYQ	N
250	JSAM	JES3 SPOOL I/O	IATDMGB	N
240	CONSDM	REMOTE MESSAGE SPOOLING	IATCNDM	N
240	RJPCONS	RJP MESSAGE PROC.	IATRJPC	N
53	MAIN	CTC I/O USER ADDRESS SPACE	IATMSMI	I
51	MAIN	GMS - JOB SCHEDULING	IATMSMS	I
40	TRAKALOC	FSS TRACK ALLOCATION	IATDMTA	N
35	DYNAL	DYNAMIC ALLOCATIONS	IATDYDR	N
34	ARMDRVR	AUTOMATIC RESTART MGMT.	IATGRPJ	N
32	CIDRVR	CI FSS PROCESSING	IATIICD	I
30	OUTSERV	OUTPUT SERVICE	IATOSDR	N
30	VERIFY	VERIFY PROCESSING	IATLVVR	N
30	SETUP	MAIN DEVICE SCHEDULING	IATMDDR	N

Resident FCTs

The visual shows the FCTs that are resident in the JES3 nucleus in module IATGRPT. There should be a minimum of 30 FCT entries in an active system. FCT entries are created for each active device in the system, printers and punches, both local and remote. Also consider the number of NJE and RJP lines active.

The next visual is a continuation of the resident FCT chain.

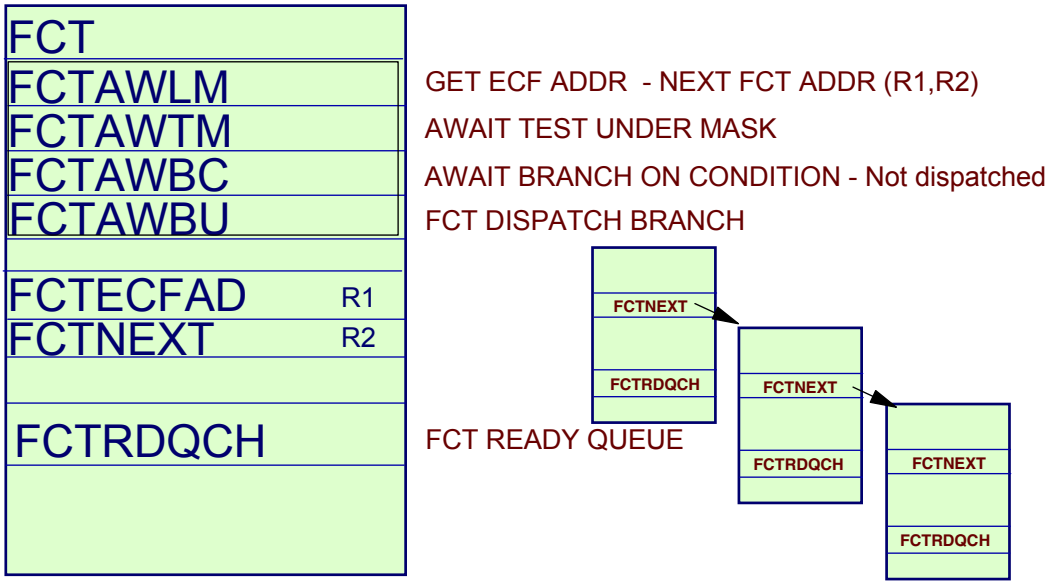
1.29 Resident FCT Chain

PRTY	FCTENTRY	DESCRIPTION	DRIVER MOD. - LOAD	
25	BDT	BULK DATA TRANSFER	IATBDCI	N
20	SAPIDRV	SAPI DRIVER	IATOSSR	N
15	MODDRVR	OPERATOR MODIFY PROC.	IATMODV	N
14	INQDRVR	OPERATOR INQUIRY PROC.	IATIQDV	N
14	ENSTDRV	EXTENDED STATUS	IATGRES	N
12	WTDDRVR	WORK-TO-DO PROCESSING	IATGRWD	N
12	PSODRVR	PROCESS SYSOUT REQUEST	IATGRWP	N
12	TSODRVR	TSO OUTPUT REQUESTS	IATGRWQ	N
11	PSODSP	PRPCESS SYSOUT DSP	IATOSPD	I
5	LOCATE	LOCATE PROCESSING	IATLVIN	N
5	LMLC	LOCAL MODULE LOAD/CALL	IATGRLMC	N
4	MSGC	MAIN SERVICE GENERAL COLL.	IATMSGC	N
2	JSS	JOB SEGMENT SCHEDULER	IATGRJS	N
2	GENSERV	GENERAL SERVICES	IATGRSR	N
1	FAILSOFT	FAILSOFT RECOVERY	IATFSLG	N
0	WAIT	WAIT ENTRY TO MFM	IATWAIT	N

Resident FCTs continued

This visual is a continuation of the previous visual and shows the remainder of the resident FCTs which are available at the end of JES3 initialization.

1.30 FCT dispatching




```
   LM    R1,R2,FCTECFAD-FCTSTART(R2)  GET CURR FCT ECF ADDR
*                                      AND NEXT FCT ADDRESS
   TM    0(R1),FF                      READY TO GO ?
   BC    NO, FCTAWLM-FCTSTART(,R2)     TRY NEXT FCT
   BR    R14                           DISPATCH
```

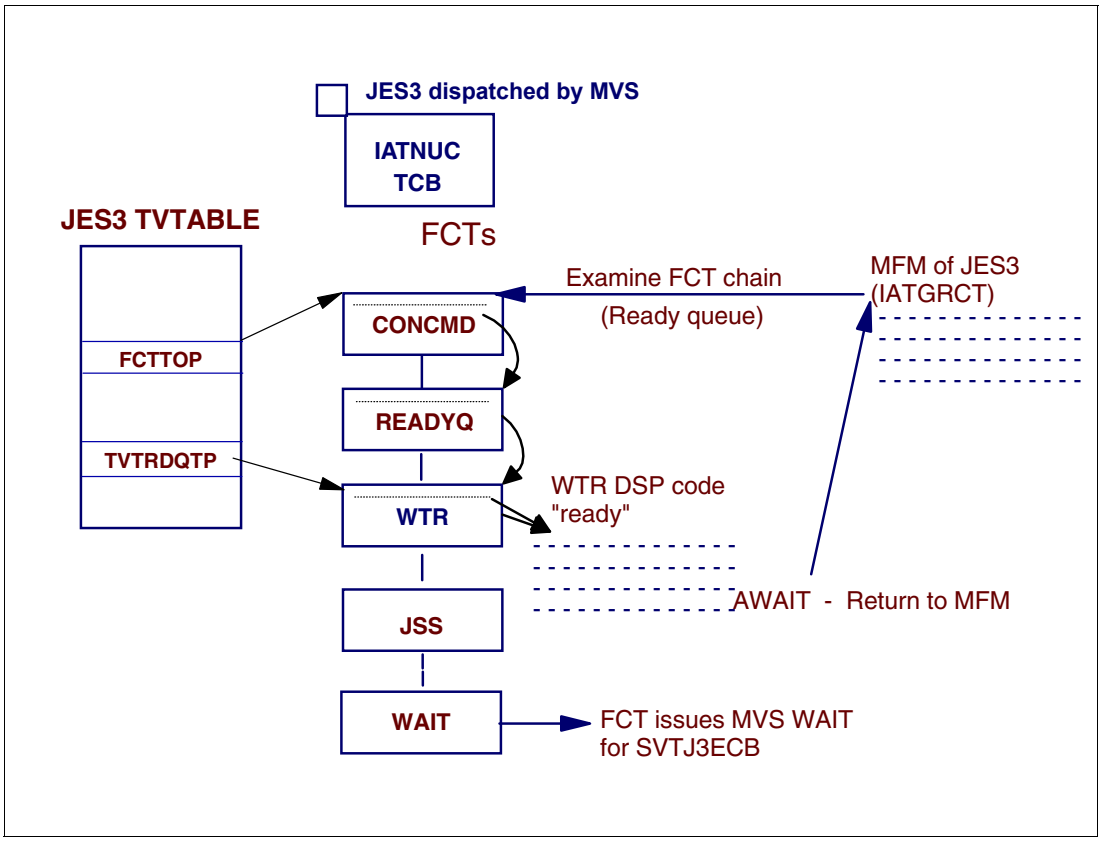
FCT dispatching

When JES3 has built the FCT chain (which is an ongoing activity as more jobs are read into the system), JES3 begins to allow certain FCT entries to be dispatched for execution. The part of JES3 that handles this dispatching is the multifunction monitor (MFM). The multifunction monitor performs the following steps:

Scans the FCT chain for an FCT entry eligible for dispatching. Each scan begins at the top of the chain, with the highest-priority FCT entry. If no entries are ready for dispatching, the WAIT DSP is dispatched.

After processing the two console-related FCTs, the TIMER FCT, (and perhaps the DSI FCT) the multifunction monitor checks the READYQ FCT (representing the READYQ DSP). The visual shows the beginning of the normal FCT chain and the position of the READYQ FCT on the chain immediately following the two console-related FCTs and the TIMER FCT. If the READYQ FCT indicates the presence of FCTs on the ready queue, then the multifunction monitor examines the ready queue FCTs before resuming the scan of the normal FCT chain. JES3 builds the FCT ready queue in last-in-first-out order.

1.31 Example - MFM dispatch of WTR FCT



MFM dispatch of WTR FCT

When an DSP cannot proceed until one or more events have occurred, it issues an AWAIT macro. The macro parameters specify the completion conditions of an AWAIT (ON - any of the bits in the mask on; OFF - all of the bits in the mask off), the mask bits in the ECF to be tested, and whether a single ECF or a list of ECFs is used. An ECF is any byte in storage. JES3 does not have a formal POST macro. Any instruction that changes the bit settings of an ECF can be used for ECF posting.

To speed up the process of locating a “ready” FCT to be dispatched, JES3 uses a separate queue of FCTs (TVTRDQTP) from the normal FCT chain. This separate queue, called “FCT ready queue”, contains FCTs for JSAM I/O completion (AWRITE, WRTCHAIN, and JESREAD) that are posted and dispatchable. JES3 uses the READYQ FCT entry on the normal FCT chain to process the ready queue. When ever an FCT is added to the ready queue, the priority 254 READYQ FCT is posted. When the MFM encounters the READYQ FCT posted, it scans the ready queue before it scans the normal FCT chain. When all FCTs on the ready queue are processed, MFM starts again the FCT chain scan from the top (priority 255).

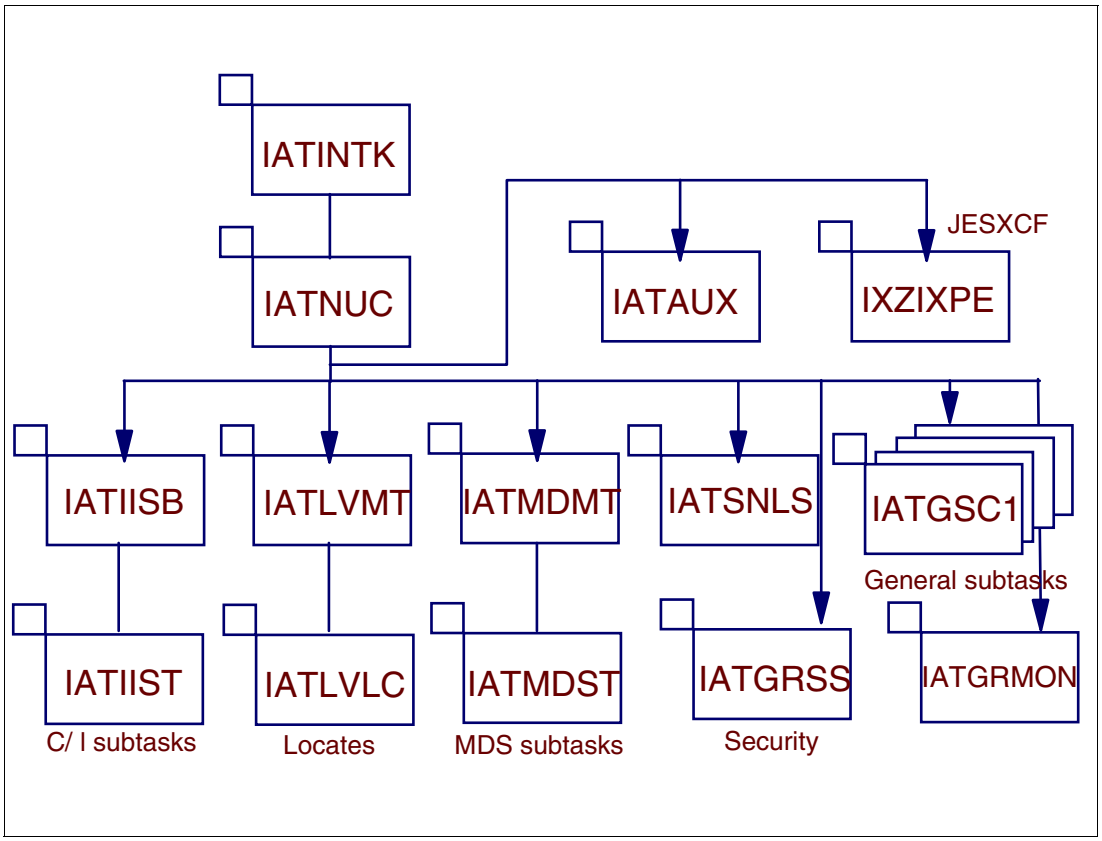
The multifunction monitor (MFM) performs the following steps:

1. Scans the FCT chain for an FCT entry eligible for dispatching. Each scan begins at the top of the chain, with the highest-priority FCT entry. If no entries are ready for dispatching, the WAIT DSP is dispatched. The priority of an FCT entry is determined by the response requirements of the JES3 function involved. The last (lowest priority) FCT entry represents the WAIT DSP. When dispatched, the WAIT DSP issues an MVS WAIT macro indicating

that there are no more dispatchable FCT entries on the FCT chain. In other words, JES3 has no more work to do.

2. An FCT is eligible to get control if its AWAIT ECF is indicates completion. MFM tries to dispatches an FCT entry by branching to the first FCT in the chain. Each FCT contains code to test its Event Control Flag (ECF) and depending on whether or not the ECF is posted, the FCT branches to the next FCT in the chain (ECF not posted) or back into the MFM (ECF posted). At this point MFM determines if the "ready" FCT is the JES3 WAIT FCT, in which case MVS WAIT on master ECB is issued, otherwise the "ready-to-go" FCT is given control by restoring its register contents from the FCT and by branching to the address pointed by register 14.

1.32 JES3 TCB structure



JES3 TCB structure

The bulk of the JES3 processing executes under the IATNUC TCB. When the IATNUC TCB is given control by the MVS dispatcher, the process that had been interrupted resumes. If the JES3 “master” ECB (SVTJ3ECB - pointed also from ASPECB in TVT) was posted control is given to MFM. MFM gives control to the highest priority “ready” FCT. JES3 DSPs must not issue MVS WAITs. If DSPs require MVS services that explicitly or implicitly involve MVS WAITs, they must use generalized subtasks (IATGSC1) to execute the part of the program that involves MVS WAITs. The scheduling of work to all the other functional JES3 TCBs is done from the controlling DSPs running under the IATNUC task.

IATAUX task

The writer output multi-tasking (IATAUX) is used to offload a portion of the JES3 writer processing from the IATNUC task. At the point where a JES3 writer has selected a data set for printing, the writer may switch from the FCT under the IATNUC task to the auxiliary task. The writer output multi-tasking should be used when there is a large number of local and remote RJP printers. The writer output multi-tasking is not used for FSS writers. Writer output multi-tasking is optional and is specified through the JES3 initialization OPTIONS statement, as follows:

```
OPTIONS,MT= {ON|OFF}
```

Operators can control the auxiliary task through the commands:

```
*F MT= {ON|OFF}   -  Start or stop writer output multi-tasking
*I MT             -  Query the status of writer output multi-tasking
```

Other JES3 subtasks

IATMFTM This module is the JMF timer subtask which is attached when JMF is called and becomes active in the FCT chain. When IATMFTM gets control, it re-chains the JMF TCB above the JES3 nucleus task (IATNUC). Each time the STIMER cycle value expires, the IATMFTM sets the JES3 TCBs to be non-dispatchable. It then takes samples by calling the appropriate JMF modules and then resetting the JES3 TCBs to be dispatchable.

Operators can control the auxiliary task through the commands:

```
*CALL JMF   - Invoke JES3 Monitoring Facility (JMF)
*CANCEL JMF - Cancel JMF
```

IATMDMT This module is responsible for attaching and detaching the MDS subtasks (IATMDST).

IATMDST This module is the MDS subtask used by MDS system verify and system select processing for the SSI calls to DFSMS/MVS. The number of tasks created is up to a maximum of 10. The numbers of tasks that are active is a function of the MDS processing workload for DFSMS managed data sets.

IATLVMT This module is the locate master task. It is responsible for attaching locate subtasks (IATLVLC). When an IATLVLC terminates (normally or abnormally), IATLVMT:

– Posts the DSP that was using the C/I subtask

– Posts the JES3 IATNUC task.

IATLVLC This module is the JES3 locate subtask. It is attached by the IATLVMT and posted by the locate FCT driver, IATLVIN, when there is work to do. Locate function is used by DSPs to obtain data set catalog information. Locate processing is performed on the JES3 global, JES3 local or in a C/I FSS address space.

IATGSC1 This module for the generalized subtasks are used by JES3 functions, DSPs, for calls that have implied MVS WAITs thus avoiding a wait of the IATNUC task. The maximum number of non-specific generalized subtasks is eight and specific is six.

IATIISB This module is the master task for all of the converter/interpreter (C/I) subtasks. It is responsible for attaching and detaching the C/I daughter tasks (IATIIST). When an IATIIST terminates (normally or abnormally), IATIISB:

– Posts the DSP that was using the C/I subtask

– Posts the JES3 IATNUC task.

IATIIST This module is used to drive the MVS C/I processing. The number of IATIIST TCBs is controlled by the CICNT parameter on the STANDARDS initialization statement. Operator may modify the subtask count through the command:

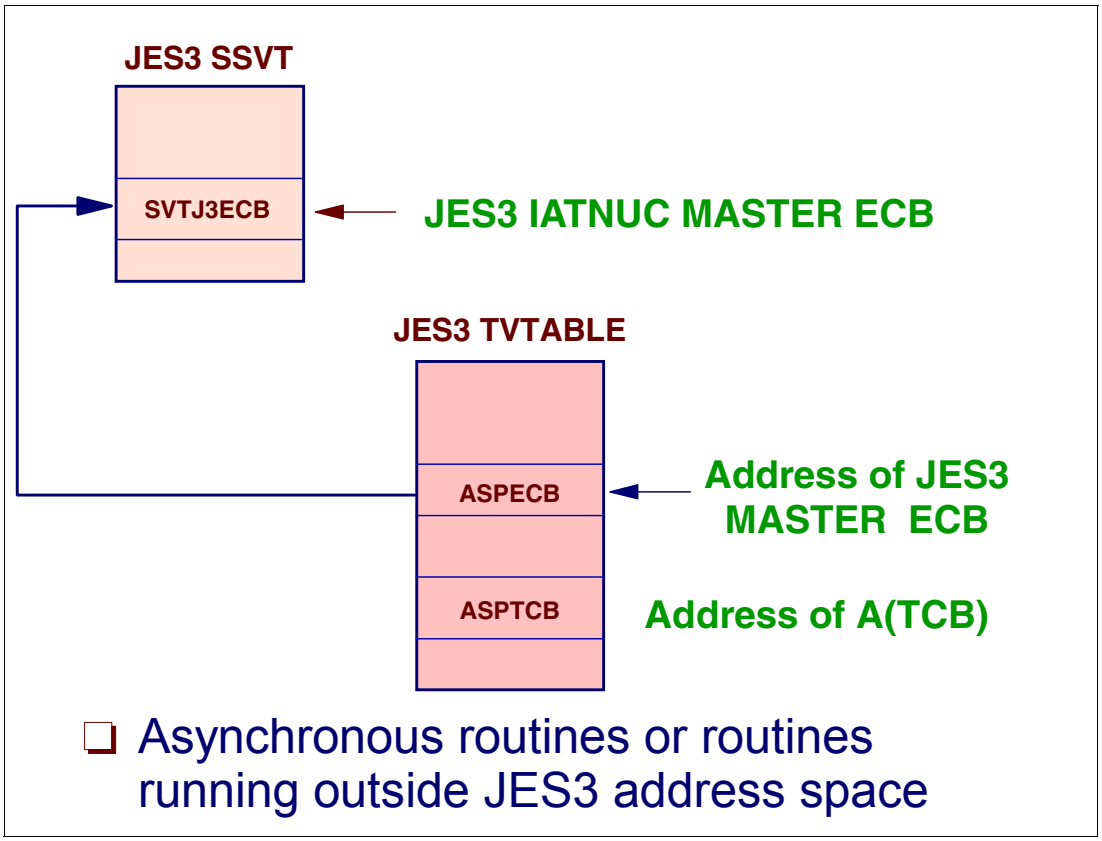
```
*F X,D=CI,MC=nn
```

IATSNLS This module consists of a subtask, which initiates the SNA RJP VTAM® communication, and VTAM exit routines. VTAM communicate with SNA RJP routines through VTAM exits, which are scheduled as IRBs running under the subtask.

IATGRSS This module is the general security subtask. It allow FCTs to asynchronously perform security checks without the AWAIT that is incurred while using a generalized subtask

IATGRMON This module is a subtask to monitor the active FCTs for the JES3 Nuc and Aux task.

1.33 Posting JES3 IATNUC ECB



Posting JES3 master ECB

The IATNUC TCB is posted by functions that are running asynchronously or initiated outside the JES3 address space, for example: JES3 I/O completion, timer pop, or staging area arrival. Each time the IATNUC TCB is posted, the ECF of the individual JES3 function (FCT) that processes the completed event is also posted. These ECFs have their addresses normally in the SSVT, DSQ or TVT control blocks. The IATNUC TCB is posted by obtaining the address of the IATNUC ECB and issuing an MVS POST macro.

Once the IATNUC TCB is posted, the task can be dispatched by the MVS dispatcher. Therefore, there is a percentage of time that the IATNUC TCB may be posted but not active while higher priority address spaces are dispatched.

JES3 DSPs must not issue MVS WAITs. If DSPs require MVS services that explicitly or implicitly involve MVS WAITs, they must use generalized subtasks (IATGSC1) to execute the part of the program that involves MVS WAITs.

1.34 Major JES3 control blocks

- JES3 functions
  - FCT - Number in storage definable
    - RESCTLBK,FCT=nnnn
- Job related
  - JCT - Stored in a separate spool data set
    - A JCT contains information pertinent to a single job
    - JCTs are resides in a dataspace when in-storage
  - JQE - Size definable via OPTIONS statement
    - OPTIONS,...JOBNO=(,,joblim)
  - RQ - Size determined by JES3
    - Cell pools built at initialization and dynamically
- JES3 address space
  - TVT - TVTABLE in JES3 nucleus
    - Contains pointers to almost everything

Function control table (FCT)

The FCT is a chain of entries, each representing work to do in the form of a DSP. Every DSP corresponds to one or more entries on the FCT chain. The FCT entry holds DSP status, DSP priority, the status of resources needed by the job, and other job information.

You should use the resident control block (RESCTLBK) initialization statement to preallocate storage for the highly used JES3 FCT entries: RESCTLBK,FCT=nnn. Using high enough RESCTLBK FCT count can reduce GETMAIN overhead and the associated paging overhead.

Job queue element (JQE)

JQE is a control block containing summary information a job description (JCT). JQEs are storage resident and are used by JES3 instead of JCTs when scheduling work.

The size of the JQE tables are determined by the number of jobs allowed in the system and the range of job numbers as defined on the OPTIONS initialization statement.

```
OPTIONS,.,JOBNO=(lowest,highest,joblim)
```

Resqueue element (RQ)

Resident queue (RQ) is a storage resident control block that is created by the job segment scheduler (JSS) to represent current JES3 processing step (scheduler element) of jobs. It contains status information and queue pointers. a particular DSP.

An RQ consists of a fixed section (common to all functions), plus a function dependent variable section. There are five variable sections:

- OUTSERV variable section - contains data used by output service
- MDS variable section - contains data used by MDS for SETUP processing. The MDS variable section starts after the OUTSERV variable section.
- GMS variable section - contains data used by GMS after MDS SETUP is complete. The GMS variable section starts after the MDS variable section.
- CI variable section - contains data used by the CI and postscan DSPs. The CI variable section starts after the GMS variable section.
- Common variable section - contains data used by all other functions. The common variable section starts after the CI variable section.

Scheduler element

A scheduler element is a part of the JCT entry and represents one or more DSPs needed for JES3 processing of a job. A scheduler element defines a piece of work for JES3 to do. Job segment scheduler (JSS) is a DSP that scans the JCT table to locate scheduler elements eligible for processing, and then builds RQ and FCT entries so that the corresponding DSPs can be dispatched. JSS DSP itself is running under its own FCT.)

There are no external means to influence the size or number of extents in each RQ cell pool.

Job control table (JCT)

One of JES3 job's control blocks, the JCT entry, resides in storage until JES3 writes it into a unique data set on DASD and in a data space. This data set is called the JCT data set and it contains JCT entries for all jobs currently known to JES3. The data set is defined in the JES3 JCL procedure and is allocated during the JES3 address space initialization. The JCT data set does not reside within the DASD space allocated for spool. The JCT entries describe the jobs to be processed by JES3 and for each job the scheduler elements representing the DSPs required to process the job.

Transfer vector table (TVT)

Two control blocks, the subsystem vector table (SSVT) and the transfer vector table (TVT), anchor all the data areas and service routines required for JES3 processing. They exist in storage in two distinct areas; SSVT in MVS common storage (CSA below the line) and TVT in JES3 address space’s private area.

The JES3 private address space contains the control blocks for all global, local or C/I FSS data used for DSP executions. This data is accessed through the TVT. Most JES3 global services are accessed through the TVT. It includes the following information:

- The addresses of most of the control blocks in the JES3 private address space.
- The entry point addresses of most of the JES3 services (resident code).
- Status information for all JES3 functions.

1.35 JES3 DSP dictionary

- Resides in IATNUC - IATGRPT
  - Contains resident FCT table
  - Contains DSP dictionary entries
  - Contains DSP device requirement lists
    - IATYDSD macro - Add DSP to system
    - IATYFCD macro - Add resident FCT
  - User exit address lists

Dynamic support program (DSP). Multiprogrammed system components that are scheduled by the job segment scheduler (JSS) and implement a specified JES3 function. A DSP can be related to job execution, such as main service or output service, or it can be a background utility, such as the dump job facility.

JES3 DSP dictionary

The DSP dictionary provides information about all DSPs that can become active in the JES3 system. Such information is required to dynamically build non-resident FCTs.

The IATYDSD macro generates an entry for a dynamic support program (DSP) in the DSP dictionary (module IATGRPT or, in a C/I FSS address space, module IATGRPTF). An entry in the table is required for each DSP in order for it to be recognized as part of JES3. The IATYFCD macro generates the permanent function control table (FCT) entries in the DSP dictionary for JES3 functions such as: console service and the main device scheduler (MDS).

Dynamic support program (DSP)

A dynamic support programs are multiprogrammed system components that are scheduled by the job segment scheduler (JSS) and implement a JES3 function. A DSP can be related to job execution, such as main service or output service, or it can be a background utility, such as the dump job facility.

Job segment scheduler (JSS)

A dynamic support program (DSP) that selects scheduler elements that are ready for processing and then builds corresponding entries in the function control table (FCT).

Scheduler element

Scheduler element is the part of the job control table (JCT) entry that represents one or more dynamic support programs (DSPs) needed for processing of jobs by JES3.

1.36 JCT control block

- Job Control Table (JCT) control blocks
  - Entry created when a job enters system
  - Entries in job priority order
- JCT spool data set
  - Contains a record for every job in system
- Most important control block in system
  - Spool data set on volume is critical for restarts
- JCT utility program (IATUTJCT)
  - Copy JCT data set to a new JCT data set

Job control table (JCT) control block

The JES3 job is defined by means of a spool resident control block called a job control table (JCT) entry. The JCT also serves as a continually-updated checkpoint of the status of the job as it progresses through the system. A job exists in the system when it JCT entry is created and ceases to exist when the JCT entry is removed. The JCT entry is part of a major JES3 control block called the job control table (JCT) which is a data set on a DASD device. The name of the data set is JES3JCT. You define the JCT data set during JES3 initialization.

The JCT data set contains a record for every job in the JES3 complex. Many JES3 functions, such as inquiry and modify, access the JCT. The job segment scheduler (JSS) is the JES3 function that accesses and updates the JCT most frequently.

Each JCT entry contains information about the job and an entry for each SE. Each SE entry contains flags indicating that SE's status: inactive, active, or complete. JSS uses these flags to determine which SE is next for scheduling/ending function: the first SE that is not marked complete is selected, and is marked either complete or active, for either ending function or scheduling. The JCT also serves as a continually-updated checkpoint of the status of the job as it progresses through the system. A job exists in the system when its JCT entry is created and ceases to exist when the JCT entry is removed. The entry contains:

- The job name
- The job number
- The job status
- A number of SEs (one for each piece of work to be performed on behalf of the job)
- Spool addresses of other job-related control blocks

1.37 Job queue element - JQE

- IATGRJX - JQE/JCT access method services
  - IATXJCT and IATXJQE macros
- JQE entries created when jobs enters the system
  - Contains a summary of information from a JCT entry
- Contains a list of functions for job scheduling
  - Function is called a scheduler element
  - Contains status of function - active or waiting to be scheduled
  - Mapped by IATYJQE

Job queue element (JQE) control block

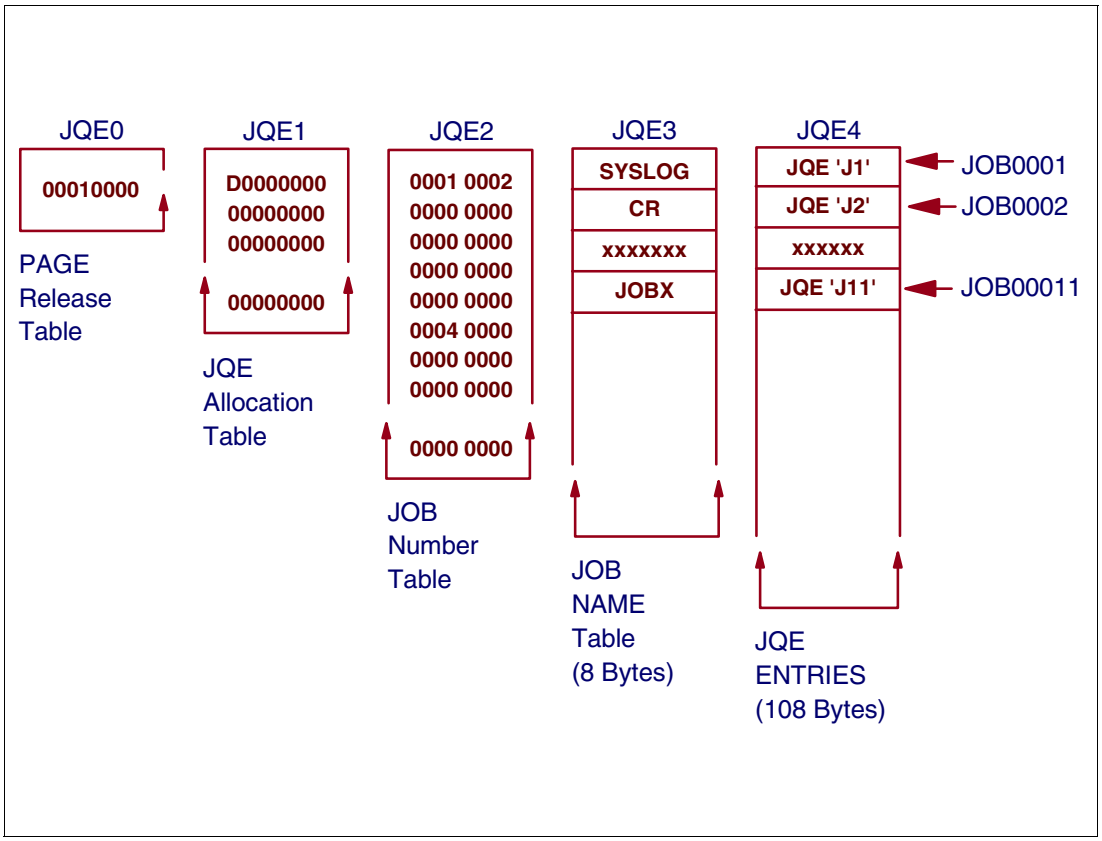
IATGRJX module services the IATXJCT and IATXJQE macro requests. These macros are used to access the JCT and JQE structure respectively.

A JQE is 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 JES3 processing.

When a job enters the system, JES3 assigns a job number and a JQE to it. When the JQE is assigned, it is added to the appropriate priority chain anchored in the JCT access method data area (JQX), based on the priority specified in the job's JCL.

Once a JQE is associated with a job, that JQE is always on a priority chain (although it may not always be on the same priority chain). The job scheduler (JSS) searches these priority chains for scheduler elements (SEs) ready for scheduling.

1.38 JQE table



Job queue element (JQE) table

The job queue element (JQE), another JSS-related control block, is also closely related to a job (JCT) but is kept in storage. The JQE contains almost all the data critical to JSS processing decisions and exists to reduce the I/O activity that would result from frequent examination of JCT entries for the scheduling of work. It is a highly abbreviated collection of information about the job, mostly copied from the job's JCT entry. It contains only that information needed to make scheduling decisions and maintain in-storage job status.

It is the JQE that is first examined by the job segment scheduler when it searches for work to be done. The JQE is also used to answer most system or operator inquiries about jobs, further removing a need for JCT access. JQEs are constructed during JES3 initialization and allocated for a job when its JCT entry is added to the job queue. The JQE structure includes five control blocks imbedded in one data area. JQE0, JQE1, JQE2, and JQE3 contain information to manage JQE4.

JQE0 JQE0 is a table for managing the virtual storage in which the entire JQE structure resides.

JQE1 JQE1 is the allocation table. Each bit corresponds to an entry in the jqe3 and JQE4 tables. There is one bit per job number. When a JQE needs to be allocated the bit map is searched for the first unused entry. The JQE3 and JQE4 that correspond to this entry are then used for the job.

JQE2 JQE2 is the job number table. There is a halfword for each possible job number. That is, the first JQE2 corresponds to the first job number, the second JQE2 to the second job number etc. The JQE2 contains the index (relative JQE

number) into the JQE3 and JQE4 tables of the entries that correspond to that job number.

JQE3 JQE3 is a jobname/userid table. There is an 16 byte entry for each job name in the system. The JQE3 is allocated using the JQE1 table.

JQE4 JQE4 contains the summary information from the JCT entry it is associated with. The number of JQE4 slots equals the maximum number of jobs that can be in the job queue at any given time. The JCT/JQE access method routine in IATGRJX updates JQE4.

The JQE tables are obtained (using the AGETMAIN macro) in contiguous storage during JES3 initialization. The JQE4 table, which is mapped by IATYJQE, contains an entry for each job. The job number range is defined in the initialization stream. Each JQE4 entry contains a subset of the information contained in the JCT for that job. The JQE4 entries also appear on many queues used by JSS for scheduling decisions, and therefore also contain chaining fields in addition to job information.

1.39 RESQUEUE control block - RQ

- Created when a job becomes active
  - Each time a scheduler element for job becomes active by JSS
- Contains all pointers to other job control blocks
- Key control block for user exits
  - Mapped by IATYRSQ

Resident Queue (RESQUEUE). A control block built in storage by the job segment scheduler (JSS) to represent a scheduler element during the life of the scheduler element.

RESQUEUE control block

The job scheduler (JSS) creates a new RQ each time it schedules a phase (SE) of a job; the RQ lasts only as long as the SE. Therefore, during its life, one job will have many RQs, as the JSS proceeds to schedule its SEs. A RQ entry may be concurrently used by many JES3 functions. This is accomplished by a series of chaining fields in each entry. However, each function has only one chaining field. All RQ entries are chained from the transfer vector table (TVT). The RQ is chained from the FCT when one FCT is responsible for processing a single job. The RQ entry has pointers to all of the job's control blocks making it an important control block for user exits.

1.40 RESQUEUE types and sections

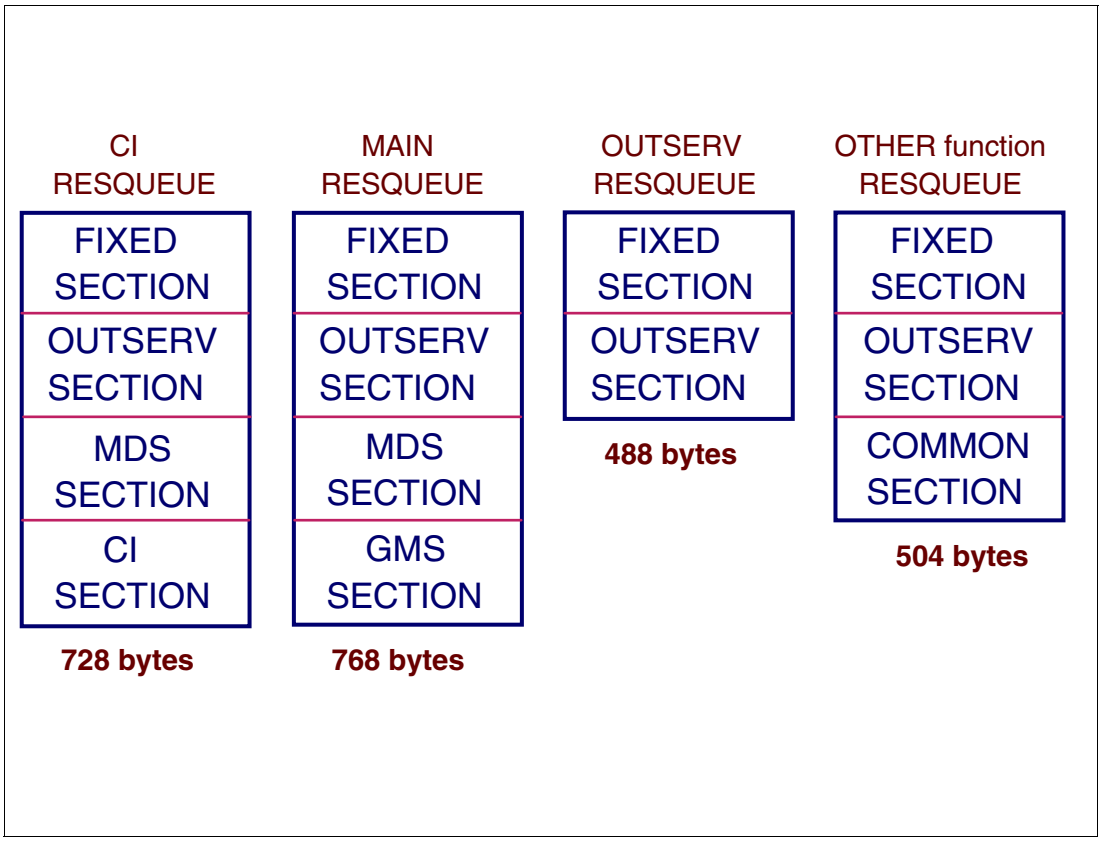
- 6 different RQ sections - HJS7750 size
  - FIXED section - 416 bytes
  - OUTSERV section - 72 bytes
  - MDS section - 200 bytes
  - GMS section - 80 bytes
  - CI section - 40 bytes
  - COMMON section - 16 bytes
- 4 RQ types
  - CI - MAIN - OUTSERV - OTHER function

RESQUEUE types and sections

Because many functions must concurrently access information in the job queue, some JCT entry information resides in main storage. This resident portion is called the resident job queue element (RQ). This control block is built by the job segment scheduler to represent an SE during its processing. It lasts only for the life of an SE. Each RQ contains pointers to the primary control blocks of a job on spool (JDAB, JST, JMR, JOBTAT, JDS, and OSE).

When the job segment scheduler is dispatched, it examines JQEs to find an eligible job so it can schedule a DSP. Part of this processing is construction of an FCT entry when the DSP being scheduled is not represented by a resident FCT entry. In addition to constructing and enqueuing an FCT entry (if necessary), the job segment scheduler always builds a resident job queue element. Spool addresses from the JCT entry to the job's other single-record files (SRFs) are extracted and placed into the RQ, which also contain status flags, job information fields, and queueing pointers.

1.41 RQ control blocks



RQ control block types

The RESQUEUE control block consists of 6 different sections. This change to the structure of the RESQUEUE is for storage constraint. The six different sections are:

Fixed This area contains data common to all functions and exists in every entry. It primarily contains data from the JCT.

OUTSERV This section contains data used by output processing and always follow the fixed section for every RESQUEUE type.

MDS This section contains data related to setup processing. It exists in the RESQUEUE entry following the output section and is present under the MAIN and CI scheduler elements.

GMS This section exists only under the CI scheduler element. It follows the MDS section in the entry.

CI This section exists only under the CI scheduler element. It follows the MDS section in the entry.

Common This section exists for all scheduler elements other than the CI, MAIN, and OUTSERV. It follows an OUTSERV entry in the entry.

1.42 JES3 naming conventions

- JES3 naming conventions
  - Module names : IAT ff xx
    - ff : functional area
    - xx : function performed
  - Module examples :
    - IATISDV - IATOSDR - IATSICA
    - IATISDR - IATOSSC - IATSI34
  - Coding macros - IATX...
  - Data macros - IATY...
  - JES3 nucleus - IATNUC

JES3 naming conventions

Most JES3 module names are seven characters long and begin with the characters IAT. The fourth and fifth characters are a mnemonic that represents a specific function. The sixth and seventh characters indicate a service performed by that module within the function. For example IATISDV:

- IAT signifies that this is a JES3 module.
- IS signifies that this is an input service module.
- DV signifies that this is the device driver module.

The mnemonic in the fourth and fifth character indicates any of the following module functions:

```
AB - abnormal termination
BD - MVS/Bulk Data Transfer
CN - console services
CS - callable services
DC - dependent job control
DJ - dump job
DL - deadline scheduling
DM - spool data management
DS - dynamic system interchange
DY - dynamic allocation fastpath
FC/FP - functional subsystem interface
FS - failsoft
GR - general routines
```

```
GS - generalized subtask service
II - converter/interpreter service
IN - initialization
IP - interactive problem control system
IQ - operator inquiry commands
IS - input service
JV - job validation
LV - locate/verify
MD - main device scheduler
MF - JES3 monitoring facility (JMF)
MO - operator modify commands
MS - main service and generalized main scheduling
NT - JES3 networking
OD - output service data
OF - initialization offset table
OS - output service
PU - purge
RJ - binary synchronous communication remote job processing
SN - systems network architecture remote job processing
SS/SI - subsystem interface
UT - operator utilities
UX - user exit
```

The sixth and seventh characters restrict any of the above mnemonics to further classification of a specific task, and indicate (but are not limited to) any of the following:

```
XM - cross memory processing
DV or DR - driver module for a particular DSP
DT or DA - data CSECT for a particular DSP
CR - card reader processing
MN - monitor
```

Due to the limited number of combinations provided by 2 characters, there are exceptions to these conventions.

Additional conventions are:

- The fourth character of most JES3 executable macros is an X (but there are executable macros that do not follow this convention):

  ```
  IATX__ __ __
  ```

- The fourth character of a JES3 data area mapping macro is a Y:

  ```
  IATY__ __ __
  ```

- The fourth character of a JES3 macro that expands within other JES3 macros is a Z:

  ```
  IATZ__ __ __
  ```

The JES3 nucleus load modules are named:

IATNUC - JES3 address space
IATNUCF - C/I FSS address space
IATNUCI - Initialization stream checker

2

Spool data sets and checkpoint

Spooling refers to process of communicating data to another program by placing it in a temporary working area, where the other program can access it at some later point in time. Spooling is often used when devices access data at different rates. The temporary working area provides a waiting station where data can reside while a slower device can process it at its own rate.

In a JES3 system, spool serves:

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

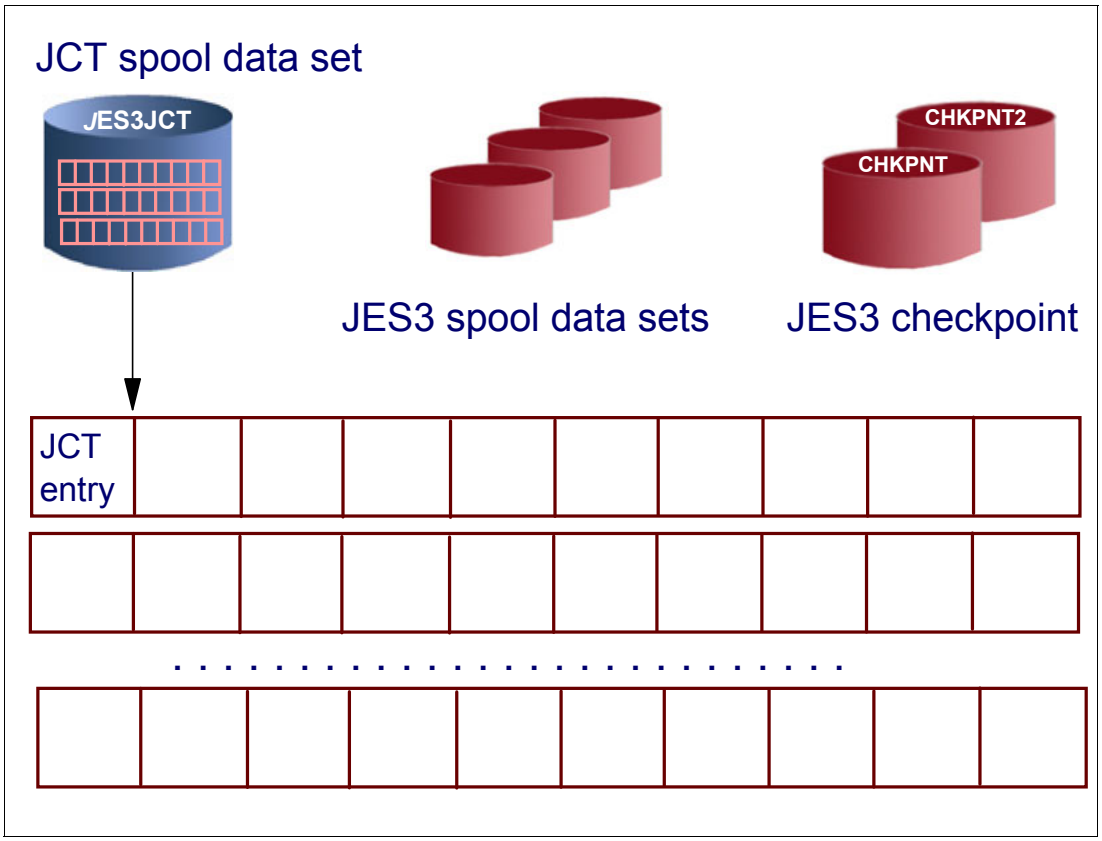
A spool device must be a DASD. Spooling refers only to use of DASD by JES3 for storage of jobs and job-related data. Use of the same device(s) for other purposes is not spooling.

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

The JES3 checkpoint data set contains the information required to initialize either a global or local JES3 processor which let JES3 to start either a global or local JES3 processor with little or no loss of system information.

The JES3 checkpoint facility writes job-related control block information to the JES3 checkpoint data set(s) at appropriate points in time during system processing; that is, as information changes in the system. This control block information is restored to the system after performing a hot or warm start. All other information is lost.

2.1 Spool data sets and checkpoint



JES3 spool and checkpoint

The JES3 spool consists of one or more single extent DASD data sets. Each data set is typically a full volume that starts and ends on a cylinder boundary. The data sets are logically divided into track groups, the unit of spool space allocation for jobs.

The system programmer defines a spool data set to JES3 by a TRACK or FORMAT statement in the JES3 initialization stream. Each TRACK or FORMAT statement defines a unique ddname of the DD statement or the ddname on a DYNALLOC statement that defines the spool data set. Spool data sets must be MVS allocated to each JES3 processor so that any main in the JES3 complex can perform I/O to a any spool data set.

JES3JCT is a the ddname given to the DD statement of job control table (JCT) data set. Other spool data sets allocated to JES3 cannot use this ddname.

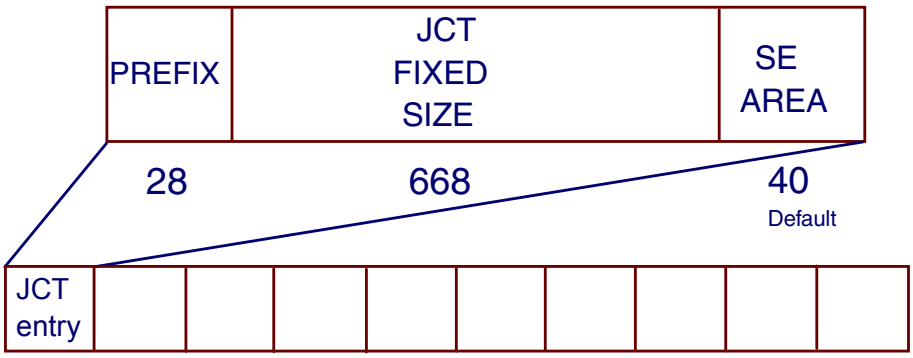
The JCT data set is the JES3 job queue. It is accessed by JES3 functions through spool data management services. The JCT data set holds JCT entries that represent jobs. The JCT data set will not contain any SYSIN or SYSOUT data.

The spool data set can be allocated either by using JCL DD statements in the JES3 start procedure or by the JES3 DYNALLOC initialization statement. To manage spool data sets, JES3 assigns a unique internal extent number to each data set during JES3 initialization.

The checkpoint data set(s) let(s) JES3 to store key information while the system is running. One or both JES3 checkpoint data sets must be defined in the JES3 start procedure using ddnames CHKPNT and CHKPNT2.

2.2 JES3JCT - JES3 job control table on spool

- JES3JCT - JCT data set - Allocate in CYLs
  - Must be allocated and cataloged before JES3 start
  - JCT entries are fixed-length, unblocked format
  - Number of cylinders determines number of jobs
- Place on lightly used volume



JES3 JCT spool data set - JES3JCT

One of the JES3 job control blocks is the JCT entry. The JCT resides in storage until JES3 writes it into the JES3JCT data set on DASD and into a dataspace. This data set is called the JCT data set and it contains all of the JCT entries. MVS allocates this data set during JES3 initialization. The JCT data does not reside within the DASD space allocated for the other spool volumes.

The JCT entries on the JCT data set are a fixed-length, unblocked format. Note that JCT entries vary in size: called jobs have two scheduler elements (SE), standard jobs have four (or more), and non-standard jobs have a variable number up to an installation-defined maximum.

The maximum number of scheduler elements is specified on the SE parameter of the OPTIONS initialization statement (10 through 90 is the valid range - 10 is the default).

Even though the JCT data set is not contained within the DASD space allocated for spool, it is read from and written to the JES I/O buffer pool by means of JSAM single-record file processing techniques. The JCT data set, depending on how much space was allocated to it, contains a fixed number of blocks.

The JCT data set, depending on how much space was allocated to it, contains a fixed number of blocks which determines the maximum number of jobs simultaneously in the JES3 complex.

2.3 JCT data Set allocation

- JES3 limits the maximum number of jobs in complex
  - OPTIONS initialization statement JOBNO= keyword
  - The range of job numbers from the JOBNO= keyword
  - The number of entries in the JCT
- How many cylinders to allocate ?
  - 3390 - 41 rec/trk - 615 rec/cyl
  - Allocating 35 cylinders - 21525 records
    - (35 x 615) - 64 = 21461 jobs
- JCT entry mapped by IATYJCT macro

```
IAT4030 0001 SPOOL DATA SET  IN USE
IAT4075 MAXIMUM NUMBER OF JOBS IS LIMITED TO 0021461 BY JCT DATA SET SIZE
*JES3 z1.10.0   SYSTEM  HOTSTART ON 2008.270 AS SC65
```

JCT data set allocation

You can allow as many as 999,999 jobs to exist in your JES3 complex at the same time. However, JES3 limits the maximum number of jobs that can be active concurrently by choosing the smallest of the following:

- The value you specify on the job limit parameter on the JOBNO= keyword of the OPTIONS initialization statement
- The range of job numbers that you define on the JOBNO= keyword of the OPTIONS initialization statement
- The number of entries in the job control table (JCT)

The JCT data set contains information about the status of jobs in the system. You must make the size of the JCT data set large enough to accommodate the maximum number of jobs that can be in your JES3 complex simultaneously. If the data set is larger than required, JES3 uses only that portion needed to hold the maximum number of jobs.

JCT size calculation

For optimum performance, minimize the size of the JCT data set as much as possible.

- Calculate the size, in bytes, of the JCT entry within the JCT data set:

  ```
  xxx= (4 x max SEs) + zzz + 28 bytes (prefix)
  where:
  ```

  - xxx is the size, in bytes, of a single JCT
  - 4 is the size of one JCT scheduler element in the JCT

- Max SEs is the maximum number of scheduler elements as specified
- zzz is the size, in bytes, of the fixed JCT. Field JCTFSIZE in the IATYJCT macro defines this value. If you have not modified the size of the JCT, the value of JCTFSIZE is 668 bytes.
- 28 is the size, in bytes, of the SRF header prefix mapped using the IATYSRF macro.

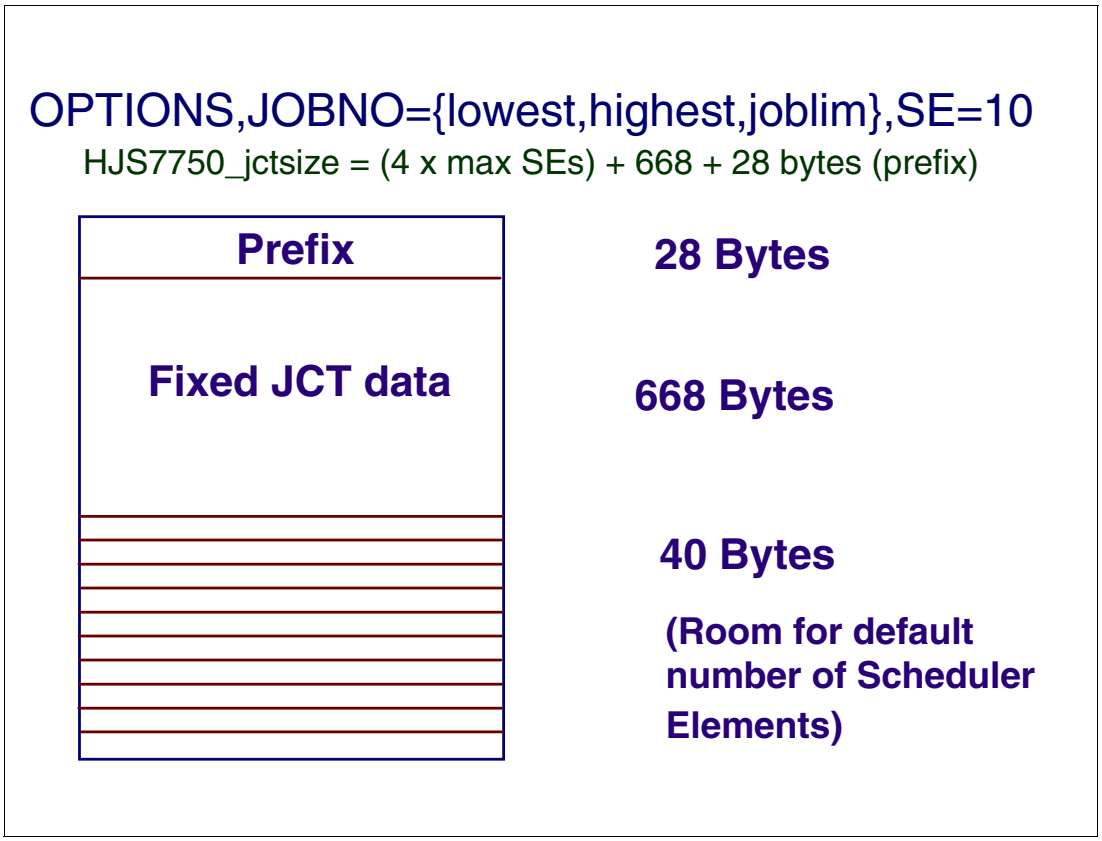
► Determine the number of JCTs per cylinder. For example, the IBM 3390 device contains 41 standard size JCT entries on a track and 615 JCT entries on 1 cylinder.

Sample JCT calculation

If you allocate 35 cylinders, then there are 21525 JCT entries allocated. JES3 reserves 64 entries for error recovery, so there are actually only 6686 JCT entries and the maximum number if jobs can only be 21461.

This number is shown during a JES3 start in message IAT4075 that are received during the JES3 start-up. Message IAT4075 is displayed only when the JES3JCT data set is not large enough to contain the JCT records for the defined maximum number of jobs or the number of job numbers in the defined job number range is less than the defined maximum number of jobs.

2.4 JCT spool record



Determine the Size of the JCT Data Set

The size of a JCT entry is (28 + 668 + 40). The 40 is 10 SEs, which is the default specified in the OPTIONS statement (SE=10). To determine the size of a JCT spool record, use the following formula:

```
JCT data set          (64 + max # of jobs)
       size      =   ----------------------------
   in cylinders       number of JCTs per cylinder
```

Where:

- The JCT data set size in cylinders must be incremented to the next whole number if the result is not a whole number.
- 64 is the number of JCTs JES3 reserves for write error recovery.
- The max # of jobs is the maximum number of jobs that can simultaneously be in your JES3 complex.
- The number of JCTs per cylinder (as explained in the device specific table). For IBM 3380 and 3390 devices, use the following documents to determine the number of JCTs that will fit on a track and cylinder.

For example, the IBM 3390 device contains 41 JCT entries per track and 615 entries per cylinder.

2.5 JCT dataspace

- JCT dataspace - name JES3JCT
  - Created by JES3 JCT initialization
  - Entire JCT data set kept in dataspace
  - Dataspace deleted when JES3 terminates
- JCT performance
  - Reduces amount of I/O to read JCT
- JCT reads
  - No I/O required if JCT in real or expanded storage
- JCT writes
  - JCT data set updated on writes
  - JCT entry in dataspace updated

JCT dataspace - JES3JCT

The JCT data set is formatted during JES3 initialization during a cold start. During initialization, JES3 places a copy of the JES3JCT data set JES3 spool into the JES3JCT dataspace. A data space is a range of up to 2 gigabytes of contiguous virtual storage. The JCT dataspace is a pageable, non-permanent dataspace. If an error occurs during initialization of the JCT dataspace, JES3 issues error messages and disables the data space. Even when the dataspace is disabled, initialization continues and JES3 uses the copy of the JCT data set on spool.

The JCT dataspace is created during the JES3 initialization. If the data space creation fails, the operator is notified and the JCT access will be to the JCT spool data set for reads and writes.

JCT performance

When the JCT dataspace is successfully created, there is no read activity shown for the JCT data set. A copy of the entire JCT data set resides in the dataspace. The purpose of this dataspace is to reduce the I/O necessary to access the JCT data set. The amount of I/O reduction obtainable is dependent on the amount of real and expanded storage.

JCT Read Requests

By keeping the JCT data set in a dataspace, all read requests for a JCT entry can be eliminated if the corresponding page that contains the entry is in real or expanded storage. I/O is necessary only if the page has been migrated to external storage. The I/O then takes place from page data set.

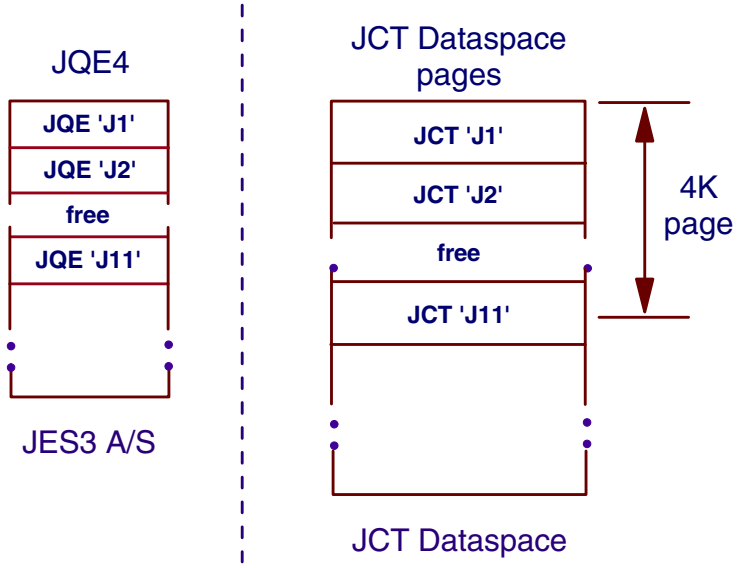
When a DSP issues a JESREAD request for a JCT entry, the data is moved from the dataspace to a JSAM buffer and the DSP's FCT is posted.

JCT Write Requests

A JCT write request updates the entry in the JCT data set on spool. The JCT entry in the JCT dataspace is also updated. The JCT entry in the Data Space is moved to a JSAM buffer for the I/O to occur.

2.6 Access to dataspace JCT entry

- JQE entry has corresponding JCT entry
  - (((JEQ_ADDR - JQE4_BASE_ADDR/JQESIZE)*JCTSIZE)+DATASPACE_ORIGIN)
- Page release table for JCT dataspace



Access to JCT entry in dataspace

JES3 initialization also creates a JCT dataspace page release table which JES3 uses to track the number of allocated JCTs in each JCT data space page. JES3 releases pages when they no longer contain allocated JCTs.

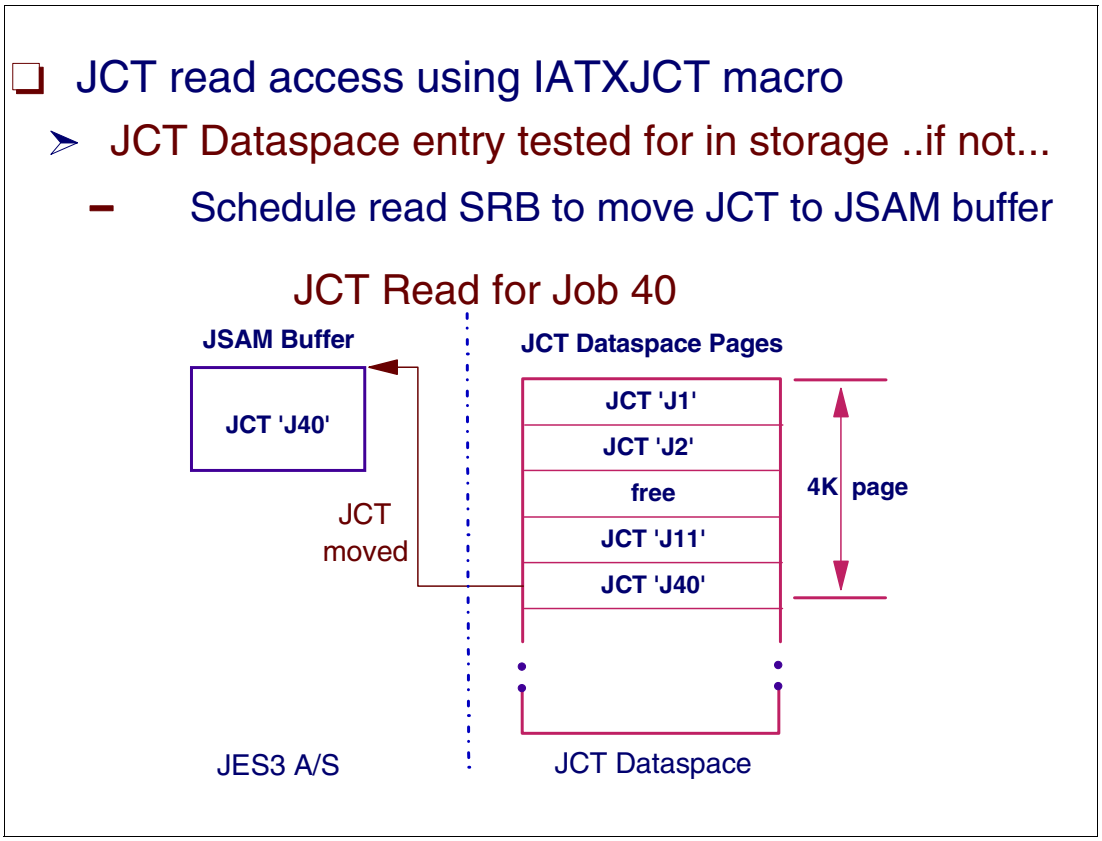
See *z/OS JES3 Initialization and Tuning Reference*, SA22-7550 for information about how to define the size of the JCT and how to tune JCT access.

JCT dataspace page release table

A page release table is maintained for the JCT dataspace to allow for a page to be released when it no longer contains any JCT entries. This avoids unnecessary I/O to the page data sets.

A JCT entry may span a page boundary. A test for a JCT entry to determine if it is still in storage and not paged out tests the first and last byte of the JCT entry since an entry may span a page boundary.

2.7 IATXJCT - JCT access



JCT access

The IATXJCT macro service adds, alters, examines, deletes, or releases a JCT by invoking the JCT access method (IATGRJX). The JCT access method accesses the JCT in the JCT data space whenever the data space is enabled. JES3 automatically enables the JCT data space during JES3 initialization on the global. If the JCT data space is not enabled, the access method accesses the JCT from the JES3JCT data set.

A move of the JCT record from the dataspace into a JSAM buffer is done by the JSAM routines.

The JESREAD macro cannot be used to read a JCT record, which is also a single record file. For a read request, if the JCT dataspace is enabled, a JSAM buffer is obtained and the dataspace addresses of the JCT entry are checked to determine if the JCT entry is in real storage:

- If the JCT entry is in real storage, it is moved directly from the data space to the JSAM buffer.
- If the JCT entry is not in real storage, the JCT read SRB is scheduled to move the JCT entry to the JSAM buffer. The SRB routine isolates JES3 nucleus task from page faults when accessing the JCT entry.

A job's JCT entry address in the dataspace is computed from the job's JQE address:

```
(((JEQ_ADDR - JQE4_BASE_ADDR/JQESIZE)*JCTSIZE)+DATASPACE_ORIGIN)
```

2.8 IATXJCT macro

- **IATXJCT** macro for JQE/JCT access method
  - **TYPE = CHKPNT** - update JCT on spool and keep control
- Access Example:
  - **IATXJCT TYPE = RW** (read JCT and change)
  - **IATXJCT TYPE = CHKPNT** (update and keep in JSAM buffer)
- Keyword **SOURCE = DSPACE|JCTDS**

```
LA     R1,JCTBLD                    ADDRESS OF JCT BUILD AREA
IATXJCT  TYPE=ADD,JCT=(R1),ERROR=CALL035     ADD JCT
```

JCT access macro - IATXJCT

The IATXJCT macro adds, alters, examines, deletes, or releases a JCT.

The are various types of JCT access as follows:

RW Access an existing JCT to allow the JCT to be read or written to (read/write access). On return, register 1 contains the address of the JCT, register 0 contains the address of the JQE.

CHKPNT Write the JCT to the JCT data set but do not relinquish control of the JCT. The buffer containing the JCT will not be freed, and the FCT will still be enqueued on the JCT. Previous read/write access must have been obtained to invoke this checkpoint function. JES3 does not update the JCT data space for TYPE=CHKPNT requests until after it writes the JCT to the JCT data set.

SOURCE DSPACE specifies that the JCT should be obtained from the JCT data space. If the JCT data space is not enabled, JES3 obtains the JCT from the JCT data set.

JCTDS specifies that the JCT should be obtained from the JCT data set. This is the default if the JCT data space is not enabled. SOURCE=JCTDS should only be used for doing problem analysis when

ADD Add a new JCT to the JCT data set. On return, register 0 contains the address of the JQE.

2.9 JCT utility

- JCT utility - Moves JES3 JCT data set from one volume to another volume without JES3 cold start
  - New JCT data set can be larger or smaller than original
  - The new JCT data set may reside on a volume of different device type than the original JCT data set
  - New primary JES3 checkpoint data set is created
  - Existing JES3 data sets (JCT or checkpoint) are not altered - If utility fails, recovery is via a JES3 hot start
  - Use of the new JCT data set and checkpoint data set requires a warm start of JES3 rather than a cold start

JCT utility IATUTJCT

For various reasons you might need to replace an existing JCT data set. In order to allow a JCT data set to be copied without a cold start, JES3 provides a program called the JCT utility. The utility highlights include:

- The new JCT data set may be larger or smaller than original JCT data set. IATUTJCT runs as a started task under the MSTR subsystem and copies JCT entries from one JCT data set to another while updating the device dependent information at the same time. IATUTJCT also copies the JCT information that is kept in the checkpoint data set.
- The new JCT data set may reside on a volume of different device type than the original JCT data set. Because the contents of the JCT data set are dependent on the device type and the location of the JCT data set on its volume, an existing JCT data set cannot be copied by a utility such as IEBGENER. Furthermore, there is information about the JCT data set in the checkpoint data set which cannot be copied.
- A new primary JES3 checkpoint data set is created to reflect the location and name of the new JCT data se and may reside on the volume of choice. IATUTJCT copies JCT entries from one JCT data set to another, and updates the device-dependent information at the same time. IATUTJCT also copies the JCT information that is kept in the checkpoint data set.
- The existing JES3 data sets (JCT, spool or checkpoint) are not altered. Should the utility fail, recovery is via a JES3 hot start.
- Use of the new JCT data set and checkpoint data set requires a warm start of JES3 rather than a cold start.

2.10 IATUTJCT utility

- Requires a procedure in SYS1.PROCLIB
  - IATUTJCT is started using SUB=MSTR
- Define procedure to:
  - ICHRIN03 or STARTED class
  - Procedure userid must have access to JCT data set
- IATUTJCT must be in APF authorized library
  - Or in link list with LNKAUTH=LNKLS
- Sample procedure in SYS1.SIATSAMP - (IATUTJCS)

IATUTJCT utility

You run IATUTJCT by invoking a started task that runs the IATUTJCT program. Such a JCL procedure must reside in SYS1.PROCLIB. Member IATUTJCS of the z/OS SIATSAMP data set also provides a sample JCL procedure for IATUTJCT. IATUTJCT must be started using SUB=MSTR.

Define the procedure to the security product:

- Define either a profile for the started task in the STARTED security class or an entry in the started task table (ICHRIN03).
- Provide the userid under which the procedure is to run with at least read access to the old JCT and checkpoint data sets and update access to the new JCT and primary checkpoint data set.
- Place the load library containing IATUTJCT and IATGRCK in the LNKLST concatenation or ensure they are in STEPLIB. JES3 must find these load modules in a normal library search. Ensure that the library containing the IATUTJCT and IATGRCK modules is APF authorized.

IATUTJCT procedure

You run IATUTJCT by invoking a started task that runs the IATUTJCT program. Below is a sample JCL procedure you might use to run IATUTJCT. Such a JCL procedure must be in SYS1.PROCLIB. Member IATUTJCS of the SIATSAMP data set also provides a sample JCL procedure for IATUTJCT. IATUTJCT must be started using SUB=MSTR.

2.11 Using IATUTJCT

- Copy a JCT data set without migration - test mode
  - S procname,SUB=MSTR
  - No changes to CHKPNT, CHKPNT2, JES3JCT
  - Why test:
    - Make sure data sets allocated correctly
    - Check authority to data sets
    - Make sure JCT data set size is correct
- Copy a JCT data set with migration
  - S procname,SUB=MSTR,P=MIGRATE
- JES3 start options with new JCT data set
  - Hot start - Hot start with refresh - Warm start

Using IATUTJCT

The IATUTJCT functions are to:

- Utilize IATUTJCT to migrate the individual JCT entries and the same time creating a new JCT data set and checkpoint data sets.
- You need to allocate a larger JCT data set in order to provide capacity for more jobs in the JES3 job queue.
- You need to move your JCT data set to a new volume. The new volume can be the same device type as the one where your JCT data set currently resides, or it can be a different one.

You can run the IATUTJCT utility as a test while JES3 is up and running. Do not change any definitions in the JES3 startup procedure or CIFSS procedures. The test options are:

- If testing the migration from a lower version to a higher one, use the P=MIGRATE parameter.
- You need to test IATUTJCT only once in a JES3 complex, and you can test it on the global or any local.
- You do not need to specify SUB=MSTR parameter when testing IATUTJCT with JES3 up and running.

Starting JES3

If IATUTJCT fails, or if you decide to not switch to the new data sets, you must restart JES3 with the old data sets. You can perform a hot start, hot start with refresh, or IPL with a warm start. If you have changed the initialization stream (the JES3JCT DYNALLOC statement to

point to the new JCT data set) and you perform a hot start with refresh or a warm start, do not use the initialization stream member containing the change.

When using IATUTJCT to move the JCT data set, you can pick up the new data set with any type of JES3 start; however, if you perform a hot start with refresh or a warm start, and you use the DYNALLOC statement to define the JES3JCT DD name, you must change the DYNALLOC statement to point to the new JCT data set.

Note: See “JES3 start types” on page 206 for information of the JES3 starts.

2.12 IATUTJCT JCL DD statements

- IATUTJCT requires following DD statements :
  - JES3JCT - Old JCT data set
  - NJES3JCT - New JCT data set
  - CHKPNT - Old primary checkpoint data set
  - NCHKPNT - New primary checkpoint data set
  - CHKPNT2 - Allocate a new empty data set

Running the IATUTJCT utility

In the IATUTJCTJCL, do the following:

- Ensure that the JES3JCT, NJES3JCT, CHKPNT, and NCHKPNT DD statements are specified. The CHKPNT2 DD statement is optional and should be omitted if you are not using an alternate checkpoint data set or if you will not be using one after changing JES3 to use the new data sets.
- Ensure that proper CHKPNT and CHKPNT2 DD statements are specified. There is no NCHKPNT2 DD statement. If you are using an alternate checkpoint data set, allocate a new one and leave it empty. When you restart JES3 to pick up the new primary and alternate checkpoint data sets, JES3 will automatically synchronize the new alternate checkpoint data set to the new primary checkpoint data set. Do not use the old alternate checkpoint data set as the new alternate checkpoint data set.

2.13 Running IATUTJCT utility considerations

- JES3 global and locals and all CIFSS address spaces must be stopped
  - They use CHKPNT, CHKPNT2, and JES3JCT
  - S iatutjct_proc,SUB=MSTR,P=MIGRATE
- Modify JES3 and CIFSS JCL procedudes
  - Use symbolic parameters in JES3 procedure

```
//JES3   PROC CK=CHKPNT,CK2=CHKPNT2,JCT=JES3JCT,P=
//JES3   EXEC PGM=IATINTK,DPRTY=(15,15),TIME=1440,
//           REGION=0M,PARM=&P
 ...
//CHKPNT    DD    DISP=SHR,DSN=SYS1.&CK
//CHKPNT2   DD    DISP=SHR,DSN=SYS1.&CK2
//JES3JCT   DD    DISP=SHR,DSN=SYS1.&JES3JCT
 ...
```

S JES3,CK=NEWCHK,CK2=NEWCHK2,JCT=NEW3JCT,P=NOREQ

IATUTJCT utility considerations

When you are ready to copy the checkpoint and JCT data sets to the new ones, stop all CIFSS address spaces and bring JES3 down on the global processor and all local processors before you run the IATUTJCT utility. If you have any other started tasks or jobs that reference the JCT and/or checkpoint data sets, such as user-written spool browsers, bring them down also.

JES3 and all converter/interpreter FSS (CIFSS) startup procedures must be updated to point to the new checkpoint data sets. Note that you cannot use symbolic parameters in the CIFSS startup procedures. You must change the DD statements in the CIFSS procedures.

In the JES3 procedure you can use symbolic parameters to specify the checkpoint and JCT data set names. Note, if you use the DYNALLOC statement to define the JES3JCT DD name, you must create a new initialization stream member that points to the new JCT data set.

Considerations if cold starting

You can choose to cold start a HJS6606 release of JES3 instead of run IATUTJCT. However, if you plan to save and restore jobs using the Dump Job dynamic support program (DJ DSP), do not use the TRANS=YES parameter. Also, if you cold start to fall back to the lower JES3 release you will need to run the Dump Job DSP to restore jobs without using the TRANS=YES parameter. The dump job DSP tolerates the different versions of the JCT record provided that the JES3 release is at least HJS6606, or at least HJS4421 (with the migration PTF installed).

IATUTJCT does not allow the JES3 installation to migrate "down" from a higher JCT version to a lower one; however, if IATUTJCT was previously run with P=MIGRATE and the lower JES3 release has the migration PTF for APAR OW30849 installed, a fallback to that "lower" release is possible using the higher version of the JCT data sets. If the migration PTF is not installed on the lower release, a cold start is required if you need to fall back to the lower release.

JCT expansion

The actual migrating of the JCT entry and creation of a new JCT data set is accomplished using a utility called IATUTJCT. This utility currently allows an installation to move/copy the JCT data set without a cold start to a device of different geometry and/or data set of different size. The utility will be incorporated and standardized into the JES3 product and modified to allow the migration of JCT entries.

JCT record access

Fields JCTFSIZE and JCTFEND should no longer be used to address the variable (scheduler element) portion of the JCT, instead, use JCTFIXL.

2.14 JES3 Checkpoint data set

- At least one of the checkpoint data sets must be used for JES3
- One or both JES3 checkpoint data sets defined in the JES3 procedure
  - The ddnames are CHKPNT and CHKPNT2
- Either checkpoint data can be added or replaced over a JES3 restart
- Size and placement of the checkpoint data sets
  - For size considerations - see *JES3 Initialization and Tuning Guide*
  - If either checkpoint data set runs out of space, it must be replaced
  - Allocate the checkpoint data sets on
    - Different volumes
    - Different channel paths
    - Different control units
- IATXCKPT checkpoint access macro
  - Interfaces to the checkpoint record access method functions

Allocating the JES3 checkpoint data set(s)

The JES3 checkpoint data set(s) let(s) you warm start or hot start the JES3 system with little or no loss of system information. One or both JES3 checkpoint data sets must be defined in the JES3 start procedure using ddnames of CHKPNT, CHKPNT2, or both. If your installation defines any C/I FSS address spaces, define the JES3 checkpoint data set(s) in the C/I FSS start procedure exactly as the JES3 start procedure defines them, except substitute DISP=SHR for DISP=OLD.

You cannot dynamically allocate the checkpoint data sets. You must allocate and catalog one or both checkpoint data sets before JES3 operation. At least one of the checkpoint data sets must be available to JES3 during initialization processing. You can add or replace either checkpoint data set over a JES3 restart of any kind with no effect on JES3 processing. Both checkpoint data sets contain identical information; to ensure against loss of checkpointed data, allocate both data sets.

Determining the size and placement of the checkpoint data sets

Each checkpoint data set must be allocated as a single extent which begin and ends on a cylinder boundary. The size of each checkpoint data set should be at least two cylinders on a direct access storage device. To determine how much space your installation's checkpoint data sets require consider the following factors:

- Each checkpoint record type begins on a track boundary. Each track contains a 128-byte track header record.

- The checkpoint data set track map, the complex status record, the initialization checkpoint record, and the JESCKPNT checkpoint record each need one track.
- The dynamic allocation checkpoint record requires 44 bytes plus an additional 92 bytes for each DYNALLOC initialization statement.
- The spool volumes checkpoint record requires 64 bytes plus an additional 80 bytes for each TRACK and FORMAT initialization statement. Add additional bytes as reserve for spool expansion.
- The spool partition checkpoint record requires 64 bytes plus an additional 96 bytes for each SPART initialization statement.
- The partition TAT checkpoint record space requirement is calculated using a complex algorithm involving many different factors. Allow about 512 bytes for each spool data set.
- The BADTRACK checkpoint record requires 44 bytes plus an additional 6 bytes for each BADTRACK initialization statement. Every entry in the BADTRACK checkpoint record requires an additional 64 bytes. Thus, the size of this data area varies with the number of tracks having I/O errors at one time.
- Allocate enough free space to permit growth in the complex without reallocating the checkpoint data sets. Checkpoint data sets are limited to 65,535 tracks.

If either checkpoint data set runs out of space, the data set must be replaced. Recalculate the amount of space the checkpoint data set needs and allocate a new checkpoint data set that is larger than the old one.

To minimize the effect of the loss of any one DASD volume, control unit, or channel path, allocate the checkpoint data sets on different volumes, channel paths, and control units. The volumes may be different device types.

Recovering from permanent errors on a checkpoint data set

If a permanent I/O error occurs on one of the JES3 checkpoint data sets, recovery actions depend on whether the installation allocated one or both checkpoint data sets.

If the only checkpoint data set develops a permanent I/O error:

1. Perform a **cold start.**

If one of the two checkpoint data sets develops the error:

1. Enter a DISPLAY command for the data set that is **not** in error. In the display, look for messages indicating problems in the data set.
2. If the display shows problems, perform a **cold start**.
3. If the display shows no problems, perform a hot or warm start to replace the data set with the error.

   In this start, JES3 copies the checkpoint records it finds on the old checkpoint data set onto the new checkpoint data set. When the hot or warm start is finished, JES3 has two complete checkpoint data sets once again.

Recovering when a checkpoint data set is out of space

If either checkpoint data set runs out of space, replace the data set. Recalculate the amount of space the checkpoint data set needs. Then allocate a new checkpoint data set that is larger than the old one.

IATXCKPT macro

The IATXCKPT macro requests checkpoint record access method functions.

2.15 JES3 spool data sets

- Allocating spool data sets
  - On any DASD device
    - Include a DD statement in the JES3 procedure
    - Omit the DD statement and include a DYNALLOC initialization statement
  - No secondary extents
  - Can be greater than 65,535 tracks
    - DSNTYPE=LARGE on the JCL DD statement
- No more than one spool data set per volume
- JES3 spool data sets
  - Location-dependent and unmovable

Allocating JES3 spool data sets

To allocate a spool data set, include a DD statement for the data set in the JES3 start procedure. To dynamically allocate the spool data set, omit the DD statement and include a DYNALLOC statement for the data set in the JES3 initialization stream. Dynamic allocation provides an easier method for changing your spool configuration than allocating the spool data sets through the JES3 start procedure with DD statements.

Several factors determine the number and size of spool data sets that you should allocate:

- The amount of spool space that all jobs in the complex may need at any one time. Allocate enough spool space to handle peak usage.
- The number of processors in the complex competing for available control units and devices. The more processors competing for control units and devices, the more spool data sets needed for reasonable performance. A spool data set should not be busy more than 30 to 40 percent of the time or there will be too much contention for the data set.
- The type of work being carried out in the complex. Jobs that include large amounts of output handled by JES3 need more spool space than jobs with little output, such as TSO jobs.

DASD volume types

You can allocate a spool data set on any direct access storage device (DASD). The volume on which a spool data set is allocated must be accessible to the global processor and to all local processors. Each spool data set must be contained in a single extent. (A single extent is one adjoining group of tracks or cylinders.) You cannot allocate any secondary extents. A

spool data set can be greater than 65,535 tracks only if you specify the DSNTYPE=LARGE keyword on the JCL DD statement that is used to create the data set.

To avoid degradation of JES3 performance and possible lockouts, do not allocate more than one spool data set per volume. When a volume contains more than one spool data set, the average seek time to access the data increases. Similarly, do not allocate data sets to a volume that JES3 rarely accesses.

Note: This suggestion does not apply to parallel access devices such as ESS. With the parallel access devices, a spool data set is not allocated on a single physical device but on a logical device located anywhere in the storage cluster.

Spool data set placement

JES3 spool data sets are location-dependent and unmovable. Utility programs that defragment should not be allowed to move these data sets. Spool data sets can be marked unmovable by coding DSORG=PSU on the DCB parameter of the DD statement.

2.16 Formatting spool data sets

- Spool data set formatting
  - During JES3 Initialization
    - FORMAT initialization statement
  - IEBDG
    - TRACK initialization statement
  - Reformatting
    - Only when spool data set is damaged
    - When BUFSIZE= on the BUFFER statement is changed

```
//JOB1     JOB   .....
//FORMAT    EXEC  PGM=IEBDG
//SPXTNT    DD    DSN=spool_data_set_name,DISP=OLD,
//              DCB=(RECFM=U,BLKSIZE=nnn)
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD  *
DSD       OUTPUT=(SPXTNT)
FD        NAME=SPOOL,FILL=X'FF',LENGTH=nnn
CREATE    NAME=(SPOOL),QUANTITY=2000000000
END
```

Formatting spool data sets

Before JES3 can use a spool data set, you must format the spool data set. Two ways to do this are:

- Format it during JES3 initialization by including a FORMAT statement in the JES3 initialization stream.
- Format it by executing the utility program IEBDG as a batch job. This fills the extent with hexadecimal "FF" data.

If you format a spool data set during JES3 initialization, JES3 can use the spool data set after initialization completes. If you use IEBDG to format a spool data set, you must then do a warm start or cold start so JES3 can use the data set.

Formatting during JES3 initialization

To format a spool data set during JES3 initialization, include a FORMAT statement for the spool data set in the JES3 initialization stream. Then start JES3 using that initialization stream.

The type of start you use depends on why you are formatting the spool data set:

- If you have changed the BUFSIZE= parameter on the BUFFER statement, use a cold start (C). (In this case, you must format all spool data sets.)
- If you are replacing a spool data set, use a warm start to replace a spool data set (WR). If you also want an analysis of the jobs in the job queue, use a warm start with analysis to replace a spool data set (WAR).

- If you are adding a spool data set, use a warm start (W) or a warm start with analysis (WA).

After JES3 processes the initialization stream, replace the FORMAT statement with a TRACK statement. If the FORMAT statement contained the STT or STTL parameter, also code this parameter on the TRACK statement.

If you use a warm start and the initialization stream contains a FORMAT statement for a spool data set that is already formatted, JES3 issues a warning message. JES3 continues with initialization, however, and does not reformat the spool data set.

Formatting with IEBDG

You can use the utility program IEBDG to format a data set that you plan to use as a spool data set. z/OS DFSMSdfp Utilities explains how to use IEBDG. An example:

```
//JOB1      JOB   .....
//FORMAT    EXEC  PGM=IEBDG
//SPXTNT    DD    DSN=spool_data_set_name,DISP=OLD,
//                UNIT=SYSDA,VOL=SER=serial_number,
//                DCB=(RECFM=U,BLKSIZE=nnn)
//SYSPRINT  DD    SYSOUT=A
//SYSIN     DD  *
DSD        OUTPUT=(SPXTNT)
FD         NAME=SPOOL,FILL=X'FF',LENGTH=nnn
CREATE     NAME=(SPOOL),QUANTITY=2000000000
END
/*
```

The value of the variable *nnn* must equal the value of the BUFSIZE= parameter on the BUFFER initialization statement. The variable *nnn* appears on both the SPXTNT DD statement and on the FD utility program control statement.

The value of the QUANTITY=2000000000 on the CREATE statement should be specified to ensure that IEBDG formats the entire data set.

If IEBDG successfully formats the entire spool data set, the formatting job ends with an abend code of D37. In addition, MVS issues message IEC031I. Ignore the corrective action specified in the message.

After the spool data set has been formatted, include a TRACK statement for it in the initialization stream. To make the spool data set available to JES3, restart JES3. The type of restart you use depends on why you are formatting the spool data set:

- If you have changed the BUFSIZE= parameter on the BUFFER statement, use a cold start (C). In this case, you must format all spool data sets.
- If you are replacing a spool data set, use a warm start to replace a spool data set (WR) or a warm start with analysis to replace a spool data set (WAR).
- If you are adding a spool data set, use a warm start (W) or a warm start with analysis (WA).

Reformatting a spool data set

When formatted, reformat a data set only when:

- The spool data set has been damaged
- You change the BUFSIZE= parameter on the BUFFER initialization statement (in this case, you must reformat all spool data sets)

To reformat a spool data set, use either of the procedures just described.

3

JES3 job flow and scheduling

JES3 job management consists of the following phases for jobs that go to MVS execution:

- Input service
- Converter/interpreter service
- Resource allocation
- Job selection and scheduling
- Output service
- Purge

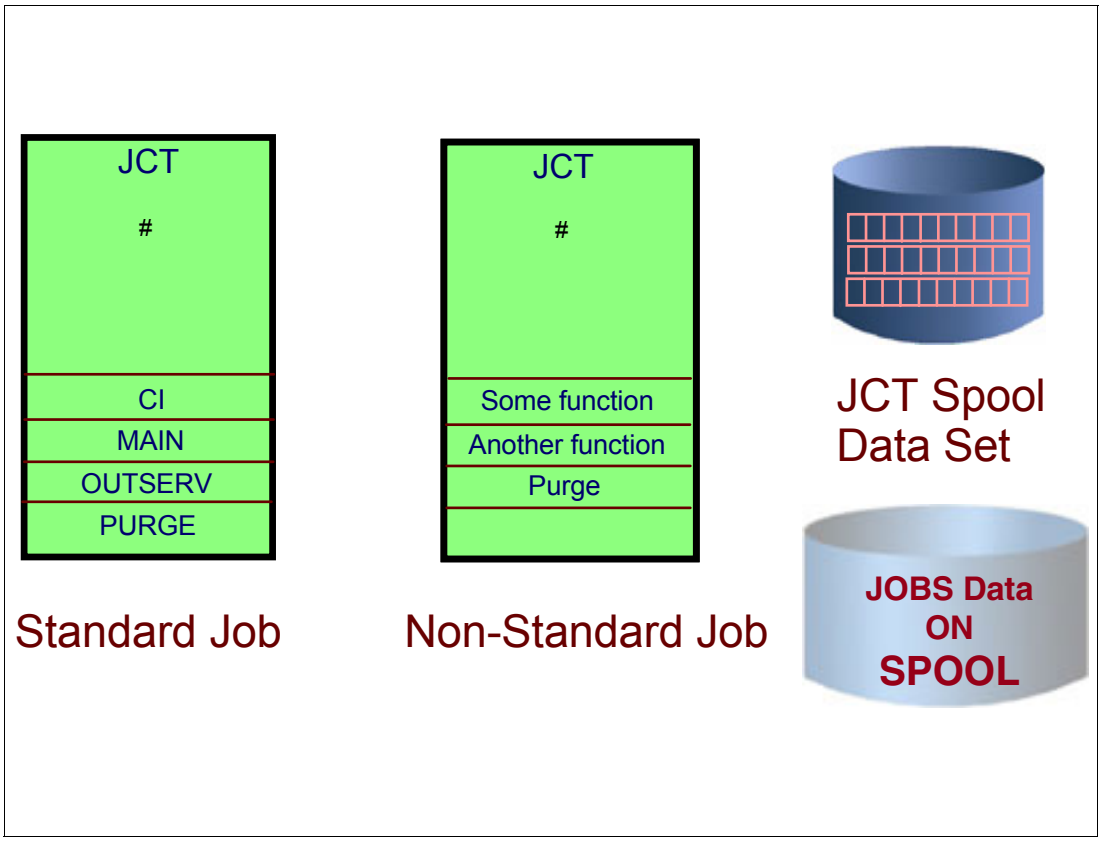
Input service, the first phase of JES3 job management, reads MVS jobs and places each job into a queue for subsequent processing by other phases. These types of jobs include MVS batch jobs, started tasks, and TSO logons.

Another group of JES3 jobs consists of callable DSPs. These jobs include operator called DSPs, JES3 called DSPs.

This section describes the following:

- Types of JES3 jobs.
- Sources of JES3 jobs.
- JES3 job flow.
- The job segment scheduler (JSS)

3.1 JES3 jobs



Types of JES3 jobs

A JES3 job represents unit of work that JES3 is to process. MVS jobs (jobs defined with JCL) can come from a card reader, a tape unit, a disk reader or from a remote workstation. The reader phase treats MVS jobs from a remote workstation as though the job came from a card reader.

When a job enters a JES3 complex, it is a assigned a job number that distinguishes it from other jobs. The two types of MVS jobs are standard jobs and non-standard jobs.

Standard job

The standard job - If a user job contains no //*PROCESS control statements JES3 defines the standard sequence of scheduler elements (SEs). Scheduler elements make up the part of the job control table (JCT) entry that represents one or more dynamic support programs (DSPs) needed for processing of jobs by JES3.

Non-standard job

If a MVS job contains one or more //*PROCESS statements, JES3 defines the job as requiring the sequence of scheduler elements named on the //*PROCESS statements. The job has been created for a callable DSP.

JES3 JCT and spool data sets

All jobs are registered on the JES3 JCT spool data set. The jobs' SYSIN and SYSOUT data sets and all job related control blocks reside on the JES3 spool data sets.

3.2 Scheduler elements

- Basic unit of work for JES3 job management
  - Scheduled synchronously - one at a time
- Scheduled by Job Segment Scheduler - JSS
- Job - Any combination of scheduler elements
  - JES3 produces a standard set for MVS jobs
  - User may define via a JES3 control statement
    - //*PROCESS
  - JES3 produces SEs for internally created jobs
    - Called DSPs
  - Installation standard set using user exit
    - User exit 17

Scheduler elements

All jobs that are read into JES3 are placed in the JES3 job queue, which resides on a direct-access storage device (DASD). Once in the job queue, the job is subdivided into two or more processing segments called scheduler elements. A job submitted in the normal way has scheduler elements (SEs) consisting of a standard set representing the converter/interpreter (C/I) for input processing, main service (MAIN) processing, output service (OUTSERV) for processing the job's output, and PURGE for purging a job from the system.

Job segment scheduler (JSS)

JES3 does not schedule jobs in the MVS sense of scheduling jobs; it schedules the work that must be done on behalf of MVS jobs and callable DSPs by using scheduler elements. The job segment scheduler (JSS) selects scheduler elements (perhaps from several jobs) that are ready for processing, and builds entries in the FCT table so that DSPs can be dispatched to the work.

//*PROCESS statements

This statement can be used to control how JES3 processes a job. A job that contains //*PROCESS statements receives only the JES3 processing specified on the //*PROCESS statements plus certain required processing.

Note: The //*PROCESS statement consists of the characters //* in columns 1 through 3, PROCESS in columns 4 through 10, a blank in column 11, and the DSP name beginning in column 12. The rest of the columns must be blank.

JES3 created scheduler elements

JES3 offers utility programs, or utilities, for handling the routine work related to running JES3. These are internal jobs created by JES3 to do special processing on behalf of many JES3 functions.

Called DSPS

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

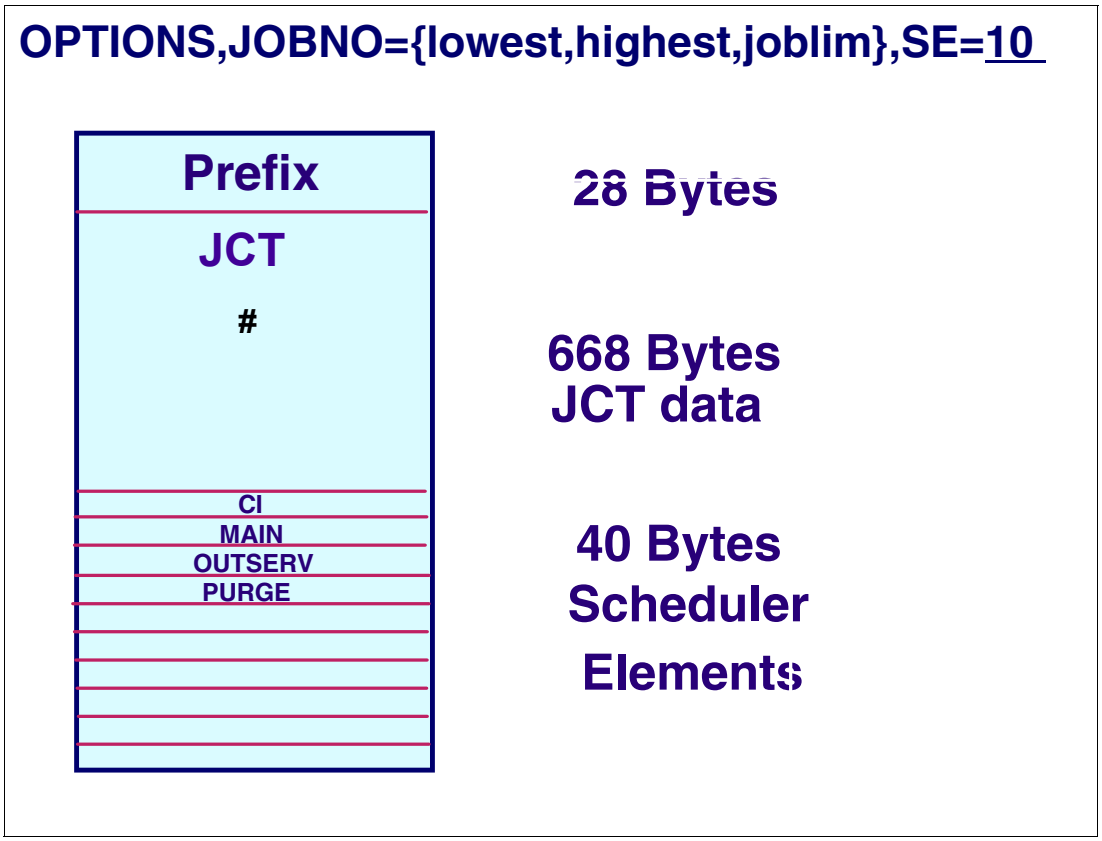
Many DSPs are called by the operator to do specific functions.Many are called utility DSPs.

IATUX17 user exit

In summary, each job is composed of the number of SEs required perform its JES3 work. There may be as few as two scheduler elements; JES3 provides four as default for MVS jobs.

This exit, entered at the beginning of input service processing during job statement processing (module IATISJB), permits you to create the initial set of scheduler elements for each job. If this installation exit is not taken, the standard list of scheduler elements--converter/interpreter, main service, output service, and purge--is used. On completion, you must have defined the required set of scheduler elements.

3.3 JES3 JCT and job number range



Job number range

The number of jobs concurrently allowed in a JES3 complex depends on the size of the JCT data set and JOBNO= keyword parameters specified on the OPTIONS statement in the JES3 initialization. The JOBNO= keyword has the following format.

```
OPTIONS,JOBNO=(low_job_number,high_job_number,max_number_jobs)
```

The job number range is specified by *low_job_number* and *high_job_number* values. The number of concurrent jobs allowed in the JES3 complex is the minimum of the following values:

- Number of records in the JCT data set
- Maximum number of jobs from the JOBNO= keyword
- Difference between low and high job numbers from the JOBNO= keyword

The SE= keyword parameters specified on the OPTIONS statement specifies the maximum number of scheduler elements (SEs) that may be constructed for any job. The value chosen should equal the number of scheduler elements required to support the largest job (in terms of SEs) to be run in the complex. A number between 10 and 90 may be specified. This specification is used to calculate the maximum size for a JCT and its SEs (JCT record size).

JES3 JCT

The JES3JCT data set contains a record for every job in the JES3 complex.The job segment scheduler accesses the JCT when scheduling jobs for various phases of JES3 processing.

Every job containing JCL, but containing no special JES3 control statements, is considered a standard job. JES3 places into the JCT entries of all standard jobs the same four scheduler elements in the same order:

- Converter/interpreter (CI)
- Main service (MAIN)
- Output service (OUTSERV)
- Purge (PURGE)

Scheduler elements

Each JCT entry contains information about the job and an entry for each SE. Each SE entry contains flags indicating that SE's status:

- Inactive
- Active
- Complete

JSS uses these flags to determine which SE is next for ending function/scheduling function as follows:

- The first SE that is not marked complete is selected, and is marked either complete or active.or either ending function or scheduling.
- The JCT also serves as a continually-updated checkpoint of the status of the job as it progresses through the system.

JCT data

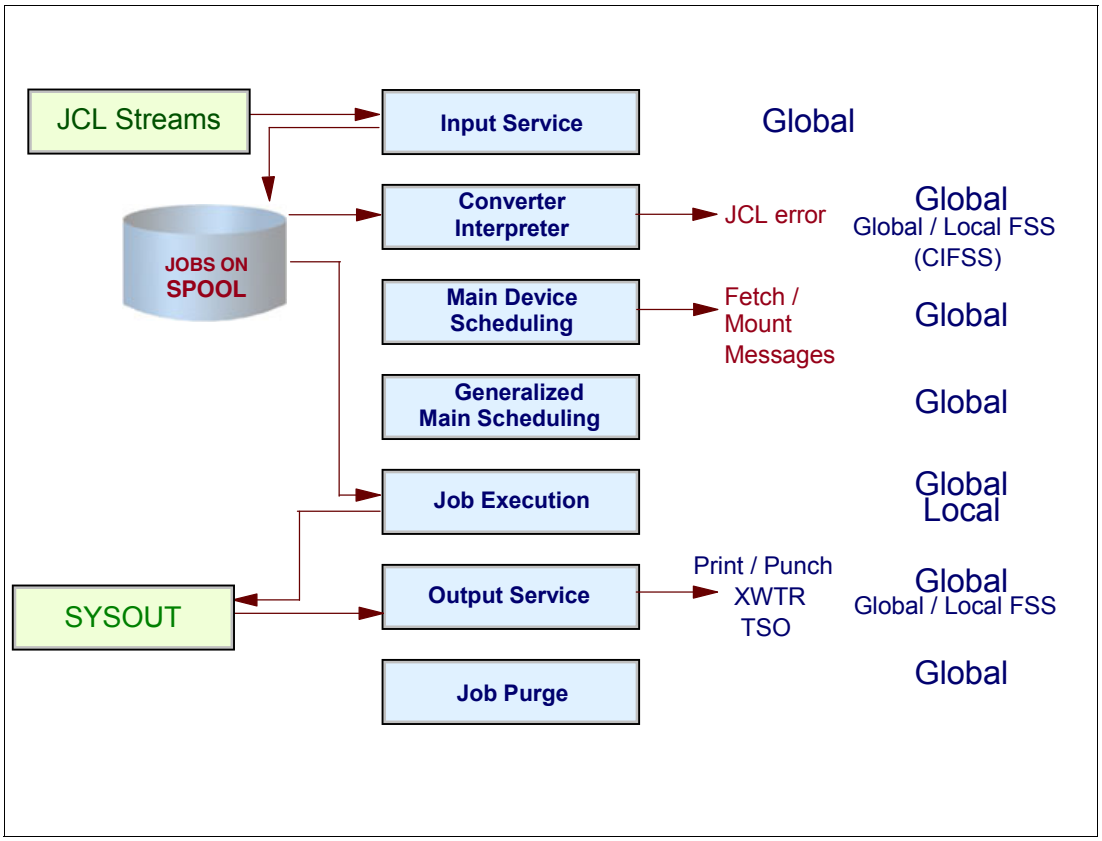
A job exists in the JES3 complex when its JCT entry is created and ceases to exist when the JCT entry is removed.

The entry data includes:

- The job name
- The job number
- The job status
- A number of SEs (one for each piece of work to be performed on behalf of the job)
- Spool addresses of other job-related control blocks

The JES3 user exit IATUX17 (Define Set of Scheduler Elements), entered at the beginning of input service processing during job statement processing (module IATISJB), permits you to create the initial set of scheduler elements for each job. If this installation exit is not taken, the standard list of scheduler elements --converter/interpreter, main service, output service, and purge-- is used.

3.4 JES3 job flow



JES3 Job Flow

The term “job” has been traditionally defined as unit of work signified by the presence of a JOB JCL statement in the input stream. Within each MVS job, the control statements are grouped into job steps. A job step consists of all the control statements needed to run one program. If a job needs to run more than one program, the job would contain a different job step for each of those programs.

JCL streams

In general, statements in the MVS job control language (JCL) and in the job entry control language (JECL) for JES2 and JES3 subsystems, are used by the users to define a job’s execution steps and resource requirements. The MVS system requires an internal format a job's JCL statements before it can process it. MVS converter/interpreter transforms the external (JCL) format job description into the internal format called the scheduler work area (SWA). Every MVS job must contain a minimum of the following two types of control statements:

- A JOB statement, to mark the beginning of a job and assign a name to the job. The JOB statement is also used to provide certain administrative information, including security, accounting, and identification information. Every job has one and only one JOB statement.
- An EXEC (execute) statement, to mark the beginning of a job step, to assign a name to the step, and to identify the program or procedure to be executed in the step. You can add various parameters to the EXEC statement to customize the way the program executes. Every job has at least one EXEC statement.
- One or more DD (data definition) statements, to identify and describe the input and output data to be used in the step. The DD statement may be used to request an existing data

set, to define a new data set, to define a temporary data set, or to define and specify the characteristics of the output.

Input service

Whatever the source of the input, JES3 is signalled that an input stream is to be read. This begins a chain of events that includes:

- Creation and scheduling of a card reader DSP - a JES3 job
- Reading of the input stream by a DSP
- Building of the JES3 control blocks describing the MVS job to JES3. JCT entries for each job in the input stream are added to the JES3JCT data set.
- Execution of DSPs represented by scheduler elements in the JCT entries for each job

The DSPs that provide the JES3 input service control the processing of a typical MVS job at the beginning. Input service routines create scheduler elements that represent a job's flow through the various JES3 processing phases. Input service, active on the global processor, accepts and queues all jobs entering the JES3 complex. The global processor accepts MVS jobs into the system from:

- A TSO SUBMIT command
- A local card reader (CR DSP)
- A local tape reader (TR DSP)
- A disk reader (DR DSP)
- A remote work station (RJP/SNARJP DSPs)
- Another node in a job entry network (NJE DSPs)
- The internal reader (INTRDR DSP)

Once an MVS job is registered into the JES3, the JES3 job segment scheduler DSP (JSS) will be responsible for the flow of the job through the JES3 processing phases.

Converter interpreter processing

The converter/interpreter (C/I) is the first scheduler element for every standard job. After a job passes through this phase of processing, JES3 knows what resources the job will require during execution. C/I routines provide input to the main device scheduling (MDS) routines by determining jobs' devices, volumes, and data sets requirements. The C/I routines, that execute after the MVS converter/interpreter invocation, extract from the jobs' scheduler work area (SWA) the jobs' resource requirements and create control blocks for MDS. Jobs with JCL errors are flushed. Main device scheduling provides for the effective use of system resources.

Main device scheduling (MDS)

JES3 MDS, commonly referred to as "setup", ensures the operative use of non-sharable mountable volumes, eliminates operator intervention during job execution, and performs JES3 data set serialization. It oversees specific types of pre-execution job setup and generally prepares all necessary resources required to execute the job. The resource tables and allocation algorithms to satisfy a job's requirements through the MDS allocation of volumes and devices are established at JES3 initialization.

Generalized main scheduling (GMS)

JES3 generalized main scheduling (GMS) is the group of routines that govern where and when MVS execution of a job occurs. Job scheduling controls the order and execution of jobs running within the JES3 complex.

Job execution

Job execution is under the control of JES3 main service which selects jobs to be processed by MVS initiators. Main service selects a job for execution using the job selection algorithms established at JES3 initialization. MAINPROC, SELECT, CLASS, and GROUP initialization statements control the key variables in the job scheduling and job execution process.

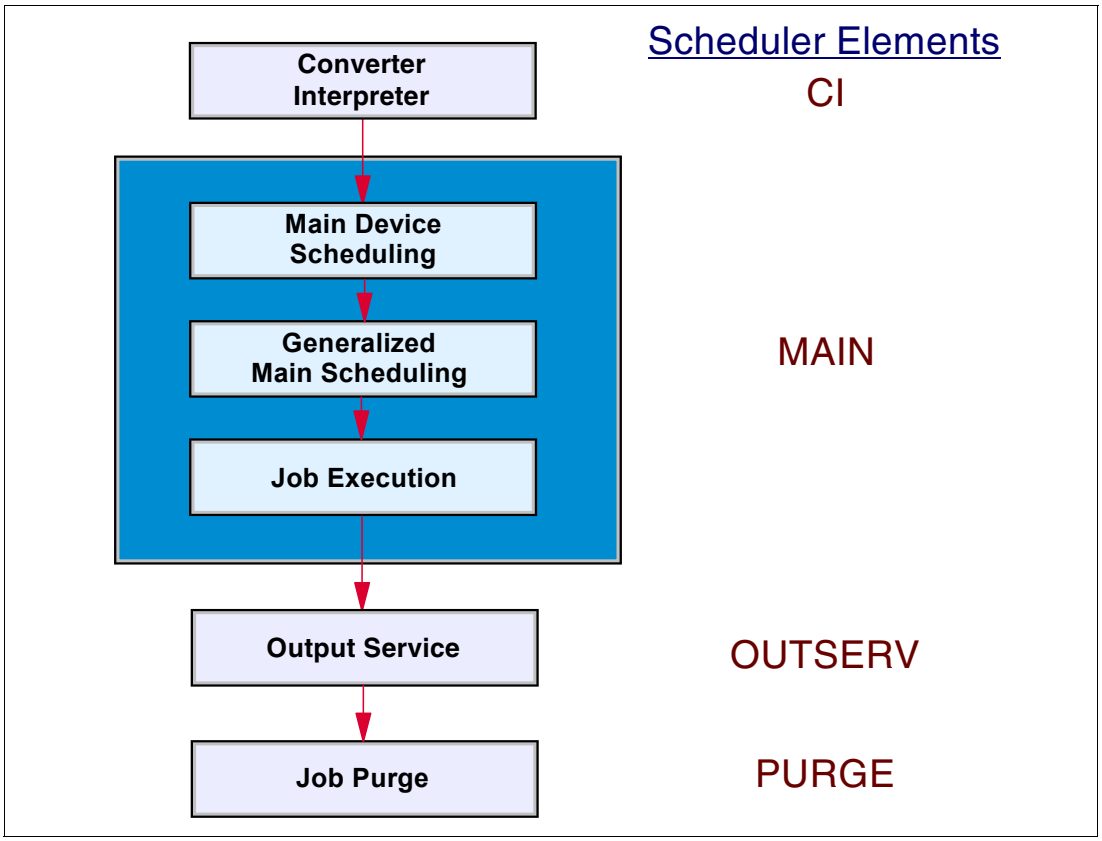
Output service

Output service routines operate in various phases to process sysout data sets destined for local or remote print or punch devices, TSO users, external writers, and writer functional subsystems.

Job purge

Purge processing represents the last scheduler element for all JES3 job. That is, the last processing phases for the jobs. It releases the JES3 resources allocated for the job and cuts the System Management Facility (SMF) to records

3.5 JES3 MVS job flow - schedular elements



JES3 MVS job flow

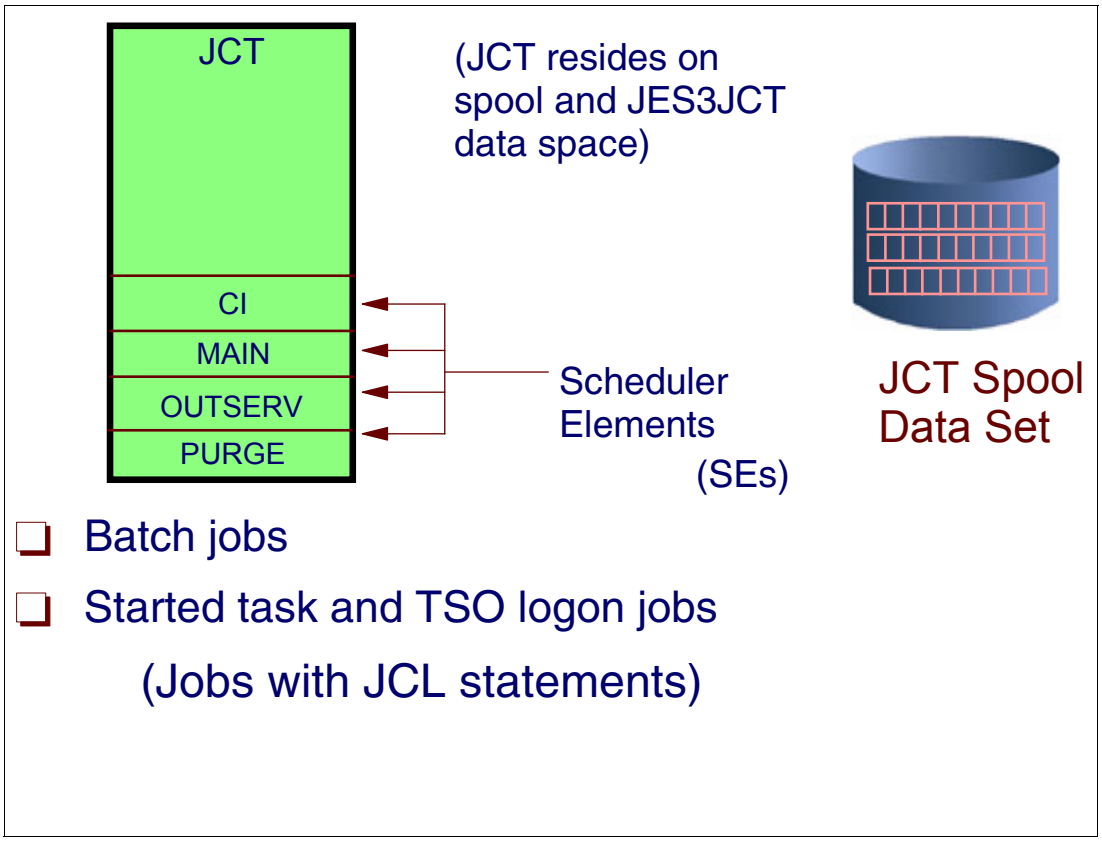
JES3 processing performed on behalf of jobs consists of distinct phases. A typical MVS job entering the JES3 complex is set by the input service to go through the following phases (scheduler elements (SE)):

- Converter/Interpreter (CI) - Interpretation of the job's JCL to a form usable by MVS (SWA - Scheduler Work Area).
- Main Device Scheduling (MDS) - Pre-execution resource allocation to insure I/O resources needed by the job are available, followed by passing (Generalized Main Scheduling - GMS) the job to MVS execution.
- Output Service - Output processing that handles the job's SYSOUT data.
- Purge - Removal the job from the system.

Schedular elements

Each of the four phases of a job is represented by a scheduler element (SE) in the JCT entry. Every scheduler element denotes a unit of JES3 work (DSP) JES3 must perform when processing the job. The JES3 job segment scheduler (JSS) selects SEs (perhaps from several jobs) that are ready for processing, obtains an RQ for the job, and builds entries in the function control table (FCT) (if they do not already exist) so that DSP can be despatched to do the work.

3.6 JES3 standard job JCT



JES3 standard job JCT

Every job containing JCL, but containing no special JES3 control statements, is considered a standard job. JES3 provides the standard job with a default set of scheduler elements. JES3 places into the JCT entries of all standard jobs the same four scheduler elements in the same order:

- Converter/interpreter (CI)
- Main service (MAIN)
- Output service (OUTSERV)
- Purge service (PURGE)

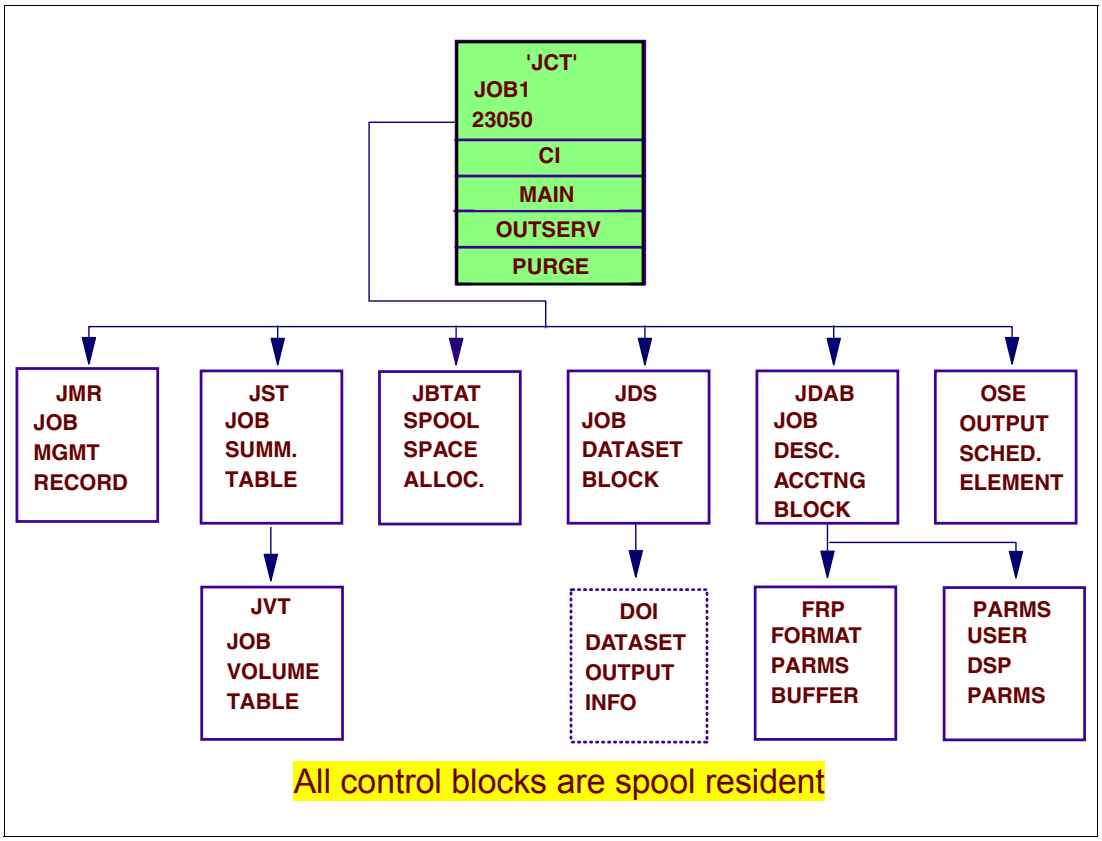
The names in parentheses are formal names of the DSPs that are executed to perform the work. The number or type of scheduler elements to be used for standard jobs can be changed by means of a user exit routine; such a change would apply to all standard jobs.

Batch jobs and started tasks

When you determine where and when you want a set of JCL to run, you will consider using batch jobs or started tasks. Batch jobs are scheduled by a job entry subsystem (JES) and are scheduled to run based on the resources they require and their availability, or based on controls you put on the batch system. Controlling where and when a batch job runs is more complex than using a started task.

A started task is a set of JCL that is run immediately as the result of a START command. Started tasks are generally used for critical applications.

3.7 Job's JES3 control blocks



Job's JES3 control blocks

When JES3 input service creates a job, it stores the job's information in a number of spool resident control blocks. These are described as follows:

JCT The JCT (Job Control Table - IATYJCT) entry containing information about the job's characteristics and file description blocks (FDB) for the other spool resident control blocks.

JDAB The JDAB (Job Description and Accounting Block - IATYJDA) includes information used by DSPs in processing the job. The JDAB is constructed simultaneously with a JCT entry. The JCT and JDAB have much information in common. JDAB gives DSPs a source of data about a job and makes JCT access unnecessary. Entries representing each scheduler element in a JCT entry are appended to the JDAB to provide DSP accounting for each DSP needed for the job.

JMR The JMR (Job Management Record - IATYJMR) is built during input service processing. It contains the job accounting information necessary to construct a System Management Facilities (SMF) type 26 accounting record for the job.

JBTAT The JBTAT (Job/Dataset Track Allocation Table - IATYJBT) contains information about the track groups of spool space allocated to the job.

JDS The JDS (Job Data Set Block - IATYJDS) contains information about all multi-record (e.g. SYSIN/SYSOUT) files owned by the job.

FRP The FRP (Format Parameter Buffer - IATYFRP) includes information supplied through //*FORMAT JECL control statements. If the user indicates, by using the

JES3 FORMAT control statement, that there is to be special handling for any of the SYSOUT data sets, a FRP is constructed for the job during input service processing. The FRP is used to store the information, taken from FORMAT statements, relating to individual SYSOUT data sets.

JST The JST (Job Summary Table - IATYJST) contains setup information for the job

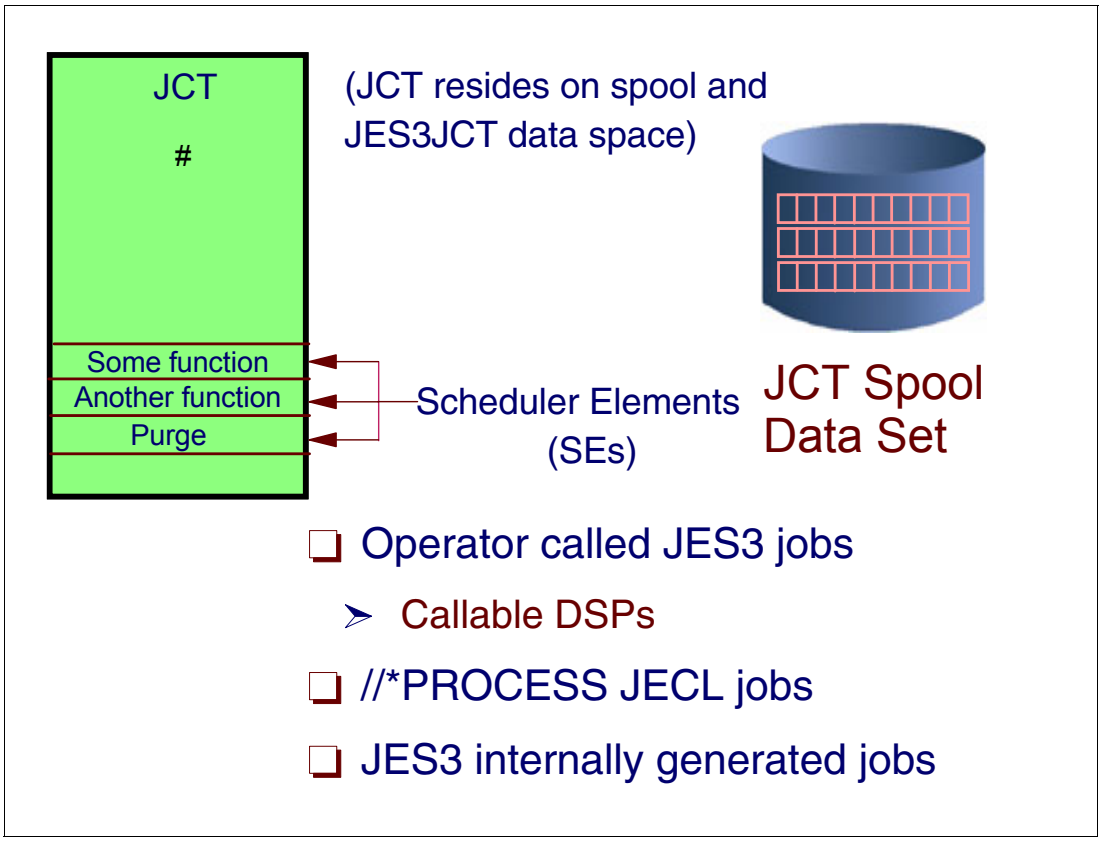
JVT The JVT (Job Volume Table - IATYJVT) entries identify by serial number the volumes a job will require during MVS execution. The JVT is built during converter/interpreter processing.

PARMS The JES3 PROCESS control statement allows the user to define a non-standard job. (That is, it allows design of the set of scheduler elements to be placed in the JCT entry for a job.) Each PROCESS statement may be followed by parameter information to be passed to the DSP named the PROCESS statement. That parameter information is stored in the user parameter buffer (PARMS).

OSE The OSE (Output Service Element - IATYOSE) contains the characteristics of SYSOUT data sets for the job. OSEs contain a set of SYSOUT data set characteristics relating to one or more data sets to be managed during output service processing. The first OSE for a job is constructed during input service processing and written to a spool volume. Later, during output service processing, this OSE is completed and others may be generated for the job. These elements become the "output queue”.

DOI The DOI (Dataset Output Information - IATYDOI) contains dynamically defined characteristics of SYSOUT data sets.

3.8 JES3 non-standard job



JES3 non-standard job

A non-standard job is a job for which the user determines the number of scheduler elements or a job called by the operator in which the job does not contain the four standard job scheduler elements as described in “JES3 standard job JCT” on page 103.

Operator called jobs

Called jobs, which are non-standard, are created by operator request. The operator uses JES3 `*X` commands to make the requests. Called jobs are unique because they are not defined by JCL. These jobs are internally generated by JES3 in response to the `*X` command, and their JCT entries always contain two scheduler elements:

- One to represent the DSP needed for the request
- One to remove the called job from the system (PURGE)

//* PROCESS statement

To specify a different set of scheduler elements on a job-by-job basis, code //*PROCESS JES3 JECL control statements and include them with JCL statements for the jobs.

3.9 JES3 operator callable DSPs

```
WTR      - Writer DSP          NJEROUT  - NJE output rerouting DSP
CR       - Card reader DSP     SNARJP   - SNA RJP DSP
TR       - Tape reader DSP     NJERDR   - Networking reader DSP
DR       - Disk reader DSP     TCP      - TCP/IP networking DSP
MONITOR  - Monitor DSP         TST      - Test DSP
DJ       - Dump job DSP        TST01    - Test DSP
DISPLAY  - Display DSP
DC       - Dump core DSP
RJP      - Remote job processing DSP
DEADLINE - Deadline scheduling DSP
JESNEWS  - JESNEWS DSP
DISPDJC  - Display DJC DSP
RJPSNPS  - RJP snap dump DSP
IC       - Iteration counter DSP
FSIO     - Spool I/O error generator DSP
JMF      - JES3 monitoring facility DSP
NJE      - Networking line manager DSP
NJECONS  - Networking console support DSP
```

Five categories of DSPs:
Global function DSPs
Operator utility DSPs
Diagnostic aid DSPs
Job processing DSPs
Environment DSPs

JES3 DSPs

The visual shows the names of the JES3 callable DSPs. Each small piece of work that JES3 performs when processing a job is accomplished with a JES3 program called a dynamic support program, or DSP. Each DSP is represented on the FCT chain by one or more FCT entries or elements. The elements on the FCT chain are executed according to their priority, and are placed on the FCT chain with the high priority elements first. The higher priority elements are executed before the lower priority elements.

There are 3 types of DSPs:

- Resident DSPs that are fixed parts of JES3 processing.
- DSPs that are invoked by an operator command (`*X` dspname). These DSPs usually involve utilities, input and output services, and diagnostic aids.
- DSPs that process pieces of work required by a job.

There are five categories of DSPs:

- Global function DSPs
- Operator utility DSPs
- Diagnostic aid DSPs
- Job processing DSPs
- Environment DSPs

Global function DSPs

- Console command service - CONCMD
- Console service - CONSERV

- Ready queue - READYQ
- Console spool I/O - CONSDM
- RJP message processor - RJPCONS
- Modify commands processor - MODDRVR
- Inquiry commands processor - INQDRVR
- Work-to-do driver - WTDDRVR
- Job segment scheduler - JSS
- Networking BSC line manager - NJE
- Networking console support - NJECONS
- Networking BSC sender - NJESND
- Networking dummy - NJEDUMY
- Networking store and forward - NJESF
- Networking rerouting - NJEROUT
- Remote job processing - RJP
- SNA/NJE reader - NJERDR
- SNA RJP - SNARJP
- SMF interface - SMFWGST
- Spool data management USAM JDS access interface - DMJA
- Track allocation - TRAKALOC
- MVS/BDT communications - BDTCOMM
- MVS/BDT command processor - BDT
- Set print - SETPR
- DASD dynamic device reconfiguration - DSDDDR
- Tape/unit record DDR - TAPDDR

Operator utility DSPs

- Dump job - DJ
- Tape-to-printer - TP
- Tape-to-tape - TT
- Tape label - TL
- Tape dump - TD
- Spool partition inquiry on track groups - PIG
- Vary assignable device online on a local - VARYL

Diagnostic aid DSPs

- Control block print - CBPRINT
- Display - DISPLAY
- Display DJC - DISPDJC
- Dump core - DC
- Job validation snap routine - INJOBSNP
- JES3 monitoring facility - JMF
- Local JMF - LJMF
- Failsoft test - FSTS
- RJP dump - RJPSNPS
- Iteration counter - IC

Job processing DSPs

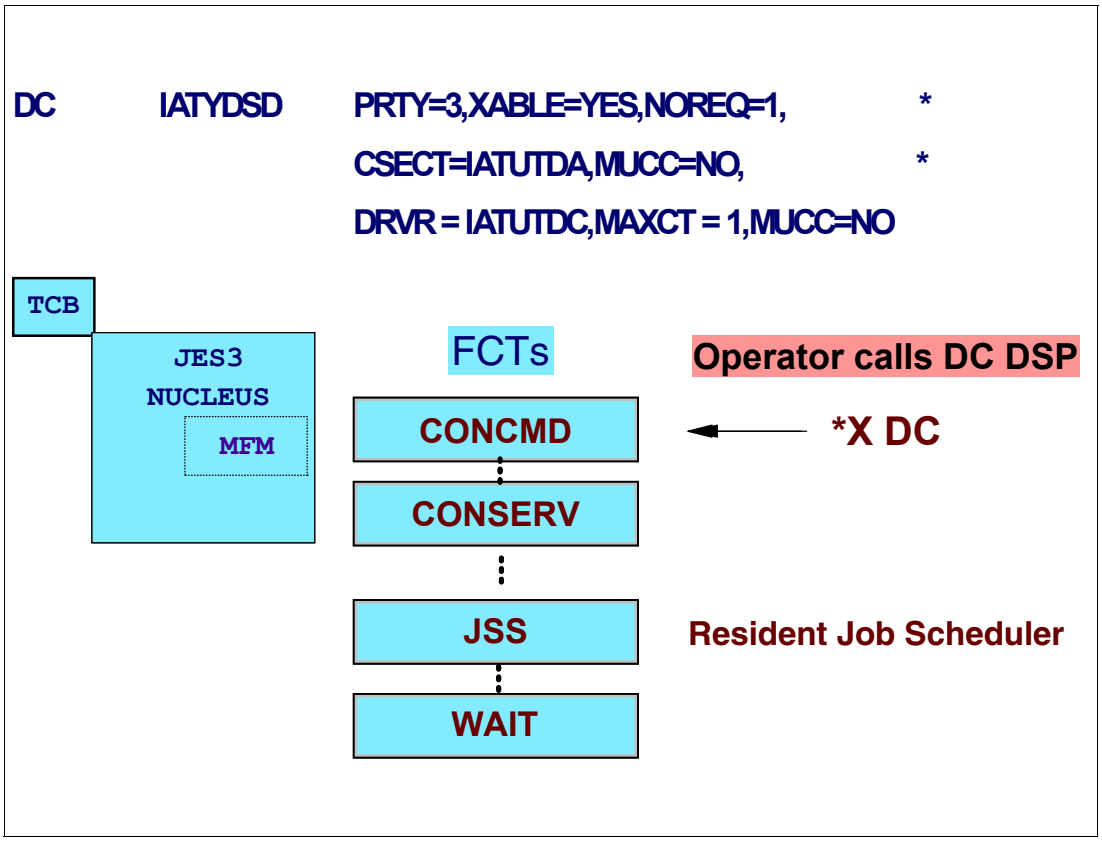
- Purge - PURGE
- Job validation during initialization - INJOBVAL
- Dynamic allocation - DYNAL
- Output service - OUTSERV
- Main device scheduler - SETUP
- MDS system resource scheduler - MDSSRS
- MDS cleanup - MDSCLNUP

- Main service - MAIN
- Writer - WTR
- Card reader - CR
- Tape reader - TR
- Disk reader - DR
- Converter/interpreter - CI
- Postscan - POSTSCAN
- Converter/interpreter cleanup - CICLENUP
- Converter/interpreter driver - CIDRVR
- Enable - ENABLE
- Disable - DISABLE
- Input service - ISDRVR
- JES3 verify - VERIFY
- JES3 locate - LOCATE
- Deadline scheduling - DEADLINE
- //*PROCESS DJC update - DJCUPDAT
- DJC operator utility - DISPDJC
- Internal reader - INTRDR
- DJC update - DJCPROC
- JES3 news data set - JESNEWS

Environment DSPs

- JES3 restart/connect DSP #1 - IATMSR1
- JES3 restart/connect DSP #2 - IATMSR2
- JES3 restart/connect DSP #3 - IATMSR3
- JSAM I/O - JSAM
- Main service general collection - MSGC
- General service - GENSERV
- Failsoft - FAILSOFT

3.10 DSP dictionary and scheduling



Scheduling and DSP dictionary

Each small piece of work that JES3 performs when processing a job is accomplished with a JES3 a dynamic support program (DSP). Each active DSP is represented on the FCT chain by one or more FCT entries. The entries on the FCT chain are executed according to their priority, and are placed on the FCT chain with the high priority element first. The higher priority elements are executed before the lower priority elements.The part of JES3 that handles the FCT scheduling is called multifunction monitor (MFM).

The term scheduling, as it applies to JES3, means the placing of an FCT entry on the FCT chain. The work to be scheduled is defined by scheduler elements which are contained within JCT entries for individual jobs. So the process of scheduling consists of a search of JCT entries to locate scheduler elements for unstarted work, and the building of corresponding FCT entries.

IATYDSD macro

The IATYDSD macro generates an entry for a dynamic support program (DSP) in the DSP dictionary (module IATGRPT or, in a C/I FSS address space, module IATGRPTF). An entry in the table is required for each DSP in order for it to be recognized as part of JES3. The following are some considerations:

DC Specifies the 1- to 8-character name of the DSP whose entry is being created by this macro. If the DSP is to be callable, this name can appear as the argument of an *X,dspname command. If the DSP is to be processable, this name will appear in the //*PROCESS dspname JCL statements. The label is required.

PRTY Specifies the priority to be assigned to the DSP, in the range from 1 through 255. This priority becomes the FCT priority when the DSP is activated.

REENT Specifies whether the DSP is reenterable.

MUCC Specifies whether the maximum use count specified by MAXCT can be altered by using the *MODIFY command.

DRVR Specifies the name of the DSP driver (main line) module to be loaded, if necessary, for each use of the DSP.

CSECT Specifies the name of the data CSECT (dynamic work area) to be loaded by the job segment scheduler driver (module IATGRJR) for each use of the DSP.

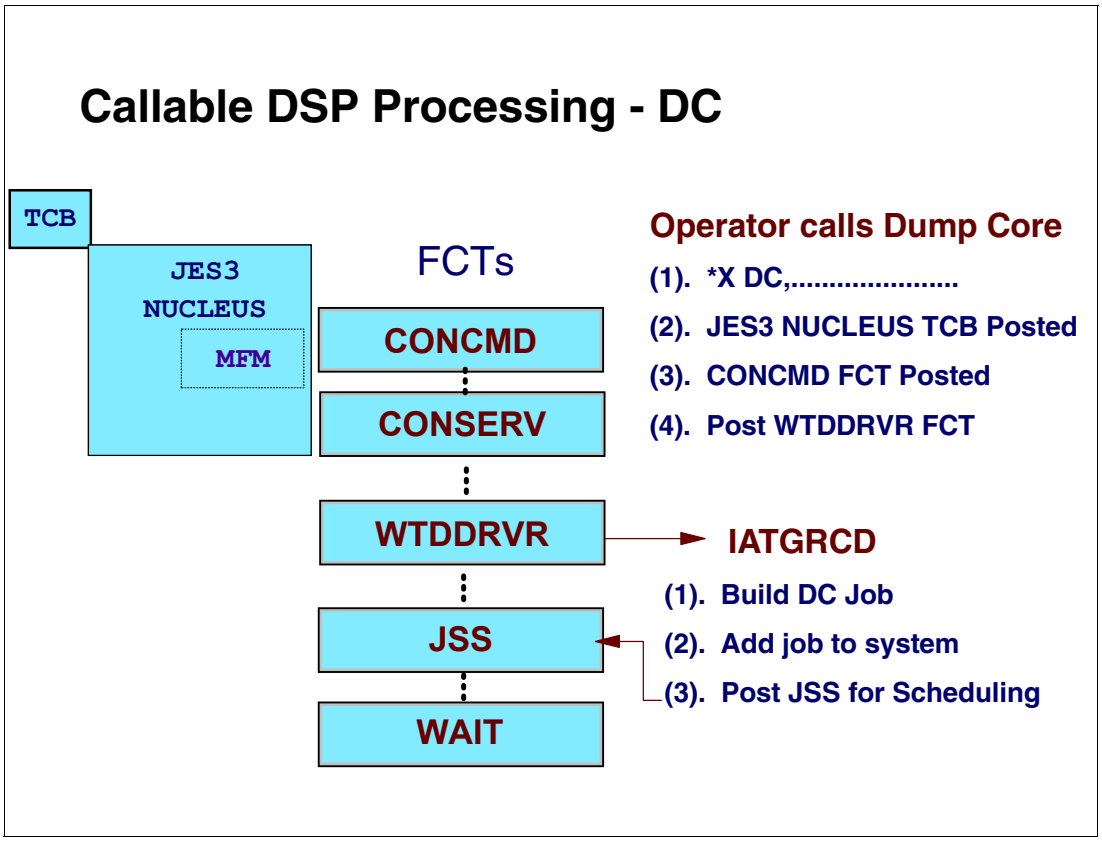
MAXCT Specifies the maximum number of copies of this DSP that may be concurrently active. The number must be within the range 1 through 65535. If this parameter is not specified for a DSP specifying REENT=YES, no MAXCT limit is imposed. This parameter is dynamically alterable by using the *MODIFY command unless you specify MUCC=NO.

XABLE Yes specifies that the DSP can be called by using the **`*X`** command. NO specifies that the DSP cannot be called by using the **`*X`** command.

Job segment scheduler (JSS)

Once the DSP is called by the operator a non-standard job JCT is created. JSS then schedules the DSP into execution by building an FCT and adding the FCT to the FCT chain in priority sequence.

3.11 Callable DSP processing - DC



Callable DSP processing - DC

The visual describes callable DSP processing by using as an example of an operator issuing a **`*X DC`** command to signal JES3 that the DUMP Core utility is be started. This begins a chain of events that includes the following steps:

- The **`*X DC`** command issued by the operator is read by a DSP named CONCMD, which is responsible for JES3 command processing.
- The CONCMD DSP is represented by a resident FCT entry, so it will be dispatched (if there is work to do) when the multifunction monitor reaches the entry.
- The CONCMD DSP simply routes the *X DC command to another DSP, the work-to-do drive (WTDDRVR).
- The work-to-do (WTDDRVR) driver routine (IATGRWD) services requests from JES3 functions that cannot tolerate potential delays in accessing the JCT entries.
- The JES3 function which requests WTDDRVR service constructs a WTD control block and adds it to the WTD queue in the TVT.

WTDDRVR requests

WTD requests come from the following JES3 functions:

- Inquiry/modify
- Console service
- Deadline scheduling

- Dependent job control
- Main service
- MVS TSO

WTDDRVR DSP

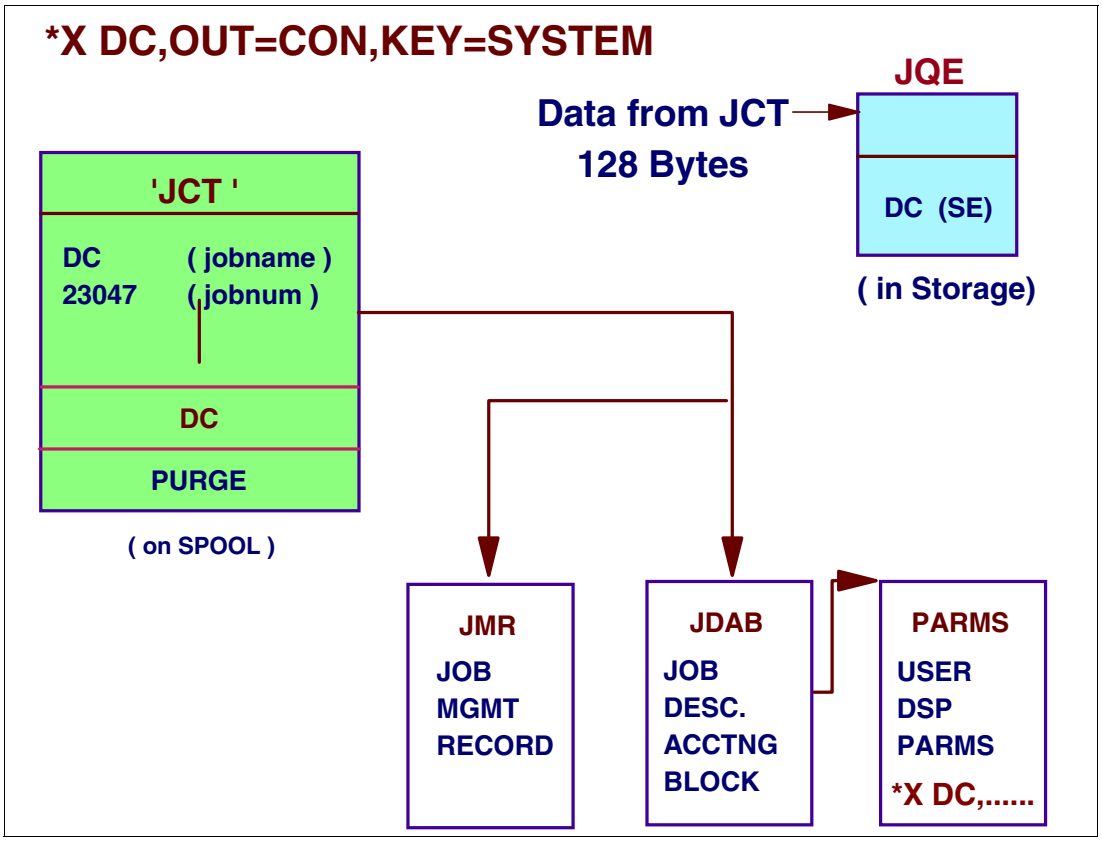
Like the CONCMD DSP, the WTDDRVR DSP is represented by a resident entry on the FCT chain. It is the WTDDRVR DSP which actually begins adding the called DSP to the job queue by passing the console buffer containig the command to the “*X DSP command processing module” (IATGRCD).

IATGRCD processing

Module IATGRCD builds a job structure for the DR DSP and adds the job to the JES3 job queue.

JES3 user exit IATUX27 (Examine/Alter the JDAB, JCT, and JMR) is called from the “*X DSP command processing”. It created the control blocks for the job description accounting block (JDAB), job control table (JCT), and job management record (JMR).

3.12 DC DSP job structure



DC DSP job structure

The DC DSP job has two scheduler elements, DC and PURGE. The JCT entry is added to the JES3JCT data set (spool extent) and an in-storage JQE entry is built. The JQE's current scheduler element number is set to index the DR SE in the JCT. JSS uses the in-storage JQEs when scheduling work.

Callable DSP control blocks

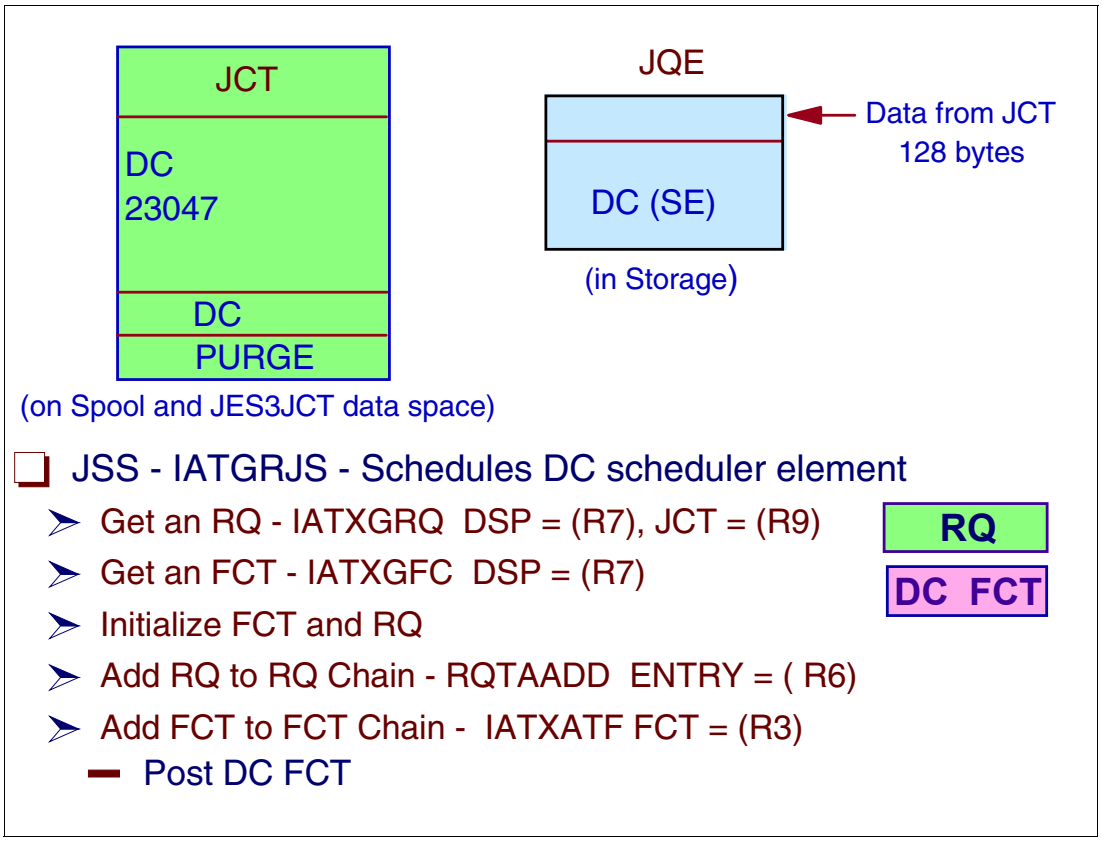
All called jobs have three spool control blocks, a JMR, JDAB, and a parmater buffer (PARMS) where the operator call command is placed.

JDAB The job description and accounting block (JDAB) is constructed simultaneously with a JCT entry. The JCT and JDAB have much information in common. JDABs give DSPs a source of data about jobs and make JCT access unnecessary. Entries representing each scheduler element in a JCT entry are appended to the JDAB to provide DSP accounting for each DSP needed for the job.

JMR The job management record (JMR) is built during input service processing. It contains the job accounting information necessary to construct a System Management Facilities (SMF) type 26 accounting record for the job.

PARMS The called DSP parameter information is stored in the parameter buffer (PARM). The user parameter buffer is generated for called jobs to contain the CALL command.

3.13 JSS scheduling - DC DSP



JSS scheduling - DC DSP

When the **`*X DSP`** command processing adds the JCT and JQE for the newly created DR job into the JES3 complex, the JSS routine is called (using the IATXJSS macro) which adds the JQE to the JSS ready queue and posts JSS for work (JSSWORKQ bit).

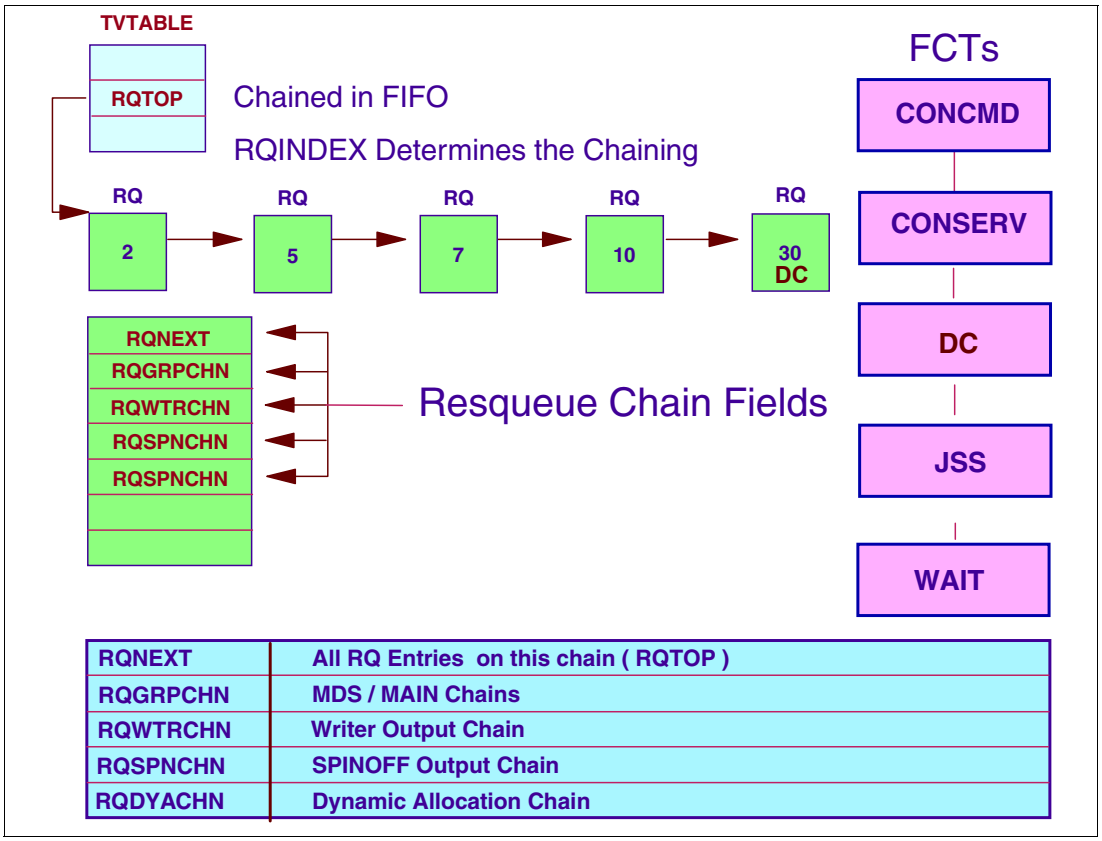
The job segment scheduler FCT is dispatched by the multifunction monitor, locates the JCT entry for the DC job on the JSS ready queue, and schedules the first scheduler element, DC.

RQ and FCT processing

The job segment scheduler creates an FCT entry to represent the DC DSP, places the entry on the FCT chain, builds an resident queue element (RQ) entry to represent the DC job, and passes the RQ entry to the DC DSP for processing. JSS posts the new DC FCT which then will be dispatched by the MFM. The job segment scheduler has now completed its scheduling activities, so it returns to the multifunction monitor, which begins a scan of the FCT chain.

The RQ represents a scheduling chain within a given DSP. It contains a summary of the information the DSP needs to accomplish its function, plus additional information. The JSS builds the RQ from information in the JCT entry. Pointers from the JCT entry to the job's other single-record files (SRFs) are extracted and placed into the RQ. The RQ contains spool information for the job that is executing and holds even more information about the job than the JCT entry.

3.14 RESQUEUE chaining



RESQUEUE (RQ) chaining

When the job segment scheduler (JSS) is dispatched, it examines JQEs to find an eligible job so it can schedule a DSP. Part of the processing is construction of an FCT entry when the DSP being scheduled is not represented by a resident FCT entry. In addition to constructing and enqueuing an FCT entry (if necessary), the job segment scheduler always builds a resident queue element (RQ or RESQUEUE) except for resident FCT entries. The RQ is built after the job is scheduled and it thus becomes the basis for DSP processing.

The RQ is a large control block containing status flags, job information fields, and queuing pointers. RQs last only for the life of a scheduler element. The RQ represents a scheduling chain within a given DSP. It contains a summary of the information the DSP needs to process.

RESQUEUE chain pointers

All RESQUEUE entries are chained together in priority sequence and the anchor is RQTOP in the TVTABLE (TVT). A RESQUEUE when created is assigned a RQINDEX value and RESQUEUE then also reside on a subchain.

RQNEXT All RESQUEUE are chained together using the RQNEXT field. No matter which subchain, RQNEXT points to the next RQ on that chain.

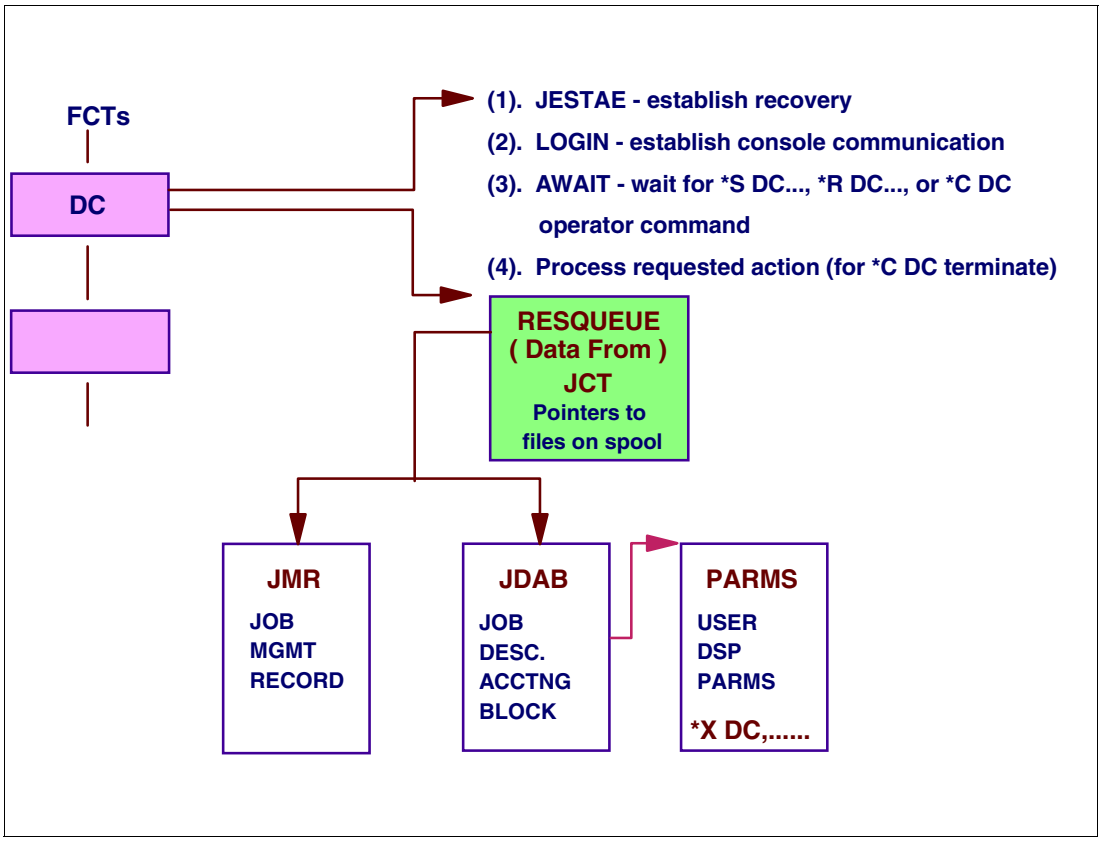
RQRPNCHN When the MAIN scheduler element is active, all RQs are on this chain. The RQINDEX value is used tom place the RQs on their individual chains.

RQSPNCHN Points to the next RQ that requires spin-off processing.

RQWTRCHN Points to the next RQ that requires output service processing.

RQDYACHN Dynamic allocations that occur when jobs are executing have the request placed on this RQ chain.

3.15 Scheduler element active - DC



Scheduler element active - DC

The dump core (DC) utility does the following:

- Display and then modify data in main storage
- Intercept program flow during processing
- Format control blocks for debugging purposes
- Find the location of a module in storage
- Display a requested portion of JES3's storage
- Display the contents of a spool record.

DSP processing initialization

An example of standard DSP required tasks for the initialization phase of a DSP initialization that occur when the job segment scheduler (JSS) schedule and gives control to the DSP:

JESTAE The JESTAE macro defines a DSP abnormal exit routine. The JESTAE exit routine must be resident throughout the life of the JESTAE request. A DSP can issue more than one JESTAE macro. All JESTAE requests issued by programs running under the same FCT are queued so that the exit established by the most recent JESTAE request will be the first to get control. If this exit fails or requests that the abnormal termination continue, the exit established by the previous JESTAE request will get control. This process is called JESTAE percolation.

LOGIN The LOGIN macro establishes communication and transfer of data between console service and the dynamic support program (DSP) using the macro. This macro must be executed by each DSP that allows the receipt of messages and responses from the consoles.

3.16 DC DSP console commands

```
*x dc
IAT6306 JOB (JOB01472) IS DC      , CALLED BY ROGERS
*IAT7921 ISSUE START/CANCEL/RESTART DC REQUEST

*i j=dc
 IAT8674 JOB DC       (JOB01472) P=15 DC(ACTIVE)
 IAT8699 INQUIRY ON JOB STATUS COMPLETE,       1 JOB  DISPLAYED

*i j=1472
 IAT8674 JOB DC       (JOB01472) P=15 DC(ACTIVE)
 IAT8699 INQUIRY ON JOB STATUS COMPLETE,       1 JOB  DISPLAYED

*S DC,OPTION=INS
                 INTERNAL READER ANCHOR BLOCK
     LOC         IRE       ISCD       DICT    ACTIVE   IDLE    HI-WATER
   245CD098   245AB6F8   24C591A0   00000000   0002    0002      0002
                 INTERNAL READER ELEMENT CHAIN
     LOC        RESQ       NEXT       PREV       ECFA    FLAG   INTRDR
   245AB6F8   24992000   245AB728   00000000   24C92F6A   80   JOB28541
   245AB728   249921C8   00000000   245AB6F8   24C8FF6A   80   JOB11479
*IAT7921 ISSUE START/CANCEL/RESTART DC REQUEST
```

DSP commands

Once the DC DSP, or any DSP becomes active by using the ***X** command, as shown in the visual, the following operator commands can be used to communicate with the active DSP.

To inquire on the DC job, you can use the ***I** command, as follows:

```
*I J=DC
```

The response shown that the DC job is active with the first scheduler element DC.

If you know the DC job number, then the inquiry command can specify the job number, as follows:

```
*I J=1472
```

Display core DSP (DC)

Now that the DC job is active, you can issue ***S** commands. The visual shows a ***S** command to the DC DSP, OPTION=INS, which will list all of the internal reader control blocks waiting for internal reader processing. The name of dump contents of storage option(s) to be displayed. The format for most control blocks that are dumped by the ***S DC,OPTION=options** are:

DMP Causes all the standard formatting to be performed just as though a completely formatted JES3 dump was requested without system intervention.

INS Displays information about internal reader control blocks.

SNP= *name* - Causes a dump of job-related control blocks to be recorded on the output device. SNP=name causes a dump of a specific job-related control block to be

recorded on the output device. You can request any one of the following control blocks:

Where *name* is:

ARL - Allocation resource list

ASR - Available spool records

JDAB - Job description accounting block

JDS - Job data set control block

RQ - Resqueue

FRP - Format parameter buffer

JST - Job summary table

JVT - Job volume table

TAT - Job or data set track allocation table

MOSE - Master output service element

OSE - Output service element

With the SNP=name, you can also specify J=ALL or J=jobno, for example:

```
*S DC,OPTION=SNP=name,J=ALL
```

SNP= JCT - JCT specifies a dump of the JCT, use the SOURCE= parameter to specify whether dump core should obtain a copy of the JCT from:

The JCT data space - (Specify JCTDS)

The JCT data set - (Specify DSPACE)

Both the JCT data space and the data set - (Specify ALL)

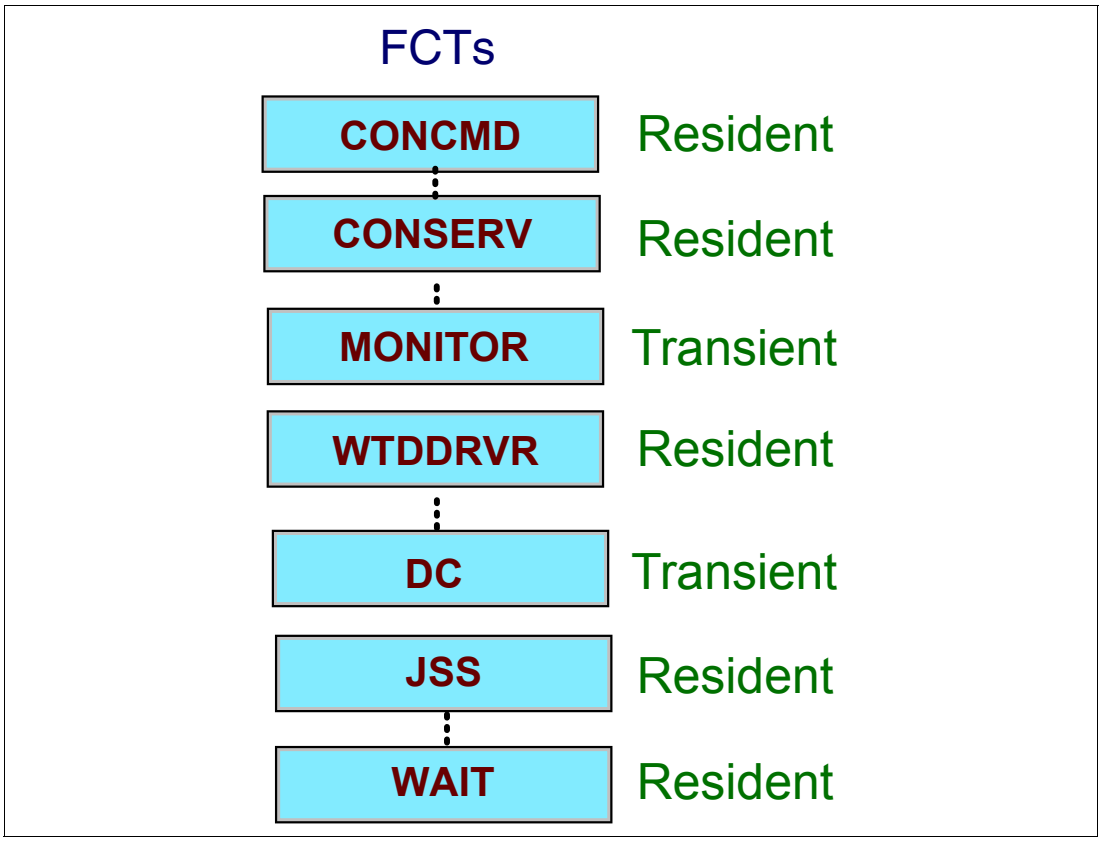
For example: `*S DC,OPTION=SNP=JCT,SOURCE=ALL`

SNP= OSE,DIAG - DIAG displays a formatted OSE, which includes such information as:

class
forms
queue
destination

SNP, J=jobno|ALL - J=jobno or ALL indicates either the number of the job whose control blocks are to be dumped. The default (ALL) causes all jobs' control blocks to be dumped (that is, the JES3 job queue is dumped). You can use this parameter to select a job number only when the OPTION= parameter is set to SNP.

3.17 FCT chain



FCT chain

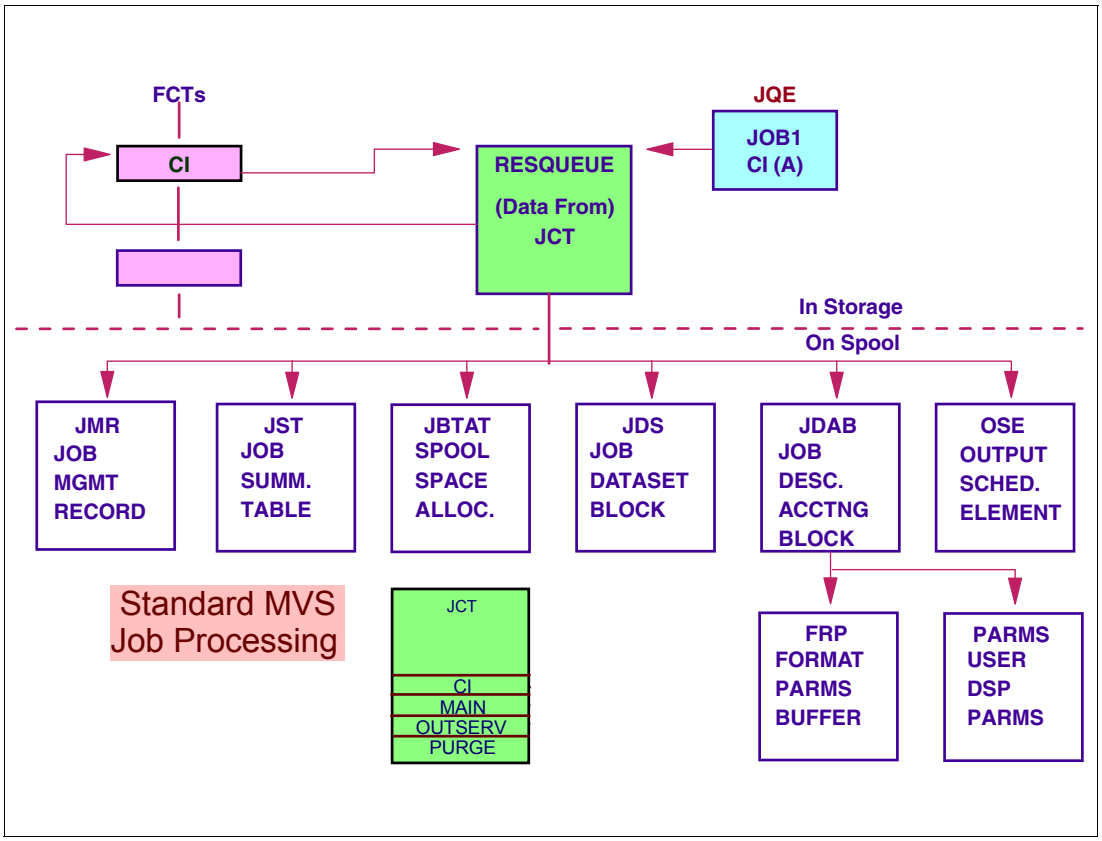
This visual shows some of the FCTs that exist in an FCT chain and it shows two of the FCTs previously discussed in this topic.

The FCT chain elements are arranged according to the priority of the DSP, going from the highest to the lowest. Certain resident FCT entries that represent required DSPs have a higher priority than other FCT entries, however, priority is not related to residence.

The priority of an FCT entry is determined by the response requirements of the JES3 function involved. The last (lowest priority) FCT entry represents the WAIT DSP. When dispatched, the WAIT DSP issues an MVS WAIT macro indicating that there are no more dispatchable FCT entries on the FCT chain. In other words, JES3 has no more work to do.

Non-resident, or transient, FCT entries, on the other hand, are added and deleted from the dispatching queue as needed. These non-resident FCT entries, such as DC, perform operator utility functions as well as other basic JES3 functions.

3.18 Converter/Interpreter processing



Converter/Interpreter processing

The CI DSP exists for one main reason: to gather information concerning the job's device, volume, and data set requirements prior to MVS execution of the job. This information is necessary for pre-execution setup, one of the major facilities of JES3. The information is extracted from DD statements in the job's JCL. JSS schedules the CI processing using the CI element in the JQE and builds the FCT and RQ.

JES3 CI FCT

The JES3 CI FCT invokes the MVS converter/interpreter (C/I) function to process the JCL statements in a job stream. Converter processing converts JCL to C/I internal text (or C/I text). If the converter detects any JCL syntax or logic errors, the job is printed and purged (and the MAIN scheduler element is bypassed). If there are no errors, the interpreter routines then convert the C/I text to scheduler work area (SWA) control blocks. The interpreter routines build all SWA control blocks above 16 megabytes in virtual storage in the JES3 global and C/I FSS address spaces. The scheduler control blocks are moved from the SWA to JES3 spool.

CI scheduler element in the JCT

When the CI scheduler element is scheduled and a RQ entry is created, the RQ has a pointer to all of the job's control blocks. When a job enters the MAIN scheduler element, the JES3 initiator support routines move the scheduler control blocks from spool to the SWA in the user's address space. You can modify the scheduler control blocks before they are moved to the SWA by implementing installation exit routine IATUX26.

3.19 Converter/Intepreter processing

- First scheduler element for standard job
  - Interface to MVS converter/interpreter
  - Construct SWA - write to spool
  - Flush jobs with JCL errors
  - Determine devices required
  - Determine volumes required
  - Establish data set awareness
  - 13 user exits
  - Uses subtasks - number is user defined
    - Controllable by operator

Converter/Intepreter Processing

The CI DSP serves as an interface between JES3 and the JES3 CI subtasks that invoke the MVS converter/interpreter subtasks. The interface responsibilities of the CI DSP are:

- To fail the job if JCL errors are detected by the MVS converter/interpreter.
- To flush any job that contains more JCL statements than the limit allows.
- To delay processing of any job that would temporarily cause the C/I subtasks to process more JCL statements than the system limit allows
- To construct the SWA control blocks that are produced by the MVS converter/interpreter and write them onto the spool.

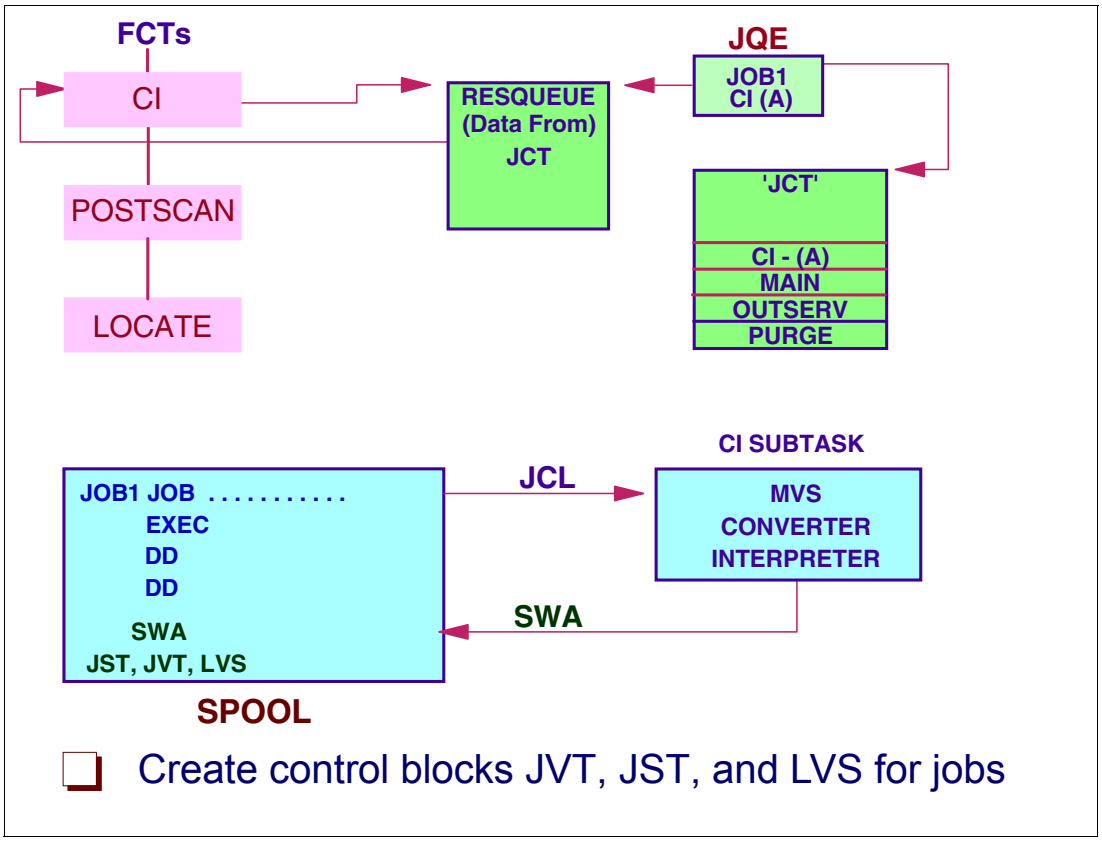
CI DSP processing

The CI DSP stores I/O requirements in job-related control blocks called the job summary table (JST) and the job volume table (JVT). It writes these control blocks to spool along with others for the job. Devices, volumes, and data sets required by the job are determined during this processing. The information related to data sets and volumes are kept in main storage allows JES3 to maintain a complex-wide awareness of the status of volumes and data sets.

CI subtasks

During JES3 initialization, JES3 attaches an installation-defined number of converter/interpreter (C/I) tasks. These C/I tasks are subtasks to the JES3 primary task. When a C/I subtask processes a job, it stores the interpreted job's JCL into the scheduler work area (SWA) in the address space in which the service is invoked (global or functional subsystem) above the 16 megabytes line.

3.20 CI scheduler element processing FCTs



CI scheduler element processing FCTs

The CI subtask is responsible for invoking the MVS converter to convert JCL into internal text and to the MVS interpreter to create from the internal text the scheduler control blocks (SWA).

Prescan phase of the CI scheduler element processing extracts job requirements from the scheduler control blocks for use in the postscan phase. At the end of the prescan phase, the scheduler control blocks are moved from the SWA to JES3 spool.

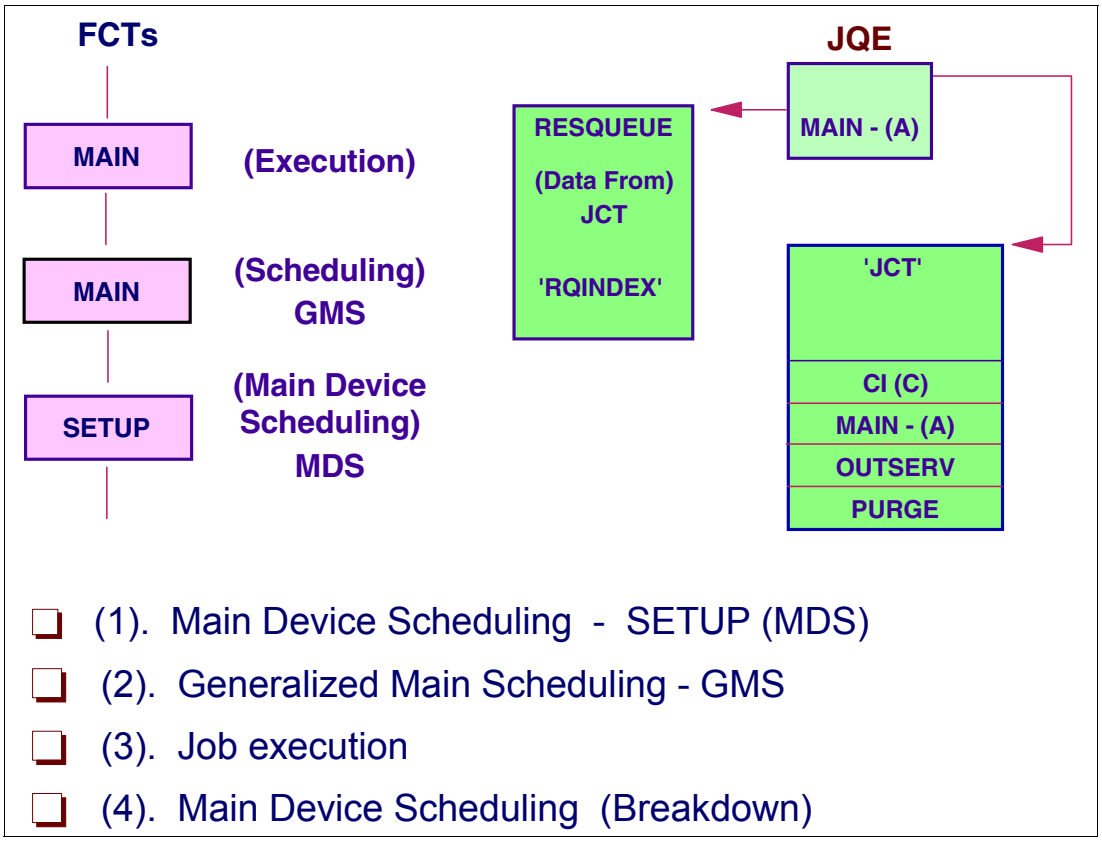
The CI DSP stores I/O requirements in job-related control blocks called the job summary table (JST) and the job volume table (JVT). It writes these control blocks to spool along with others for the job.

JST The job summary table (JST) is a spool-resident control block that contains a summary of the job's unit, volume and data set requirements. It is used by main device scheduling (MDS) to allocate the job's resources. It is created during the postscan phase from the IJS and the JVT and may span one or more buffers, depending on the job's setup requirements.

JVT The job volume table (JVT) contains an entry for each volume (needed by the job) that requires a JES3-managed device. Each JVT entry is associated with the IJS entry for that DD statement.

LVS The locate request table (LVS) contains an entry for each data set that requires a catalog search to obtain unit and volume information. The data set is assumed to require a volume on a JES3-managed device until the catalog is searched. Each LVS entry is associated with the IJS entry for that data set.

3.21 Main scheduler element processing



Main scheduler element processing

Main service processing is the allocation of device, volume, and data set resources to jobs, the selection of jobs to be passed to MVS initiators for execution, and the freeing of allocated resources after the jobs are executed.

For a standard jobs, main service processing begins when the job segment scheduler reaches the MAIN scheduler element in the job's JCT entry. The MAIN scheduler element represents two DSPs: setup (SETUP) and generalized main scheduling (MAIN). Both of these DSPs are resident and each has a permanent entry on the FCT chain, so the job segment scheduler need not construct the FCT entries.

MAIN and SETUP FCTs

The work performed by the SETUP and MAIN DSPs is crucial to JES3 processing. The goals of the SETUP and MAIN DSPs are effective resource utilization and maximum job throughput. The processing sequence is:

1. Initial setup processing to prepare I/O resources
2. Generalized main scheduling to select and pass a job to an initiator
3. Job execution
4. Breakdown processing to relinquish I/O resources

   MDS allocates I/O resources among competing jobs and releases resources after they have been used. The type of JES3 setup processing is defined by JES3 initialization statement parameters, JECL control statements, and JES3 operator commands.

3.22 Main device scheduling - MDS

- MDS processing options - SETPARAM statement
  - Volume fetch
  - Job setup
  - High watermark setup (HWS)
  - Explicit setup - //*MAIN SETUP=
  - NONE - (makes MDS optional)
- Schedules resources to jobs
  - Devices - volumes - data sets
- Objectives
  - Satisfy resource I/O requirements before execution

Main device scheduling - MDS

Main device scheduling (MDS) represents the second phase of setup processing. converter/interpreter is the first phase. The converter/interpreter routines construct a job summary table (JST) that lists required data sets and devices, and a job volume table (JVT) that describes the volumes the main device scheduling routines will fetch and allocate. Volumes that are mounted will be verified.

Main device scheduling functions are optional and may be bypassed for jobs as if SETUP=NONE is specified on the SETPARAM statement to indicate no MDS, and all devices are allocated, mounted, and deallocated by MVS. The SETPARAM initialization statement specifies parameters that the JES3 main device scheduler (MDS) and the DYNAL DSP uses in allocation, mounting, and deallocation of devices for jobs run on all mains.SETPARAM also indicates how MDS is to manage devices through the following options:

- Volume fetch - Specifies whether fetch messages for mountable devices are written into the JESMSGLG data set.
- Job setup - Requests setup to allocate all JES3-managed devices required in the job before the job executes.
- High-watermark setup - Requests setup to allocate the minimum number of devices required to run the job.
- Explicit setup - Modifies the standard setup algorithm used in assigning devices to a job before its execution.

Schedule resource to jobs

JES3 provides a device management facility called the main device scheduler (MDS) that can wholly or partially support the MVS allocation process. The purpose of MDS is to satisfy job resource requirements (the devices, volumes, and data sets needed) before and during job execution, thus allowing execution to proceed without allocation delays. MDS also allows controlled multisystem access to commonly accessible data sets in the loosely coupled environment.

MDS objectives

MVS and MDS allocation consider a job's resource requirements at different levels. MVS allocation considers job requirements one step at a time for the processor executing the job; MDS considers the resource requirements for all the steps in a job for all processors in the JES3 complex.

The SETNAME and DEVICE statements are used with the SETPARAM statements. SETNAME and DEVICE identify the devices to be managed by MDS. MDS processing is performed in phases.

3.23 Generalized main scheduling - GMS

- Generalized main scheduling - GMS
  - User defined algorithms - jobs to initiators
- JES3 or MVS workload manager (WLM) or both manage initiators
- Job execution
  - Monitor jobs during execution
  - Operator communication
- Main device scheduling - Breakdown
  - Return resources - devices, volumes, data sets

Generalized main scheduling - GMS

Each time an MVS initiator requests work, generalized main scheduling (GMS) selects and schedules a job for execution. The job that GMS selects depends primarily upon initialization parameters that you have specified.

Initiator management

An installation can have JES3 or MVS workload manager (WLM) or both manage initiators for batch jobs. JES3 or WLM control of initiators is at the job class group level. To specify whether JES3 or WLM manages initiators for a job class group, the installation uses the MODE parameter on the GROUP initialization statement. If MODE=JES is specified or defaulted, the initiators are managed by JES3. If MODE=WLM is specified, the initiators are managed by WLM. The MODE parameter can also be changed dynamically using the `*F G` command. A group must be either WLM managed or JES3 managed; it cannot be WLM managed on one system and JES3 managed on another. When triggered by an MVS initiator's request for a job, the MAIN DSP chooses a job to give to the initiator. The selection of the job is influenced by the installation specifications.

JES3-managed initiators

For JES3-managed initiators, jobs are selected within a job class group by priority. If there is more than one job with the same priority, the jobs are ordered by the time they are ready to be selected to run (that is, when they arrive on the GMS select queue). Changing a job's priority affects when the job is selected to run.

GMS selects jobs to be processed by MVS initiators. For JES3-managed initiators, GMS selects a job for execution using the job selection algorithms established at JES3 initialization. MAINPROC, SELECT, CLASS, and GROUP initialization statements control the key variables in the job scheduling and job execution process. The selection variables include:

- Eligibility relationships between jobs and processors based on:
  - Job classes
  - The number of active initiators for the jobs (JES3 managed initiators). JES3-managed initiators (how many, when to start them, and where to start them) are controlled by JES3 based on the EXRESC parameter of the GROUP initialization statement and through the use of the *MODIFY,G command.
  - Whether processors are online or offline
  - I/O rates of jobs in execution (JES3 managed initiators)
  - Virtual storage requirements relative to the working set size of address spaces given to jobs (JES3 managed initiators)
  - Job priorities, to the extent that the installation wishes to honor priorities

WLM-managed initiators

For WLM-managed initiators, jobs are selected within a service class by their **main service arrival time**. Main service arrival time is the elapse time for C/I processing of a job. Job priority does not influence whether a job is selected to run first; therefore changing the priority for a job has no effect unless such a change causes the job's service class to change to one having more aggressive goals.

WLM classification assigns a service class and optionally a report class to a job based on the classification rules in the WLM policy. A service class is a group of work which has the same performance goals, resource requirements, or business importance. For workload management, you assign a service goal and optionally a resource group to a service class. WLM starts initiators on a service class basis. WLM determines how many initiators to start, when to start them, and where to start them based on performance goals in the WLM policy, backlog of jobs, and system capacity.

When GMS receives a request for work from a WLM-managed initiator, the set of jobs to be selected narrows the choice in two ways:

1. The service class identification of the initiator limits the choice to jobs with that service class.
2. The processor on which the initiator is started limits the choice to jobs that can run on that processor.

Even though WLM management operates at a job class group level, WLM-managed initiators select work by service class and not by job class group like JES-managed initiators. During execution, MVS work load management (WLM), JES3, and the operator monitor jobs.

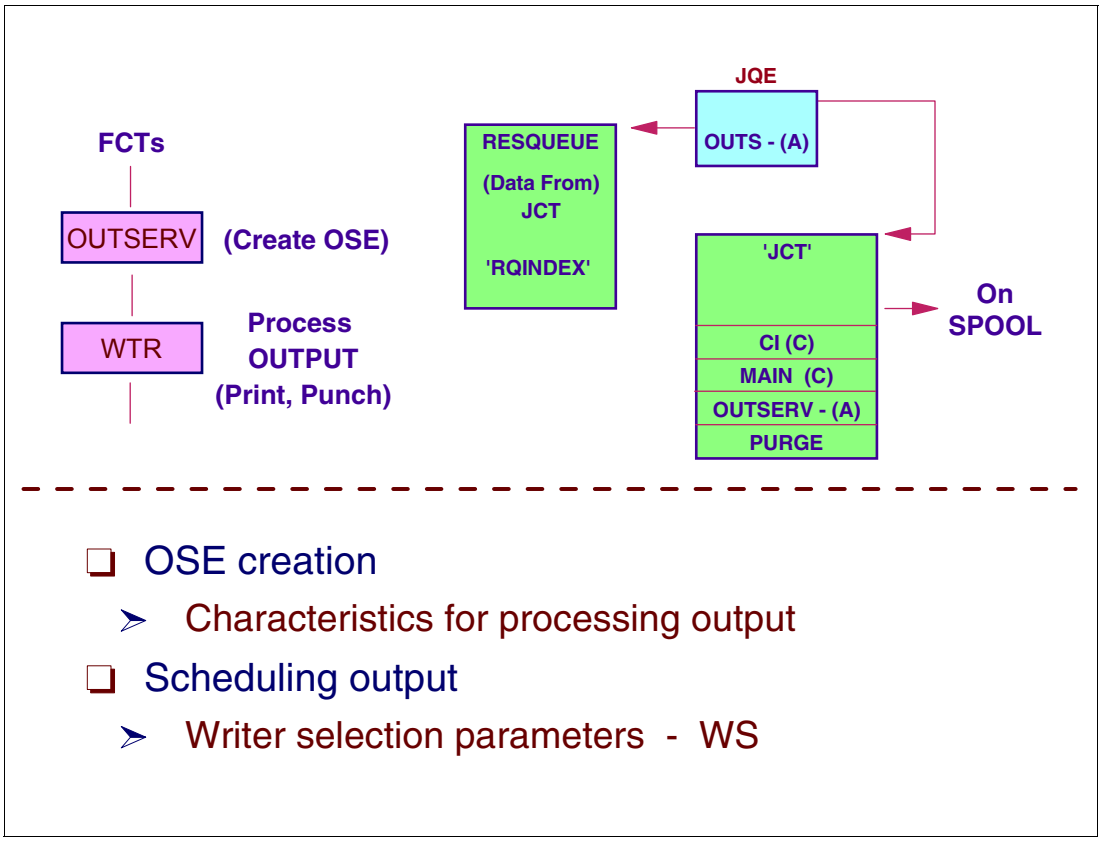
Job execution

Job scheduling controls the order and execution of jobs running within the JES3 complex. Job scheduling involves the routines invoked by the MAIN DSPs, which are represented by the MAIN scheduler elements on the job control table entry.

MDS breakdown

When a step completes or the job ends, MDS breakdown occurs which frees the devices, volumes, and data sets still held by the job.

3.24 OUTSERV scheduler element processing



OUTSERV scheduler element processing

Output service executes on the global and processes SYSOUT data sets destined for JES3 managed devices, PSF managed devices, system application printer interface (SAPI) applications, NJE (BSC, TCP, and SNA), and Process SYSOUT (PSO).

The output service driver receives control after a job completes breakdown in main service, after a job spins off an output data set, or after JES3 spins off an output data set.

JES3 output service performs three distinct phases:

1. Phase 1 - queuing of output
2. Phase 2 - scheduling of output
3. Phase 3 - printing and punching of output

Phases 1 and 2 occur in the JES3 global address space on the global processor. Phase 3 can run in the global address space under the global primary task, the global auxiliary task, or functional subsystems.

OUTSERV DSP

The OUTSERV DSP summarizes output data sets at two points of processing:

- For normal jobs, when a job completes main service processing
- For spin-off data sets when a job or a DSP moves a data set while executing directly to output service for processing.

The RESQUEUE entry for the job to be processed by the queueing function is placed on a queue of output service work. When the queueing function receives control, it dequeues the next job on the work queue.

OSE creation

The queueing function of output service accesses the job data set (JDS) for the job or for the spin data sets of a job. The queueing function builds output scheduling elements (OSEs) from the JDS. One OSE is built for each group of data sets that have unique writer requirements.

The output scheduling elements (OSEs) contain SYSOUT data set characteristics extracted from JCL parameters on the SYSOUT DD and OUTPUT JCL statements, the //*FORMAT JES3 control statements, and the JES3 SYSOUT class table.

The installation can change the characteristics in the OSE by coding the installation exit routine IATUX19 (Examine/Modify Temporary OSE) or IATUX72 (Examine/Modify a Temporary OSE or an OSE Moved to Writer Queue).

Scheduling output

JES3 output service schedules OSEs to writers in one of two ways:

- An OSE is used to scan the 'writers waiting for work' queue and the available-devices queue to find a device that can process the OSE.
- A set of writer scheduling parameters (WS) is used to search the OSEs for the first perfect-fit OSE or for the OSE which best fits the requirements of the writer requesting a job. You can specify these parameters on the DEVICE or OUTSERV initialization statement by coding the WC and WS parameters. The operator can change these parameters when calling, starting, or restarting a writer.

  If two or more OSEs fit the requirements of this writer equally well, JES3 schedules the OSE with the highest JES3 job queue priority. The JES3 job queue priority is based on the job priority specified on the JCL JOB statement.

3.25 Output service processing

- Output writers
  - Hot writers - *X WTR,.....
  - Dynamic writers
- Output types processed
  - Print and punch
  - External writer - PSO
  - TSO users via OUTPUT command - PSO
  - SYSOUT Application Program Interface - SAPI
- User exits
- SMF type 6 records

Hot writers

Hot writers give operations personnel total control of output handling. Operators enter commands to call and control hot writers. Hot writers remain active, even when there is nothing to print.

Dynamic writers

Dynamic writers are often used for volume printing on stock paper. These writers allow JES3 to control changing the setup characteristics for devices thereby reducing the amount of control operators have over when and how writing is performed.

Output types

These characteristics indicate which type of device is to receive the output. Data sets that were defined as "print type" will be sent to printers, "punch type" will be routed to punches, and "sys type" will be routed to TSO or to external writers.

User exits

There are 10 user exits during the OUTSERV DSP.

PSO interface

The Process SYSOUT Data Sets call (PSO - SSI function code 1) allows a user-supplied program to access JES SYSOUT data sets independently from the normal functions (such as print, network) JES provides, so that the characteristics of the SYSOUT data sets can be either retrieved or updated. The program using this interface is called an external writer.

SAPI interface

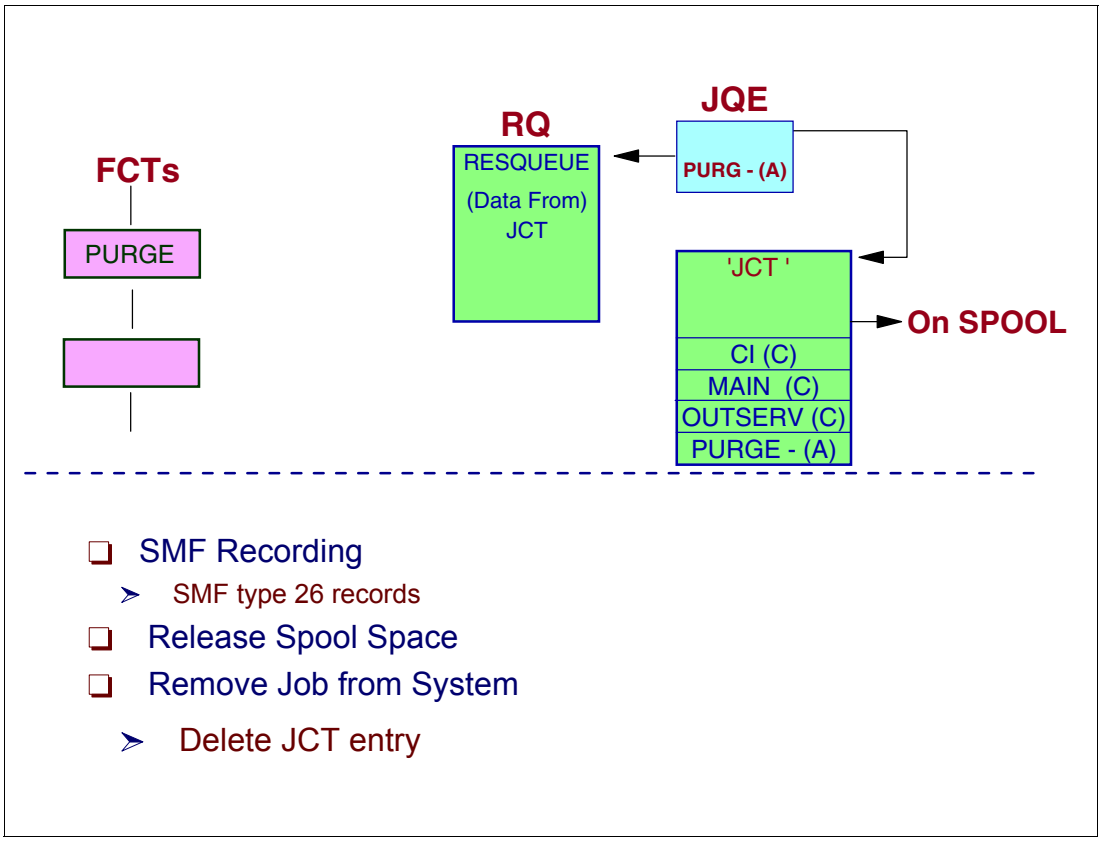
The SYSOUT Application Program Interface (SAPI - SSI function code 79) allows JES3 to function as a server for applications needing to process SYSOUT data sets residing on JES spool.

SMF type 6

Output service creates an SMF type 6 (JES3 Output Writer) record for each data set processed. One type 6 record is written for each data set section within an output scheduler element (OSE). It contains information on the output writer activity such as:

- The number of logical records processed
- Number of data sets processed
- Output service start time and date
- I/O status indicators
- Data set control indicators
- JES3 logical output device name
- Output activity.

3.26 Purge processing



Purge processing

Purge is the last processing function for a job in the JES3 system. It releases all JES3 DASD space assigned to the job, making it available for allocation to subsequent jobs. A message is issued to the operator indicating that the job has been purged from the system.

Purge processing consists of the following steps:

- JES3 releases all spool space assigned to the job and updates resident control blocks.
- JES3 writes SMF record type 25 which contains device allocation information.
- JES3 writes SMF record type 26 which contains final job accounting information.
- JES3 informs the operator that the job has been purged.
- The JCT entry is deleted.

SMF recording

The JES3 SMF record type 25 is written for each job that completed JES3 converter/interpreter (C/I) processing. One type 25 record is written for each job, whether the job contains DD statements. A separate type 25 record is written for each job that uses a private catalog. A separate type 25 record is written for each main device scheduling (MDS) dynamic unallocation request.

The SMF type 25 record contains allocation-related information such as the number of tape and disk volumes fetched and mounted, and the time and date of JES3 device verification.

The job name, time, and date that the reader recognized the JOB card (for the job) constitute the job log identification.

JES3 SMF record type 26

This record contains operating information such as:

- Message class
- Job class
- JES3 job selection priority
- JES3 logical input device name
- Processing time
- Output lines
- Output punched cards
- Deadline schedule type
- Deadline schedule time and date
- Start and stop times for:
  - The reader
  - The Converter
  - The Execution processor
  - The output processor.

3.27 JES3 JECL statements

- Control statements types
  - //*OPERATOR — Messages to Operator
  - //*MAIN — Job Processing Options
  - //*FORMAT — Output Processing Options
  - //*NET — Dependent Job Control - DJC
  - //*PROCESS — Non-standard Jobs
  - //*DATASET — In-stream data
- Control statement usage
  - Non - Standard job scheduling
  - Overriding installation standard defaults
  - Invoking optional JES3 facilities
    - Dependent Job Control - DJC
    - Deadline scheduling

JES3 JECL statements

JES3 JECL control statements are processed during the input service ISDRVR phase of processing for a job and are as follows:

//*OPERATOR	This statement allows operator commands to be in the input stream.
//*MAIN	This statement defines the processor requirements for the current job. Many parameters are used to override parameters on the JES3 STANDARDS initialization statement.
//*FORMAT	Each //*FORMAT statement specifies the job setup requirements and the information is stored in the format parameter buffer (FRP).
//*NET	Each //*NET statement defines a dependent job control (DJC) network.
//*PROCESS	For each //*PROCESS statement, which specifies the scheduler element, input logic determines if the specified DSP name is valid and adds the DSP to the scheduler element list for the job.
//*DATASET	Allows an input data set to be specified in the input stream.
//PAUSE**	Halt an input reader temporarily.
//*ROUTE XEQ	Send an input stream to a network node.

Control statement placement

Place JES3 control statements, except the command and //**PAUSE statements, after the JOB statement and its continuations. JES3 ignores JES3 control statements, except the

command and //**PAUSE statements, that appear before the JOB statement or between continued JOB statements.

Do not include JES3 control statements in a cataloged or in-stream procedure. JES3 ignores JES3 control statements in a procedure.

JES3 control statements have no function in an APPC scheduling environment. If you code them, the system will ignore them, and they will appear as comments in the job listing. Use of JECL statements in started task JCL will result in JES3 failing the job.

//*MAIN parameter

Deadline scheduling is defined via this statement. The deadline specifies when the job is required to be finished executing. Deadline statements are as follows:

```
DEADLINE=(time,type[,date])
DEADLINE=(time,type[,rel,cycle])
```

3.28 JES3 global failure options

- System failure option
  - CANCEL - With Print
  - RESTART - From beginning of job
  - HOLD - Job Restarts
  - PRINT - Print job, operator hold
- Specifiable by:

User control Card	**//*MAIN FAILURE =**
Job Class	**CLASS, FAILURE =**
Initialization Standards	**STANDARDS, FAILURE =**

JES3 global failure options

The system failure option specifies the job recovery option that JES3 is to use when a system failure affects jobs.

CANCEL Print any job output that is in a SYSOUT class that is specified as TYPE=PRINT. After printing the output, cancel the job.

RESTART Restart the job from the first step. The job will be restarted on the processor on which it was active.

HOLD Place the job into the hold queue.

PRINT Print any job output that is in a SYSOUT class that is specified as TYPE=PRINT. Then place the job into the hold queue.

User control statements

A user can specify the failure option by using the //*MAIN control statement.

The installation can set default failure options by job class using the CLASS initialization statement or on the STANDARDS initialization statement.

3.29 JES3 inquiry (*i) commands

- *i a - Active jobs
- *i b - Backlog of jobs
- *i c - I/O buffers (spool)
- *i d,d= - Devices (DASD, tape, printers RJP)
- *i f,f= - Functional Subsystems (printers, CI)
- *i g,sysname,... - GMS - generalized main scheduling
- *i j= num | name - Inquiry jobno or jobname
- *i p= - Jobs at given priority
- *i q - Job queue
- *i s - Job setup resource management
- *i u .... - Output and jobs with output
- *i x .... - JES3 DSP info

JES3 inquiry commands (*i)

The **`*I`** command is used to obtain information about the processing status of given jobs, JES3 system devices, and resources. Information obtained can be categorized for all active jobs on the system, all jobs of a stated priority, all jobs submitted from a particular remote job processing (RJP) work station, or for all jobs that use a specified dynamic support program (DSP). Information concerning the status or environment of the processors, hardware, and job processing functions can also be obtained.

The **`*I`** commands can be used to display the status of JES3 resources, DSPs and jobs.

When you enter **`*I`** commands, you can specify he number of entries to display with the N= parameter. N=10 is the default if not specified. You can specify a number (nnnnnn) or ALL.

nnnnnn - specifies the number of detail lines (0 through 999999) to be displayed on the console or that all lines of the response to the **`*I`** command are to be displayed. If you omit the N= keyword, a maximum of ten lines is displayed.

3.30 Modify (*F) commands

- *f f,f= - Functional Subsystems (printers, CI)
- *f g,sysname,.... - GMS - generalized main scheduling
- *f j=num | name - Modify job
- *f config - Make configuration changes dynamically
- *f s - Job setup resource management
- *f u .... - Output and jobs with output
- *f x .... - Change DSPs and modules

JES3 modify (*F) command

The ***F** (modify) command is used to change the specified objects. The ***F** command allows the alteration of a job, its status, or some aspect of the JES3 environment.

The operator or system programmer issues modify commands to change a characteristic of a job or the processing of a JES3 functional area.

The modify command (***F**...) may be used to change the processing in any one of these JES3 functional areas:

- Generalized main scheduling
- Deadline scheduling
- Dependent job control
- Network job entry
- Consoles
- Spool data management
- Remote job processing
- Output service
- Converter interpreter service
- Main device scheduling

3.31 JES3 commands for DSPs

- JES3 commands for DSPs
  - *S - Start to a function
  - *R - Restart to a function
  - *C - Cancel to a function
  - *X - Start a DSP by calling
- JES3 commands to JES3 address space
  - *RETURN
  - *DUMP
  - *FAIL

JES3 commands

You communicate with JES3 dynamic support programs (DSPs) using JES3 commands. When a JES3 DSP initiates execution, it must identify itself to console service. You can refer to JES3 DSPs by the name or number of the device assigned to the dynamic support program.

Commands to DSPs

The ***C**, or ***R**, or ***S** command can be issued to a DSP to either cancel, restart, or issue a start to the DSP. JES3 commands are issued by the operator and consist of commands directed to various DSPs within JES3. JES3 commands can be entered from consoles attached to any system in the sysplex. The exception to this is the JES3 non-directed ***FREE** command which must be entered from an RJP console.

Start a DSP

DSPs that are invoked by an operator command (***X** dspname). These DSPs usually involve utilities, input and output services, and diagnostic aids. The parameters required for the execution of each DSP are entered as text with the ***X** command and subsequent ***S** commands.

Commands to address spaces

When a ***RETURN** or ***DUMP** command is issued the operator performs an orderly shutdown.

- If a ***RETURN** is entered, JES3 is terminated without a dump.

- If a **`*DUMP`** is entered as an operator command, the JES3 tables are formatted, and MVS takes a dump of JES3, or an unformatted dump is taken to the SYS1.DUMPxx data set, depending on how the DUMP=parameter on the OPTIONS initialization statement is specified, after which JES3 is terminated.

The operator issued a **`*FAIL`** command to end the specified DSP.

4

JES3 spool data management

In JES3 spool data management is defined as the recording and retrieval of data on the spool data set and the management of space within the spool data set.

The spool serves:

- For jobs in execution as a buffer between input devices and routines that read input data, and between routines that write output data and output devices
- As a storage place for the control blocks and data that JES3 needs to process jobs.

A spool must be allocated on a DASD device. Transfer of job data to and from the spool device is called spooling. Spooling refers only to use of a DASD devices by JES3 for storage of jobs and job-related data. Use of the same devices for other purposes is not spooling.

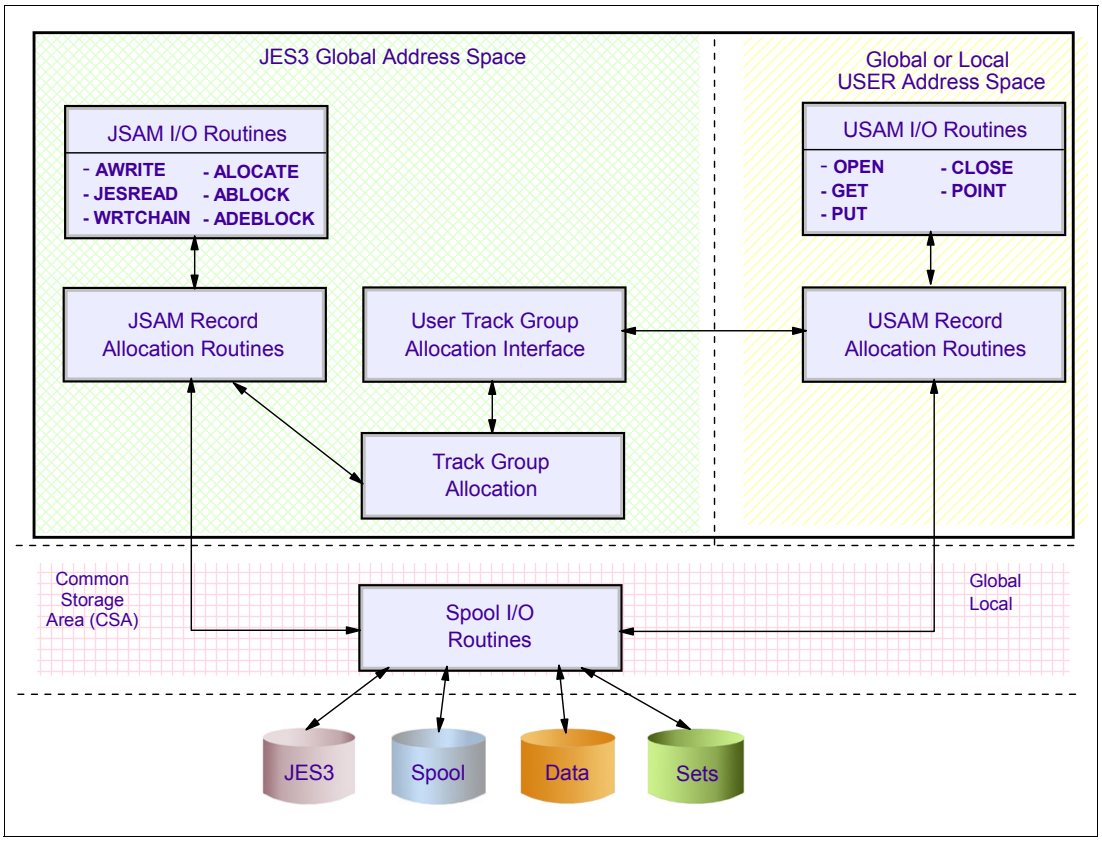
Spooling allows jobs to read SYSIN or write SYSOUT data at or near the full speed of DASD devices.

JSAM is a collection of routines invoked by DSPs by macro calls to acquire spool space, create spool files, read and write records, and to purge SRFs and MRFs when they are no longer needed. JSAM buffer pool management procures space for SRFs and MRFs in JES3 private storage.

The single track table is a section of spool space used exclusively for JES3 control blocks not associated with a particular job, such as control blocks used to track JES3 functions and to save status. The allocation mechanism of the single track table is by record as opposed to track group, in contrast to the rest of JES3 spool space allocation.

Dynamic support programs (DSPs) perform the work that JES3 is required to do in order to run jobs. The DSPs that use the JES3 spool access method (JSAM) run in either a CI FSS or JES3 address space. A DSP needs to access a job's control blocks that are found on spool.

4.1 JES3 spool I/O overview



JES3 spool I/O overview

The spool data management function controls the JES3 spool and data. The term "spool" used in the JES3 environment is an acronym for simultaneous-peripheral-operations-online and refers to that set of direct access storage devices (DASD) especially set aside for JES3. Spool data management routines control the allocation, access, and deallocation of space on these designated direct access storage devices (DASD).

JES3 spool access topics

Spool data management can be divided into three topics:

1. Spool space management, which deals with the allocation of space on JES3 DASD.
2. The access methods:
   a. The JES3 spool access method (JSAM)
   a. The user spool access method (USAM)
   a. The block spooler
3. Spool I/O scheduling.

Spool space

The management of spool space involves the assigning of spool space to jobs and JES3 functions and the recording of the ownership of that space. Spool space comprises data sets on specified DASD assigned to JES3. The summary below examines the different gradations of JES3 spool space from the smallest to the largest segment.

- A record is a physical block on a disk; the single smallest unit of spool space.
- A track group is a contiguous group of records within a spool data set. The track group is the basic unit of spool space management. (The JES3 initialization stream provides options to control how spool space will be allocated and options to define the size of a track group.)
- A spool data set (extent) consists of contiguous tracks on DASD devices. Each data set is generally an entire DASD volume.
- A partition is a group of spool data sets. The partition is used by spool space management to manage track groups within the data sets.

JES3 spool access methods

The spool access methods have many common characteristics, although the actual mechanism used to get into the allocation function is different for each of the three access methods. They all use the base unit of spool space - the track group; they all associate data sets with named partitions. When a track group is allocated in each case, the partition must be specified.

JES3 two techniques for recording spooled data: single record files and multi-record files. Control blocks are recorded as single-record files (SRFs). This means that a control block is recorded as a single buffer image (a block of data on spool). While most of the control blocks will fit into a single buffer image, some (due to their variable lengths and multiple parts) may extend across multiple buffers. When this occurs, the buffers are chained together into chained single record files.

Multi-record files (MRFs) consist of JCL, SYSIN, and SYSOUT recorded as data records, placed into spool buffers, chained together, and written to spool.

The recording and retrieval of spooled data are the responsibilities of three special access methods:

1. JES spool access method (JSAM), used by JES3 modules and user-written DSPs to read and write SRFs and MRFs.
2. User spool access method (USAM), used by programs in the user address spaces to read SYSIN data and write SYSOUT data.
3. The block spooler, is used to:
   a. read and write spool data in a writer FSS address space
   a. read the LVS for locate subtasks
   a. read and write the JST for MDS subtask purposes.

Spool I/O routines

JES3 initiates I/O to the spool device by the STARTIO macro, and thus JES3 routines use directly the services of the MVS input/output supervisor (IOS). The JES3 routine that schedules spool I/O operations is common to JSAM, USAM, and the block spooler. This routine is responsible for initiating and terminating I/O requests. When JSAM requests are completed, the common routine notifies JSAM to signal the I/O completion to the appropriate DSP. USAM and block spooler I/O completions cause the routine to notify the appropriate user address space that I/O activity is complete.

Spool space is allocated to a user's address space in track groups. The job track allocation table (JBT) is used to hold the track groups allocated to a job.

4.2 JES3 spool environment

- JES3 spool extents
  - All job related information
    - Job control blocks
    - SYSIN data sets
    - SYSOUT data sets
- JES3 JCT data set
  - JCT record for each job
  - A separate MVS data set from spool extents

JES3 spool environment

JES3 allocates space on spool to a job upon entry to the system and deallocates spool space when the job leaves the system. An initial quantity of spool space is allocated to hold job-related control blocks and a job's SYSIN and SYSOUT data. Additional spool space may be required during the execution of a job to hold SYSOUT data.

JES3 records several types of data on the spool data sets:

- Information (originally taken from initialization statements) necessary to initialize JES3 in the global and local processors.
- JES3 control blocks that define the scheduling and operational characteristics of the jobs.
- SYSIN (DD * or DD DATA) data sets and SYSOUT data sets for jobs.

JCT data set

The job control table (JCT) is a major JES3 control block on a DASD device. The data set is defined in the JES3 start procedure. The DDNAME is JES3JCT. The JES3 JCT data set does not reside within the DASD space allocated for the spool extents and is allocated by MVS during JES3 start.

The JCT data set contains a JCT entry or every job in the JES3 complex. Many JES3 functions, such as inquiry and modify, access the JCT. But the JES3 function that uses the JCT the most is the job segment scheduler (JSS).

4.3 Spool volumes

- Up to 32767 definable
- Mixed device types supported
- Define contiguous extent
  - Specify cylinders - SPACE = (CYL,xxx)
  - Can be greater than 65,535 tracks
    - Requires DSNTYPE=LARGE on the DD statement
- Understand geometry of device
  - Spool space allocation - GRPSZ
- Place on lightly loaded DASD strings
- TRACK or FORMAT initialization statements

```
*I Q SP=ALL
IAT8980 DRAINED  HAS NO SPOOL DATA SETS
IAT8980 UNAVAIL  HAS NO SPOOL DATA SETS
IAT8509 JES3PART: 215,295 GRPS, 205,719 LEFT ( 96%); MIN  5%, MRG  5%, DEF, INIT
IAT8607 INQUIRY ON SPOOL PARTITION STATUS COMPLETE
```

Spool extents

The JES3 spool consists of one or more single extent DASD data sets. Up to 32,767 data sets (spool extents) can be defined. Mixed device types are supported. Each data set is typically a full volume that starts and ends on a cylinder boundary. The data sets are logically divided into track groups -- the JES3 allocation unit of spool space. Since track groups, specified using the GRPSZ keyword on the BUFFER statement in the initialization stream, is the space allocation unit for jobs, it is important to understand the device geometry.

The volume on which a spool data set is allocated must be accessible to the global processor and to all local processors. Each spool data set must be contained in a single extent. (A single extent is one adjoining group of tracks or cylinders.) You cannot allocate any secondary extents.

To avoid degradation of JES3 performance and possible lockouts, do not allocate more than one spool data set per volume. When a volume contains more than one spool data set, the average seek time to access the data increases. Similarly, do not allocate data sets to a volume that JES3 rarely accesses.

Spool space GRPSZ

A track group is the number of records that JES3 treats as a unit when allocating spool space. You can specify the size of a track group using the GRPSZ parameter, choosing a value from 1 to 999.

Spool placement

Spool performance is important for overall JES3 performance, so place the spool volumes on the same DASD string and on a lightly used string in relation to any other non-spool devices on the same DASD string.

Spool partitions

A spool partition is a logical grouping of spool data sets. You control five factors:

- The number of spool partitions used
- The number of spool data sets that are in each spool partition
- The work load distribution across spool partitions
- The type(s) of spool data to be included in each spool partition
- The size of a track group for each partition

These factors influence the reliability, availability, and serviceability (RAS) of spool data sets and the performance impact of accessing a spool data set. Spool partitioning allows you to isolate different types of work in specific partitions. Isolating spool data in separate partitions can help you improve spool performance, spool recovery procedures, and spool space management.

To provide for occasions when a requested spool partition is full, you can specify where each spool partition's overflow data should go. To do this, use the OVRFL parameter on the SPART initialization statement.

For each spool partition, you can specify the spool extents that are to be included into that partition. You do this by specifying the name of the spool partition on the FORMAT or TRACK statement associated with the spool extent. A spool extent can be included to one spool partition. If you do not specify spool partitions, JES3 defines a one and names it to JES3PART. All spool extents are included into this spool partition.

JES3 data sets that belong to the default partition contain:

- Control blocks
- SYSIN data for jobs
- Single track table (STT) expansion
- Initialization data

*INQUIRY,Q

Use the *INQUIRY,Q command to display among other things:

- The size of the spool partitions and the amount of space currently available
- The amount of space available on all the JES3 spool data sets in the complex

4.4 Defining spool space allocation units

```
BUFFER    ,BUFSIZE = nnnn | 4084
          ,PAGES = ( {global} {,local} {,cifss} )
          ,GRPSZ = nnn | 30
          ,MINBUF = nn | 16
          ,SPLIM = ( {min} , {marg} )
          ,TRUNC={YES|NO}
```

Defaults:

BUFFER,BUFSIZE = 4084,PAGES = (1024,128,512),

GRPSZ =30,MINBUF=16,SPLIM=(10,25),TRUNC=YES

- Take defaults for others
- MINBUF = 16,SPLIM = (10,25)

JES3 BUFFER initialization statement

The basic unit of spool space allocation is called a track group. A track group is a group of spool records, with the size of each spool record equal to the size of a JES3 buffer. JES3 buffer size is defined by the BUFSIZE parameter on the BUFFER initialization statement You can define the number of spool records in a track group using the GRPSZ parameter on the BUFFER statement or on the SPART statement.

BUFSIZE parameter

The size of records on the spool data sets (except the JCT) is defined by the **BUFSIZE** parameter on the **BUFFER** initialization statement, and it may range from **1952** to **4084**. Each buffer is preceded by 12 bytes of control data, and the buffer and its prefix must be totally contained within a single page of virtual storage. The recommended buffer size is 4084 bytes.

Device	Record Size	Records/Track	Records/Cylinder
3380	4K	10	150
3390	4K	12	180

JES3 allocates one or more track groups to a job when the job needs spool space. The number of track groups allocated on each request depends on the number defined using the TRKGRPS parameter for the job's SYSOUT class, //*MAIN JES3 control statement, job class, or assigned processor.

PAGES parameter

The amount of virtual storage allocated to buffers is specified by the **PAGES** parameter, and acquired dynamically (AGETMAIN) at JES3 initialization. If the number of available buffers

goes below a value calculated from the **MINBUF** parameter, JES3 identifies the condition by adding a user-specified identifier to all console messages. JES3 uses buffers to read and write data from and to spool. A buffer is an area of storage that is temporarily reserved for use in performing an input/output operation, into which data is read or from which data is written.

GRPSZ parameter

The **GRPSZ** parameter on the BUFFER and SPART initialization statements Specifies the number of spool records in each track group. (The BUFSIZE parameter determines the size of each spool record.) The number must not be greater than 999. JES3 rounds the specified value up to the number of records in the nearest whole physical track for the selected spool device type. Track group specification is important for spool space utilization. Each job entering the system is assigned initially a track group and during its life in the system is assigned additional track groups. Do not over specify this parameter. The default value is okay.

MINBUF parameter

MINBUF specifies the value that JES3 uses to calculate the acceptable minimum number of free JSAM buffers. The value you specify can be any decimal integer between 8 and 64, inclusive.

SPLIM parameter

The **SPLIM** parameter on the BUFFER and SPART initialization statements specifies the minimum and marginal percentages of spool space still available in active spool partitions. An active spool partition is one containing at least one spool data set. If the minimal or marginal percentages of spool space are reached, indicating that a spool partition is nearly full, JES3 issues action messages to the operator.

- **min** -- When a spool partition reaches a minimum spool space condition, JES3 issues a message stating that this condition has occurred. The message alerts the operator to inquire whether the spool partition automatically overflows into another partition. If the spool partition does overflow, no operator action is required. Otherwise, the operator can use JES3 commands to take appropriate actions.If a minimum spool space condition arises on the default spool partition, JES3 suspends all SYSOUT buffer processing. JES3 does not resume SYSOUT buffer processing until enough spool space is freed to reach a marginal spool space condition.
- **marg** -- When a spool partition reaches a marginal spool space condition, JES3 issues a message stating that this condition has occurred. The message alerts the operator to inquire whether the spool partition automatically overflows into another partition. If the spool partition does overflow, the operator need not take any action. Otherwise, the operator can use JES3 commands to take appropriate actions. When a marginal spool space condition arises, job selection is suspended.

TRUNC parameter

The **TRUNC** parameter on the BUFFER and SPART initialization statements specifies whether or not you want JES3 to truncate trailing blanks from all SYSOUT data that is produced in the complex. YES Indicates that you want JES3 to truncate trailing blanks from all SYSOUT. NO Indicates that you do not want JES3 to truncate trailing blanks from all SYSOUT.

Note: JES3 saves the original record length of a record, prior to truncation of X'40' bytes, allowing applications that retrieve the record via the functional subsystem interface to reconstruct the original record. This is primarily of interest for files in which the X'40' character is considered a significant data character, rather than a blank.

4.5 Operator command for buffer usage

- JSAM buffers in use

```
*I C
IAT8506 JSAM BUFFER USAGE
IAT8725 TOTAL NUMBER OF JSAM BUFFERS ............ 01000
IAT8508 CURRENT NUMBER IN USE ................... 00041
IAT8510 MAXIMUM NUMBER USED ..................... 00292
IAT8722 PRIMARY EXTENT SIZE ..................... 01000
IAT8723 SECONDARY EXTENT SIZE ................... 00500
IAT8724 SECONDARY EXTENTS ALLOWED ................ 0004
IAT8501 CURRENT SECONDARY EXTENTS IN USE ......... 0000
IAT8512 NUMBER OF AWAITS FOR AVAILABLE BUFFER .... 0000
```

JSAM buffers in use

Use the **`*I C`** command to display the current status of the JSAM buffer pool. The display includes statistics such as current and maximum use counts for these pools.

Secondary extent size

JES3 automatically expands the size of the JSAM buffer pool when it detects a buffer shortage. For each expansion, JES3 increases the size of the buffer pool by one half of the value you specify on this parameter up to a maximum of four times or 32,767, whichever occurs first. For additional information about determining the size of the JSAM buffer pool.

When JES3 determines that all of the buffers in the primary allocation have been exhausted, it automatically expands the buffer pool by allocating additional buffers. Although JES3 provides a secondary allocation of buffers each time an out-of-buffer conditions exists, JES3 will expand the buffer pool up to a maximum of 4 times or 32767 bytes, whichever occurs first. The number of buffers that JES3 allocates for each secondary allocation is one-half the number specified on the PAGES= keyword of the BUFFER initialization statement.

JES3 frees the secondary allocation(s) of buffers when the number of available buffers is greater than or equal to the number of buffers in the secondary allocation plus the acceptable minimum number of JSAM buffers. To find the minimum number for your installation, divide the total number of JSAM buffers (the primary allocation buffers from the PAGES= keyword plus all secondary buffer allocations) by the MINBUF= parameter.

4.6 Selecting GRPSZ value

- Select **GRPSZ** based on
  - Spool space used by typical job
  - Select number on track boundary
- Spool OUTPUT of a JOB
  - All control blocks - JDAB,JMR, etc.
  - SWA
  - Input files
  - JESMSGLG, JESYSMSG, JESJCL
  - Output data sets
  - J3SCINFO, J3JBINFO, JCBLOCK, JOURNAL

Selecting GRPSZ value

Make sure that the track group size (GRPSZ) is on a track boundary. The default of 30 is 3 tracks on a 3380 device. The default track group size of 30 allocates approximately 120K bytes of data. Again, the default value of 30 is adequate

Note: Current DASD volumes are type 3390. Using the default of 30 would cause it to be rounded up to 36 as the 3390 has 12 records per track.

GRPSZ value and a typical job

The GRPSZ value should correspond to the spool space requirements of a typical job in the JES3 complex. For example:

- If the workload consists mainly of small jobs, specify a GRPSZ value of less than half the number of records per cylinder for the spool device type. The small value uses spool space efficiently and reduces access time on moveable head devices.
- If the workload consists mainly of jobs producing much output, specify a GRPSZ value of roughly the number of records per cylinder. The large value reduces the number of requests for additional spool space and thus reduces the amount of time the job must wait to be allocated additional space. (The number of allocation requests also depends on the track group allocation size defined by the TRKGRPS parameter.

If you have defined spool partitions to isolate different types of data, specify a "tailored" GRPSZ value on the appropriate SPART statement. Tailoring the GRPSZ value this way

helps especially when part(s) of the installation's workload requires much less or much greater spool space than average.

Output on spool

Records on JES3 spool data sets are processed by JES3 routines running in either the JES3 address space on the global processor, user address spaces on either the global or a local processor, or FSS address spaces on the global or a local processor. Information held on spool data sets includes:

- JES3 control blocks
- In-stream data sets (DD *, DD DATA, //*DATASET)
- SYSOUT data sets
- Job journal
- JESMSG, JCLIN, JESJCL, JCBLOCK, and SYSMSG data sets
- J3SCINFO (DFSMS scheduling information)
- J3JBINFO (DFSMS job information)
- JES3 job queue

4.7 GRPSZ selection

BUFFER,.....,GRPSZ = 36

- 3390 - 4K records - 12 records/TRK -180 records/CYL
- 3 Tracks of Data - 36 Records - 3390

Record 1											Record 12
Record 13											Record 24
Record 25											Record 36

- 36 records at 4084 bytes = approx.144K Bytes

Selected GRPSZ

The selected GRPSZ on the BUFFER statement in the visual is 36. This is not the default. On a 3390 spool device, this represents 3 tracks with 12 records on a track. Therefore, each track group when assigned to a job will have 30 spool buffers written to it. These spool records can be control blocks, SYSIN data, SYSOUT data, and other control information to the job.

The GRPSZ is the number of spool records in each track group. (The BUFSIZE parameter determines the size of each spool record.) The number must not be greater than 999. JES3 rounds the specified value up to the number of records in the nearest whole physical track for the selected spool device type.

Note: If you change the group size parameter, you must perform a warm start. See z/OS JES3 Commands for information about how to replace a spool data set (WR).

If you omit this parameter, the default value is taken from the GRPSZ parameter of the BUFFER statement. If omitted from both statements, the default is 30.

If you specify an invalid subparameter, JES3 uses the parameter default.

4.8 Single track table

- Spool space for small jobs (DSPs)
- Specify on TRACK | FORMAT statement
  - STT=(cylnum,cylnum)
  - STTL={(cylnum,numtrkgps)}
- Default STTL
  - 2 track groups in center of default partition
- Display STT - *S DC OPTION=STT
  - May create voluminous output

Single track table

The single track table is a section of spool space used exclusively for JES3 control blocks not associated with a particular job, such as control blocks used to track JES3 functions and to save status. The allocation mechanism of the single track table is by record as opposed to track group, in contrast to the rest of JES3 spool space allocation.

There are two types of STTs used in the JES3 environment:

- The job control table (JCT) STT
- The system STT

Job control table single track table (JCT STT)

The job control table single track table (JCT STT) is built during JES3 initialization. The JCT STT is a fixed length table and contains a bit map used to allocate JCTs.

System single track table (STT)

The system STT is used to allocate single record files (SRFs) on a individual record basis. The system programmer specifies the spool data sets that contain the STTs to be allocated using the STT= or STTL= parameter on the TRACK or FORMAT statement. The first system STT is called the primary STT while STTs that are built later are called expansion STTs. The primary STT and any expansion STTs are linked together to form a single linked list.

```
FORMAT,DDNAME=SPOOL1,STT=(400,401)
FORMAT,DDNAME=SPOOL2,STTL=(400,2)
```

STT Specifies the range of cylinders you want allocated to the single track table (STT). This range must be within the extent allocated to the data set. The value of *cylnum* specifies an absolute cylinder number. (Absolute cylinder numbers are device-dependent; the component description for the device describes the numbering scheme.)

STTL Specifies the location and number of **track groups** to allocate to the single track table (STT). These track groups must be within the extent allocated to the data set. The value for *cylnum* specifies an absolute cylinder number indicating the beginning cylinder number of the STT allocation in this extent. (Absolute cylinder numbers are device-dependent; the component description for the device describes the numbering scheme.) The value for *numtrkgps* specifies the number of track groups to allocate to this extent, beginning with the first track group that is located completely in cylinder *cylnum*. The maximum number of track groups that may be allocated to the STT is 9999.

STT dynamic expansion

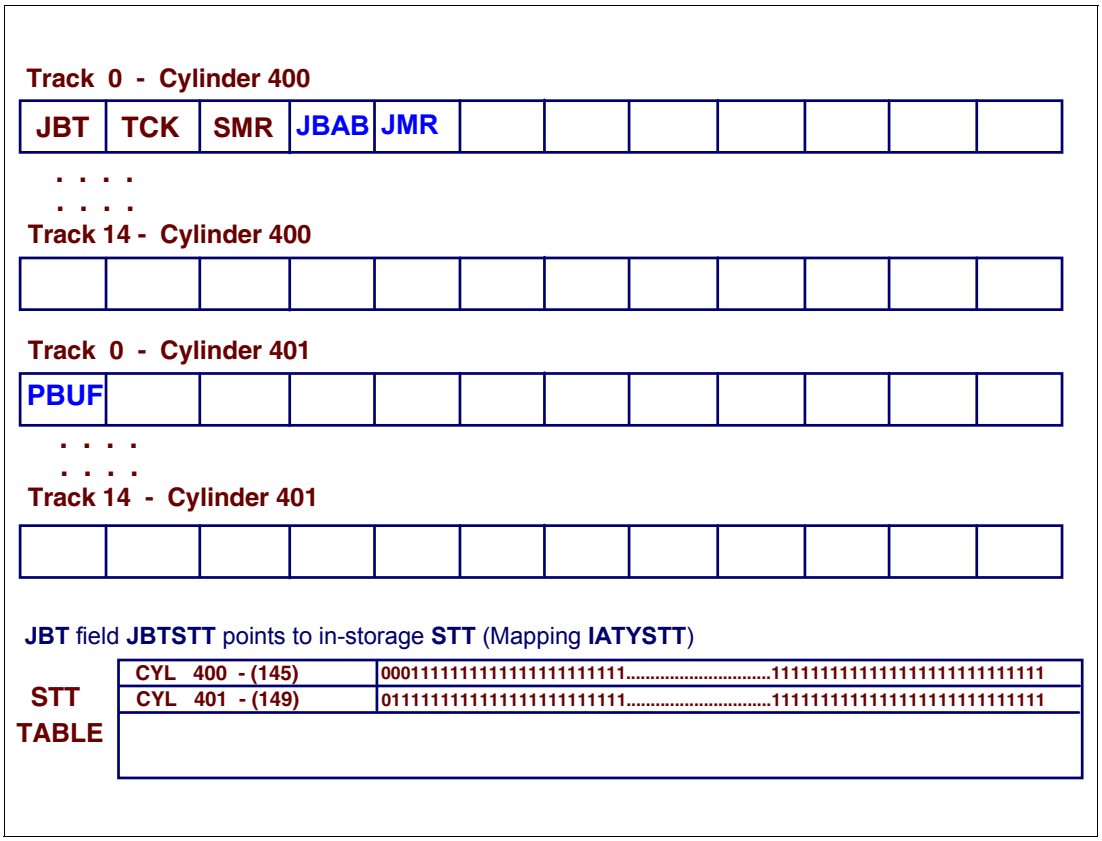
If all STT space is used, the STT is dynamically expanded. STT space is accounted for on a record basis via a control block called the STT. In the STT control block, there is one bit for each STT record.

Replacing or deleting a spool data set

If you replace a spool data set, JES3 cancels all jobs with data on the replaced spool data set. If the replaced data set contains STT records, JES3 might lose information that could result in the loss of jobs in the system. STT records include information such as the status of devices, DJC network data, deadline queue data, volume unavailable data, dynamic allocation checkpoint data, output service checkpoint data, JESNEWS, device fencing data, virtual unit status, GMS status, and FSS checkpoint data. If STT data is lost, JES3 issues messages that allow you to take the appropriate recovery actions.

If you delete a spool data set, JES3 cancels all jobs in the system that have spool data or allocation tables on the affected data set. Try not to delete a data set that contains important information (for example, the single track table (STT) or the JESNEWS data set). If this information is lost, the system issues messages giving you the opportunity to take appropriate actions.

4.9 Single track table allocation



Single track table allocation

Shown in the figure are the tracks for 2 cylinders of STT space with 15 tracks in each cylinder and 10 records on a track. The STT table for the STT space is shown below the tracks. This STT table is in storage for contains a bit for each record in the 2 cylinders that are allocated. When a bit is on (1) the record is not allocated. When a bit is off (0) the record is allocated and has data in it.

This figure shows the allocation of a called DSP and its three spool control blocks, the JMR, JDAB, and PBUF or PARMS. The STT table consists of two cylinders and the three bits in the STT table are off (zeros) representing the allocated space. When the DSP job purges, the space is returned to the STT and is available for use by another user of STT track space.

The IATYSTT macro maps the STT table.

Writing a single record file to spool

A DSP issues an AWRITE macro to write a SRF to spool. For an SRF, a spool record address is allocated from either the space allocated to a job (the addresses for this spool space is found in the job's JBT) or from the system single track allocation table (STT). Records that have previously been written to spool are written to the same record address.

4.10 Spool data management access methods

- 3 access methods
  - JES3 spool access method - (JSAM)
    - JES3 data management used by JES3 modules and user-written DSPs to read and write SRFs and MRFs
  - User spool access method - (USAM)
    - JES3 data management routines used by programs in user address spaces to read SYSIN data and write SYSOUT data.
  - Block spooler - (Read spool for PSF)
    - The block spooler enables spool access from a JES3 subtask or other address space that cannot use JSAM services to access spool

Spool data management access methods

JES3 uses two techniques for storing spooled data: single record files and multi-record files. Control blocks are recorded as single-record files (SRFs). This means that a control block is recorded as a single buffer image (a block of data on a spool volume). While most of the control blocks will fit into a single buffer image, some (due to their variable lengths and multiple parts) may extend across multiple buffers. When this occurs, the buffers are chained together into chained single record files.

Multi-record files (MRFs) consist mainly of JCL, SYSIN, and SYSOUT data records, placed into spool buffers, chained together, and written to spool.

The recording and retrieval of spooled data are the responsibilities of three special access methods:

- JES spool access method (JSAM), used by JES3 modules and user-written DSPs to read and write SRFs and MRFs.
- User spool access method (USAM), used by programs in the user address spaces to read SYSIN data and write SYSOUT data.
- The block spooler. The block spooler is used in FSS address spaces to read data buffers for PSF printers.

Any main in the complex can perform I/O to a spool data set. Requests for spool space can originate from the global or a local, but only the global manages spool space.

JSAM

JSAM is a collection of routines invoked by DSPs by macro calls to acquire spool space, create spool files, read and write records, and to purge SRFs and MRFs when they are no longer needed. JSAM buffer pool management procures space for SRFs and MRFs in JES3 private storage.

The single track table is a section of spool space used exclusively for JES3 control blocks not associated with a particular job, such as control blocks used to track JES3 functions and to save status. The allocation mechanism of the single track table is by record as opposed to track group, in contrast to the rest of JES3 spool space allocation.

USAM

USAM provides user access to SYSIN data and the creation of SYSOUT data sets. MVS data management access method macros (BSAM, QSAM) pass control to JES3 by the compatibility interface to allocate spool space for output data sets. USAM thereby provides for the opening, closing, reading, and writing of data sets to JES3 spool.

USAM buffer pool management involves user buffer pools. The user storage buffer pools are inside the user address space and consist of either one page of virtual storage for a SYSIN data set or multiple pages for SYSOUT data sets. The contents of user storage buffers are not transmitted directly to or from spool volumes, but rather are moved to or from the USAM protected buffer pools that reside in the common service area (CSA) or in a JES3 auxiliary address space (AUX).

Block spooler

The block spooler enables a program to access spool data sets from a JES3 subtask or other address space that cannot use JSAM services to access spool.

For example, writer FSS input routines invoke the block spooler to read blocks of SYSOUT data from spool and to read and write writer checkpoint records that are used by FSS writers. The block spooler reads data one track at a time (when possible) into buffers in the USAM buffer pool.

The block spooler, is also used to read the LVS for locate subtasks and read and write the JST for MDS subtasks.

4.11 JES3 spool access method - JSAM

- JES3 nucleus modules for creation and access
  - All spool files
    - Control blocks - Single record files (SRF)
    - SYSIN / SYSOUT files - Multiple record files (MRF)
    - JCT record access
- Resident FCT for posting I/O completion
- Spool access control block - FDB
  - Required to do spool I/O

JES3 spool access method - JSAM

A JES3 address space or C/I FSS address space uses JES3 spool access method (JSAM) to create new single-record files (SRFs) and multi-record files (MRFs) or to access existing SRFs and MRFs containing job-related and system-related control blocks and data sets.

All spool files which include SYSIN, SYSOUT, job control blocks, and writing and reading JCT records is done using the JSAM access method in the JES3 address space.

JSAM resident FCT

In the resident FCT table is the JSAM FCT which:

- Processes track group allocation requests from user, local, or CI/FSS address spaces.
- Schedules the spool I/O error FCT.
- Calls the track group bypass table update routine.
- Returns buffers to the buffer pool and initiates I/O for a JSAM I/O request.
- schedules the tape and DASD DDR FCTs.
- Issues the JSAM buffer shortage message.
- Frees secondary extents of the JSAM buffer pool.
- Issues spool partition related messages.

The JES3 address space or a C/I FSS address space uses JSAM to read and write data in the form of SRFs and MRFs. JES3 maintains a file directory (FD) which contains a list of all

the open MRFs, and any SRF requests waiting for spool space or buffers. Each entry in the FD contains information on the open file, such as:

- The address of the FDB for the open file
- The address of the JBT for an open SYSOUT file
- Address of the FCT that issued the I/O request
- A flag to indicate the status of any I/O

JSAM I/O completion

JES3 spool data management routines initiate spool I/O which requests directly the MVS I/O supervisor (IOS) to perform the actual I/O operation. The JES3 spool data management routines provide a compatibility interface to aid in the processing of spool I/O. Spool I/O is asynchronously performed with other JES3 functions.

When IOS has processed all the buffers in the I/O request, a JES3 I/O termination routine, called a disabled interrupt exit (DIE), is called to begin termination processing for the I/O request. If the I/O request completes successfully, MVS calls termination processing for the queue of buffers. JES3's JSAM I/O termination for a single record file request updates the file's FDB and then adds the FCT that issued the I/O request to the FCT Ready Queue.

The termination routines run in either a JES3 or CI FSS address space. If the buffer is for a JSAM, block spooler or USAM I/O request, the termination routine will run in a JES3 address space. For CI JSAM requests, the termination routine will run in a CI FSS address space.

FDB control block

A file description block (FDB) is used by JSAM and describes a single record file (a control block) or a multi-record file (SYSIN or SYSOUT data). An FDB contains the spool record address of the data or, if the file is in storage, the address of the buffer containing the in-storage spool record.

Job and data set track allocation tables (JBTs) are SRFs used by the JES3 global address space to record spool space ownership.

JCT and JQE access method

Two facilities are provided for accessing and deleting a job from the JES3 job queue: the JCT access method and the JQE access method. Module IATGRJX services requests to access the JQE and JCT structures. The JCT access method, through the IATXJCT macro, allows a DSP to:

- Add an entry to the JCT data set and JCT data space
- Read in a JCT for a specific job.
- Release a JCT entry previously read in using the access method.
- Delete a JCT entry.

The JQE access method, like the JCT access method, provides access to job information through the IATXJQE macro.

4.12 File description block (FDB)

- SRF- FDB (12 bytes)
  - XL(FDBSRFL)'00'
- MRF-FDB (32 bytes)
  - XL(FDBMRFL)'00'
- Special FDB - JBTAT-FDB (28 bytes)
  - XL(FDBJBTL)'00'
- IATYFDB - File Description Block mapping

File description block

A file description block (FDB) is used by JSAM and describes a single record file (a control block) or a multi-record file (SYSIN or SYSOUT data). An FDB contains the spool record address of the data or, if the file is in storage, the address of the buffer containing the data.

SRF FDB

SRFs are created by the DSP acquiring a buffer, filling the buffer from byte 12 onwards, building a 12 byte FDB with the buffer address in the first 4 bytes, then issuing the AWRITE macro.

MRF FDB

JSAM uses a 32-byte control block called the FDB to process MRFs. This block is built or made available by the requesting DSP. It holds information regarding the status of the MRF, and is updated by JSAM as processing proceeds.

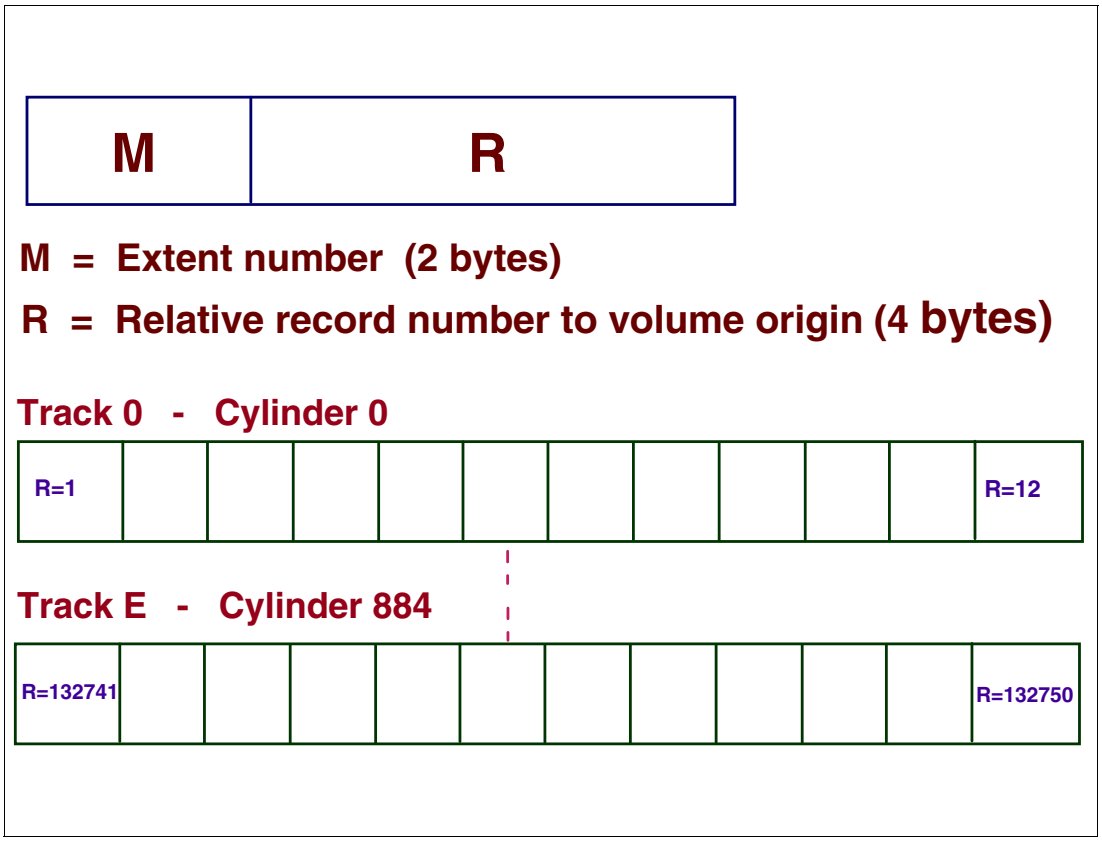
JBTAT FDB

The JBTAT-FDB is a special FDB for the spool track space allocation. While the JBTAT is a single record file, the FDB is 28 bytes because it has additional information about the file itself.

IATYFDB

File description block mapping macro.

4.13 Spool record address - M.R



Spool record address - M.R

JES3 addresses spool records by using its own addressing scheme. Each spool record in the spool environment has its own unique record address and they are assigned sequentially to each record. The spool record address, (M.R), is 6 bytes as follows:

M This is the 2 byte extent number. If you have 2 spool volumes or extents, then this number is 2 or 3. Spool volume extent number 1 is always the JCT spool extent.

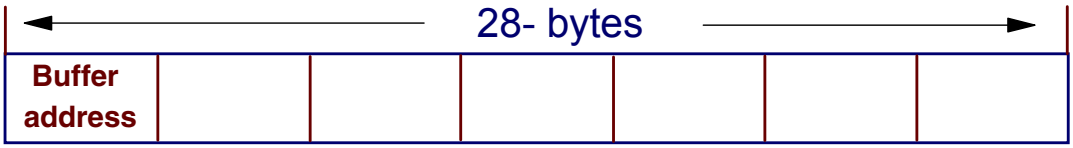
R This value is a number relative to the spool volume origin. On track 0 cylinder 0 on the spool volume, the R value is 1 and the numbers grow consecutively in ascending order to the last cylinder and track on the spool volume.

Spool writes

JES3 can write either JES3 control blocks or sequential job SYSIN and SYSOUT data to spool. JES3 usually records control blocks as single record files (SRFs) that occupy a single spool record. If a control block is large and requires more than one spool record, another record is allocated and is chained to the previous spool record. JES3 records SYSIN or SYSOUT data as multiple record files (MRFs). For large amounts of data, a MRF allows more efficient packaging of data than SRFs. An MRF consists of a set of logical records that occupy single or multiple physical records. These records are chained together with backward and forward pointers.

4.14 Spool space allocation - JBT

- JBT - FDB is for a SRF
  - Includes an extension for M.R allocation
- Input service creates for each job
  - Contains information of all spool space owned by a job



(when buffer in storage)

```
        AGETBUF                    Get spool buffer
        ST    R0,WATAT             Store in FDB.
        .....
        AWRITE   FDB=WATAT,TATPTR=WATAT,ID=JBT
WATAT   DC    XL(FDBJBTL)'00'      JBTAT work fdb
```

Spool space allocation block - JBT

This file is built during input service processing, the job track allocation table (JBTAT) is a list of spool track groups allocated to a job. The table is updated to reflect any subsequent allocations as required. During purge processing, track groups are given back to the system for reassignment to other jobs.

JBTAT contents - X.Gs

Job and data set track allocation tables (JBT) are used to contain information about the space allocated to jobs, spin-off data sets, single track tables, and so on. Spool space is allocated to a job or data set in track groups in the form of X.Gs (track group addresses) in the job. (The 2 byte X portion of the X.G's represent a spool extent, the 4 byte G is the track group number within the extent.) The X.Gs allocated to a job or data set are stored in entries in the variable section of the JBT. If all the entries in the JBT are filled, a new JBT record is created and chained to the previous record.

Tip: M.R conversion to X.G is based on the following computation:

```
'X' = 'M'
'G' = 1 + ('R' - ‘R_OF_1ST_RECORD_IN_EXTENT) / NUMBER_OF_RECORDS_PER_TRACK_GROUP
```

JBTAT FDB

Each JBT is represented by a 28 byte FDB which contains the following information:

- The spool record address (M.R) of the JBT or its buffer address if the JBT is in storage

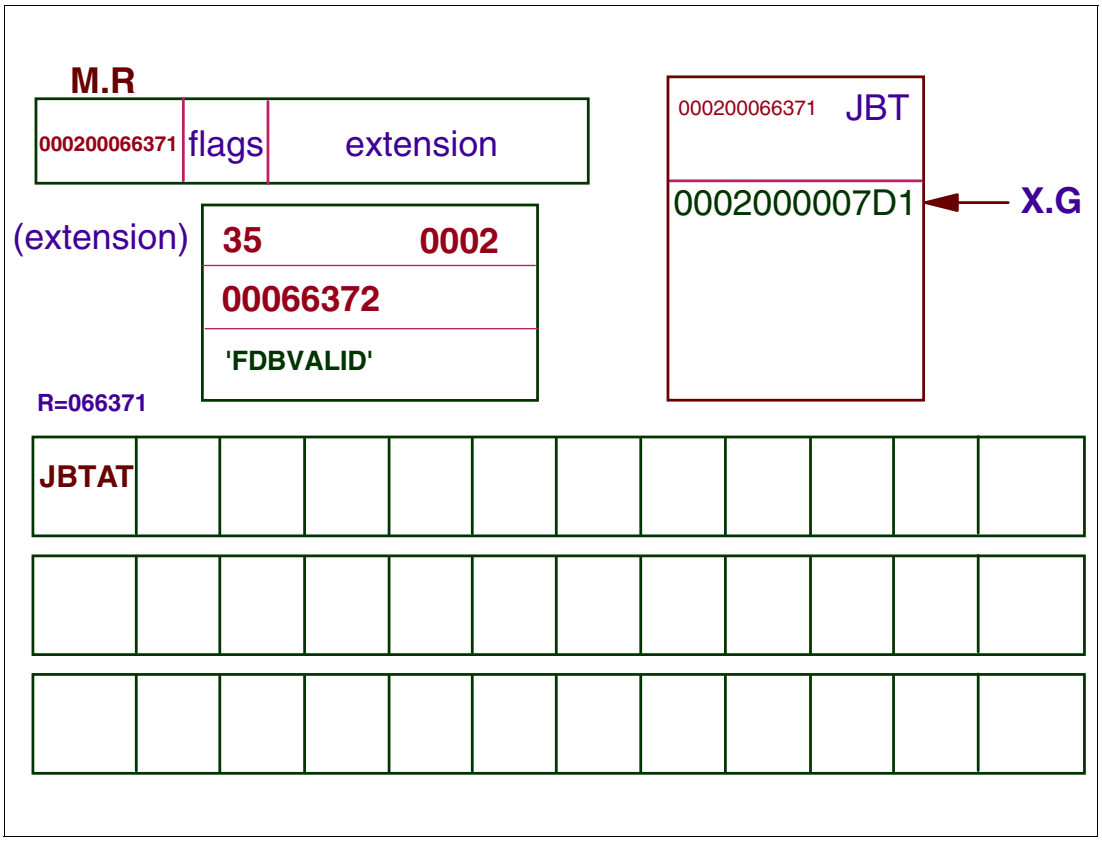
- An available track group that contains a count of the number of available records and the next available record address
- The spool partition index
- Status flags
- A validation field used for multi-record files

JBTAT creation

The JBTAT is obtained during input service. It will eventually contain all the spool space owned by the job. It is written to the spool using the AWRITE macro and the spool space used to write it to the spool comes from the space owned by the job (the first spool record address in the first track group owned by the job).

A user's address space allocates spool space from a record allocation block (RAB). The RAB contains the spool record addresses (M.Rs) used to write buffers to spool. The job track allocation table (JBT) is used to hold the track groups allocated to a job.

4.15 JBT description



JBT description

Spool space management functions keep track of the ownership of spool space. To accomplish this, each track group has its own unique 6-byte address that consists of the extent number and the track group sequence number within the spool data set.

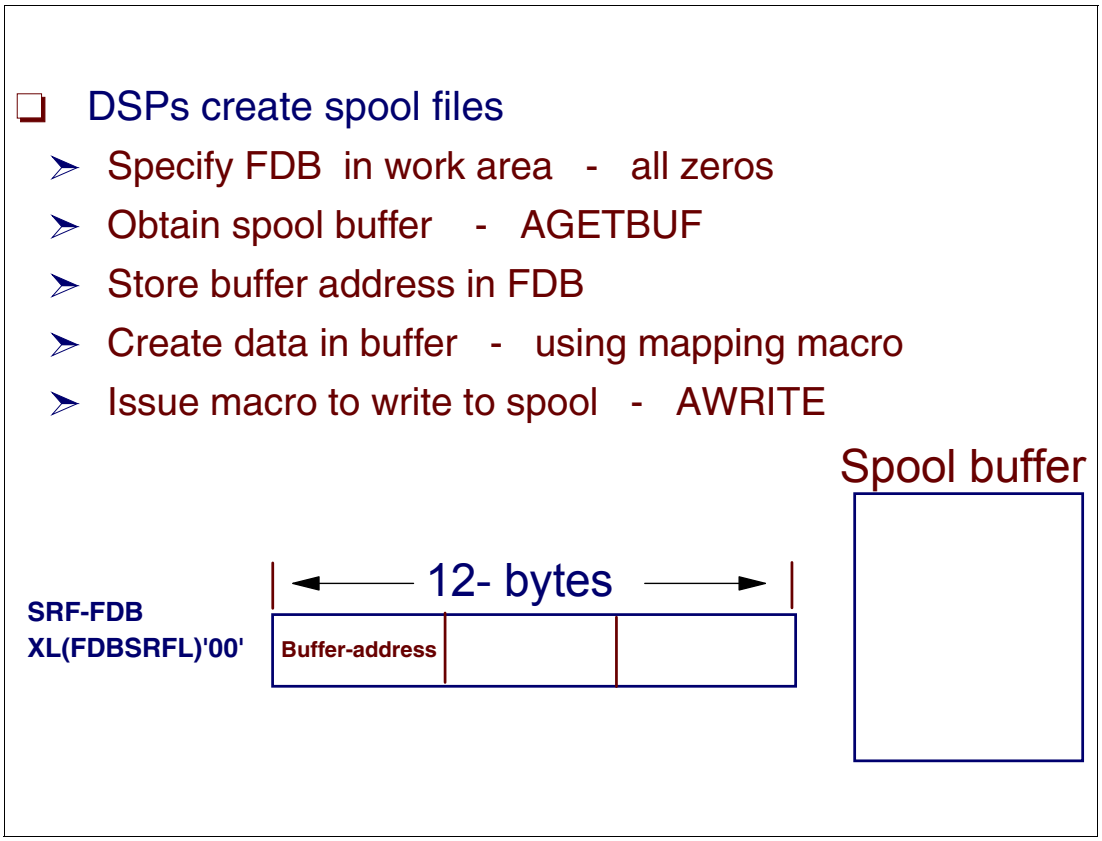
The "X" portion of an X.G identifies the spool extent‘ where the track group is located and "G" portion is used to locate a track group on the specified spool data set. Track group addresses (X.Gs) that are allocated to a job or data set are recorded in a job or data set track allocation table (JBT). One to nine track groups can be allocated to a job or data set for each spool space request. The number of track groups assigned to each job or data set is determined by the TRKGRPS parameter. The TRKGRPS parameter can be specified on the MAINPROC, CLASS and SYSOUT initialization statements and on the //*MAIN JES3 control statement.

New JBT record

If all the entries in the JBT are filled, a new JBT record is created and chained to the previous record. Each JBT is represented by a 28 byte FDB which contains the following information:

- The spool record address (M.R) of the JBT or its buffer address if the JBT is in storage (in the figure: M.R=000200066371).
- An available track group that contains a count of the number of available records (35) and the next available record address (R=00066372)
- A spool partition index is spool partitioning is in use
- Status flags
- A validation field used for multi-record files, FDBVALID

4.16 Creating spool files - SRFs



Creating spool files - SRFs

To create a spool file control block, the following steps are required:

- An FDB for a single record file must be cleared to all zeros in a work area. When creating this file, create space in a work area using:

  ```
  XL(FDBSRFL)'00'
  ```

- A spool buffer is required to enter the data and write it to the spool. The JSAM access provides a macro to obtain the buffer, AGETBUF (Get Buffer from JSAM Buffer Pool). On return from the AGETBUF, register 0 contains the buffer address.
- This buffer address must be stored in the FDB.
- The data to be written is then build into the buffer using a mapping macro that defines the format of the data.
- When the buffer is completely filled with the data, use the JSAM AWRITE macro to write the data to the spool.
- When the file is written, the I/O routines place the track address (spool record address) in the first 6-bytes of the FDB.

Spool I/O macros

JES3 records control blocks as single record files (SRF) in a single record on a spool data set. For single record files, JES3:

- Writes data to spool using the AWRITE macro
- Reads data from spool using the JESREAD macro

4.17 Creating a SRF control block

```
WAJDAB  DC     XL(FDBSRFL)'00'          12 byte work FDB

        AGETBUF                         Get spool buffer
        ST     R0,WAJDAB                Store in FDB.
*       [Buffer-address | | ]
*
        AWRITE   FDB=WAJDAB,ID=JDAB,                          *
                 TATPTR=WATAT,AWAIT=YES
*       [ M.R | flags .... | ]
*
        APUTBUF                         Put spool buffer
```

Creating a SRF control block

Creating a single record file requires a work area for the FDB (location WAJDAB). The work area can be defined using the XL(FDBSRFL)'00' specification.

AGETBUF macro

The AGETBUF macro obtains a buffer from the JSAM buffer pool of the calling routine's address space. If a buffer is not available, then the routine will wait until one becomes available unless you specify the BUSY parameter. The buffer is cleared to binary zeros before control returns to the calling module, unless bit AIOGETBF is set in the TVT (which specifies it is not necessary to set the buffer to zero).

The buffer address is returned in register 0. The returned buffer address in R0 is stored into the FDB.

AWRITE macro

The AWRITE (write a single-record file) macro writes a single record file (SRF) to the spool data set. A DSP issues an AWRITE macro to write a SRF to spool. For an SRF, a spool record address is allocated from either the space allocated to a job (the addresses for this spool space is found in the job's JBT) or from the system single track allocation table (STT). Records that have previously been written to spool are written to the same record address.

The AWRITE macro FDB parameter specifies the address of the FDB for the SRF to be written.

TATPTR parameter

The TATPTR parameter specifies the address of the track allocation table FDB. To write an SRF into the single track table (STT), code TATPTR=MNTRKFDB. If you do not specify the TATPTR parameter, JES3 will use the TAT FDB from the resqueue (RQ) as pointed to by the FCT (FCTRQAD). Make sure that the FCT is associated with the job whose control block is being written, even if the AWRITE is updating an already existing record on spool. In case of a recoverable I/O error, the TAT FDB will be used to find another record on spool. If the TAT is from a different job, the spool space will be purged independent of the current job which may lead to spool overlays.

ID parameter

The ID parameter specifies the name to be associated with the SRF. The number of characters is limited to four. ID is compared to the ID field in the buffer prior to the write. If they are not the same, the error exit is taken.

AWAIT parameter

The DSP executing under an FCT must wait for the I/O to complete before it gets control again.

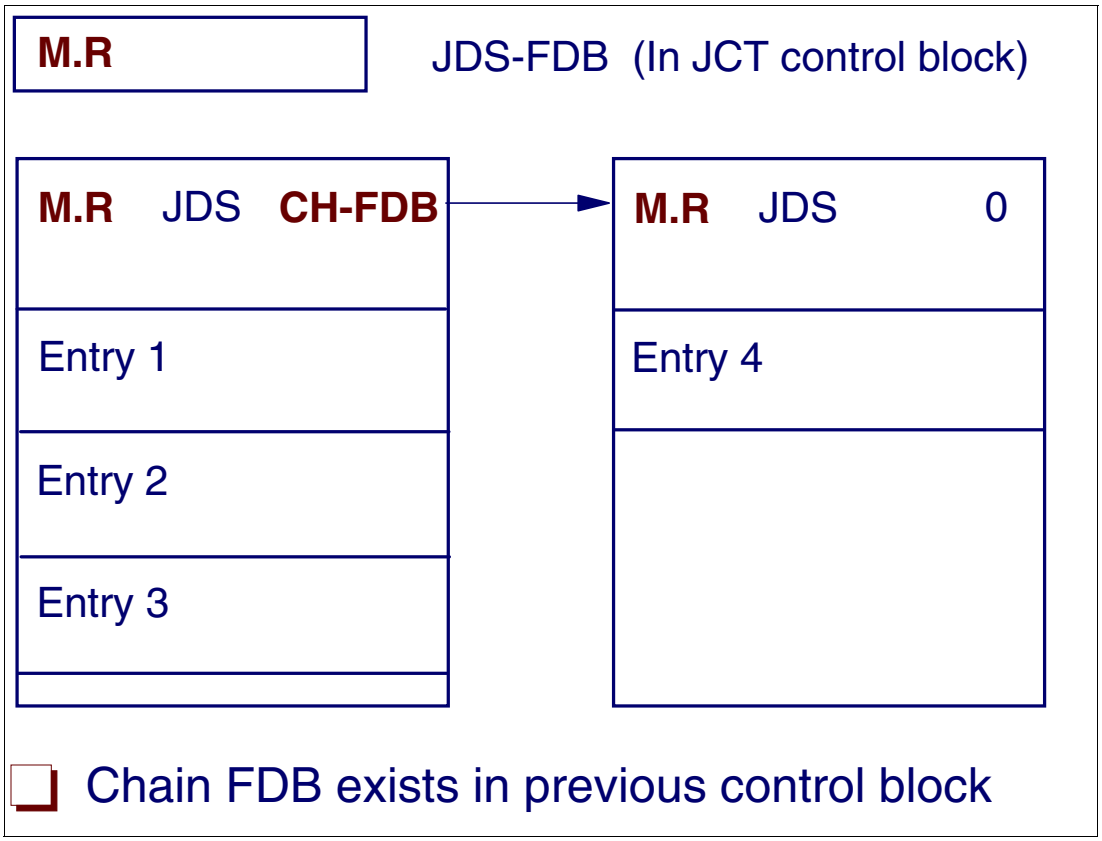
AWAIT=YES specifies that AWRITE should wait for the I/O to complete. The JSAM FCT posts the waiting FCT that issued the I/O request when the I/O completes.

When the I/O is complete, the JSAM routines place the spool record address (M.R) in the first 6 bytes of the FDB.

APUTBUF macro

The APUTBUF (Return a JSAM Buffer to the JSAM Buffer Pool) macro returns a JSAM buffer, obtained by a previous AGETBUF macro, to the JSAM buffer pool.

4.18 Chained single record files



Chained single record files

While most of the JES3 control blocks will fit into a single buffer image, some (due to their variable lengths and multiple parts) may extend across multiple buffers. When this occurs, the buffers are chained together into chained single record files. These buffers are chained together by placing the FDB of the next buffer in the previous buffer.

Reading a chained single record file from spool is done using the JESREAD (Read Single Record File) macro. JESREAD is invoked to read a record that is part of a chained single record file. To read the next file in the chain, the FDB for that file is in the file just read.

Writing or releasing chained SRFs

After accessing chained control blocks, such as a JDS, the IATXRELC (Restore Spool Address in FDB) macro should be used when chained control blocks have been read into storage, and has not been modified, and are no longer needed in storage. If a control block has been modified, the WRTCHAIN (write chained single record files) macro should be used to update the control blocks on the spool.

```
label   WRTCHAIN FDB=(R1)|address ,TATPTR=(R0)|address
                 ,DISP=12|(reg)|number ,PUTBUF=YES|NO ,ID=NONE|(reg)|name
                 ,OPTION=MODONLY ,ERROR= address ,NORMAL=address
```

When writing a chained SRF to spool using the WRTCHAIN macro, the DISP parameter of the WRTCHAIN macro specifies the displacement into the SRF of the chain FDB.

Other chained file macros

Other chained chained single-record file processing macros:

- IATXRDCH

  Use IATXRDCH to read a chained, single record file (SRF) into storage. If an error occurs, the caller is responsible for issuing an IATXRELC request to release any buffers that JES3 might have read. This macro uses register 2 in its expansion.

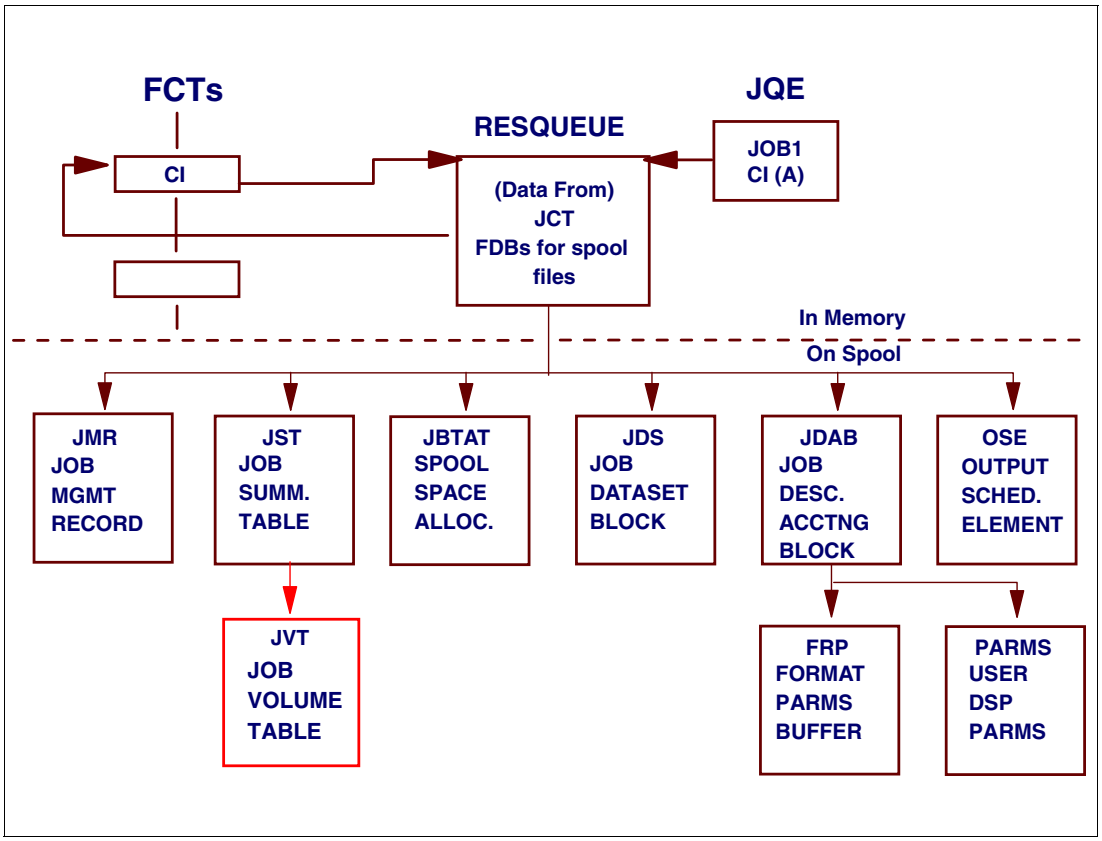
- PURCHAIN

  The PURCHAIN macro purges chained single record files (SRFs).

- IATXERCV

  The IATXERCV macro corrects the chaining of the file description blocks (FDBs) of a chained single-record file (SRF) after a write error. Register 15 must contain the error code from the AWRITE macro.

4.19 Accessing job control blocks



Accessing job control blocks

When a scheduler element is made active (the CI scheduler element in the figure) the FCT field FCTRQAD contains pointer to the RESQUEUE.

- The RESQUEUE has pointers to the FCTs of the job's SRFs

```
*        CONTROL BLOCK FDBS (FROM THE JCT)                    *
RQFDALIN DS    OF                ALIGNMENT ON A FULLWORD
RQJDBFDB DS    XL(FDBSRFL)       JDAB FDB
RQJSTFDB DS    XL(FDBSRFL)       JST FDB
RQJDSFDB DS    XL(FDBSRFL)       JDS FDB
RQJMRFDB DS    XL(FDBSRFL)       JMR FDB
RQJOBTAT DS    XL(FDBJBTL)       JOBTAT FDB
RQOSEFDB DS    XL(FDBSRFL)       OSE FDB
```

- The JQE contains a pointer to the job's JCT FCT and address of the RESQUEUE:

```
JQEFDB   DC    XL(FDBSRFL)'00'   FDB FOR JCT I/O REQUESTS
JQERESQ  DC    F'0'              ADDR OF RESQ
```

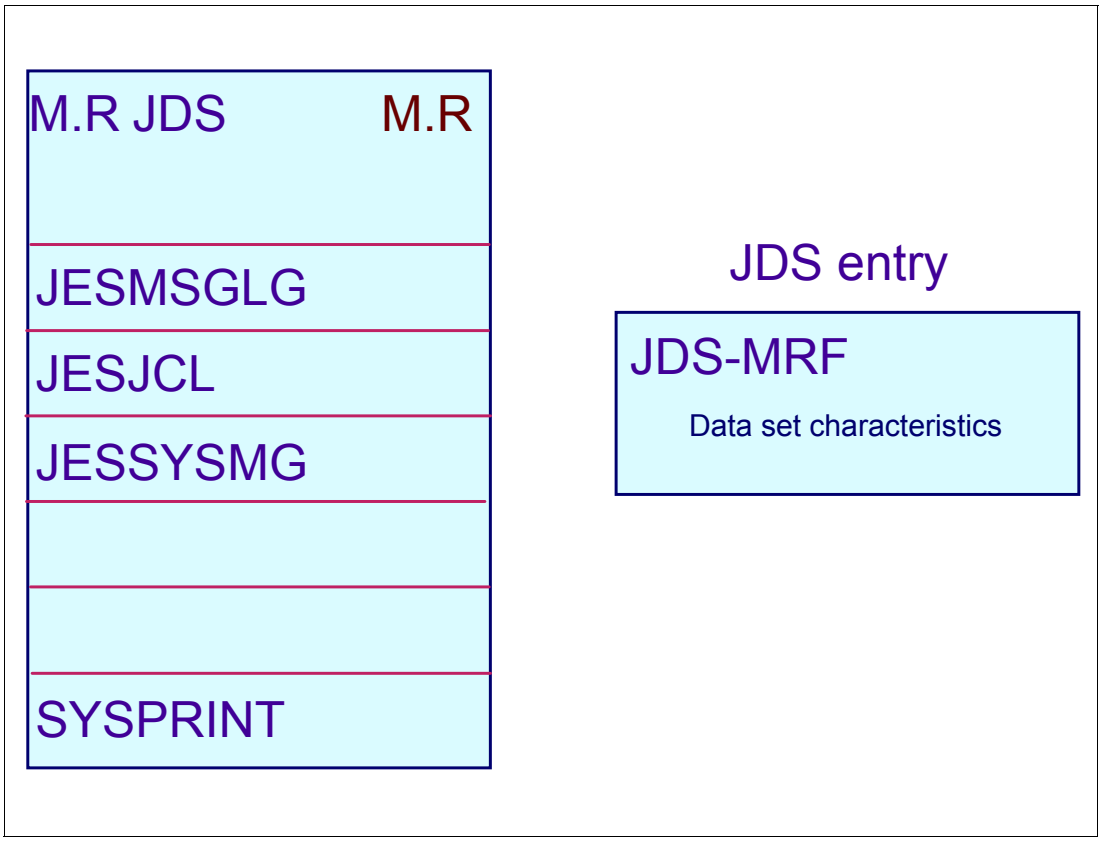
- The JST data includes an FCT for the JVT:

```
JSTJVTFD DS    XL(FDBSRFL)       JVT FDB
```

- The JDAB data includes FCTs for the FRP and PARM buffer:

```
JDABOSFD DS    XL(FDBSRFL)       FDB FOR OUTPUT SERVICE FRPS
JDABPFDB DS    XL(FDBSRFL)       PARAMETER FDB OR INDEX TO PARMS IN JDAB
```

4.20 JDS control block



JDS control block description

All of a job's multi-record files are recorded in the job data sets block (JDS). The JDS contains FDB's for all data sets associated with the job. This block, built while the JES3 input service job is being processed, initially contains the following:

```
JESMSGLG
JESJCL
JESYSMSG
J3SCINFO
J3JBINFO
JCBLOCK
JOURNAL
```

The standard data sets created for a job are:

- The data set which contains the JCL for the job. (JESJCL)
- All SYSIN data sets for the job.
- Two message data sets: one for messages generated by DSPs and one for system-generated messages. (JESMSGLG and JESSYSMG)
- JOURNAL data set for checkpoint/restart
- The VATs and SWA control blocks to the job's JCBLOCK data set.
- The JES3 Spool Access Facility Tokens for J3SCINFO and J3JBINFO are the addresses of the SPAF parameter lists in the ICT. When SMS wants to access the JES3 spool, it

passes the SPAF token to JES3. JES3 uses this token to access the proper control blocks to allow SMS to write to the JES3 spool.

The J3SCINFO and J3JBINFO JDS entries are updated in the JDS if the job has SMS managed requests. The MVS converter and interpreter writes to these data sets, and the JDS entry must be updated to show the current status of the data sets.

Data sets created during execution

Entries for SYSOUT data sets are added to the JDS when the data sets are opened during MVS execution such as for this job:

```
SYSPRINT
SYSUT2
```

4.21 Reading spool control blocks

- Reading a control block

RQJDSFDB (JDS-FDB)

M.R	flags	

```
LA          R1,RQJDSFDB
JESREAD  FDB=(R1),ID=JDS
```

Buffer-address	flags	

(After successful JESREAD FDB contains JDS buffer address)

- If you modify JDS control block

```
AWRITE   FDB=RQJDSFDB,ID=JDS
```

- If JDS control block not modified

```
ARELEASE   FDB=RQJDSFDB
```

Reading spool control b2locks

The JESREAD macro reads a single record file (SRF).

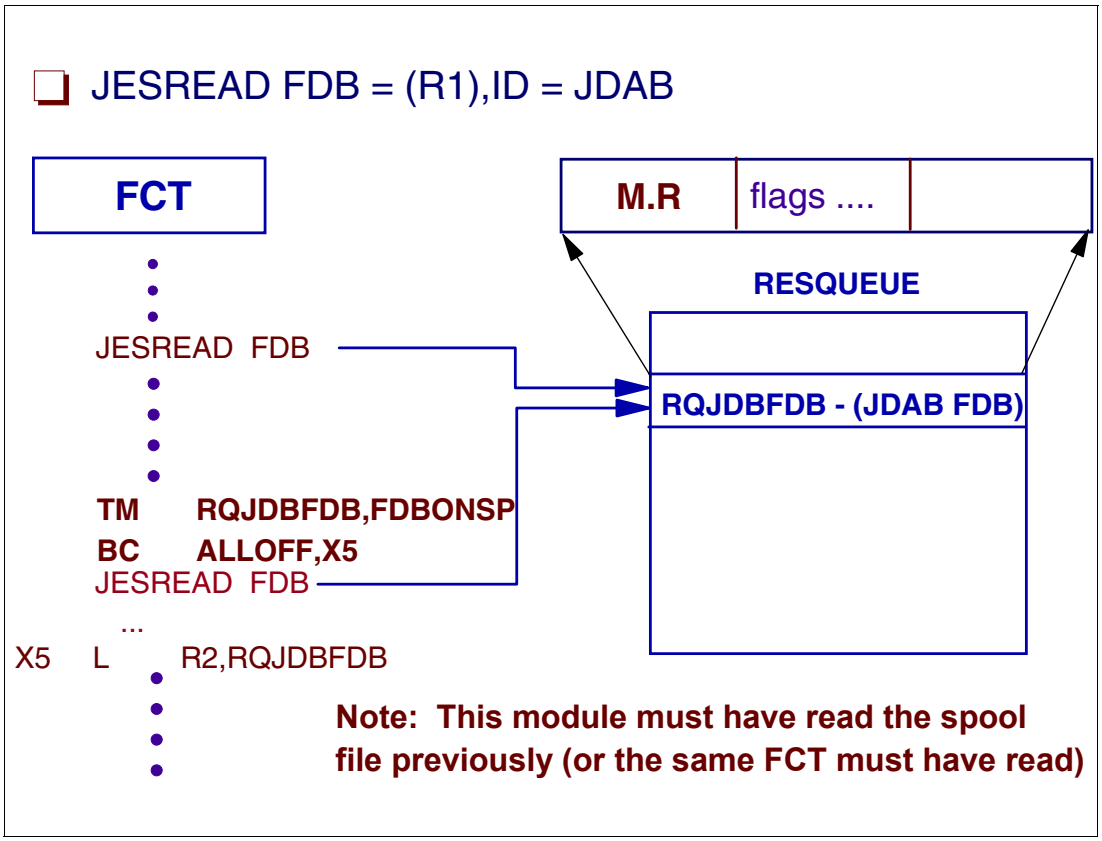
To read a spool control block for the job, do the following:

- Use a register to point at the FDB if the control block you wish to read. The FDBs of a job's SRFs can be found from RESQUEUE. The ID= parameter can be used for validity check of the buffer when issuing the JESREAD macro.
- When the JESREAD request completes, the FDB will contain the address of the spool buffer obtained by the JSAM routines.

  After accessing a control block, such as a JDS, the ARELEASE or IATXRELC macro should be used when a control block is read into storage, and has not been modified, and is no longer needed in storage. If a control block has been modified, the AWRITE or WRTCHAIN macro should be used to update the control block on the spool.
- If you modify anything fields in the buffer, you must issue an AWRITE macro to cause the updated fields to be stored on the spool.
- If you do not change any data in the buffer, you must issue an ARELEASE macro to free the spool buffer and cause the JSAM routines to put the M.R back in the FDB.

Note: Under no circumstances should any function move the FDB from the RESQUEUE to a work area and then read the file. This could cause an integrity problem where two or more DSPs could then read the same file at the same time.

4.22 Reading spool files (one FCT)



Reading spool files

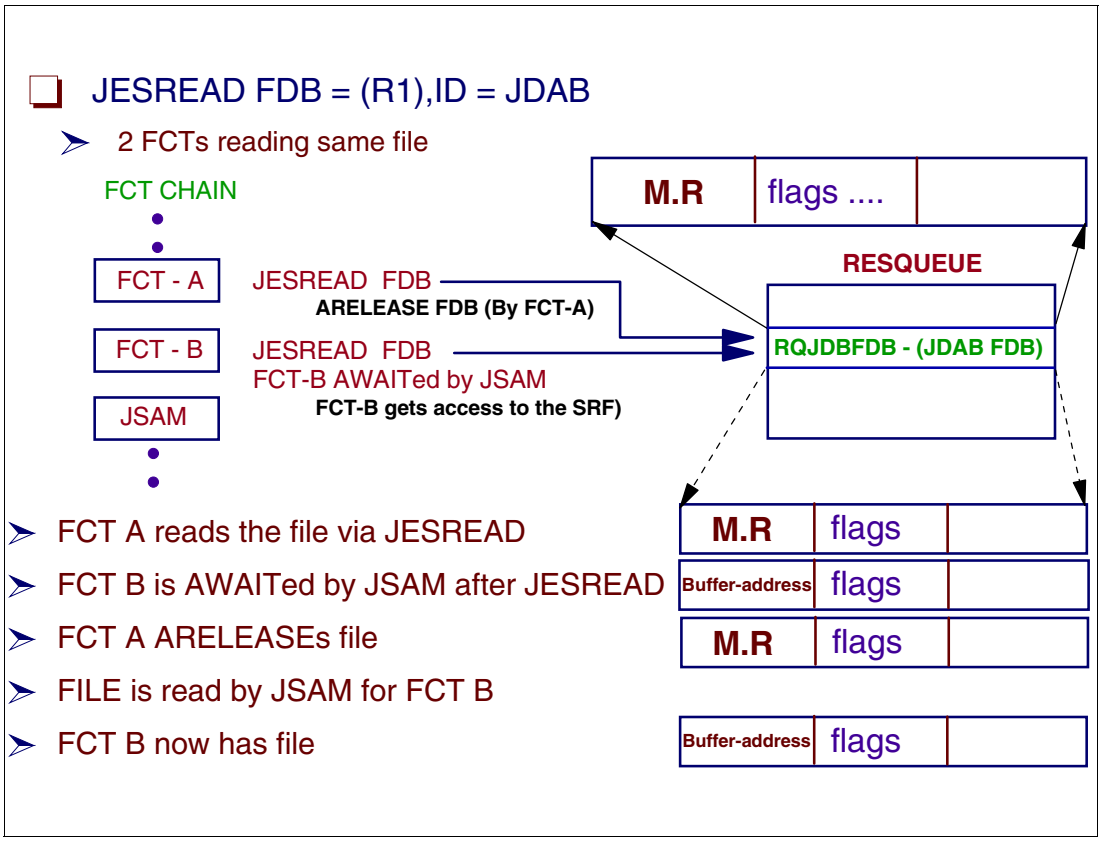
When reading a spool file using an FDB from the RESQUEUE entry, a DSP must always be careful that it does not try to read a spool file (control block) that it has is already read into storage.

The FDBONSP flag in the FDB of the file being accessed indicates whether the FDB contains a spool address. Code to test the flag;

```
        TM    RQJDBFDB,FDBONSP
        BC    ALLOFF,X5              "ON SPOOL" not set - Don't read.
           :
           :
  X5    DS    0H
```

Note: A DSP cannot use the FDBONSP to test whether it has read the spool file or another DSP running under a different FCT has accessed the spool file. The FDBONSP flag should be used on DSP cleanup processing to indicate whether a file should be released.

4.23 Contention accessing JES3 spool files



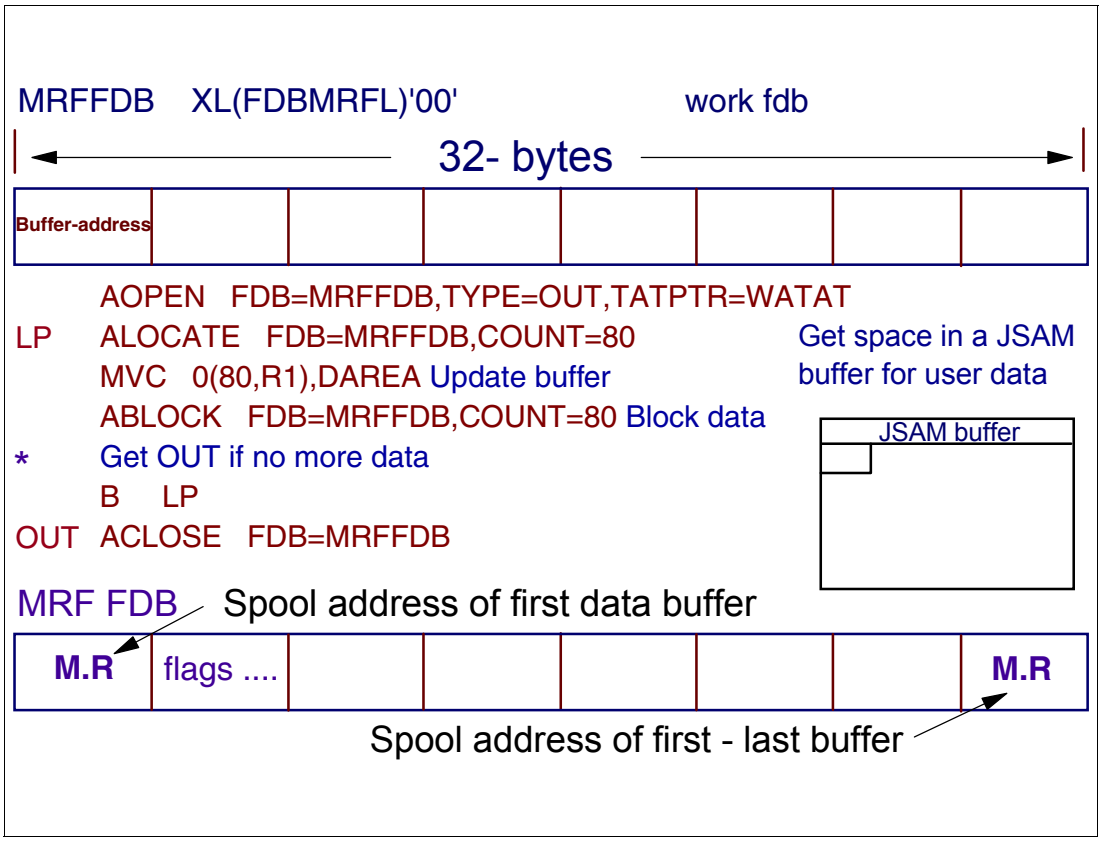
Contention accessing JES3 spool files

This visual shows two FCTs trying to access the same spool file. When FCT-A reads the file, the FDB now contains a buffer address. When FCT-B tries to read the file, the JSAM routines place FCT-B in an AWAIT condition, waiting for the file to be released by FCT-A if AWAIT=YES was specified on the JESREAD macro. If AWAIT=NO,BUSY=address is specified, the routine at address is invoked if I/O is not possible. The 31-bit ECF address in register 0 and the ECF mask in register 1 are returned to the busy routine.

> **Note:** Under no circumstances should FCT-B be checking that the FDB contains a buffer address, to try to access the buffer. This could cause an integrity problem since FCT-A is using the file.

When FCT-A does an ARELEASE or an AWRITE that returns the file, FCT-B now has its AWAIT condition satisfied and the JSAM routines read the file again for FCT-B and posts the AWAIT and FCT-B gets control from the MFM.

4.24 Creating multi-record files



Creating multi-record files

Multi-record files (MRFs) consist of JCL, SYSIN, and SYSOUT recorded as data records, placed into spool buffers, chained together, and written to spool. (MRFs). The MRF can occupy a single spool record or many spool records. Each spool record may contain one logical record, part of a logical record or many logical records. Various JSAM multi-record file services are invoked to create and access MRFs:

- The AOPEN (open a multi-record output file) macro sets up for the reading or writing of a MRF.
- AOPEND (open a multi-record output file) macro opens an existing multirecord file (MRF) for the later addition of more records at the end-of-file (EOF).
- The ALOCATE (locate space for a logical record) and ABLOCK add record to a multi-record file) macros write data to a MRF.
- The ADEBLOCK (retrieve a record from a multi-record file) macro reads data from a MRF.
- The ACLOSE (close a mult-record file) macro terminates the processing of a MRF.

Prior to reading from or writing to a multi-record file, the AOPEN routine is invoked to prepare the file for access. The processing performed by the AOPEN routine depends on whether the file is opened for input or output.

AOPEN macro

JES3 DSPs use the JSAM access method to create multi-record files. The AOPEN macro opens a multirecord file (MRF) for later output (blocking) or input (deblocking) of data. The

DSP points at the FDB, specifies TYPE=OUT for an output file, and points at a JBT FDB for the spool space needed to write the file to spool.

AOPEN for new multi-record file requests (the MRF FDB is zero), an entry is added to the file directory (FD) which will remain there until the file is closed. For multi-record file output requests the disk I/O routine is called to initiate the I/O processing.

AOPEND macro

The AOPEND macro opens an existing multirecord file (MRF) for the later addition of more records at the end-of-file (EOF).

ALOCATE macro

The ALOCATE macro asks the JSAM routines for pointer to where a new output record t can be stored. The JSAM routines get a buffer and provide the pointer and the DSP moves the data into the buffer.

The ALOCATE routine finds space for the logical record in the multi-record file, and returns the address of the space to the caller. If the current buffer does not have enough space for the record, the ALOCATE obtains a new buffer. The address returned to the caller is the area in the buffer where the user can move data. After issuing the alocate macro and moving the data into the buffer, the caller issues the ABLOCK macro to block the data into the file.

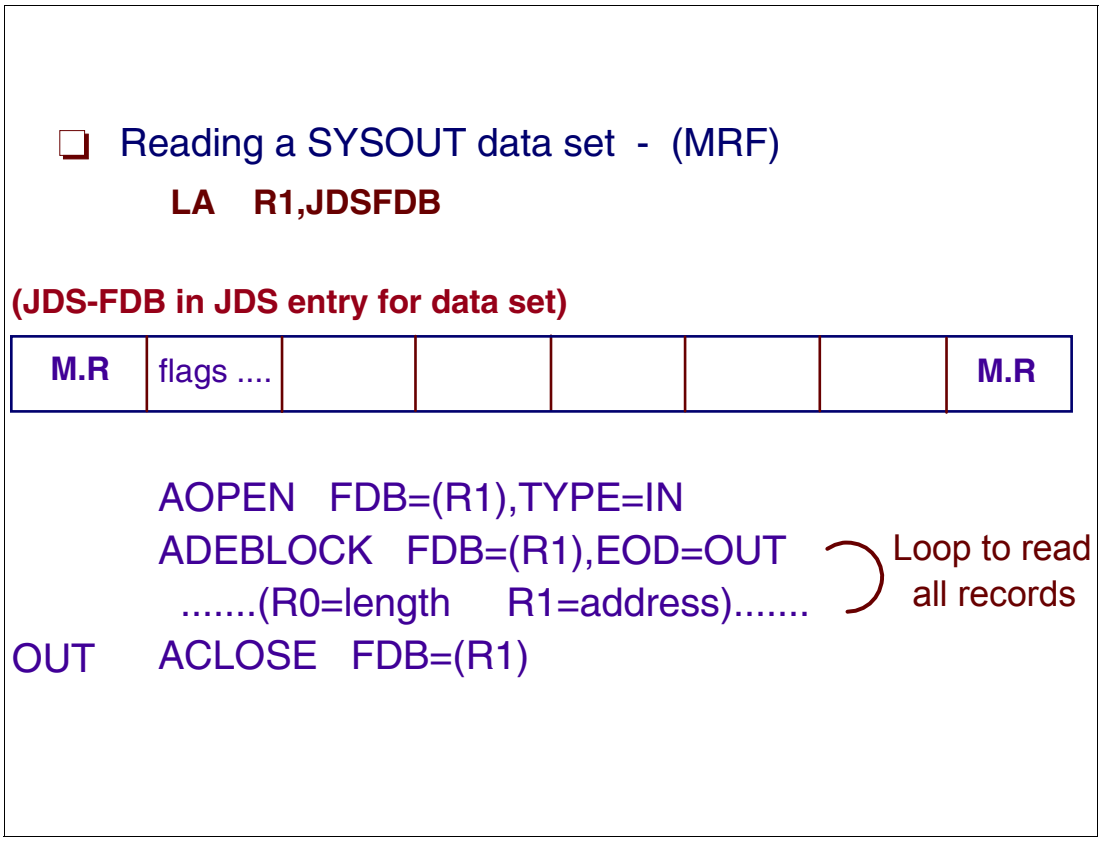
ABLOCK macro

The ABLOCK macro tells the JSAM routines the data is in the buffer and the JSAM routines then update the DATCC (JES3's record descriptor) field in front of the record with the record size plus the DATCC byte count.

ACLOSE macro

The ACLOSE macro causes the file to be closed. It then sets an end-of-data indicator (X'FFFFFFFF') into the buffer and writes the data buffers to the spool.

4.25 Reading multi-record files



Reading MRF files

Reading MRF in a JES3 DSP is done as follows:

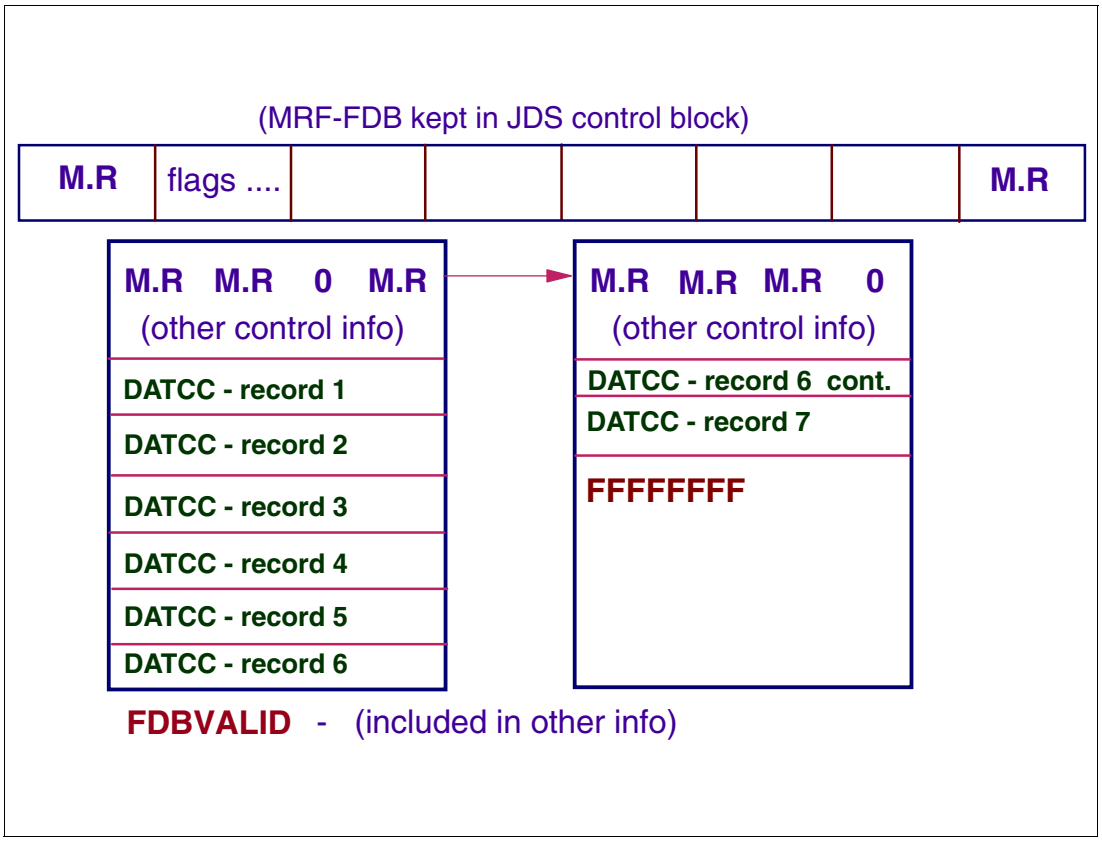
- The AOPEN macro is used with TYPE=IN. We are now reading a file that once was an output file. The FDB used to create the file as an output file must be used to read the file. FDB=(R1) points at the MRF-FDB.
- The JSAM routines during the AOPEN use the M.R in the FDB to read in the first spool buffer for the file.

Note: SYSOUT data set control information is kept in the JDS control block for each job.

- The ADEBLOCK (retrieve a record from a multi-record file) macro is used to obtain a record from the spool file. Always point at the MRF-FDB when using this macro. The EOD= specifies where to branch to when the end-of-data mark in the last buffer is reached meaning all the records have been accessed.
- When EOD occurs, or when you want to stop reading the file, the file must be closed using the ACLOSE macro. The JSAM routines replaces the M.R in the first 6 bytes of the FDB and turns on the close bit in the FDB.

Note: The MRF-FDB is used by JSAM for flags and storing of buffer addresses during the reading of the file. The FDB is a work area for the JSAM spool routines.

4.26 Multi-record file buffers



Multi-record file buffers

IATYDAT macro maps the MRF spool I/O buffer containing user data, forward, and backward M.R chains.

The job data set (JDS) control block (built during input service processing) contains information from the SYSOUT DD and the OUTPUT JCL statements. The job data set entry for a SYSOUT data set is built by module IATSIOR during OPEN processing. JDS control block is spool-resident.

A MRF FDB contains in the first 6 bytes the spool record address (M.R) of the first spool buffer. The last 6 bytes of the MRF-FDB contains the spool record address (M.R) of the last spool buffer in the data set.

The four M.Rs shown in each spool buffer are:

```
DATTHIS  DC    XL(L'FDBSPADR)'0' - SPOOL ADDRESS FOR THIS BUFF,
DATFIRST DC    XL(L'FDBSPADR)'0' - SPOOL ADDR FOR 1ST BUFFER
DATPREV  DC    XL(L'FDBSPADR)'0' - SPOOL ADDR FOR PRIOR BUFFER
DATNEXT  DC    XL(L'FDBSPADR)'0' - SPOOL ADDR FOR NEXT BUFFER
```

Each record in the spool buffer is preceded by the DATCC and the count in this field is filled in by the ABLOCK routine.

The x’FFFFFFFF’ at the end of the records in the last spool buffer is an end-of-data mark placed there by the ACLOSE routine.

The structure of JSAM/USAM buffer with MRF records looks like this:

```
 __________ _____ ______ ____ _____ ______ ____    ________
|DAT header|DATCC|DATCCX|data|DATCC|DATCCX|data|...|        |
|          |     |      |    |     |      |    |   |00FF0000|
|__________|_____|______|____|_____|______|____|   |________|
                                                    RL= 5 or 6
                                                 End of Buffer

 __________ _____ ______ ____ _____ ______ ____    ______
|DAT header|DATCC|DATCCX|data|DATCC|DATCCX|data|.....|      |
|          |     |      |    |     |      |    |     |      |
|__________|_____|______|____|_____|______|____|     |______|
                                                     RL < 5
                                                 End of Buffer

 __________ _____ ______ ____    ________
|DAT header|DATCC|DATCCX|data|....|        |
|          |     |      |    |    |FFFFFFFF|
|__________|_____|______|____|    |________|
                                 End of File
```

Space in spool buffers

If the room left is less than 5 bytes, there is no need to put the E-O-B identifier (x'00FF0000') and the room left will be left as it is (x'0000'). If the room left is greater than or equal to 5 bytes and still does not have enough room to put at least 1 byte of data, then put the E-O-B identifier.

Spool buffer control information

Either end-of-buffer identifier (x’00FF0000’) or room left less than 5 bytes indicates an end of buffer condition.

The DATCCX is a DATCC extension. If present it contains the original record length. The DATDATX flag indicates the presence of the DATCC extension. IATXDATX (Process DATCC with/without extension) macro is used to address user data following a DATCC with or without extension. It also provides the data length and logical record length of the current record. The logical record length returned is zero if DATCC extension (DATCCX) does not exist.

The FDBVALID (validation field for the MRF) is propagated into each MRF buffer (DATVALID). It is used to detect an abnormal end of a MRF in the case where DATNEXT points to a spool record which was never written as part of the MRF.

The 40 bytes of control information at the top of every spool buffer contains the following:

This TA M.R of this spool buffer

First TA M.R of the first spool buffer

Prev TA M.R of the previous spool buffer

Next TA M.R of the next spool buffer

DATOLIN The line number of the first record in this buffer.

DATLOREC The record number of the first record in this buffer.

DATLOPAG The page number of the first record in this buffer.

DATVALID A time stamp created for the job that owns this spool file. This DATVALID is in every MRF spool buffer owned by the job.

4.27 DATCC field information

- **DATCC - 4 BYTES**

- **DATCC high-order byte (flags)**

  DATCPDS - DATMAC - DATASA - DATOPTCD

  DATSPLTB - DATCON - DATSPAN - DATDATX

- **DATCCX - 2 BYTES**

  DATOLRCL

- **IATXDATX (Process DATCC with/without extension)**

DATCC field information

The DATCC is 4 bytes and contains the following information:

Byte 1 This byte contains a set of flags the determine the kind of data record; page data (DATCPDS), the carriage control character is in machine code (DATMAC), or the carriage control character is in ASA code (DATASA).

The DATOPTCD flag specifies that the MRF to be opened contains table reference characters.

The DATSPLTB flag indicate that this record is split between this buffer and the next buffer.

The DATCON flag indicates that this record is a continuation from the previous buffer.

The DATSPAN flag indicates that this record spans this buffer, the previous buffer, and the next buffer.

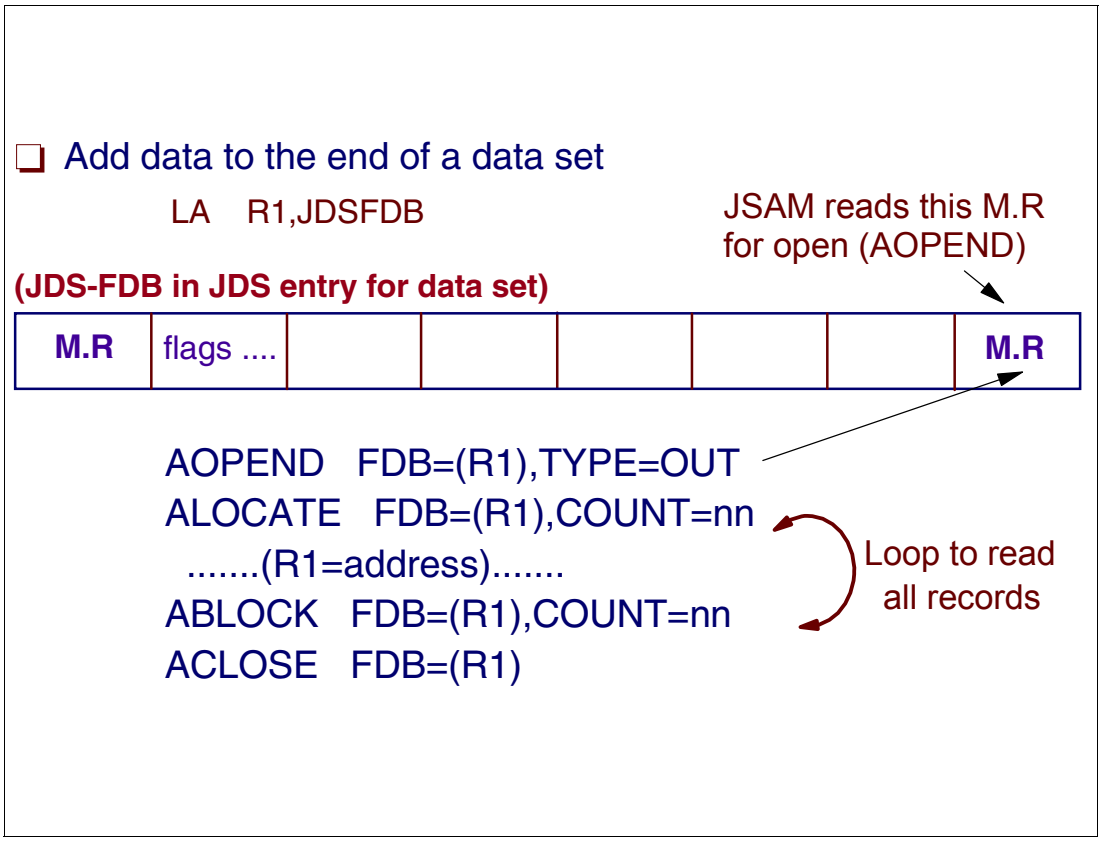
The DATDATX flag indicates the presence of DATCC extension (JES3 V1R5)

Byte 2 Reserved by IBM for future use.

Byte 3-4 These two bytes contain the length of the spool record plus four bytes which is the length of the DATCC.

The DATCCX is 2 bytes and contain the original logical record length of a record. If blank truncation is used, this is the original record length before truncation.

4.28 Updating a MRF files



Opening MRFs at the end

Many times it is necessary to add more records to a file that is already closed. This happens with the JESMSGLG data set. To add records to an existing closed file:

- Use the OPEND macro pointing at the MRF-FDB and specifying TYPE=OUT. The JSAM routines use the M.R in the last 6 bytes of the FDB which is the track address of the last dat buffer in the file.
- The ALOCATE macro is the used to get a pointer to the next available space for the new record to be added to the file. COUNT= specifies the record count. The data is then moved to the pointer given on return from the ALOCATE.
- The ABLOCK is issued to cause the DATCC to be filled in.
- When all the new records are added to the file, issue the ACLOSE to close the file and cause the FFFF to be placed at the end of the last record.

4.29 JES3 SYSOUT data set names

DD-name	dsnumber
JESMSGLG	D0000002
JESJCL	D0000003
JESYSMSG	D0000004
J3SCINFO	D0000005
J3JBINFO	D0000006
JCBLOCK	D0000007
JOURNAL	D0000008

dsnumber is the unique data set number JES3 assignes to spool data sets. A "D" is the first character of this qualifier.
- Used in JESSPOOL RACF profiles for the spool data sets
- Operator commands
- JDSGET and IATXBDSN macros

```
 IAT6140 JOB ORIGIN FROM GROUP=ANYLOCAL, DSP=IR , DEVICE=INTRDR  , 0000
 15:05:52 ---- IAT6853 THE CURRENT DATE IS SATURDAY,  25 OCT 2008 ----
IRR010I  USERID VAINI    IS ASSIGNED TO THIS JOB.
 15:05:52 IAT6312 THE FOLLOWING 000001 MESSAGES MAY BE LOGGED OUT OF SEQUENCE
 15:05:52  ICH70001I VAINI    LAST ACCESS AT 08:58:23 ON SATURDAY, OCTOBER 25, 2008
 15:05:52  IAT2000 JOB NOTHING  (JOB24957) SELECTED SC65     GRP=A
 15:05:52  ICH70001I VAINI    LAST ACCESS AT 15:05:52 ON SATURDAY, OCTOBER 25, 2008
 15:05:52  IEF403I NOTHING - STARTED - TIME=15.05.52 - ASID=003F - SC65
 15:05:52  IEF404I NOTHING - ENDED - TIME=15.05.52 - ASID=003F - SC65
//NOTHING JOB (999,POK),'DO NOTHING',CLASS=A,REGION=2M,            *
//     MSGCLASS=T,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//NOP     EXEC PGM=IEFBR14,REGION=1K
      1 //NOTHING JOB (999,POK),'DO NOTHING',CLASS=A,REGION=2M,            *
        //     MSGCLASS=T,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
        IEFC653I SUBSTITUTION JCL - (999,POK),'DO NOTHING',CLASS=A,REGION=2M,MSGCL
        NOTIFY=VAINI
      2 //NOP     EXEC PGM=IEFBR14,REGION=1K
ICH70001I VAINI    LAST ACCESS AT 15:05:52 ON SATURDAY, OCTOBER 25, 2008
IEF142I NOTHING NOP - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP/NOP     /START 2008299.1505
IEF374I STEP/NOP     /STOP  2008299.1505 CPU    0MIN 00.00SEC SRB    0MIN 00.00SEC VIRT     4K SYS   264K EXT       0K
IEF375I  JOB/NOTHING /START 2008299.1505
IEF376I  JOB/NOTHING /STOP  2008299.1505 CPU    0MIN 00.00SEC SRB    0MIN 00.00SEC
```

JESMSGLG

Data set names

The IATXBDSN macro creates qualified data sets names consisting of multiple qualifiers. A five-part qualified data set names is used to specify the resource name. The names have the following format:

```
userid.jobname.jobid.dsnumber.dsid

DSNAME=userid.jobname.jobid.dsnumber.name, if dsnumber exists,
(JCTOUSID, JCTJBNAM, JCTJOB, JDSDSNUM, JDSDSNAM) otherwise,

DDNAME=PROCSTEPNAME.STEPNAME.
DDNAME
 (JDSPROCN, JDSSTEPN, JDSDDNAM)
```

Messages to JESMSLG data set

Several DSPs and execution of the job write into the JESMSGLG data set.

Messages to JESMSGLG from reader processing

Input service writes into the JESMSGLG data set

Messages to JESMSGLG from CI processing

Messages are written into JESMSGLG data set by using the IATXIWT and JESMSG macros.

Messages from execution

If the message requires special processing while a job is execution to the JESMSGLG, module IATSIWO sends the message to the WTO destination queue in the JES3 global's address space to be handled by the CONSERV DSP driver module, IATCNSV which places the message into the JESMSGLG data set.

5

JES3 initialization

Before jobs enter the system, the job entry subsystem (JES3) must be initialized to process work. Initialization of JES3 is the process of establishing control blocks used for JES3 processing. JES3 initialization takes place between the time that the **START JES3** command is issued and the time that the processing of jobs begins.

Depending on the type of start the operator specifies, JES3 initialization is a three or four phase process. The phases are:

- Phase 1 determines the types of starts that are allowed for the main and asks the operator to select a start type.
- Phase 2 reads the statements from the initialization stream that initialize the control blocks to manage spool space and, in the event of a restart, validates jobs that are in the JES3 job queue.
- Phase 3 reads the initialization stream and converts information supplied on each initialization statement to intermediate spool files. These intermediate spool files are written to spool.
- Phase 4 uses the intermediate spool files built in phase 3 to build the final control blocks necessary for job execution. Phase 4 informs the operator that JES3 initialization is complete.

5.1 Initialization stream

DYNALLOC		Allocate Data Sets
BUFFER TRACK FORMAT BADTRACK OPTIONS	 req. req. 	JES3 Spool Environment
ENDJSAM		Card Required here
DEVICE MAINPROC	req. req.	Rest of Statements in any order
ENDINISH		Card Required here

Initialization stream

An initialization stream is a collection of JES3 initialization statements used to define the JES3 system configuration. It also defines how JES3 manages resources and jobs. These statements tell JES3 how to manage the following:

- Jobs, job classes, and job class groups
- Mains (global and local)
- I/O devices
- Main and external storage
- The system log
- Communication lines and/or protocols
- Operator communication

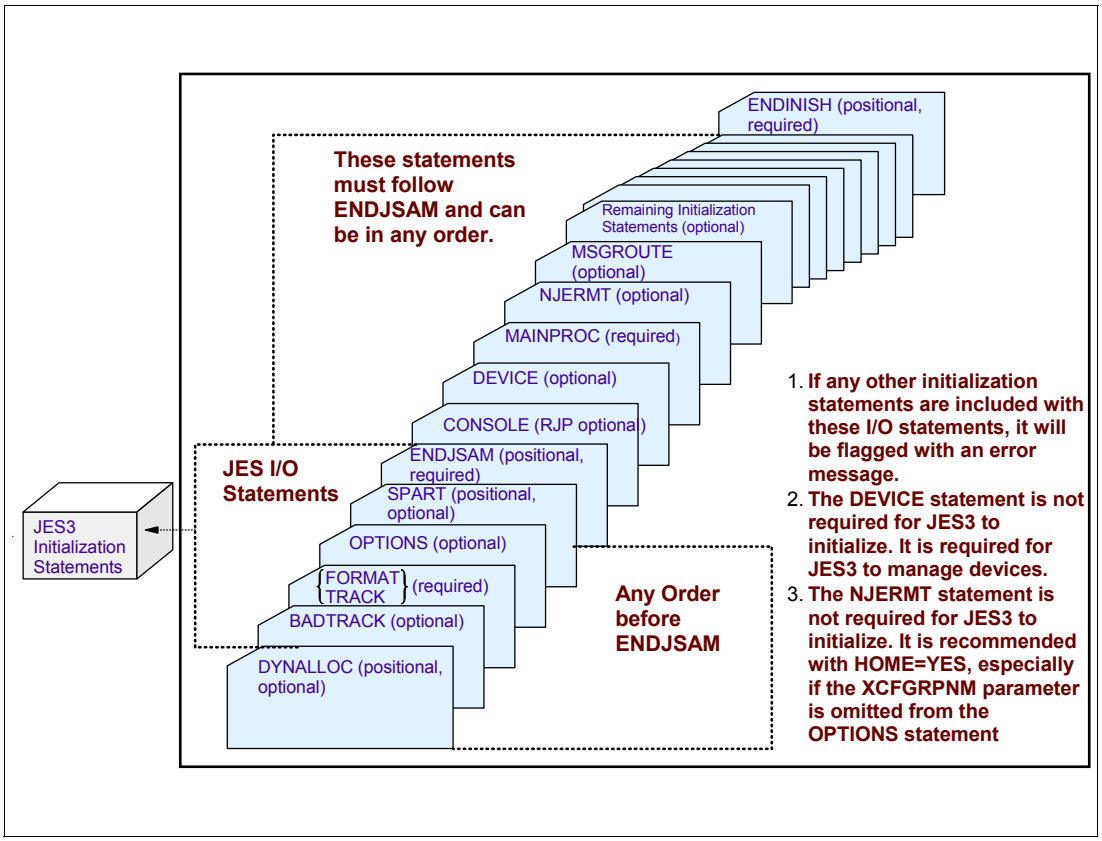
The following statements must be included in all initialization streams:

- ENDINISH
- ENDJSAM
- FORMAT or TRACK
- MAINPROC

The job management information that you code on initialization statements specify how you want JES3 to:

- Process job input - Interpret JES3 control statements
- Select and schedule jobs for execution - Process job output

5.2 JES3 initialization statements



JES3 initialization statements

You must observe the following rules when coding initialization statements:

- Code the statement name and parameters in columns 1-71. Column 72 must be blank if the statement is complete; column 72 may be blank or non-blank if the statement is continued. JES3 ignores columns 73-80.
- Use commas to separate the statement name from the first parameter and to separate one parameter from another.
- If you code a keyword parameter more than once, JES3 usually uses the last value coded. In some cases, however, JES3 treats a duplicate keyword as a continuation of a previously coded keyword of the same name. In other cases, when JES3 encounters a duplicate keyword, it issues a diagnostic message.
- If you wish to continue an initialization statement (individual statement descriptions specify whether a statement can be continued).

Phases of JES3 initialization

Depending on the type of start the operator specifies, JES3 initialization is a three or four phase process. The phases are:

Phase 1 Phase 1 determines the types of starts that are allowed for the main and asks the operator to select a start type.

Phase 2 Phase 2 reads the statements from the initialization stream that initialize the control blocks to manage spool space and, in the event of a restart, validates jobs that are in the JES3 job queue.

Phase 3 Phase 3 reads the initialization stream and converts information supplied on each initialization statement to intermediate spool files. These intermediate spool files are written to spool.

Phase 4 Phase 4 uses the intermediate spool files built in phase 3 to build the final control blocks necessary for job execution. Phase 4 informs the operator that JES3 initialization is complete.

When modifying or creating an initialization stream, you must be aware of related initialization statement parameters. With related parameters, the value you code for one parameter influences the value you code for another parameter. *z/OS JES3 Initialization and Tuning Reference*, SA22-7550 contains tables that list the interdependent parameters.

5.3 JES3 initialization statements

ACCOUNT (Job Accounting)
BADTRACK (Bypass Defective Tracks)
BUFFER (JES3 Spool Work Buffers)
CIPARM (Converter/Interpreter Parameters)
CLASS (JES3 Job Class Definition)
COMMDEFN (Communication SSI Definition)
COMMENT (*)
COMPACT (Compaction Table Definition)
CONSOLE (RJP Operator Consoles)
CONSTD (Console Service Standards)
DEADLINE (Deadline Type Definition)
DESTDEF (NJE Define Destinations for SYSOUT)
DEVICE (Processor Status, BSC line, I/O Device)
DYNALLOC (Dynamically Allocate Data Sets and Devices)
DYNALDSN (Dynamically Allocated Data Set Integrity)
ENDINISH (End of Initialization Stream)
ENDJSAM (End of JES3 I/O Statements)
FORMAT (Format Spool Data Set)
FSSDEF (Functional Subsystem Definition)
GROUP (Job-Class Group Definition)
HWSNAME (High Watermark Setup Names)
INCLUDE (Include Initialization Stream Member)
INTDEBUG (Initialization Debugging Facility)
MAINPROC (Define a JES3 Main)
MSGROUTE (MVS Message Route Table)
NETSERV (TCP/IP/NJE Network Server)
NJECONS (Console for NJE)
NJERMT (JES3 Network Node Definition)
OPTIONS (JES3 Options)
OUTSERV (Output Service Defaults and Standards)
RESCTLBK (Resident Control Block)
RESDSN (Resident Data Set Names)
RJPLINE (BSC Remote Job Processing Line)
RJPTERM (BSC Remote Job Processing Terminal)
RJPWS (SNA Work Station Characteristics)
SELECT (Job Selection Mode)
SETACC (Accessibility to Direct-Access Volumes)
SETNAME (Set JES3 Device Names)
SETPARAM (Set MDS Parameters)
SETRES (Mount Direct-Access Volumes)
SOCKET (TCP/IP socket connection)
SPART (Spool Partition Definition)
STANDARDS (Installation Defaults and Standards)
SYSID (Define the Default MVS/BDT Node)
SYSOUT (SYSOUT Class Characteristics)
TRACK (Preformatted Spool Data Set)

JES3 initialization statements

JES3IN DD-statement in the in the JES3 start procedure defines the data set containing the JES3 initialization stream. This data set must be a blocked or unblocked partitioned data set.

ACCOUNT (Job Accounting): Use the ACCOUNT initialization statement to define default job accounting information. JES3 assigns this default if the operator omits the ACCT parameter on a JES3 *CALL command.

BADTRACK (Bypass Defective Tracks): Use the BADTRACK statement to identify defective tracks on a spool volume. JES3 dynamically adds an entry to the BADTRACK table when a defective track is discovered and issues a message to the console operator that identifies the defective track. If possible, add a BADTRACK statement to your initialization stream at that time so that JES3 keeps a record of the defective track across a warm or cold start. If you cannot add a BADTRACK statement immediately, ensure that you add a BADTRACK statement before the next warm or cold start.

BUFFER (JES3 Spool Work Buffers): Use the BUFFER statement to define the size of the JES3 buffer pool and the length of JES3 buffers and spool data set records.

CIPARM (Converter/Interpreter Parameters): Use the CIPARM statement to specify the options to be used by the MVS converter/interpreter (C/I). These options are used as system defaults applied to certain JCL statement parameters and other options for jobs scheduled on any main.

CLASS (JES3 Job Class Definition): Use the CLASS initialization statement to define the characteristics of JES3 job classes. A CLASS statement must define each class that can appear on the //*MAIN control statement.

COMMDEFN (Communication SSI Definition): Use the COMMDEFN statement to specify the optional user communication subsystem interface (VTAM) parameters.

COMMENT (*): Use the comment statement to include comments in a JES3 initialization stream.

COMPACT (Compaction Table Definition): Use the COMPACT statement to define a compaction table to JES3. The compaction table is a set of characters which can be transmitted as a compacted character string.

CONSOLE (RJP Operator Consoles): Use this CONSOLE statement to define the characteristics of a console for an RJP workstation. This statement assigns message destinations to these type of consoles.

CONSTD (Console Service Standards): Use the CONSTD statement to define standards for your console configuration. These standards include JES3 command prefix characters, hardcopy log configuration, and special characters to be used in editing commands processed by JES3 console services.

DEADLINE (Deadline Type Definition): Use the DEADLINE statement to define a deadline type for job scheduling; each type determines how JES3 increases the priority of a job so the job is scheduled within a specified time limit.

DESTDEF (NJE Define Destinations for SYSOUT): Use the DESTDEF initialization statement to specify how inbound SYSOUT data sets from other NJE nodes are to be processed at this node.

DEVICE (Processor Status, BSC line, I/O Device): Define Processor Status -- Use this form of the DEVICE statement (there are two other forms of the DEVICE statement) to define the initial status of mains in a JES3 complex. If you omit this DEVICE, the processor in question is initialized online to every processor in the complex.

Define a Network BSC line or CTC Connection -- Use this form of the DEVICE statement to define a BSC line or a CTC connection that connects your node to another node in a network. You must code a DEVICE statement for each such line. or connection.

Define I/O Devices -- Use this form of the DEVICE statement to define a device that JES3 can use:

- To satisfy its own functions (JES3 device).
- To satisfy the needs of a job (execution device).
- As a JES3 device or as an execution device (shared device).

DYNALDSN (Dynamically Allocated Data Set Integrity): Use the DYNALDSN statement to specify which data sets on permanently resident or reserved DASD volumes require data set integrity protection when the data set is dynamically allocated.

DYNALLOC (Dynamically Allocate Data Sets and Devices): Use the DYNALLOC statement to specify a data set or a device that you want dynamically allocated to JES3 during initialization. The DYNALLOC statement allows you to allocate a data set or device without changing the JES3 cataloged procedure.

ENDINISH (End of Initialization Stream): Use the ENDINISH statement to identify the end of the initialization statements in the initialization stream.

ENDJSAM (End of JES3 I/O Statements): Use the ENDJSAM initialization statement to indicate the end of the JES3 spool initialization statements.

FORMAT (Format Spool Data Set): Use the FORMAT statement to specify formatting for a data set residing on a direct-access spool volume during initialization. Specify this statement only when introducing an unformatted volume into a JES3 system or when you change the BUFSIZE parameter on the JES3 BUFFER initialization statement.

FSSDEF (Functional Subsystem Definition): Use the FSSDEF statement to define the characteristics of a functional subsystem (FSS) which operates in its own address space. Use a FSSDEF statement for either of the following:

- To define one or more C/I FSSs.
- To define one or more output writer FSSs for printers that you define to run in FSS mode (via the DEVICE initialization statement). You can define more than one printer to run under the control of a single output writer FSS. If you do not define an output writer FSS for each printer that requires one, JES3 creates an FSS using default values.

GROUP (Job-Class Group Definition): Use the GROUP initialization statement to define the characteristics of a JES3 job-class group and whether the initiators managed by this group are WLM managed or JES3 managed. A GROUP statement must define each job class group (except for the default group, JS3BATCH) named on a CLASS initialization statement.

HWSNAME (High Watermark Setup Names): Use the HWSNAME statement to:

- Define, to JES3, all names by which users can reference a given device type to enable high watermark setup (HWS) processing.
- Identify the characteristics of each device name for the specific JES3 complex. This statement can be used to define which device names are subsets of other device names. In general, the fewer the number of alternate names, the more restrictive the device name being defined. This ensures that initial allocation for devices that are reused from step to step is the most restrictive device. It also ensures that attempts to override passed or cataloged unit names are processed correctly. Non-HWS users are encouraged to supply HWSNAME information to take advantage of this function.

INCLUDE (Include Initialization Stream Member): Use the INCLUDE statement to include a member in the initialization stream member. Different sections of the initialization stream can be put into different members and included in the primary initialization stream member. The member is the PDS member name within the data set specified on the JES3IN DD statement in the JES3 procedure to be included. Up to 4 member levels can be used (the primary initialization stream member and up to 3 INCLUDE level members). Use the INCLUDE statement anywhere after the DYNALLOC statements.

INTDEBUG (Initialization Debugging Facility): Use the INTDEBUG statement to specify error message text and an index value. If the specified message text is issued the number of times indicated by the index value, JES3 issues a U005 JES3 user abnormal end and takes a storage dump.

MAINPROC (Define a JES3 Main): Use the MAINPROC initialization statement to define a processor as a JES3 main. The initialization stream must include one MAINPROC statement for each main that you wish to define to JES3.

MSGROUTE (MVS Message Route Table): Use the MSGROUTE statement to control the routing of subsystem modifiable messages (such as most MVS-issued messages). If you do not include a MSGROUTE statement, the routing attributes of the messages that originate from that processor are not modified by JES3 MSGROUTE processing. Even though MSGROUTE processing may not make modifications, a message is still eligible for other forms of JES3 message routing.

NETSERV: Use the NETSERV initialization statement to define the attributes of a TCP/IP/NJE Network Server (NETSERV) address space.

NJECONS (Console for NJE): Use the NJECONS initialization statement to specify the message class to which JES3 is to send messages about the JES3 job entry network.

NJERMT (JES3 Network Node Definition): Use the NJERMT initialization statement to define a node in the JES3 job entry network.

OPTIONS (JES3 Options): Use the OPTIONS initialization statement to specify:

- The type of MVS system dump to be taken, if needed.
- Whether or not a dump should be taken when a termination condition exists.
- The job numbering limits for JES3 jobs.
- Whether you want the writer output multitasking facility enabled or disabled.
- The number of scheduler elements needed to support the largest job that will be run in the JES3 complex.

OUTSERV (Output Service Defaults and Standards): The OUTSERV initialization statement specifies default values and standards for the output service element (OSE) to be used on output devices; for example: printers, punches, or RJP (remote job processing). These defaults apply to every built OSE, regardless of the device that handles the output, provided other overrides do not take effect.

RESCTLBK (Resident Control Block): Use the RESCTLBK initialization statement to preallocate storage for the highly used JES3 function control table (FCT) entries.

RESDSN (Resident Data Set Names): Use the RESDSN statement to name permanently resident data sets for which JES3 is to bypass setup processing. JES3 bypasses setup processing whenever the named data sets appear as cataloged references (no UNIT or VOLUME parameters are specified) on the DD statement of a job.

RJPLINE (BSC Remote Job Processing Line): Use the RJPLINE initialization statement to define the characteristics of a single BSC line (and its respective adapter) that will be used by the JES3 global for remote job processing. You can also use this statement to assign a specific RJP work station, defined by the N parameter of an RJPTERM statement, to this line.

RJPTERM (BSC Remote Job Processing Terminal): Use the RJPTERM initialization statement to define a single remote BSC work station to the JES3 system. This statement causes a default description to be provided for each work station device (printer, punch, or card reader) indicated by the PR, PU, or RD parameters along with the operating characteristics of the work station. If the JES3 default characteristics for a remote printer or punch device are not acceptable, a DEVICE statement should be coded to indicate desired characteristics. If a work station is to have the facilities of a JES3 operator console, then a CONSOLE statement must be coded.

RJPWS (SNA Work Station Characteristics): Use the RJPWS initialization statement to describe each SNA work station's characteristics to the JES3 system. This statement causes a default description to be provided for each work station device (printer, punch, or card reader) indicated by the PR, PU, or RD parameter along with the operating characteristics of the work station.

SELECT (Job Selection Mode): Use the SELECT initialization statement to define scheduling controls you want associated with a particular job selection mode. The initial job selection mode is assigned to a JES3 main using the SELECT parameter on the JES3 MAINPROC initialization statement. If a MAINPROC statement does not indicate a selection mode, the SELECT statement default values are assigned to that main. Each select mode defined can

be dynamically changed using the **`*MODIFY,G,main,S`** operator command. In addition, the commands *`MODIFY,G,main,G` or **`*MODIFY,G,main,C`** can indirectly affect the select mode. A SELECT statement must be specified for each select mode indicated on a MAINPROC statement or in a **`*MODIFY,G,main,S`** command.

SETACC (Accessibility to Direct-Access Volumes): Use the SETACC initialization statement to identify those mains that normally have access to a permanently resident direct-access volume. The SETACC statement identifies the location of a volume on the uninitialized mains in a JES3 complex. SETACC prevents JES3 from setting up a job that needs the mounted volume until the main is initialized. When all mains are initialized or the volume is found, the SETACC definition is no longer used and normal JES3 management of the volume and device occurs. The devices on which the volumes reside are defined on a DEVICE statement with an XTYPE parameter and PR subparameter.

SETNAME (Set JES3 Device Names): Use the SETNAME initialization statement to specify all user-assigned names and device type names associated with MDS-managed devices.

SETPARAM (Set MDS Parameters): Use the SETPARAM initialization statement to specify parameters that the JES3 main device scheduler (MDS) and the DYNAL DSP uses in allocation, mounting, and deallocation of devices for jobs run on all mains. The SETNAME and DEVICE statements are used with the SETPARAM statements. SETNAME and DEVICE identify the devices to be managed by MDS. SETPARAM also indicates how MDS is to manage devices.

SETRES (Mount Direct-Access Volumes): The SETRES statement identifies frequently used direct-access volumes which are not permanently resident. The SETRES statement specifies volumes which may reside on devices at main initialization time. When a specified volume is found to be present on an MDS-managed, removable, direct-access device during main initialization, the volume is considered mounted by MDS, without a MOUNT command being necessary.

SOCKET: Use the SOCKET initialization statement to describe a TCP/IP socket connection that is used to communicate with an NJE node using the TCP/IP protocol.

SPART (Spool Partition Definition): The SPART statement defines one spool partition and specifies:

- The name of the partition
- Whether JES3 is to use the partition as the default partition
- Whether JES3 is to write initialization information to the named partition
- Whether the partition is to overflow into another partition
- The number of records in each track group

STANDARDS (Installation Defaults and Standards): Use the STANDARDS initialization statement to specify default values for information not provided on other JES3 initialization statements or on the //*FORMAT JES3 control statement. It also provides standards to be applied to all jobs entering the system.

SYSID (Define the Default MVS/BDT Node): Use the SYSID initialization statement to define the default MVS/Bulk Data Transfer (BDT) node for this JES3 complex. If the JES3 complex includes one or more MVS/BDT facilities (program product 5665-302), you must include this statement in the JES3 initialization stream. JES3 submits MVS/BDT commands and transactions to the MVS/BDT node defined by this statement unless otherwise specified on the command or transaction.

SYSOUT (SYSOUT Class Characteristics): Use the SYSOUT initialization statement to define SYSOUT class characteristics. The SYSOUT statement is required for each JES3

output class that requires other than TYPE=PRINT processing (JES3 initially sets all SYSOUT classes to TYPE=PRINT).

TRACK (Preformatted Spool Data Set): Use the TRACK initialization statement to replace a corresponding FORMAT statement in an initialization stream after the spool data set specified by the FORMAT statement has been formatted. The TRACK statement indicates that the corresponding data set has been formatted.

SYSOUT data sets

The SYSOUT statement parameters are applicable to all SYSOUT data sets created by JES3. Also, be aware that if the SYSOUT is associated with an output descriptor that is defined by the OUTPUT JCL statement or TSO OUTDES command, then the output characteristics are merged for SYSOUT on the HOLD queue.

5.4 Segmented initialization streams

- Allows initialization stream statements to be placed in different members
- Can group related information:
  - All GMS parameters
  - All tape devices
  - All RJP workstations and consoles
- Syntax: INCLUDE,MEMBER = member
- If JES3IN concatenated, only members from first data set included
- Up to 4 member levels are supported:
  - 1 primary member
  - 3 include members

Segmented initialization streams

JES3 allows initialization stream segmentation. That is, different sections of the initialization stream can be put into different members of the JES3IN data set and included in the primary initialization stream member. For example, all RJPWS definitions can be put into a member RJPWS and included in the primary member.

INCLUDE statement

The INCLUDE initialization statement is used to include a member in the initialization stream. The syntax of the INCLUDE statement:

```
INCLUDE,MEMBER=member
```

Where "member" is the name of the member in the data set specified on the JES3IN DD statement in the JES3 procedure to be included. Up to four member levels are supported. You can add the INCLUDE statement anywhere after the DYNALLOC statements. INCLUDE is not supported if the JES3IN DD statement is concatenated, and the members to be included are in a data set other than the first in the concatenation.The JES3IN DD statement must refer to a cataloged data set and be part of the cataloged JES3 start procedure.

INCLUDE statement members

You can have as many INCLUDE statements as you want in your initialization stream. You can have up to four member levels:

- The primary initialization member
- Up to 3 levels of INCLUDE statements.

For example, if the members of a PDS are JES3INXX, A, B, and C you can specify your INCLUDE initialization statements as follows:

```
Member JES3INXX             INCLUDE,member=A
Member A                    INCLUDE,member=B
Member B                    INCLUDE,member=C
Member C
```

5.5 Segmented initialization stream

```
DYNALLOC,...
  .

INCLUDE,MEMBER=JSAM     Include JSAM related statements up to ENDJSAM
INCLUDE,MEMBER=MAINS    Include main processor information (MAINPROC)
*
INCLUDE,MEMBER=BSCRJP   Include BSC related information
*                         (RJPLINE, RJPTERM, CONSOLE, DEVICE)
INCLUDE,MEMBER=CI       Include C/I information (CIPARM, HWSNAME, RESDSN)
INCLUDE,MEMBER=CONSOLES Include consoles information (CONSTD, MSGROUTE)
INCLUDE,MEMBER=DEADLINE Include deadline algorithms (DEADLINE)
INCLUDE,MEMBER=FSS      Include FSS related information (FSSDEF)
INCLUDE,MEMBER=GMS      Include GMS job classes, groups, select modes
*                         (CLASS, GROUP, SELECT)
INCLUDE,MEMBER=MDS      Include MDS information (SETPARAM, SETNAME, DYNALDSN)
INCLUDE,MEMBER=MISC     Include miscellaneous stuff(STANDARDS,SYSID,RESCTLBK)
INCLUDE,MEMBER=NJE      Include NJE definitions (NJERMT,DEVICE DTYPE=NJELINE)
INCLUDE,MEMBER=OUTSERV  Include output service definitions (OUTSERV, SYSOUT)
INCLUDE,MEMBER=SNARJP   Include SNA related information
*                         (COMMDEFN, COMPACT, RJPWS, CONSOLE, DEVICE)
INCLUDE,MEMBER=UR       Include unit-record devices
INCLUDE,MEMBER=TAPE     Include tape devices
INCLUDE,MEMBER=DASD     Include direct access devices
ENDINISH
```

Example of a segmented initialization stream

The following example illustrates how you could organize your JES3 initialization stream with INCLUDE statements.

The "member" you refer to is the name of the member in the data set specified on the JES3IN DD statement. The JES3IN DD statement must refer to a cataloged data set and be part of the cataloged JES3 start procedure. You can has as many INCLUDE statements as you wish in your initialization stream. You can have up to four member levels: the primary initialization member and up to 3 levels of INCLUDE statements.

The INCLUDE statement can occur anywhere after the DYNALLOC statements.

5.6 Initialization stream checker (IATUTIS)

- Detects most syntax errors and some logical errors in the initialization stream
- IATUTIS can run as a batch job or as a TSO job using a command list (CLIST)
- Step 1 gathers data for the initialization stream checker
- Step 2 runs the initialization stream checker.

```
//INITCHK    JOB 'ACCTINFO','NAME',MSGLEVEL=(1,1),
//                MSGCLASS=R,...
//IATUTIS    EXEC PGM=IATUTIS,PARM='P=1F1R'
//STEPLIB    DD DSN=SYS1.SIATLIB,DISP=SHR
//JESABEND   DD DUMMY
//JES3IN     DD DSN=INIT.PARMLIB(JES3IN00),DISP=SHR
//JES3OUT    DD SYSOUT=*
//STG1CODE   DD DSN=INSTALL.JES3,DISP=SHR
//IATPLBST   DD DSN=SYS1.PROCLIB,DISP=SHR
//
```

Testing the initialization stream

You use the JES3 initialization stream checker utility (IATUTIS) to test your JES3 initialization statements before you perform a hot start with refresh, warm, or cold start of JES3. The initialization stream checker detects most syntax errors and some logical errors in the initialization stream. You can run this utility as a batch job or as a TSO job using a command list (CLIST).

Steps to run the checker

You must perform two steps to use the initialization stream checker:

1. Step 1 gathers data for the initialization stream checker.

   Step 1 obtains hardware configuration data that the initialization stream checker uses to detect logical errors in the DEVICE, HWSNAME, RJPLINE, and SETNAME initialization statements. If you omit this step, the initialization stream checker performs only syntax checking.

   If you want the initialization stream checker to check for logical errors in your initialization stream, first obtain the MVS configuration data by running the hardware configuration definition (HCD) program. Otherwise, you can omit this step.

2. Step 2 runs the initialization stream checker.

   Step 2 runs the initialization stream checker. The initialization stream checker examines all initialization statements for correct syntax, except the DYNALLOC statements, and creates the JES3 intermediate initialization tables.

When you have obtained the configuration data for each system in your complex, submit the JCL that runs the initialization stream checker. Sample JCL is shown in the visual.

Using MVS hardware configuration definition (HCD)

The MVS hardware configuration definition (HCD) provides an interactive, panel-driven dialog that you is used to define hardware configurations. HCD also provides an option that creates JES3 initialization stream data (STG1CODE data) for use in step 2 of the initialization stream checker utility. Do the following steps to obtain the JES3 configuration data using HCD:

1. Select the "Activate Configuration Data" option on the HCD primary panel.
2. Select the "Create JES3 Initialization Stream Data" option from the JES3 panel.
3. Specify the following items on the JES3 panel:
   - Data set name (This name will be used later in the STG1CODE DD when running IATUTIS.)
   - Eight character MVS configuration identifier - available from **D IOS,CONFIG** output
   - Eligible device table (EDT) identifier

You can also create the initialization stream data using a batch job like the following:

```
//jobname JOB 'ACCTINFO','NAME',MSGLEVEL=(1,1),
// MSGCLASS=R
//BUILD EXEC PGM=CBDMGHCP,PARM='CONFIG,JES,cccccccc,ee'
//HCDIODFS DD DSN=SYS0.IODFxx,DISP=SHR
//HCDDECK DD DSN=dsname(member),DISP=OLD

where:
- xx is the suffix of the I/O definition file to be used as the basis for this data.
- cccccccc is the eight character MVS configuration identifier.
- ee is the eligible device table (EDT) identifier.
- dsname is the data set name to contain the output data (and to be used later in the
  STG1CODE DD when running IATUTIS).
- member is the member of the output data set to contain the data for a particular
  processor.
```

You must create this data for all processors defined in the initialization stream. Each member has the same name as one of these processors.

What the checker looks for

The initialization stream checker detects logical errors in the DEVICE, HWSNAME, RJPLINE, and SETNAME statements by comparing the JES3 initialization data with the configuration data that you obtain in step 1. The initialization stream checker can detect the following types of logical errors:

- Subgeneric splits; for example, devices defined to JES3 as belonging to one subgeneric group and defined to MVS as belonging to a different subgeneric group.
- Writer burster-trimmer-stacker mismatches for the 3800 printer.
- Missing or incorrect parameters required to define hardware.

5.7 INTDEBUG initialization statement

- All initialization statements and error messages
  - JESOUT data set
- Use INTDEBUG if a DUMP is required for analysis
- Monitors all occurrences of error message
- Place in stream before where error occurs
- Dump taken on number of occurrences of message
  - INTDEBUG,3,IAT3602 MAIN
  - INTDEBUG,1,IAT3251 BAD KEYWORD,
- JES3 INITIALIZATION STREAM PROCESSOR (IATUTIS)
  - Performs a syntax check of an initialization stream

INTDEBUG initialization statement

The INTDEBUG statement monitors a specified message during initialization. The messages are placed into the JES3OUT data set. These statements should be placed in the initialization stream before the place where the error statement occurred. They should be used to produce a dump when no other method of determining the error can be seen.

A dump can be taken depending on how many times the message occurred, as shown in the figure.

JES3 initialization stream checker

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

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

Testing the initialization stream

You use the JES3 initialization stream checker utility to test your JES3 initialization statements before you perform a hot start with refresh, warm, or cold start of JES3. The initialization stream checker detects most syntax errors and some logical errors in the

initialization stream. You can run this utility as a batch job or as a TSO job using a command list (CLIST).

Running the checker

You must perform two steps to use the initialization stream checker:

1. Step 1 obtains hardware configuration data that the initialization stream checker uses to detect logical errors in the DEVICE, HWSNAME, RJPLINE, and SETNAME initialization statements. If you omit this step, the initialization stream checker performs only syntax checking. The following sections contain sample JCL for obtaining configuration data.
2. Step 2 runs the initialization stream checker. The initialization stream checker examines all initialization statements for correct syntax, except the DYNALLOC statements, and creates the JES3 intermediate initialization tables.

Checker errors

The initialization stream checker detects logical errors in the DEVICE, HWSNAME, RJPLINE, and SETNAME statements by comparing the JES3 initialization data with the configuration data that you obtain in step 1. The initialization stream checker can detect the following types of logical errors:

- Subgeneric splits; for example, devices defined to JES3 as belonging to one subgeneric group and defined to MVS as belonging to a different subgeneric group.
- Writer burster-trimmer-stacker mismatches for the 3800 printer.
- Missing or incorrect parameters required to define hardware.

5.8 JES3 start procedure

```
//JES3  PROC JES=JES3,ID=01
//JES3  EXEC PGM=IATINTK,DPRTY=(15,15),TIME=1440,REGION=0M
//* ----------------------------------------------------*
//*   JES3 PROCEDURE:  JES3
//*       CHKPT:   SYS1.JES3CKPT
//*                SYS1.JES3CKP2
//*       DISK RDR:SYS1.JES3DR
//*       JCT:     SYS1.JES3JCT
//*       SPOOL:   SYS1.JES3SPL1
//*       JES3OUT: SYS1.JES3OUR
//*       DUMPS:   SYS1.JES3DUMP
//* ----------------------------------------------------*
//CHKPNT   DD DSN=SYS1.&JES.CKPT,DISP=SHR          R,S
//CHKPNT2  DD DSN=SYS1.&JES.CKP2,DISP=SHR          O,S
//JES3DRDS DD DSN=SYS1.JES3DR,DISP=SHR             O
//JES3JCT  DD DSN=SYS1.&JES.JCT,DISP=SHR           R,D,S
//SPOOL1   DD DSN=SYS1.&JES.SPL1,DISP=SHR          R,D,S
//JES3OUT  DD DSN=SYS1.&JES.OUT,DISP=SHR           R,D
//JES3SNAP DD DUMMY                                O,D
//SYSMDUMP DD DSN=SYS1.JES3DUMP,DISP=SHR
//IATPLBST DD DSN=SYS1.&PROCLIB1..PROCLIB,DISP=SHR R,D
//         DD DSN=SYS1.PROCLIB,DISP=SHR            R,D
//         DD DSN=ESA.SYS1.PROCLIB,DISP=SHR        R,D
//JESABEND DD DUMMY                                O,D
//JES3IN   DD DSN=SYS1.PARMLIB(JES3IN&ID),DISP=SHR R
//*R-Required, D-DYNALLOC eligible, S-Global/Locals share,
//* O-Optional
```

JES3 start procedure

The JES3 cataloged start procedure contains the job control language (JCL) statements needed to allocate the data sets required by JES3. IBM provides a basic cataloged start procedure shipped with JES3 and stored in JES3 member of the SYS1.PROCLIB data set.

The visual shows a sample of the JES3 procedure. This sample contains all of the required JCL statements.

If you introduce an error while changing the procedure, JES3 cannot be restarted. In this case, you must use another system (for example, the starter system) to change the procedure.

DD statements in procedure

JES3 cataloged start procedure DD-statements:

- CHKPNT and CHKPNT2: Defines the JES3 checkpoint data set(s). At least one of the two checkpoint data sets must be allocated and cataloged prior to JES3 operation. Each checkpoint data set must be allocated as a single extent which begins and ends on a cylinder boundary.
- JES3JCT: Defines the JES3 job control table (JCT) data set. This data set must be allocated and cataloged prior to JES3 operation. The data set must be large enough to accommodate the maximum number of JCT records to be allocated concurrently during normal system operation.

- spool1 to spoolnn: Defines the spool data sets. The installation selects the ddnames and data set names for these statements. The ddname for this statement must be the same ddname specified on the BADTRACK, FORMAT, or TRACK initialization statements. Spool data sets must be allocated and cataloged prior to JES3 operation. (These data sets may be any size; however, a minimum of 100 cylinders is recommended.)
- JES3OUT: Defines the data set upon which the JES3 initialization stream and initialization error messages are printed. This data set is de-allocated after initialization completes. You can tailor the block size (BLKSIZE) and logical record length (LRECL) values to improve performance. The values you can specify are device-dependent.
- JES3SNAP: Defines the data set used if JES3 produces a dump during a hot start, hot start with analysis, hot start with refresh, hot start with refresh and analysis, warm start, or warm start with analysis. This data set contains important diagnostic information. The information will not be available if you define JES3SNAP as a dummy data set.
- JESABEND: Defines the data set used for a JES3 formatted dump. If omitted, a formatted dump of JES3 control information will not be produced.
- SYSABEND, SYSUDUMP, or SYSMDUMP: Defines the data set for JES3 system dumps.
- IATPLBST: Defines the installation's standard procedure library concatenation.

  Note: Converter/Interpreter functional subsystems (C/I FSS) and the PROCLIB update function will obtain unit and volume information for the procedure libraries from the catalog. For these functions, JES3 ignores unit and volume information that you specify in the JES3 start-up procedure or on a DYNALLOC initialization statement.

- IATPLBnn: Defines the installation's other procedure library concatenation.

  Note: If a data set is dynamically allocated as both a JES3 DISKRDR data set and a JES3 PROCLIB data set, the UPDATE= parameter on the JES3 //*MAIN statement (JES3 procedure library update facility) can not be used to move the data set.

- JES3IN: Defines the data set containing the JES3 initialization stream. This data set must be a blocked or unblocked partitioned data set. The default initialization stream is read from SYS1.SIATSAMP(member JES3IN00).
- JES3DRDS: Defines the partitioned data sets containing input for the JES3 disk reader facility. The maximum block size for this data set is 3200. Concatenated data sets may be used.

Be aware that JES3 does not hold any data set ENQUEUE (major name=SYSDSN, minor name=dsname) while it is running regardless of the type of allocation (JCL or dynamic). JCL allocation ENQUEUEs are based on the DSI subparameter of the PPT parameter of the SCHEDxx member of SYS1.PARMLIB. Dynamic allocations by JES3 use an equivalent parameter to prevent a data set ENQUEUE.

5.9 JES3 start types

Hot start (H)
Hot start with analysis (HA)
Hot start with refresh (HR)
Hot start with refresh and analysis (HAR)
Cold start (C)
Warm start (W)
Warm start with analysis (WA)
Warm start to replace a spool data set (WR)
Warm start with analysis - replace a spool data set (WAR)

Local start

JES3 start types

The type of start you select depends on why you need to start or restart JES3. During phase 1 of JES3 initialization, a message is issued prompting the operator to enter the start type. The operator responds with a C, W, WA, WR, WAR, H or HA to initialize the processor that will contain the JES3 global address space.

```
IAT3011 SPECIFY JES3 START TYPE
*100 IAT3011 (C, L, H, HA, HR, HAR, W, WA, WR, WAR, OR CANCEL)
```

After the global is initialized, each additional processor can be initialized by issuing the **START JES3** command on that processor. When JES3 issues a message for the start type, the operator enters an L to initialize each subsequent processor as a local.

Hot start (H)

Use a hot start to start JES3 on the global:

- After an orderly shutdown
- After a JES3 failure on the global from which JES3 cannot automatically recover
- After an MVS failure that terminates all functions in the global
- To replace one of your checkpoint data sets.

Hot start with analysis (HA)

Use hot start with analysis after a hot start fails and/or you suspect problems with the JES3 job queue. A hot start with analysis performs the same functions as a hot start. In addition, JES3 also:

- Performs further job validation
- Gives the operator an opportunity to delete invalid jobs.

You are not required to IPL or to restart the local mains, although you may optionally do so.

Hot start with refresh (HR)

Use hot start with refresh when you want to change the initialization stream without having to IPL the entire complex. Performing a hot start with refresh avoids disrupting your system since you need only restart the JES3 global address space. You do not need to IPL all processors in the complex like you do with a warm start.

Hot start with refresh and analysis (HAR)

Use hot start with refresh and analysis after a hot start with refresh fails and/or you suspect problems with the JES3 job queue. A hot start with refresh and analysis performs the same functions as a hot start with refresh.

Warm start (W)

Use a warm start to restart JES3 on the global:

- After either type of hot start or hot start with refresh fails.
- After a failure of the global because of a software/hardware failure.
- When you want to change the initialization stream and the changes cannot be performed with a hot start with refresh.

Warm start with analysis (WA)

Use warm start with analysis when JES3 terminates abnormally and you suspect problems with the JES3 job queue, or when you change the level of JES3 and want to verify the integrity of the JES3 job queue across the change.

Warm start to replace a spool data set (WR)

Use warm start to replace a spool data set when you want to replace a spool data set. This type of start performs the same function as a warm start in addition to allowing you to replace a spool data set.

Warm start with analysis (WAR)

Use **warm start with analysis to replace** a spool data set when you suspect problems with the JES3 job queue and you want to replace a spool data set. This type of start performs the same function as a warm start.

Cold start (C)

Use a cold star**t** to start JES3 on the global when all types of warm starts are unsuccessful.

Local start (L)

Use a **local start** to start JES3 on a local main:

5.10 JES3 hot start with refresh

```
O  *RETURN
G *099 IAT3807 CONFIRM "RETURN" COMMAND FOR **GLOBAL** SC65 (CONTINUE (U) OR CANCEL)
O  R 99,U
G  IXZ0003I CONNECTION TO JESXCF COMPONENT BROKEN 167
G        GROUP WTSCPLX4 MEMBER SC65
G  IEF196I IEF142I JES3 JES3 - STEP WAS EXECUTED - COND CODE 0000
O  S JES3
G  IAT3040 STATUS OF JES3 PROCESSORS IN JESXCF GROUP WTSCPLX4
G  IAT3040 SC64    ( ), SC70    (UP), SC65    +  +
G  IAT3011 SPECIFY JES3 START TYPE
G *100 IAT3011 (C, L, H, HA, HR, HAR, W, WA, WR, WAR, OR CANCEL)
O  R 100,HR
G*101 IAT3012 SELECT JES3 INISH ORIGIN (N OR M=), AND OPTIONAL EXIT PARM (,P=) OR CANCEL
O  R 101,M=04
G  IAT4030 0001 SPOOL DATA SET  IN USE
G  IAT4075 MAXIMUM NUMBER OF JOBS IS LIMITED TO 0021461 BY JCT DATA SET SIZE
G  IXZ0001I CONNECTION TO JESXCF COMPONENT ESTABLISHED, 312
G        GROUP WTSCPLX4 MEMBER SC65
G*IAT3100 JES3 z1.5.0   SYSTEM  HOTSTART ON 2004.072 AS SC65
O *S JSS
L  IXZ0003I CONNECTION TO JESXCF COMPONENT BROKEN 014
L        GROUP WTSCPLX4 MEMBER SC70
L *IAT3098 JES3 LOCAL ON SC70 IS RESTARTING DUE TO CONFIGURATION CHANGE
L  IAT3040 STATUS OF JES3 PROCESSORS IN JESXCF GROUP WTSCPLX4
L  IAT3040 SC64    ( ), SC70    <  >, SC65    *UP*
L  IAT4030 0001 SPOOL DATA SET  IN USE
L  IXZ0001I CONNECTION TO JESXCF COMPONENT ESTABLISHED, 019
L        GROUP WTSCPLX4 MEMBER SC70
G IAT2645 ****** SC65     CONNECT COMPLETE ******
L  IAT2645 ****** SC70     CONNECT COMPLETE ******
```

JES3 start types

JES3 start types determine the type of initialization process to be performed:

Cold start Initialization reads the initialization stream to start the JES3 global processor for the first time or to restart after a warm start or hot start has failed. A cold start requires an MVS IPL on all processors. JES3 local processors can be started after the global processor cold start has completed. All jobs previously in the job queue are lost.

Warm start Initialization *reads the initialization stream* to restart the JES3 global processor after an orderly shutdown, after JES3 abnormally ends, or after a hot start fails. A warm start must be preceded by an MVS IPL of the global and local processors. Most job processing resumes after a warm start.

Hot start with refresh Initialization reads the initialization stream to restart the JES3 global processor after an orderly shutdown, after JES3 abnormally ends when you want to change the initialization stream without having to IPL the entire complex.

Performing a hot start with refresh avoids disrupting your system since you need only restart the JES3 global address space. You do not need to IPL all processors in the complex like you do with a warm start.

The visual shows the messages issued during JES3 global ho start with refresh.

Hot start Initialization is used to restart the JES3 global processor after an orderly shutdown or after JES3 ends abnormally. The initialization stream is not reread, therefore the initialization parameters remain the same as before JES3 ended. A hot start need not be preceded by an MVS IPL of the global and local processors. Most job processing resumes after a hot start (IPL) or continues through the hot start (no IPL).

Note: During JES3 hot starts and warm starts, JES3 performs validation checking on the jobs that were active when the system ended. JES3 examines the control blocks for the active jobs and if the information in the control blocks is sufficient for the jobs to continue, JES3 allows the jobs to resume processing. If not, JES3 asks the system operator whether or not the job should be cancelled. Depending on how critical the job is, the operator may have to stop JES3 initialization and then restart JES3.

Local start Initialization is identical to a hot start on the global processor except that global-only functions are not initialized.

Hot start with analysis Hot Start with Refresh and Analysis, and Warm Start with Analysis perform the same functions as the requested start type implies. In addition, JES3 also:

- Performs further job validation
- Gives the operator an opportunity to delete invalid jobs.

JES3 initialization processing

JES3 initialization processing occurs in four phases:

1. Phase one - Operator dialogue occurs as the operator is prompted to specify whether the start type is hot, warm, cold, or local. The checkpoint data set is examined to determine what types of JES3 starts are possible on each processor and what restrictions must be considered. All four types of JES3 starts include this phase.
2. Phase two - Spool initialization is another required segment of initialization processing. Control blocks and tables used specifically for the allocation of spool space are built. This phase involves spool data management initialization, spool space allocation initialization, and, for the global processor, reading of the JCT data set to construct the in-storage job queue (JQE structure). All four types of JES3 starts include the spool initialization phase.
3. Phase three - Initialization stream input occurs as the initialization stream is parsed and put on spool as intermediate control blocks. This phase occurs for cold, warm, and hot with refresh starts only.
4. Phase four - Table generation uses the output of phase three for the construction of in-storage data areas, for global and local communication initialization through the MVS JESXCF facility, and for the optional creation of the JES3 auxiliary address space. All four types of JES3 starts include this phase. For a hot start or a local start, table generation uses the output from the previous cold, warm, or hot with refresh start.

The last three phases of initialization execute under the job segment scheduler (JSS) entry on the FCT chain before JSS assumes the role it plays in regular JES3 processing. Parts of the JES3 initialization executes under the job segment scheduler (JSS) entry on the FCT chain before JSS assumes the role it plays in regular JES3 processing.

5.11 Hot start with refresh

- If hot start with refresh fails,
  - Next hot start uses information from last successful hot start with refresh, warm start, or cold start
- Not all statements and parameters processed
  - Those not processed are syntax checked
  - No detailed cross checking
  - Information from the previous warm or cold start used
  - Beware of dependencies between statements that can be changed and statements which cannot
    - See JES3 Initialization and Tuning Reference for details

Hot start with refresh

This is the preferred way to start JES3. Performing a hot start with refresh avoids disrupting your system since you need only restart the JES3 global address space. You do not need to IPL all processors in the complex like you do with a warm start.

JES3 requires at least a JES3 warm start (which implies a JES3 complex-wide restart with IPL of all processors) to change any keyword or parameter in the JES3 initialization stream. A hot start with refresh reads the initialization stream during a hot start without an IPL to allow many of the parameters to be changed (for example, FSSDEF, RJPWS, non-execution DEVICE).

JES3 start failures

If a hot start with refresh fails, any changes that were made to the initialization stream will not be in effect. That is, a subsequent hot start obtains the configuration specifications from the last successful cold, warm, or hot start with refresh.

Hot start with refresh considerations

A hot start with refresh will undo many of the changes made by operator commands that are related to initialization statements. For example, most parameters on the *F G command will no longer be in effect after a hot start with refresh; the information from the initialization statements will be used instead.

5.12 Hot start with refresh considerations

Initialization statements that can change during HR JES3 start

ACCOUNT	(A)	FSSDEF	(A,N)	RESDSN	(A,N)
BUFFER	(S,N)	GROUP	(A,N)	RJPLINE	(A)
CIPARM	(A,N)	HWSNAME	(A,N)	RJPTERM	(A)
CLASS	(A,N)	INCLUDE	(A)	RJPWS	(A)
COMMDEFN	(A)	INTDEBUG	(A)	SELECT	(A)
COMPACT	(A)	MAINPROC	(S,N)	SETACC	(A,N)
CONSOLE	(A,N)	MSGROUTE	(A)	SETNAME	(A,N)
DEADLINE	(A)	NJECONS	(A)	SETPARAM	(S,N)
DESTDEF	(A)	NJERMT	(S,N)	SETRES	(A)
DEVICE	(A,N)	NETSERV	(A)	SOCKET	(A)
DYNALDSN	(A)	OPTIONS	(S,N)	STANDARDS	(S,N)
DYNALLOC	(A)	OUTSERV	(A)	SYSID	(A)
ENDINISH	(A)	RESCTLBK	(A)	SYSOUT	(A,N)
ENDJSAM	(A)				

A - All parameters S - Subset of parameters N - See notes in *JES3 Initialization and Tuning Reference*

Statements that require C or W JES3 start to change

BADTRACK (Bypass Defective Tracks)
CONSTD (Console Service Standards)
FORMAT (Format Spool Data Set)
SPART (Spool Partition Definition)
TRACK (Preformatted Spool Data Set)

JES3 start type to modify initialization statements

You need to be careful when changing the initialization stream during a hot start with refresh, because not all of the initialization statements are processed. If you add or change one of the initialization statements that is not processed during a hot start with refresh, errors can occur when it is referenced by another initialization statement that is processed during a hot start with refresh.

The visual shows that during a hot start with refresh, because of dependencies between statements, many statements and parameters on statements cannot be changed. The visual is a summary of the initialization statements that are processed during a hot start with refresh. The A, S, and N on the visual indicate what statements are affected.

See *z/OS JES3 Initialization and Tuning Reference*, SA22-7550 for the specific statements and the restrictions in place concerning the dependencies.

5.13 Serialization of JES3 configuration

- For hot/refresh, global serializes against config changes
  - Prevent locals and C/I FSS from trying to read config info
- Serialization also done by locals and C/I FSS's
  - Prevent global from changing config while it's being read
  - IAT3072 main WAITING FOR EXCLUSIVE USE OF THE CONFIGURATION
  - IAT3073 ISSUE 'CANCEL' TO CANCEL WAIT FOR main
- SYSTEMS ENQ is issued:
  - Major: SYSZIAT:
  - Minor: CONFIG.CHANGE.checkpointvolser.checkpointdsname
    - Shared - for cold, warm, DSI, hot, local, C/I FSS starts
    - Exclusive - for hot starts with refresh
    - RNL=NO specified on ENQ/DEQ so SYSTEMS ENQ not converted to SYSTEM ENQ by SYSTEMS exclusion RNL

Serialization of JES3 configuration

During a hot start with refresh, the JES3 global attempts to obtain exclusive access to the JES3 configuration. This is necessary in order to make sure that any JES3 local processors or C/I FSS's are not trying to access the information in the configuration while the JES3 global is changing it. If the JES3 global cannot get access to the configuration, the following messages are issued:

```
IAT3072 main WAITING FOR EXCLUSIVE USE OF THE CONFIGURATION
IAT3073 ISSUE 'CANCEL' TO CANCEL WAIT FOR main
```

The configuration information is stored by JES3 on spool and the checkpoint data set from the initialization stream. The global JES3 processor serializes against the configuration data sets to prevent the JES3 local and C/I FSS address spaces to read the configuration information during update/modify attempts. Also, the local and C/I FSS address spaces serialize against the configuration information to prevent the global from changing the configuration while it is being processed.

SYSTEMS ENQ

Therefore, JES3 issues a SYSTEMS ENQ to allow for serialization against the JES3 configuration information. A major name of SYSZIAT, and a minor name of CONFIG.CHANGE.*checkpointvolser.checkpointdsname* is used for this ENQ, where:

checkpointvolser The volume serial number of the primary JES3 checkpoint data set

checkpointdsname The name of the primary JES3 checkpoint data set

Depending on the type of processing, one of two types of ENQ is issued:

SHARED For Cold, Warm, DSI, Hot, Local Starts, and C/I FSS starts

EXCLUSIVE For Hot Starts with Refresh and **`*F CONFIG`**

> **Note:** RNL=NO is used on the ENQ/DEQ macros to prevent a SYSTEMS ENQ from being converted to a SYSTEM ENQ by use of the SYSTEMS exclusion RNL facility.

JES3 system action

JES3 waits for the configuration to become available. If this is the JES3 global address space, and either a hot start with refresh is being performed, or a **`*F CONFIG`** command is being processed, and JES3 issues message IAT3073 to allow the operator to cancel the request.

```
IAT3072 mainname WAITING FOR EXCLUSIVE USE OF THE CONFIGURATION
IAT3073 ISSUE 'CANCEL' TO CANCEL WAIT FOR mainname
```

Operator response

The operator action to the IAT3072 message can be to determine who has control of the JES3 configuration, issue toe of the following commands:

```
D GRS,RES=(SYSZIAT,*)
D GRS,RES=(SYSZIAT, CONFIG.CHANGE*)
D GRS,RES=(SYSZIAT,CONFIG.CHANGE.volser.dsname)
```

5.14 JES3 configuration serialization

- If global cannot obtain the ENQ on hotstart / refresh
  - `IAT3072 mainname WAITING FOR EXCLUSIVE USE OF THE CONFIGURATION`
    `IAT3073 ISSUE 'CANCEL' TO CANCEL WAIT FOR mainname`
    - Messages remain highlighted
    - Issued after waiting 15 seconds
    - CANCEL allowed so that operator can cancel wait and perform hot start
- If local or C/I FSS cannot obtain the ENQ
  - `IAT3072 xxxx WAITING FOR SHARED USE OF THE CONFIGURATION`
    - Message not highlighted , but retrievable via D R,L
    - Issued after waiting 60 seconds
    - No CANCEL option since there is no alternative

Serialization of configuration

JES3 issues the IAT3072 message if the configuration serialization ENQ is not obtained. JES3 waits for the configuration to become available. If this is the JES3 global address space, and either a hot start with refresh is being performed, or a `*F CONFIG` command is being processed, and JES3 issues message IAT3073 to allow the operator to cancel the request. JES3 issues the IAT3073 message in conjunction with IAT3072 to allow the operator to cancel the wait for the configuration to become available.

A local processor is in the process of starting and waiting for the operator to respond to message IAT3011. This will prevent a global from performing a hot start with refresh because the local needs only shared access to the configuration while the global requires exclusive access. Before you perform a hot start with refresh, make sure that there are no outstanding IAT3011 messages for local processors.

An FSS address space is in the process of starting and has requested services from the JES3 global. For example, the FSS address space can be in the process of dynamically allocating a data set, and a request has been sent to the JES3 global to determine if the data set is available to be allocated. This prevents a global from performing a hot start with refresh because the FSS address space needs only shared access to the configuration while the global requires exclusive access.

The FSS address space will not release the configuration until the JES3 global has responded to the request. But the JES3 global can't respond to the request until it completes initialization.So either cancel the FSS address space to allow JES3 to continue, or respond CANCEL to IAT3073 to cancel the hot start with refresh, and then perform a hot start.

5.15 JES3 hot start with analysis

```
O  S JES3
G  IAT3040 STATUS OF JES3 PROCESSORS IN JESXCF GROUP WTSCPLX4
G  IAT3040 SC64    (  ), SC70    (UP), SC65    +  +
G  IAT3011 SPECIFY JES3 START TYPE
G*103 IAT3011 (C, L, H, HA, HR, HAR, W, WA, WR, WAR, OR CANCEL)
O  R 103,HA
G  IAT4030 0001 SPOOL DATA SET  IN USE
G  IAT4075 MAXIMUM NUMBER OF JOBS IS LIMITED TO 0021461 BY JCT DATA SET SIZE
G*104 IAT3146 ENTER JOB NUMBER OR JOB NAME TO BE PURGED PRIOR TO ANALYSIS (JOB NUMBER, JOB NAME, OR END)
O  R 104,DC
G*105 IAT3146 ENTER JOB NUMBER OR JOB NAME TO BE PURGED PRIOR TO ANALYSIS (JOB NUMBER, JOB NAME, OR END)
M  IAT6397   FCT JSS      (NODEVICE) HAS BEEN DISPATCHED CONTINUOUSLY FOR:
M  IAT6398     00000 HOURS  00 MINUTES  31 SECONDS
M  IAT6415 at 20FC67BE in LOADMOD=IATINAL , EPNAME=IATINAL ,EPADDR=20FC6688,LEN=000978
M*106 IAT6410 Reply 'JES3    ' to fail the FCT or 'nnn' to ask later
O  R 106,180
O  R 105,26674
G*107 IAT3146 ENTER JOB NUMBER OR JOB NAME TO BE PURGED PRIOR TO ANALYSIS (JOB NUMBER, JOB NAME, OR END)
O  R 107,END
G*108 IAT3151 DATA LOST AND/OR JOB(S) DELETED DURING ANALYSIS PROCESSING.  DO YOU WISH TO PROCEED?            .
(CONTINUE OR CANCEL)
O  R 108,CONTINUE
G  IAT4133 JOB WW6VAINA (JOB26674) HAS BEEN DELETED DUE TO OPERATOR ANALYSIS REQUEST
G  IAT4133 JOB DC       (JOB27270) HAS BEEN DELETED DUE TO OPERATOR ANALYSIS REQUEST
G  IXZ0001I CONNECTION TO JESXCF COMPONENT ESTABLISHED, 975
G         GROUP WTSCPLX4 MEMBER SC65
G  IAT3102 POSSIBLE IMPACT OF FUNCTION  DETECTED DURING INITIALIZATION, SEE JES3OUT. JES3 CONTINUING
G *IAT3100 JES3 z1.5.0   SYSTEM  HOTSTART ON 2004.072 AS SC65
O *S JSS
G  IAT6394 MONITOR FUNCTION ACTIVE
G  IAT7131 NJECONS NOW ACTIVE
G  IAT9225 NJERDR IS ACTIVE
G  IAT2645 ****** SC65     CONNECT COMPLETE ******
```

Current system is a global:	**+status+**
Global on another system:	***status***
Current system is a local:	**<status>**
Local on another system:	**(status)**

JES3 hot start with analysis

A JES3 on the global with analysis analyzes the JES3 job queue, cancels jobs that have control block errors, and initiates snap dumps of the incorrect control blocks.

You can use hot start with analysis to restart JES3 if either the JES3 address space or MVS fails on the global main and you suspect problems with the JES3 spool. If a normal hot start fails, use hot start with analysis.

After a power failure, you should perform a hot start with analysis, a warm start with analysis, or a hot start with refresh and analysis.

During a hot start with analysis, you can remove a spool data set from the system (as long as the data set does not contain the checkpointed initialization stream) or reinstate a spool data set that was previously removed.

In addition to restarting JES3 on the global processor, hot start with analysis also does the following:

- Analyzes the JES3 job queue and responds as follows for jobs that have control block errors:
  - Cancels jobs that are not running on a processor
  - Marks jobs that are running on a processor for deletion
  - Initiates snap dumps of the incorrect control blocks

When you perform a hot start with analysis, you do not have to perform an MVS IPL on either the global or the local mains and you do not have to restart JES3 on the local mains.

5.16 *S JSS - start job scheduling

- Start job scheduling after JES3 message IAT3100
  - Initialization processing is complete on the global:
    - IAT3100 JES3 V1R6 SYSTEM HOT START ON 2004.140 AS SC70
- Before starting job scheduling, you can use JES3 commands to:
  - Cancel jobs
  - Change the status of jobs
  - Change the status of devices
- After you enter the ***S JSS** command, ensure that the global is varied
  - *V,main,ON
    - IAT2645 ***** main CONNECT COMPLETE *****

Start job scheduling

When you are satisfied that the system is ready to begin processing, enter a **`*S JSS`** command to start job scheduling.

Before starting job scheduling, you can use JES3 commands to cancel jobs, change the status of jobs, and change the status of devices. During a hot start with analysis, you can release jobs in spool hold status after reinstating a spool data set that contains data for the jobs, and you can vary devices online or offline. You can make adjustments for any system data that might have been lost during the restart. You can also make any changes to the system that were made before a hot start or a warm start but did not remain in effect after the restart.

Connect mains

After you enter the **`*S JSS`** command, ensure that the global is varied online to JES3. If it is not, enter the **`*F V,main,ON`** command to vary the processor online, ensuring that the subsystem interface, the MVS system commands, and the system log are initialized. JES3 then issues the following message:

```
IAT2645     ***** main CONNECT COMPLETE *****
```

5.17 Automatic start - no *S JSS

- If you want JES3 functions to be available after JES3 initialization without requiring the *S,JSS command
  - Include the PARM=NOREQ parameter in the JES3 start procedure

//IEFPROC EXEC PGM=IATINTK,DPRTY=(15,15),PARM=NOREQ

PARM=NOREQ

PARM=NOREQ specifies that JES3 global will start JES automatically if you want JES3 functions to be available after JES3 initialization without requiring the **`*S,JSS`** command. Place this in the JES3 start procedure. This avoids having operators forget to do the *S JSS command when restarting JES3.

If you want JES3 functions to be available after JES3 initialization without requiring the **`*S,JSS`** command, include the PARM=NOREQ parameter as shown below.

```
//IEFPROC   EXEC  PGM=IATINTK,DPRTY=(15,15),PARM=NOREQ
 //STEPLIB   DD    DISP=SHR,DSN=SYS1.SIATLIB
 //CHKPNT    DD    DISP=OLD,DSN=SYS1.JES3CKPT
 //CHKPNT2   DD    DISP=OLD,DSN=SYS1.JS3CKPT2
 //JES3JCT   DD    DISP=OLD,DSN=dsn
 //spool1    DD    DISP=OLD,DSN=dsn
       .
       .
       .
 //spoolnn   DD    DISP=OLD,DSN=dsn
 //JES3OUT   DD    UNIT=OOE
```

JES3 start command

To nullify the parameter you can specify **`S JES3,PARM=`** on the start command.

5.18 MONITOR DSP

- MONITOR DSP monitors a resource or queue based on information you specify
- JES3 starts the MONITOR DSP and monitors various queues and resources automatically
- The information that the monitor DSP displays includes both of the following:
  - The queue or resource the job or FCT is waiting for
  - The amount of time the job or FCT has been waiting
- This DSP provides you with the ability to monitor how long a job or FCT has been waiting for a specific JES3 function or resource
  - *S MONITOR,DISPLAY

MONITOR DSP

The Monitor FCT (IATGRMN) is responsible for monitoring certain resources and queues based on parameters specified by the installation, and issuing messages to the operator if someone is waiting more than a specified period of time.

The **`*X MONITOR`** command is issued automatically by JES3 on every global restart if the MONITOR DSP is not already active. Therefore you will not need to issue this command unless you have previously issued the **`*C MONITOR`** command.

The MONITOR DSP's mainline routine is invoked when a **`*X MONITOR`** command is issued by the operator. The mainline routine is responsible for determining whether there are any resources or queues to be monitored, and for setting up and invoking the appropriate routines to do the actual work.

The monitor DSP provides you with the ability to monitor how long a job or FCT has been waiting for a specific JES3 function or resource. For example, if you want to know when a job has been waiting for a CI DSP for more than five minutes, you can set the monitor DSP to issue a message when five minutes have elapsed. The following is a chronological example of the monitor DSP in use:

- You issue the **`*S MONITOR,DISPLAY`** command to examine the current monitoring parameters.

The system issues the following messages:

```
*S MONITOR DISPLAY
IAT6399 ID       INTERVAL THRESHOLD COUNT SUMMARY STATUS
IAT6400 RESOURCE   005       001     ALL   YES    ACTIVE
IAT6400 LOCATE     005       001     ALL   YES    ACTIVE
IAT6400 CIFSS      005       001     ALL   YES    ACTIVE
IAT6400 SYSSELQ    005       005     ALL   YES    ACTIVE
IAT6400 ALLOCQ     000       000     ALL   YES    INACTIVE
IAT6400 VERIFYQ    000       000     ALL   YES    INACTIVE
IAT6400 SYSVERQ    005       001     ALL   YES    ACTIVE
IAT6400 DSPWAIT    005       005     ALL   ONLY   ACTIVE
IAT6400 JSSWAIT    005       005     ALL   ONLY   ACTIVE
IAT6400 IOWAIT     030       001     ALL   YES    ACTIVE
```

During the JES3 HA start, a prompt to the operator happens to spend over 30 seconds before a response was given to the IAT3146 message. The MONITOR DSP detected an long IOWAIT (interval 30) wait and issued the IAT6410 message soliciting for operator response.

Operator response

Determine if a FCT is legitimately monopolizing the JES3 Nuc or Aux tasks. If you think there is a reason for the same FCT to be active for an extended period of time, you can ignore this message or respond with a number of seconds (*nnn*) to request that JES3 wait for that period of time before issuing another message (if it finds that the same FCT is still active).

If you respond with JES3, system name of C/I FSS name, as displayed in the WTOR, JES3 will try to terminate the active FCT and eventually invoke its JESTAE recovery.

You may have to repeat the response several times if the FCT attempts to retry and enters a loop or MVS WAIT again. In that case, you may have to terminate JES3 by either responding to message IAT3822, if one is displayed, or by issuing the **FORCE JES3,ARM** command.

If you feel the default interval is too short causing the message to appear too frequently, you can adjust it by issuing the **F JES3,INT=*n*** command. *n* specifies the interval in seconds.

5.19 DSI processing

```
Assign JES3 global function to a JES3 local processor

On global
  *X DSI
 *IAT0920 DSI - CHECK GLOBAL DSI PROCEDURE FOR SC65
  *S DSI

On a local
  *X DSI
 *IAT0915 DSI - REVIEW LOCAL DSI PROCEDURE FOR SC70
  *S DSI
 *IAT0900 DSI - SWITCH GLOBAL DEVICES
  *S DSI
 *IAT0905 DSI - STARTED FOR SC70
```

DSI processing

Dynamic system interchange (DSI) is a process by which the JES3 global function can be assigned to a JES3 local processor, which then becomes the new JES3 global processor. DSI can be used when:

- The global processor is not active.
- The installation wants a local processor to assume the role of the global processor.

If the global processor is not active, the operator can invoke DSI to keep the complex running. Once DSI is complete, JES3 on the old global processor can be reinitialized as a local processor without an intervening IPL, once it becomes available for reinitialization.

If the global processor is active but the installation requires that another processor be assigned as the global processor, the operator can invoke DSI. This procedure could be used for such reasons as scheduled preventive maintenance or for alternate processor utilization.

Disable old global

When your global is inactive and you need to perform a DSI on global, you should disable the global by performing a system reset. A system reset causes MVS and JES3 to terminate. All jobs that were executing on the global are lost; JES3 will reschedule them.

All FSS address spaces on the global are also lost. You must restart all FSS address spaces that were executing at the time of the system reset.

If your global is active but you want another main to become the global, you must disable the old global by entering a **`*X DSI`** and **`*S DSI`** command on the MCS console attached to the old global. Before entering the **`*X DSI`** command, you must complete all reconfiguration tasks that require JES3, such as stopping RJP to disable communication lines.

Call DSI on local

DSI is started by entering the **`*X DSI`** command on the master console of the local main that you want to make the new global. All FSSs that were executing on local mains at the time of the DSI, including the local that is to become the new global, continue processing during and after the DSI.

If you disable the global using a **`*X DSI`** command, then any output writer FSSs that were active on the old global remain active when the new global attempts to connect to the old global. However, if the old global fails as a result of an IPL or system reset, all output writer FSSs that were active on the old global end.

If a failure occurs during DSI, you must perform a warm start.

IAT0920

DSI has been called on global processor main, and the operator is requested to review the global DSI procedure. Operator should review any installation guidelines for dynamic system interchange. When finished, enter **`*S,DSI`** to continue or *`C,DSI`** to end DSI.

IAT0915

Operator should review any installation guidelines or procedures for dynamic system interchange. When this review is complete, enter the **`*S,DSI`** command to continue, or the **`*C,DSI`** command to end.

IAT0900

Operator should set the switching devices to enable channel paths from the new global to all JES3 devices and mains as required. When this function is complete, enter the **`*S,DSI`** command to continue, or the **`*C,DSI`** command to end DSI.

IAT0905

DSI is active. The active local JES3 system main was ended. JES3 was reinitialized in global hot start mode. The operator should proceed with a JES3 global hot start. Continue normal operation on the new global. If the old global was disabled by pressing SYSTEM RESET, JES3 on the old global can be initialized by performing a local start after an MVS IPL.

If the old global was disabled by entering **`*X,DSI`** and then **`JES`**, the old global can be initialized as a local without an intervening MVS IPL.

5.20 DSI processing

```
JES3 messages for the DSI restart of the local

IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT , SBOX08,3F00.
IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT2, SBOX09,2517.
IXZ0003I CONNECTION TO JESXCF COMPONENT BROKEN 752
            GROUP WTSCPLX4 MEMBER SC70
IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT , SBOX08,3F00.

IAT3040 STATUS OF JES3 PROCESSORS IN JESXCF GROUP WTSCPLX4
IAT3040 SC64    ( ), SC70    +DS+, SC65    (DS)

 IAT4030 0001 SPOOL DATA SET  IN USE
IXZ0001I CONNECTION TO JESXCF COMPONENT ESTABLISHED, 762
            GROUP WTSCPLX4 MEMBER SC70

*IAT3100 JES3 z1.5.0   SYSTEM  HOTSTART ON 2004.085 AS SC70
```

DSI processing and startup of new global

JES3 messages begin to appear for the restart of the local as the new global.

The +DS+ status indicates that the local (SC70) is the current global in message IAT3040. The DS status (SC65) indicates the main has not completed dynamic system interchange.

When you specify a start type, the local initializes as the new global and message *IAT3100 appears.

Jobs with //*MAIN statements

Jobs queued as a result of the SYSTEM=JGLOBAL or SYSTEM=JLOCAL parameters on //*MAIN statements prior to DSI are not requeued to the new global or local processor after the DSI. The jobs remain queued on the processor on which they were previously queued. If that processor becomes available, the jobs can execute.

Flush old local

If you do not expect to re-IPL JES3 on the old global processor immediately after a DSI, to ensure jobs registered with automatic restart management on the old global processor are restarted on either the new global processor or another active local processor, you must issue the **`*S,main,FLUSH`** command.

5.21 Special JES3 commands

- DSP failures and recovery
  - JES3 DSP fails
  - JES3 JESTAE macro
- *DUMP
- *RETURN
- *FAIL
- *X VARYL

DSP failures and recovery

Abnormal termination recovery involves both ABEND and failsoft processing. Failsoft minimizes the impact of failures that occur during dynamic support program (DSP) execution by allowing the DSP to execute a recovery routine. This recovery routine determines whether the DSP can resume or has to be terminated.

To receive control when an ABEND occurs, DSPs issue the JESTAE macro to define a DSP abnormal exit routine. The JESTAE exit routine performs pre-termination functions and diagnoses the error. The exit routine must determine whether abnormal termination should continue for the function (FCT) or whether normal processing, as represented by the function, can be resumed at some retry point.

At the time of a JES3 failure, a console message is displayed. The failure messages are assigned a unique failure numbers. Detailed failure information is also recorded into the hardcopy log. No operator intervention is required unless WANTDUMP=ASK is coded on the OPTIONS initialization statement (or WANTDUMP=YES is coded but WANTDUMP=ASK is assumed because the number of dumps in the specified interval has exceeded the specified limit).

A JES3 failure display:

```
*IAT3800 SVC DUMP IN PROGRESS BY THE JES3 ADDRESS SPACE
 IEA794I SVC DUMP HAS CAPTURED: 065
 DUMPID=001 REQUESTED BY JOB (JES3    )
 DUMP TITLE=JES3 1.10.0 FLN0=001 TST      FCT=255F1200  S0C1-000
            00001 IN IATST00  PSW=071C00008000DFD2 301/1041
```

```
IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT ,SBOX08,8074.
IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT2,SBOX09,8312.
IAT3713  FAILURE LOGOUT 068
IAT3713  ************************************************************
IAT3713  ************************************************************
IAT3713  DATE = 2008301 TIME = 1041460   JES3 z1.10.0
IAT3713  JES3 FAILURE NUMBER = 0001   ABENDED  SOC1
IAT3713  FAILURE REASON CODE = 00000001
IAT3713  ACTIVE FCT = TST        DEVICE = NONE      FCT FAIL NO = 0001
IAT3713  JOB TST      (JOB24980)
IAT3713  MODULE = IATST00 ...... MOD BASE = 0000DFD0  DISP = 000002
IAT3713  APAR NUMBER = .......   PTF NUMBER = .......
IAT3713  PSW AT TIME OF FAILURE  071C0000 8000DFD2  ILC  2
IAT3713  THE FAILING INSTRUCTION IS   0000
IAT3713  REGISTERS AT TIME OF FAILURE
IAT3713  REGS  0- 3   0000DFD0   258FA8FC   00000000   24F52350
IAT3713  REGS  4- 7   00000000   00000000   00000000   24F52340
IAT3713  REGS  8-11   00000000   00000000   24F44708   255F1200
IAT3713  REGS 12-15   24F04000   0000DD20   24F447BC   0000DFD0
IAT3713  ************************************************************
IAT3713  ************************************************************
IAT3702 TST                ABENDED  SOC1 - JES3 FAILURE NO. 0001
IAT3709 TST                IS BEING TERMINATED
IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT ,SBOX08,8074.
IAT6369  JES3 WAITING FOR CHECKPOINT DATA SET RESERVE -  CHKPNT2,SBOX09,8312.
IAT7450 JOB TST      (JOB24980) PURGED
```

Special JES3 commands

If a ***DUMP** was entered as an operator command, the JES3 tables are formatted, and MVS takes a dump of JES3, or an unformatted dump is taken to a MVS system dump data set, depending on how the DUMP=parameter on the OPTIONS initialization statement is specified, after which JES3 is terminated. Use the *DUMP to produce such an abend dump intentionally. The *DUMP command is not allowed from the input stream.

If a ***RETURN** was entered, JES3 is terminated without a dump.

When a program check or a ***FAIL** operator command is encountered, an FSWA is created and then the error is properly logged to the console error destination; that is, a unique failure number is assigned, the user/system ABEND code, the APARNUM/PTFNUM from the failing module, and the program status word (PSW) and registers at time of failure are displayed. Also displayed are the failing instruction, the location, the active FCT, and the module name, base and displacement as well as the calling sequence.

Use the ***X VARYL** command to unassign an IBM 3480 or 3490 tape drive from JES3 local domains. The VARYL dynamic support program (DSP) unassigns an IBM 3480 or 3490 tape drive from JES3 local mains. You must invoke the VARYL DSP from each local main to which the IBM 3480 or 3490 is assigned before using the device for a stand-alone dump.

5.22 OPTIONS initialization statement

```
OPTIONS,DUMP={PRDMP|MVS|JES}
        ,SDI={ON|OFF}
        ,DUMPLINS={nnnnnn|24576}
        ,WANTDUMP={YES|NO|ASK}
        ,INTRDR={20|nnn}
        ,JOBNO=({1,9999,9999})
        ,MT={ON|OFF}
        ,SE={nn|10}
        ,XCFGRPNM=groupname
```

OPTIONS statement

Use the OPTIONS initialization statement to specify:

- The type of MVS system dump to be taken, if needed.
- Whether or not a dump should be taken when a termination condition exists.
- The job numbering limits for JES3 jobs.
- Whether you want the writer output multitasking facility enabled or disabled.
- The number of scheduler elements needed to support the largest job that will be run in the JES3 complex.

DUMP= parameter

This parameter indicates the type of MVS system dump to be taken in the event of an abnormal termination of JES3 or program check. The JES3 control blocks, if written, are formatted and always written to the JESABEND data set.

PRDMP Specifies that a dump of main storage is to be written to the MVS SYS1.DUMPxx data set. To print this dump, use the MVS interactive problem control system (IPCS).

MVS Specifies that the MVS system dump written to the SYSUDUMP or SYSABEND data set contains the MVS nucleus and SQA as well as the MVS JES3-related control blocks and JES3 region.

JES Specifies that the MVS system dump written to the SYSUDUMP or SYSABEND data set contains only the MVS JES3-related control blocks and JES3 region.

5.23 OPTIONS statement - WANTDUMP parameter

- WANTDUMP= YES | NO | ASK
  - New keywords with APAR OW48161 - WANTDUMP=YES
    - LIMIT= 2-9 | 3 | 0 - Defines number of abends within INTERVAL before JES3 issues a WTOR
    - INTERVAL= 2-60 | 10 | 0 - Sets the time period for LIMIT
  - *F WANTDUMP=YES,L=limit
  - *F WANTDUMP=YES,I= interval
  - *F WDMP=NO
  - *F WANTDUMP=ASK

WANTDUMP= parameter

The WANTDUMP= parameter on the OPTIONS initialization statement specifies whether a dump should be taken. The options are NO, YES, and ASK. The ASK value results in the system asking the operator at the time of failure if he requests a dump. If the WANTDUMP= NO is specified or the operator does not request a dump, then no dump is taken. If the WANTDUMP= YES is specified or the operator requests a dump, one of the following occurs (depending on how the DUMP= parameter on the OPTIONS initialization statement is specified):

- The JES3 tables are formatted and MVS takes a dump of JES3.
- An unformatted dump is taken to the SYS1.DUMPxx data set.

Use the **`*F WANTDUMP`** command to change the settings of the WANTDUMP parameter on the OPTIONS statement.

YES Specifies that a dump should be taken when a failure occurs.

Two keywords, INTERVAL= and LIMIT=, have been provided on the OPTIONS initialization statements as additional control when WANTDUMP=YES is specified.

LIMIT The maximum number of failures within the interval before JES3 temporarily changes to WANTDUMP=ASK. The acceptable value is a number between 2 and 10 or a zero. The default value is 3. Zero (0) indicates no limit will be used.

INTERVAL A time period, in minutes, that will be used as the basis for the limit. The interval is a sliding window that ends at the time of the latest JES3 failure.

The acceptable value is a number between 2 and 60 or a zero. The default value is 10. Zero (0) indicates no limit will be used.

If you run with WANTDUMP=YES, JES3 will override it by temporarily changing to WANTDUMP=ASK if more than LIMIT failures occur within the previous INTERVAL minutes.

NO Specifies that no dump should be taken when a failure occurs.

ASK Specifies that, when a failure occurs, the operator is to be given the choice of specifying whether a dump is to be taken.

Using dumps during diagnosis

You need to consider the following recommendation in securing dump information for your diagnosis efforts:

- JES3 WANTDUMP recommendations

  IBM recommends that you allow the WANTDUMP option on the STANDARDS initialization parameter to default to "YES" instead of setting it to "ASK". Using the default of "YES" allows the system to determine what action to take when a JES3 failure condition occurs. In today's sysplex environment setting this parameter to "ASK" can cause delays in operations because the JES3 address space essentially stops functioning until you respond to the IAT3714 message. Also, certain portions of the dump, such as the system trace, are invalid because the system continues processing until you response to message IAT3714.

- Taking JES3 Dumps

  If you need to take a dump of JES3 and its related address spaces, such as FSS address spaces, consider the following hints:

  – Always include the SDATA parameters RGN, LSQA, LPA, SUM, CSA, NUC, PSA, and SWA. JES3 stores most of its control blocks and modules in the private region.
  – If you run SAPI applications, you can include SAPI dataspaces (JES3SAPI) by adding DSPNAME ('JES3'.JES3SAPI) to your dump command.

  If you need to take a dump of JESXCF, always include its dataspaces. See Information APAR II09383 for instructions on including this information.

- DUMPCORE

  DUMPCORE is an alternative to a dump for many problems associated with JES3. DUMPCORE is a tool that allows you to:

  – Examine control blocks
  – Set traps
  – Find modules
  – Zap storage

5.24 JES3 abends and DM codes

- For JES3 address space abends, examine the following
  - User completion codes
  - JES3 DM abend codes
  - System completion codes
- Following data required to diagnose abends in JES3
  - A dump produced by JES3
  - SYSLOG
- Depending on nature of abend the following additional items may be required:
  - The JCL/joblog
  - Your Initialization statements
  - JES3OUT

JES3 DM codes

The dynamic support program (DSP) failsoft feature of JES3 allows a DSP to abend without ending JES3. When a DSP encounters an error, it issues a FAILDSP macro. The FAILDSP macro ends the JES3 function but allows other functions to continue processing jobs.

The FAILDSP macro provides the user with failure codes that identify the error. The heading of the resultant dump may contain a system completion code, user completion code, and/or a JES3 failsoft DM code.

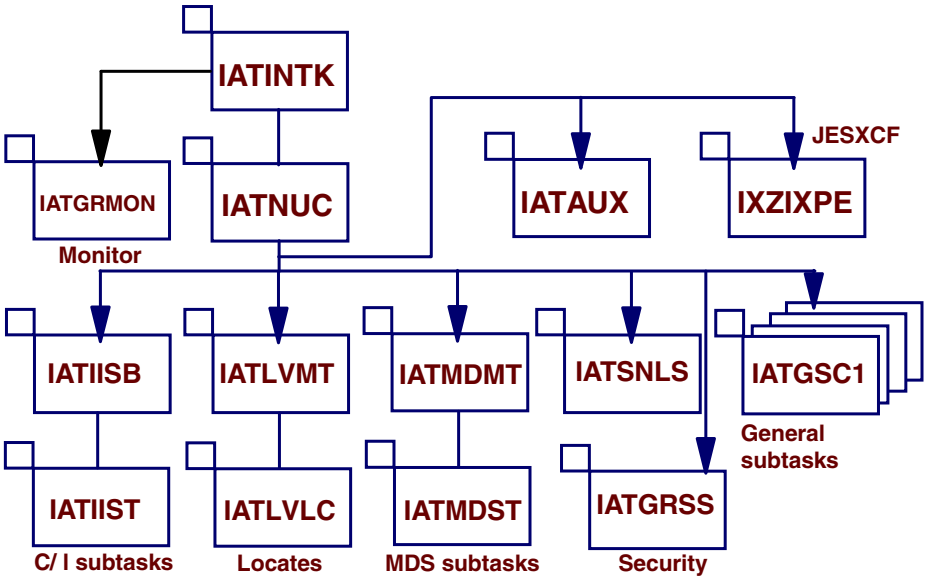
JES3 documentation

All JES3 failsoft DM codes are described in the *z/OS JES3 Diagnosis Reference*, GA22-7548 document. A DMxxx code appears as a user abend code (Uxxx) to the base control program (BCP).

The JES3 system completion codes appear in *z/OS MVS System Codes*, SA22-7626 document.

z/OS JES3 Diagnosis, GA22-7547 document provides information for debugging JES3 and installation-written extensions of JES3. It describes the tools that JES3 users can use for debugging.

5.25 JES3 initialization task - IATINTK



IATINTK processing:

- Ensures JES3 is the primary subsystem and no more than one is started
- Initializes the checkpoint access method
- Initialize subsystem vector table
- Determines the type of JES3 restart by reading all JES3 checkpoint records
- Attaches the JES3 resident nucleus (IATNUC)

JES3 initialization task - IATINTK

JES3 can be started wit a START operator command or by the system during an IPL (the START is executed automatically if it is included in the master JCL, if START=YES is specified on the IEFSSNxx parmlib member primary subsystem definition, or the START command is in COMMNDxx parmlib member). MSTR subsystem will process the START JES3 command which requires that the JES3 start procedure is in the MSTJCLxx IEFPDSI DD concatenation. The START JES3 command causes MVS to:

- Create an address space for JES3
- Execute the JES3 start procedure obtained from the procedure library data set
- Attach the initialization task (IATINTK) in the JES3 address space

IATINTK program properties

JES3 module IATINTK is given control with the following program propertied attributes (defined in the IEFSDPPT linklib member):

- Non-cancellable
- Protection key 1(one)
- Non-swappable
- System task and not timed (system task programs must be a one-step job started by a START or MOUNT command)
- No data set integrity

IATINTK processing

Module IATINTK enters supervisor state. It then initializes an SSOB and SSIB to issue a subsystem verify request to the Master subsystem. A subsystem verify request is issued to check if JES3 is a defined subsystem and to get the address of its SSCT. Also IATINTK ensures JES3 is the primary subsystem, validity checks the SVT (if one exists), and ensures that no more than one JES3 primary subsystem is started at a time.

Following the above processing, IATINTK loads and branches to IATGRCK for initialization of the checkpoint access method. Next, it loads and branches to IATINGL.

IATINGL processing

IATINGL invokes IATINSV for SSVT initialization and then determines the type of JES3 restart to be performed by reading all JES3 checkpoint records from the checkpoint data set(s) which may restrict the start type. The status of each processor in the complex is displayed and then module IATINGS is invoked to communicate with the operator and to read the DYNALLOC statements from the JES3 initialization stream.

Data sets described by the dynamic allocation checkpoint record are dynamically allocated.

The complex status record is updated to indicate that the active processor is in initialization and the record is written back to the checkpoint data set(s).

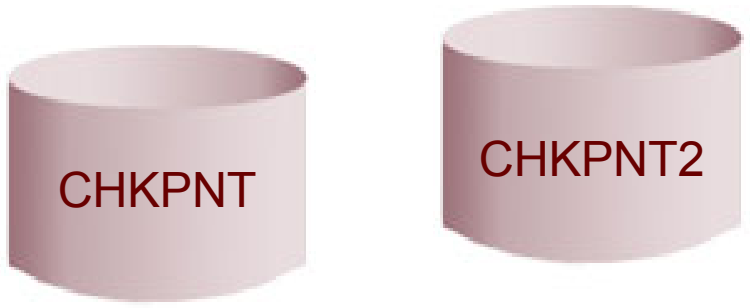
IATINGL returns to IATINTK. Upon return from IATINGL, the JES3 resident nucleus (IATNUC) is attached. Next, an SMF 43 record (subsystem start) is written. IATGRMON is attached to perform monitoring of the Nuc and Aux tasks. IATINTK then waits for IATNUC task termination or an operator command for the monitor (**`MODIFY(F) JES3,cmd`**).

JES3 nucleus

The JES3 nucleus contains modules needed throughout JES3 processing. When the nucleus is attached, control is passed to module IATINIT (which was specified as IATNUC's entry point when the nucleus was link-edited).

5.26 JES3 checkpoint data sets

- Defined in the following methods:
  - JES3 procedure or
  - Initialization stream (DYNALLOC)
- Contains 9 records
  - 1 cylinder is required
  - DDNAMES must be CHKPNT and CHKPNT2



Checkpoint data sets

The JES3 checkpoint data set(s) let(s) you warm start or hot start the JES3 system with little or no loss of system information.

The checkpoint must be allocated in the JES3 start procedure (1 cylinder is enough). The DDNAMEs must be CHKPNT and CHKPNT2. At least one of these data sets must be available to JES3 during initialization processing. The checkpoint is not heavily used, but do not place it on a spool volume.

Checkpoint considerations

You can add or replace either checkpoint data set over a JES3 restart of any kind with no effect on JES3 processing. Both checkpoint data sets contain identical information; to ensure against loss of checkpointed data, allocate both data sets.

The following rules and requirements apply to the checkpoint data set:

- The checkpoint data set must be allocated as a single extent.
- The extent must begin and end on a cylinder boundary.
- The number of 4K records must be calculated for each record type.
- The number of tracks must be calculated for each record type.
- Additional tracks should be added to the total for all record types for error recovery and possible expansion of a record type over time.

Accessing the checkpoint data set

Input or output to the checkpoint data set may be done only through the IATXCKPT macro interface. The following rules must be heeded when accessing the checkpoint record.

- Access to the checkpoint from the JES3 address space is serialized by the access method itself.
- The IATXCKPT macro to READ, WRITE, or PURGE records must be used.
- Creation of new records can only be done by the JES3 address space on the global processor.
- Requests to read or write existing records may be done in any address space or processor.
- Requests to PURGE existing records may be done only from the global JES3 address space.

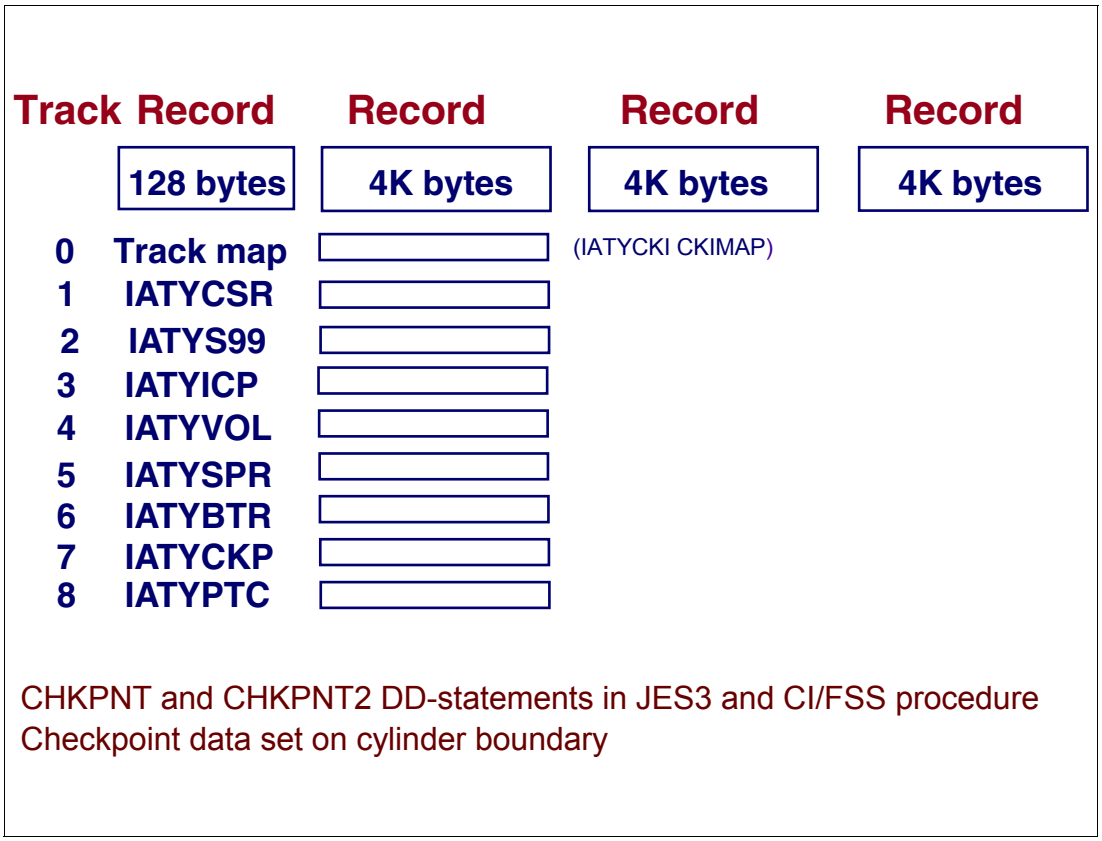
Note: If you allocate only one checkpoint data set and it develops a severe permanent I/O error, you must perform a cold start. If you allocate both checkpoint data sets and one develops a severe permanent I/O error, JES3 can continue. For recovery procedures, see *z/OS JES3 Diagnosis*, GA22-7547.

Checkpoint data set space

To determine how much space your installation's checkpoint data set(s) require, consider the following factors:

- Each checkpoint record type begins on a track boundary. Each track contains a 128-byte track header record.
- The checkpoint data set track map, the complex status record, the initialization checkpoint record, and the JESCKPNT checkpoint record each need one track.
- The dynamic allocation checkpoint record requires 44 bytes plus an additional 92 bytes for each DYNALLOC initialization statement.
- The spool volumes checkpoint record requires 64 bytes plus an additional 80 bytes for each TRACK and FORMAT initialization statement. Add additional bytes as reserve for spool expansion.
- The spool partition checkpoint record requires 64 bytes plus an additional 96 bytes for each SPART initialization statement.
- The partition TAT checkpoint record space requirement is calculated using a complex algorithm involving many different factors. Allow about 512 bytes for each spool data set.
- The BADTRACK checkpoint record requires 44 bytes plus an additional 64 bytes for each BADTRACK initialization statement. Every entry in the BADTRACK checkpoint record requires an additional 64 bytes. Thus, the size of this data area varies with the number of tracks having I/O errors at one time.
- Allocate enough free space to permit growth in the complex without reallocating the checkpoint data sets.

5.27 Checkpoint records



Checkpoint records

The figure shows the specific types of records that are written to the checkpoint data set by JES3.

CKIMAP Track Map Record

This record maps the entire checkpoint data set.

IATYCSR Complex Status Record

The complex status record (CSR) maintains the status of all the JES3 main processors in the JES3 complex. The CSR is created by IATINGL during phase 1 of a cold or warm start on the global and is updated by IATINJB when initialization completes. The record contains an entry for each main.

IATYS99 Dynamic allocation checkpoint record - DYNALLOC

DYNALLOC and JES3LIB are the first statements in the JES3 initialization stream. They provide a description of each data set or device that will be dynamically allocated during JES3 initialization. The DYNALLOC and JES3LIB checkpoint record (S99) is created or restored by module IATINGS during phase 1 of JES3 initialization to maintain a record of the dynamic allocation information.

IATYICP Initialization Checkpoint Record

The initialization checkpoint record (ICP) is a record containing portion of the:

- Transfer vector table (TVT)
- Subsystem vector table (SSVT)

- Initialization data CSECT (INT)
- Output service resident data area (OSD)

It is written to spool at the end of phase 3 by module IATINGN.

IATYVOL Spool data set Checkpoint Record

The spool data set checkpoint record (VOL) records information for all the data sets in the JES3 spool configuration. Each entry in the VOL contains a header and the following information about each data set:

- The name of the partition
- The status of each data set (DRAINED, UNAVAIL or available)
- The beginning and ending addresses of the tracks in the data set
- The size and number of records in each track group

IATYSPR Spool partition Checkpoint Record

The spool partition checkpoint record (SPR) is created during JES3 initialization. The SPR contains the following information on each spool partition:

- The name of the partition
- Initialization information for the partition

IATYBTR BADTRACK Checkpoint Record

The bad track checkpoint record (BTR) maintains a listing of defective spool tracks. The bad track record allows JES3 to add new entries to the record when a permanent I/O error occurs. The record is indexed by the extent number (the X portion of the track group address, X.G).

The information stored in a bad track checkpoint entry includes:

- The type of error encountered
- The address (CCHH) of the unusable track
- The date and time the entry was added to the record

An operator uses the inquiry command, **`*I Q,BT`**, to obtain the status of track from the BTR.

IATYCKP JESCKPT Checkpoint Record

This record is very important for JES3 restarts. It contains pointers to many functions checkpointed data,

IATYPTC Partition track allocation Checkpoint Record

The partition track allocation table checkpoint contains the jes3 spool space allocation bit maps. The IATYPTC data area consists of a header, followed by one entry per defined spool partition. Each partition entry consists of the partition name, the partition size, and the partition track allocation table (PTAT) bit map for the partition.

Checkpoint access by JES3

As information changes in the JES3 complex, the checkpoint data set is updated accordingly. The checkpoint data allows JES3 warm starts or hot starts with little or no loss of system information. At least one of the checkpoint data sets must be available to JES3 during initialization processing. You can add or replace either checkpoint data set over a JES3 restart of any kind with no effect on JES3 processing. Both checkpoint data sets contain identical information; to ensure against loss of checkpointed data, allocate both data sets.

The size of each checkpoint data set should be at least two cylinders on a direct access storage device.

5.28 IATYCKP data mapping macro

```
CKPTDATA DSECT
         IATYCKI HDR=CKP
CKPINID  DS    XL8                 INISH ID (DATE/TIME)
CKPTSTRT DS    0D -                START OF CHECKPOINT DATA
PRTYFLAG DC    16F'0'              IATYJQX-JQXPSTAT FLAGS
CKPRSVD4 DC    XL(FDBSRFL)'00'     RESERVED UNTIL NEXT
*                                    COLD START
CKPSTTFD DC    XL(FDBJBTL)'00'     SYSTEM STT JBT FDB
DLQFDB   DC    XL(FDBSRFL)'00'     DEADLINE QUEUE FDB
CKVUTFDB DC    XL(FDBSRFL)'00'     VOLUME UNAVAILABLE FDB
CKPOSFDB DC    XL(FDBSRFL)'00'     OUTPUT SERV.CKPT.FDB
*
CKPJNEWS DS    0XL(3*FDBSRFL)      JESNEWS FDBS CKPT
CKPJESDS DC    XL(FDBSRFL)'00'     JESNEWS CKPT AREA
CKPTSODS DC    XL(FDBSRFL)'00'     TSONEWS CKPT AREA
CKPRJPDS DC    XL(FDBSRFL)'00'     RJPNEWS CKPT AREA
*
CKPLOCDS DC    XL(FDBSRFL)'00'     LOCNEWS CKPT AREA
CKGMSFDB DC    XL(FDBSRFL)'00'     MAIN SCHEDULING CHECKPOINT FDB
CKDFCFDB DC    XL(FDBSRFL)'00'     DEVPOOL CHKPT FDB
CKDJCFDB DC    XL(FDBSRFL)'00'     DJC CKPT FDB
CKONLFDB DC    XL(FDBSRFL)'00'     ONLINE STATUS CHECKPNT FDB
CKDYNFDB DC    XL(FDBSRFL)'00'     DYNAMIC ALLOC CHECKPT FDB
CKFSSFDB DC    XL(FDBSRFL)'00'     FSS/FSA CHECKPT ROOT FDB
CKSNAFDB DC    XL(FDBSRFL)'00'     SNA/NJE CHECKPT ROOT FDB
CKLCPFDB DC    XL(FDBSRFL)'00'     LOCATE CHECKPOINT ROOT FDB
```

IATYCKP data mapping macro

This is the checkpointed information used to restart JES3 functions from their checkpointed data while JES3 was running. Most data areas are FDBs containing spool addresses to the various functions' checkpoint information.

IATYCKP also contains the range of job numbers assigned in the system and the JCT priority hold flags.

The JES3 checkpoint area is allocated to either one unique data set or two duplicate data sets. You can cause information to be checkpointed in the IATYCKP control block by issuing the JESCKPNT macro.

Checkpoint spool record addresses

Checkpoint data area (IATYCKP), containing the spool record addresses of single record files and multi-record files that checkpoint the status of individual functions within JES3. These individual functions and the files related to them (whose addresses are contained in IATYCKP) are described as follows:

- **Main Device Scheduler (MDS):** The MDS volume unavailable table, which contains the volume serial numbers of volumes unavailable to MDS processing, and the data areas indicating the online/offline status of real devices eligible for setup.
- **Output Service:** The job data accounting block (IATYJDA or JDAB), job data set control block (IATYJDS), and output scheduling element (IATYOSE) data areas, which contain the checkpoint data for the output service driver module.

- **Deadline Scheduling:** The deadline scheduling queue data areas.
- **JESNEWS:** The JESNEWS data set.
- **TSONEWS:** The TSONEWS data set.
- **RJPNEWS:** The RJPNEWS data set.
- **Generalized Main Scheduling:** Data areas containing information about GMS selection modes, execution resources and various GMS parameters.
- **Device fencing:** The device fencing data areas.
- **Dependent Job Control (DJC):** The checkpointed net control block (IATYNCK), which contain entries for each DJC network in the complex.
- **Functional Subsystems:** The FSS/FSA table checkpoint (IATYFCK) data area, which checkpoints functional subsystem and functional subsystem applications information.
- **JES3 Dump Suppression:** The JES3 dump suppression record (IATYDMP) which contains the list of JES3 failsoft codes whose dumps are suppressed automatically when the WANTDUMP=YES parameter is coded on the OPTIONS initialization statement.

5.29 Checkpoint problems at initialization

- COLDSTART required:
  - Both track maps invalid
  - Both checkpoint data sets bad
  - Both CSR records bad
  - Both VOL records bad
  - Both CKP records bad
  - Both SPR records bad

Checkpoint problems at initialization

JES3 creates a copy of each checkpoint record on the duplex checkpoint data set. The primary use of the duplex copy is to enhance error recovery from checkpoint errors in the primary checkpoint data set. The TOD, time of day, is saved in all records and track headers on the checkpoint data sets. The following logic is used when two copies of the checkpoint exist.

- On read requests, the most current copy of a checkpoint record is used. When it occurs that the time stamps do not match, the most current copy is then used to update the other one.
- If an I/O error occurs in accessing a checkpoint record, then the duplex checkpoint record is accessed to satisfy the request.
- If one checkpoint data set is lost, it can be replaced with a new one on the next restart. The remaining checkpoint data set is used to update the new one.
- If both checkpoint data sets are not usable, a COLDSTART is required.

5.30 Creating JES3 job zero

- Output service initialization
  - Create job structure
  - Create RESQUEUE entry
  - Job number is 00000
  - 3 control blocks
    - JDAB - OSE - JDS
    - Kept in JES3 spool - pointed from OSC checkpoint record
  - No JCT record

Creating JES3 job zero

During a JES3 cold start, output service (IATOSDR) constructs a dummy job called job zero (it has a job number of zero in its resqueue entry). There is no JCT entry. There is a resqueue that has FDBs for a JDAB, OSE, and JDS. The JDS and OSE control blocks that are initially empty.

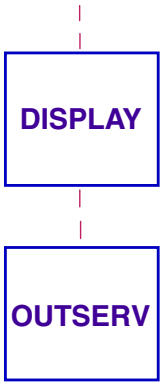
Pointers to this information is kept in the JES3 checkpoint record (IATYCKP).

JES3 restarts

The job structure for job zero is created on any cold start of JES3. On all other starts of the JES3 global, job zero already exists in the JES3 checkpoint and all entries that still exist are always there on the restart.

5.31 JES3 job zero

- No JCT for job Zero - known only in OUTSERV
  - *I U Q=WTR J=0 ....
  - *I J=0 -- IAT8610 JOB (JOB00000) NOT FOUND
- A job that holds JES3 DSPs' SPINOFF OUTPUT



***X DISPLAY,OUT=(PRT),J=DC**

(1). Module IATUTDS
Create output data set - MRF
Create JDS entry for MRF
Issue SPINOFF JDS=

(2). Module IATOSDR
Create OSE for Job zero

JES3 job zero

Job zero (0) is not a “real” job. It is known only to the JES3 output service.

Job zero is used to hold printed output created by JES3 DSPs. Any DSP does not have a print scheduler element, so any output they create has no way to be printed under the DSP’s job structure. After creating a print file using data set track allocation table, the DSP uses a JES3 SPINOFF macro service (IATOSGR) to add the print file (JDS entry) to job zero’s JDS. The job zero resqueue is added to the output service spin-off chain. The OUSERV FCT is posted.

When the OUTSERV FCT gets control, among all the resqueue entries that may be on the chain is also the job zero resqueue. When this resqueue is processed, an OSE entry is created for the spinoff JDS entry. The newly created OSE entry is added into the job zero OSE buffer.

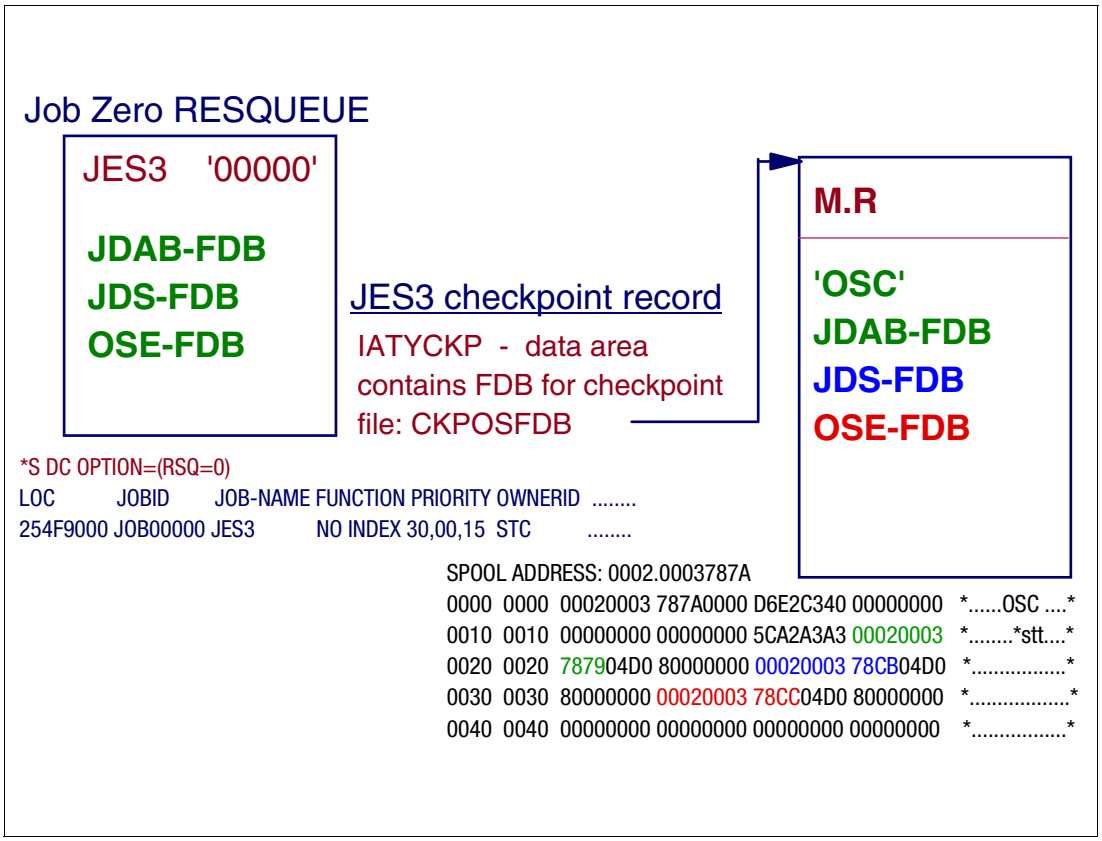
The job zero resqueue is then placed on the output service chain for jobs waiting for writer processing.

JES3 output service inquiry command for job zero:

```
*I U Q=WTR J=0
IAT8131 JOB JES3 (JOB00000), T=PRT, L=127386, PG=0, SR=127386,
IAT8131 JOB JES3 (JOB00000), BY=17879752.
IAT8119 NUMBER OF JOBS FOUND : 1

*I J=0
IAT8610 JOB (JOB00000) NOT FOUND
```

5.32 Job zero structure



Job 0 checkpoint record

In the JES3 checkpoint record, on the track that contains data mapped by IATYCKP, there is an FDB in field CKPOSFDB. This FDB when read contains the three job zero FDBs created during the last JES3 cold start.

When JES3 restarts, during initialization, JES3 takes the output service FDB from the checkpoint, reads the output service checkpoint record (OSC), constructs a resqueue entry, and moves the three job zero FDBs from the OSC record into the resqueue.

5.33 *F CONFIG - dynamically changing JES3 configuration

- *F CONFIG command to allows to add the following without restarting JES3:
  - SNA RJP workstations (RJPWS statements)
  - SNA RJP printers, and punches (DEVICE statements)
  - SNA RJP consoles (CONSOLE statement)
  - Non-channel attached printers (DEVICE statements)
  - Writer FSS's (FSSDEF statements)

*F CONFIG command

You can make changes to the JES3 configuration dynamically after the system is active by using the **`*F CONFIG`** operator command. The **`*F CONFIG`** command allows you to add the following definitions to JES3 without having to restart the JES3 global and local address spaces:

- SNA RJP work stations and their associated consoles and devices
- Non-channel attached printers

You use **`*F CONFIG`** to specify the name of a member in the data set allocated to the JES3IN DD statement in the JES3 cataloged start procedure. This member contains the initialization statements associate with definitions that you want added to the JES3 configuration. The following initialization statements can be coded in the member specified on the **`*F CONFIG`** command:

- RJPWS to define SNA/RJP work station characteristics
- CONSOLE to define SNA/RJP console
- DEVICE to define SNA/RJP devices and non-channel attached FSS managed
- Printers
- FSSDEF to define writer FSSs
- INTDEBUG to establish the Initialization Debugging Facility
- INCLUDE to include another initialization stream member

5.34 *F CONFIG command

- Changes are checkpointed
  - Changes are there at next **hot** start
  - Changes are undone if hot start with refresh, warm, or cold start
  - DO NOT forget to add the changes to initialization stream

*F CONFIG command

These changes are checkpointed so that they will remain in effect if you perform a hot start. If you perform a hot start with refresh, the changes are lost. Therefore, make sure you update your initialization stream before performing a hot start with refresh, warm, or cold start.

After the **`*F CONFIG`** command is processed, the appropriate tables are built to represent the information that was added. The changes you made are preserved across a JES3 hot start; you do not have to issue another **`*F CONFIG`** command. However, the changes are not preserved across a cold start, warm start, or hot start with refresh so remember to add the initialization statements to your initialization stream.

5.35 *F CONFIG command

- *F CONFIG,ADD = member
  , LOG={YES|NO|ERR},
  , P=parms
- Parameters can be in any order
- Only ADD parameter is required - specifies member name containing init statements to be processed
  - Member must exist in data set allocated to the JES3IN DD statement in the JES3 start procedure and name can be any format
- SAF entity name is JES3.MODIFY.CONFIG
- Operator must have at least UPDATE access to use

*F CONFIG command

The `*F CONFIG` command dynamically invokes JES3 initialization services to add additional configuration information without the need to perform any type of JES3 restarts. Following is the command syntax:

ADD=mem_name Specifies the 8-byte member name to be read from the data set allocated to the "JES3IN" DDNAME in the JES3 procedure. This member contains all initialization statements that you want to add.

LOG=YES|NO|ERR Optionally specifies whether you want to record each statement processed and any error message generated in a spin-off data set named "MODIFY CONFIG LOG." The LOG=ERR option allows a log data set to be generated only if an error occurs. The default is YES.

P=xxxxxxxx Optionally specifies a parameter string that is passed to IATUX15 as the statements are processed. This is similar to the P= parameter that can be specified in response to message IAT3012 (specify inish deck origin...).

You can protect use of the command with SAF and RACF® profiles.

5.36 P= parameter

- Specifies a one to eight character parameter string that will be passed to IATUX15 as each statement is read
- The default is no parameters are passed to IATUX15
- Similar to P = parameter in response to IAT3012

P= parameter

User exit 15, IATYX15, is entered from the operator interface module or the initialization subroutines during a warm or cold start, and permits you to scan an initialization statement immediately after the statement is read and before it is scanned by the system routines.

On completion, you have the opportunity of having the statement processed (as entered or as amended) or having the statement ignored.

Note: You can test this installation exit routine using the initialization stream checker (module IATUTIS). If you do, the exit routine must not issue any privileged instructions or a program check will result. For information on how to use that facility, see *z/OS JES3 Initialization and Tuning Reference*, SA22-7550.

When specifying the P=xxxxxxxxx option, the xxxxxxxx is a parameter string which will be passed to IATUX15 as the statements are processed. This is similar to the P= parameter that can be specified in response to message IAT3012.

```
IAT3012 SELECT JES3 INISH ORIGIN (N OR M=), AND OPTIONAL EXIT PARM (,P=) OR CANCEL
```

Register 0 points to a field containing a 2-byte length indicator, followed by 1 to 8 characters constituting the string portion of the 'P=string' value entered by the system operator in response to message IAT3012.

5.37 LOG= parameter

- Statements and error messages a log data set
  - YES - Generate the log data set - the default
  - No - No log data set - error messages on console
  - ERR - Generate log data set only if errors
- Spinoff log data set at end of *F CONFIG

LOG= parameter

Specifies whether you want a log data set generated. The log data set contains the initialization statements and any error messages that are generated. The log data set is then spun off for printing at the end of **`*F CONFIG`** processing.

YES Create the log data set.

ERR Create the log data set only if an error occurs.

NO Do not create the log data set. All error messages will be displayed on the issuing console.

This parameter is optional and any error message is generated in a spin-off data set named **`"MODIFY,CONFIG LOG`**." The default is YES.

A sample LOG output appears from JES3 job zero as follows:

```
IAT8351 *MODIFY,CONFIG - DATE = 2007.023, TIME = 14.45.27
IAT8351 DSN = SYS1.PARMLIB , MEMBER = JES3INIP, PARMS =
DEVICE,DTYPE=PRTAFP1,
     JNAME=IPDPOK,DGROUP=IPDS,DGRPONLY=NO,
     JUNIT=(,SC64,UR,,,SC63,UR,,,SC65,UR),
IAT3083 JUNIT MAIN SC63 IS NOT DEFINED AND CANNOT BE ADDED DYNAMICALLY
FSSNAME=IPDSWAY,DYNAMIC=YES,BURST=NO,FORMS=(NO,STD),
     WC=(K),WS=(CL)
IAT3088 DEVICE IPDPOK IGNORED, PREVIOUSLY DEFINED

IAT8348 WARNING LEVEL MESSAGE(S) ISSUED DURING INITIALIZATION STATEMENT PROCESS
```

5.38 Add a SNARJP workstation

*F CONFIG,ADD=ADDRJP

```
EDIT       SYS1.PARMLIB(ADDRJP) . 01.00                        Columns 000
Command ===%                                                          Scroll =
****** ********************************* Top of Data ************************
000001 *
000002 RJPWS,
000003   N=ABC12,C=S,PR=3,PU=3
000004 *
000005 CONSOLE,
000006   TYPE=RJP,JNAME=ABC12,LEVEL=15,DEST=ALL,LL=80
000007 *
000008 DEVICE,
000009   DTYPE=RMTPRINT,JNAME=ABC12PR2,BURST=NO,HEADER=NO
000010 *
****** ********************************* Bottom of Data *********************
```

*F CONFIG example

In the example, the member of parmlib to be added is ADDRJP in the PDS SYS1.PARMLIB. The member contains 3 initialization statements that define a SNARJP workstation. The statement is added dynamically to a running JES3 system. Although you will typically not use the INCLUDE statement in the member you specify on the ***F CONFIG** command, segmented initialization streams are useful when used in conjunction with ***F CONFIG** processing. For example, suppose all the RJPWS statements are in a member "RJPWS" of a PDS. And suppose your IATUX15 supports conditional logic. For example IATUX15 is set up to interrogate the parameter string that is passes and skip over certain initialization statements depending on the parameter value. To add new RJPWS statements to the initialization stream, add the new RJPWS definitions to the an existing "RJPWS" member if you are using INCLUDE statements in the initialization stream

INCLUDE statement

The INCLUDE statement is also supported during ***F CONFIG** command processing. JES3 allows the initialization stream to be segmented. Different sections of the initialization stream can be in different members of a partitioned data set and use the INCLUDE initialization statement to include the members at the appropriate places in the primary initialization stream. Each INCLUDE initialization statement refers to a member of the same partitioned data set. The member contains the initialization statements to be added to the initialization stream. The functional areas JES3 initializes depends upon the type of start being performed and the contents of the initialization stream. Each initialization statement is processed by a module that belongs to a JES3 functional area.

6

JES3 input service

Input service is the JES3 function that accepts and queues all jobs, entering the JES3 system, except those invoked via the `*X` command. When jobs are sent to a JES3 global, JES3 input service does the following:

- Processes JCL control statements
- Creates scheduler elements that represent work to be executed by JES3
- Adds jobs to the JES3 job queue.

The global processor reads the job into the system from one of the following input sources:

- A local card reader
- A local tape reader
- A disk reader
- A remote work station
- Another node in a job entry network
- The internal reader
- Started tasks

The reader phase of input service reads jobs (JCL and input stream data) and stores them on a spool data set. The only jobs not read by the reader phase are jobs from an internal reader and demand select jobs. These jobs are read directly by the control statement processing phase. Jobs can come from a card reader, a tape unit, a disk reader or from a remote workstation. The reader phase treats jobs from a remote workstation as though the job came from a card reader.

6.1 Jobs entering a JES3 system

- Jobs can be submitted into a JES3 system from:
  - Locally attached tape, disk, or card readers
    - TR, DR, and CR DSPs
  - Remote job processing (RJP) workstations
  - Time-sharing option (TSO/E) terminals
    - Submits - (INTRDR)
  - Other systems in a job entry network (NJE)
  - Internal reader submits (INTRDR)
    - DD SYSOUT=(A,INTRDR)

JES3 jobs

JES3 initially reads all jobs into the global and assigns, to each job, a unique JES3 job number from the available job number pool. Jobs can be submitted from a locally attached tape, disk, or card reader. In addition jobs can be submitted from remote job processing (RJP) workstations, time-sharing option (TSO/E) terminals, other systems in a job entry network, or by the internal reader.

Reader processing

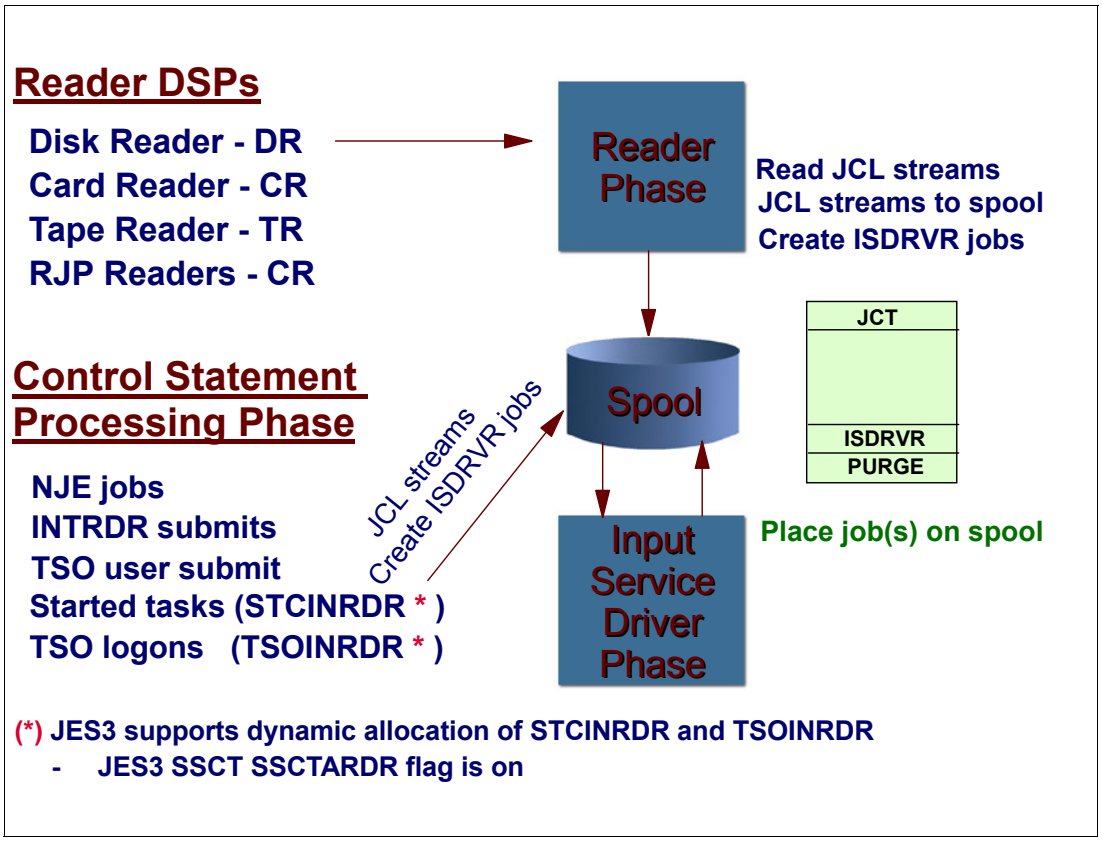
Reader processing takes place in the JES3 global address space. The reader phase reads jobs from any of the sources mentioned above and places the jobs on JES3 SPOOL in batches for later processing. For each reader batch, an input service job is created. Jobs submitted from BSC RJP remote stations are processed as if they come from a local card reader. Jobs from SNA RJP use a special logical record (LR) interface.

INTRDR processing

Input may also come from the JES3 internal reader that processes input streams contained in SYSOUT data sets obtained from MVS. The internal reader allows a JES3 output data set to be passed to JES3 input service and be processed as an input stream. In this way, jobs can also be submitted to JES3 from MVS.

START and **MOUNT** commands and TSO/E LOGONs cause jobs to be started from predefined procedures. Input service processes the JCL created for these jobs in the same manner as any other standard job. Jobs initially placed on direct-access storage devices (DASD) and subsequently analyzed by JES3 input service are placed on the JES3 spool.

6.2 JES3 input service - two phases



JES3 input service - two phases

Input service, the first phase of JES3 job management, reads jobs and places each job into a queue for subsequent processing by other phases. The modules that make up input service accept a typical MVS jobs for processing. Input routines create scheduler elements that control the flow of jobs through the JES3 processing phases, process control statements. Input service consists of two phases:

- **Reader phase** - Reader processing takes place in the JES3 global address space. The reader phase reads jobs from any of the sources mentioned above and places the jobs on spool in batches for later processing. For each reader batch a control statement processing phase (input service driver ISDRVR) job is created.
- **Control statement processing phase** - The control statement processing phase of input service begins as each batch job produced by the reader phase is read from spool. Under the ISDRVR DSP, the JES3 job control blocks for the job are built and the defaults in the control blocks are modified with the information retrieved from the job's control statements. The control blocks are written to the spool.

Reader phase

Input service queues all jobs entering the JES3 system on the global processor from:

- A card reader (CR DSP)
- A tape reader (TR DSP)
- A disk reader (DR DSP)
- A remote work station (RJP/SNARJP DSPs)

- Another node in a job entry network (NJE DSPs)
- The internal reader (INTRDR DSP)
- A TSO SUBMIT command (INTRDR DSP)
- MVS START, MOUNT, and LOGON command processing (INTRDR DSP).

Control statement processing phase

After the reader phase completes execution, the control statement processing phase (ISDRVD) receives control. This phase analyzes JES3 control statements, checks RACF authorization and writes each job to the JES3 job queue. This phase also reads jobs from the internal reader.

If the job contains no //*PROCESS control statements, the control statement processing phase defines the job as requiring the standard sequence of scheduler elements (SEs). The standard sequence is:

- Converter/interpreter (CI)
- Main device scheduler and generalized main scheduling (MAIN)
- Output service (OUTSERV)
- Purge (PURGE)

ISDRVR DSP

ISDRVR (IATISDV) is entered in one of the following ways:

- When input is through a call to a reader, (CR, TR or DR), the reader function creates an ISDRVR job and adds it to the queue.
- When the input is a demand select job, the ISDRVR job is created by module IATMSGC. When the job is scheduled the JCL for the demand select job is passed to ISDRVR in a staging area.
- When the input is through a call to the internal reader, input service runs under the internal reader's FCT. Module IATISIR, the internal reader driver, calls IATISDV. The JDS entry for the new internal reader data set is contained in the FCT's JDS.

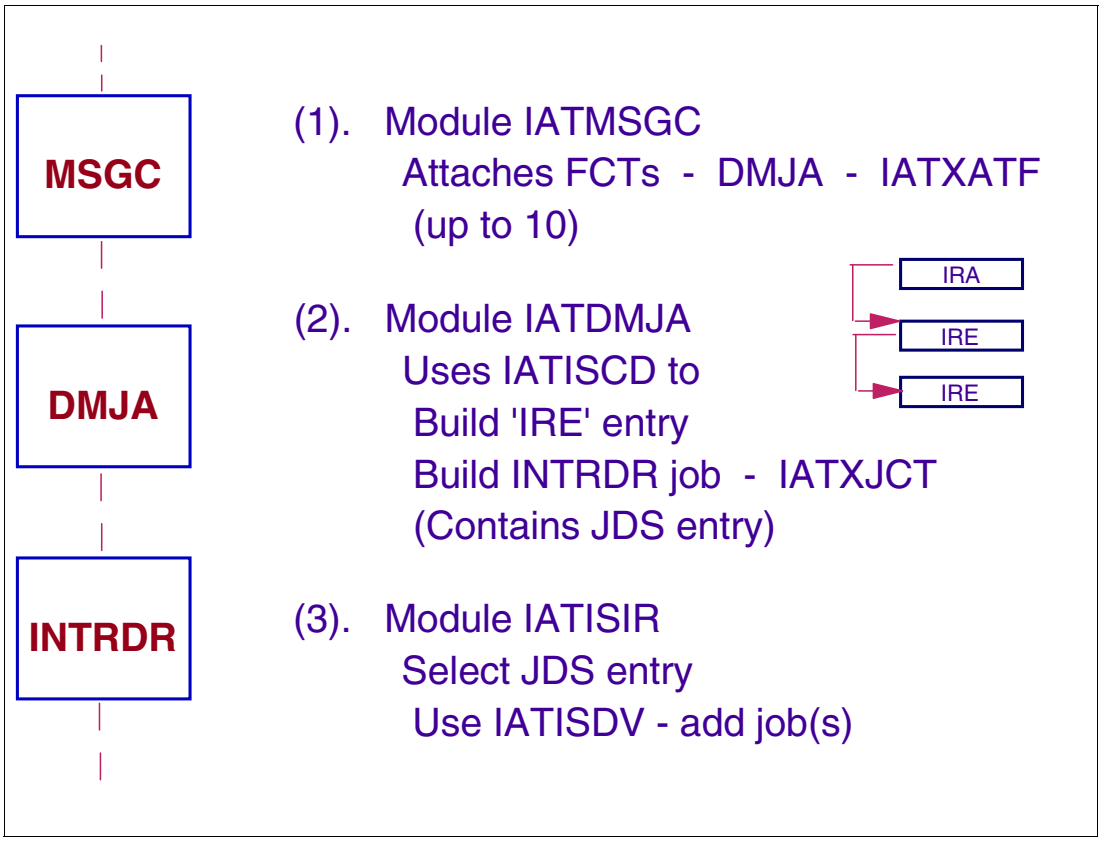
Once given control, the input service driver (ISDRVR) performs the following major functions:

- Read and pass one-at-a-time the JDS entries for the multi-record files that are the input to the this job. (There is an input job associated with each JDS entry).
- For each input job, get the buffers for and initialize the JDS and JDAB

Internal reader (INTRDR)

Internal reader routines allow TSO jobs or application programs to submit job streams to JES3 using output data sets. When a job stream enters the system, data management assigns the data sets directly to an internal reader. If an internal reader is not available, the system dynamically creates one. When JES3 schedules the internal reader, input service can proceed to process the data set as an input stream. To pass the job to the internal reader, the program must write the job to a data set for which SYSOUT=(class,INTRDR) has been specified. The internal reader DSP (IATISIR) reads the job, then gives it to JES3 where the job is processed like other reader-submitted jobs. If the submitter of an internal reader job is either a TSO user or has an associated TSO userid, the TSO userid will be propagated to the submitted job. When an internal reader job is available, the output data set is assigned directly to an internal reader by adding an entry to the internal reader job (INTRDR DSP) data sets block (JDS). If no internal reader is available, input service dynamically creates an internal reader.

6.3 INTRDR Processing



INTRDR processing

The internal reader facility is useful for several kinds of MVS applications:

- It can be used to generate another job or a series of jobs from an already-executing job. A job that produces a series of jobs can put its output to an internal reader for immediate execution.
- The operator can start utility programs to read jobs from disk or tape files and submit them to the system. The IBM-supplied procedure 'RDR' is an example of a program that does this.

An internal reader data set can be allocated in any address space either with JCL or dynamically as a SYSOUT data set with a writer name INTRDR. INTRDR is an IBM-reserved name identifying the internal reader as the program to process the SYSOUT data set after it is created, closed, and un allocated. The SYSOUT class becomes the message class for the submitted job unless overridden by MSGCLASS parameter on the JOB statement.

In additions to JCL and JECL statements, a /*DEL and /*EOF records can be sent as the last record of a job. The /*DEL record cancels the job. The /*EOF control statement delimits the current job and makes it eligible for immediate processing. The internal reader data set remains open.

When an internal reader data set is closed, the MVS data management CLOSE processing invokes JES3 SSI close routine (IATSICC). IATSICC identifies the JDS entry to be closed and passes (SSISERV) the close request to the global JES3's general collection routines

(IATMSGC). (The JDS entry was built at MVS data management OPEN time (in IATSIOR) and then passed to global to be added to the requesting job's JDS).

As soon as a the internal reader close request becomes available to the global JES3, IATMSGC (running under MAIN FCT) creates and attaches a USAM JDS access interface routine (DMJA DSP). Since the request is for an internal reader data set, IATDMJA assigns a job number to the data set, issues the IATXRABD macro to release the unused track groups, and calls IATISCD, the internal reader job scheduler, to process the data set.

Note: The internal reader anchor control block (IRA) points to a chain of internal reader elements (IREs), one for each INTRDR DSP that exists. IATISCD assumes that if there is no IRA, there are no IREs.

Module internal reader job scheduler - IATISCD

IATISCD searches for an idle internal reader job. If one is found:

If the data set originated from a TSO user, IATISCD copies the userid, MP sequence number, and the MP main name to the new JDS entry.
IATISCD inserts the new JDS entry into the JDS for the internal reader job.
IATISCD sets the work-to-do post and posts the INTRDR DSP.
If a new internal reader job must be created:

- IATISCD creates and initializes the JMR, JDAB, and the JCT JES3 control blocks. The new JDAB consists of an INTRDR and a PURGE scheduler element.
- If the data set originated from a TSO user, IATISCD copies the userid, MP sequence number, and the MP main name to the new JDS entry.
- IATISCD calls the accounting exit, IATUX27 to process the JMR, JDAB and the JCT.
- IATISCD creates and initializes the JDS entry for the new internal reader data set.
- IATISCD writes out all the new job control blocks to spool.
- IATISCD adds the new job to the JES3 job queue.

The INTRDR DSP (IATISIR) processes the internal reader data set passed by IATISCD by calling the input service driver (IATISDV).

STCINRDR and TSOINRDR data set processing

When you start a started task, the MVS writes the demand select job's JCL into the STCINDRD data set. TSO logon JCL is written into the TSOINRDR data set. The DD-staements for these data sets are defined within the MSTJCLxx member:

- //STCINRDR DD - This DD statement defines the internal reader where started tasks are to be sent.
- //TSOINRDR DD - This DD statement defines the internal reader where TSO logon started tasks are to be sent.

The ENDREQ routine in IATDMDM module initiates the submittal of a job or job stream from an internal reader. During ENDREQ processing for started tasks, mounts and tso logons, a security token is extracted for the job in whose address space IATDMDM is running. If the ENDREQ request is for a STCINRDR or TSOINRDR, the data buffer is sent to global JES3 via SSISERV for special input service processing.

Once the global JES3 General Collection Routines (IATMSGC) get control to process the demand select JCL buffer, it invokes the ISDRVR DSP.

6.4 Controlling the INTRDR DSP

- Commands to stop/start INTRDR jobs
  - *F X,D=INTRDR,HOLD
  - *F X,D=INTRDR,RELEASE
- Display IREs using DC DSP
  - *S DC,OPTION=INS
  - IREs displayed in JES3 ABEND dump
    - IRE = Internal Reader Element Block
- Defining number of INTRDR DSPs
  - OPTIONS,.....INTRDR=20|nn
    - nn - 1 to 999
- Operator command to display
  - *I A,D=INTRDR

Controlling the INTRDR DSP

Use the **`*F X,D=INTRDR`** command to change the number of internal reader DSPs that can be started. This does not effect internal readers active when the command is processed. They will complete the processing of the current input stream.

You can start or stop INTRDR processing by using the HOLD/RELEASE options on the **`*F X,D=INTRDR`** command.

The IREs queued can be displayed using the dump core (DC) DSP. They are also displayed in the JES3 formatted dump.

The **`*I A`** command allows you to display as follows:

D=INTRDR	Displays the jobs that are being processed by internal reader DSPs.
D=INTRDR,R	Resets the high water mark value to the current number of internal reader jobs either active or waiting to be processed.

The **`*I X,D=INTRDR`** command displays the INTRDR counts as shown in the figure.

The INTRDR= parameter on the OPTIONS initialization statement specifies the maximum number of internal readers that can be active concurrently. Specifying too high a value can cause a shortage of JSAM buffers. You can specify any value between 1 and 999 inclusive.

(Internal Reader Element Block (IRE) contains information used by input service in controlling the scheduling of individual internal reader jobs.)

6.5 DSP dictionary and INTRDR inquiries

```
INTRDR   IATYDSD PRTY=4,REENT=YES,NOREQ=1,XABLE=NO,PABLE=NO,
         CSECT=IATISDT,DRVR=IATISIR,HABLE=YES,JOBVAL=IATISJV,MAXCT=20

*I X,D=INTRDR
  IAT8475 (IQDX) - INTRDR   MXCT=000020 USE=000002 MOD=YES
  IAT8472 (IQDX) - INTRDR   IS NOT IN HOLD
*F X,D=INTRDR,HOLD
  IAT8497 (MODX) - INTRDR   HELD
*F X,D=INTRDR,RELEASE
  IAT8497 (MODX) - INTRDR   RELEASED
*F X D=INTRDR,MC=30
  IAT8484 (MODX) - INTRDR   OLDMXCT=000020 NEWMXCT=30
*I A,D=INTRDR
  IAT8522 JOB INTRDR   (JOB11479) ACTIVE ON INTRDR        7226.98 MIN
  IAT8522 JOB INTRDR   (JOB28541) ACTIVE ON INTRDR        7226.98 MIN
  IAT8523 INTRDR COUNTS - MAX=(020,020), ACT=0002, FCT=002, HWATER=0002
  IAT8593 INQUIRY ON ACTIVE JOBS COMPLETE,      2 JOBS DISPLAYED
*I Q,D=INTRDR
  IAT8684 NO JOBS WAITING FOR INTRDR
 IAT8595 INQUIRY ON JOB QUEUE STATUS COMPLETE,      0 JOBS DISPLAYED
```

INTRDR dictionary entry

The IATYDSD macro generates an entry for a dynamic support program (DSP) in the DSP dictionary (module IATGRPT or, in a C/I FSS address space, module IATGRPTF). An entry in the table is required for each DSP in order for it to be recognized as part of JES3.

```
HABLE= keyword
   YES - Allows the DSP's work to be held and released by the *F,X,D=dsp,HOLD and
   *F,X,D=dsp,RELEASE commands. For DSPs that have an FCT, the command takes affect
   immediately upon being issued. For DSPs having no FCT, the command prevents the job
   segment scheduler (JSS) from scheduling the DSP.
   NO - Disallows the DSP's work to be held and released by the *F,X,D=dsp,HOLD and
   *F,X,D=dsp,RELEASE commands.
```

Internal reader routines allow TSO jobs or application programs to submit job streams to JES3 using output data sets. When a job stream enters the system, data management assigns the data sets directly to an internal reader. If an internal reader is not available, the system dynamically creates one. When JES3 schedules the internal reader, input service can proceed to process the data set as an input stream.

Use the ***F X,D=INTRDR** command to change the INTRDR parameter. Use the ***F X,D=dspname** command to change the number of DSPs that can be started. Processing of this command does not affect an active DSP when the command is processed.

To display a list of all the internal readers in the system at any one time, issue the ***I A,D=INTRDR** command. . The operator can also stop the internal reader by issuing a ***C INTRDR** or a ***C J=jobno** command.

6.6 Disk reader

- Defined in JES3 PROC or Initialization Stream
  - //JES3DRDS DD DSN=SYS1.JES3DR ,DISP=SHR
  - DYNALLOC,DDN=JES3DRDS,DSN=SYS1.JES3DR
- Partitioned data set - max blocksize=3200
  - Can be concatenated
- Disk reader usage
  - Production job streams
  - Operator commands
- *X DR,M=mem,B=10,NH,J=jobnm,JOBS=nnn,PARMID=xx,K

Disk reader

If the disk reader facility (DR) is desired, specify in the JES3 start procedure:

```
//JES3DRDS   DD    DISP=SHR,DSN=dsn
```

or use the DYNALLOC statement in the initialization to specify the disk reader data set:

```
DYNALLOC,DDN=JES3DRDS,DSN=dsn
```

The disk reader data set is a the partitioned data sets containing input for the JES3 disk reader facility. The maximum block size for this data set is 3200. Concatenated data sets may be used.

You can enter all JES3 commands (except ***DUMP** and ***RETURN**) from a card reader (CR) or disk reader (DR). You can use the disk reader to enter repetitive commands based on system requirements (such as shift change). Any output messages generated from a card reader or disk reader are displayed at the console from which you called the reader.

One of the main uses of the disk reader is to read in a production JCL stream that you place into the disk reader data set.

Disk reader parameters

Use the ***X DR** command to initiate reading of jobs from the data set(s) allocated to the JES3DRDS DD-statement.

```
*X  ,DR,IN=(ANYGROUP|,group)B=(10|nnn),H|NH,JOBS=ALL|nnn,j=jobname
   ,K|KN,M=mem,P=15|nn,PARMID=01|xx
```

If the K parameter is specified on the **`*X`**, **`*S`**, or **`*C`** command, the disk reader remains active when an EOF condition is reached, and reading resumes each time an **`*S`** command is entered. To cancel a “hot DR” issue command **`*C DR,KN`**.

IN= (ANYLOCAL) or (,group) specifies the device group to which output should be routed. If omitted, the associated output is routed to printers or punches in the device group ANYLOCAL; this is not desirable if the **`*X DR`** command was entered at an RJP console.

B=10 or nnn Specifies the size of the job batches created by this reader, that is, the number of jobs that are read from the disk and placed on the spool device. The default is 10. Valid values are 1 through 255.

H or HN Specifies whether the JES3 control-card processor is to be put in the hold state (H) or allowed to process the jobs after the batch has been created (HN). The default is HN.

J=jobname Specifies the name of the job with which input is to begin. When this parameter is specified, the JOBS= default is changed from ALL to 1.

JOBS=nnn or ALL Specifies the number of jobs to be read from the input device and placed onto spool devices. ALL permits reading until a physical EOF is reached. When you specify J=jobname (see preceding), the ALL default for JOBS= is changed to 1. The maximum value allowed is 254.

K or KN Specifies whether the designated reader is to be kept active (K) when an EOF condition occurs or is to be allowed to purge (KN). Reading is resumed when you enter an *START command. The default is KN.

M =mem Specifies the member of the disk reader data set that is to be read.

P=15 or nn Specifies the scheduling priority of the control-card processor job (without affecting the priorities of the individual jobs being read in). Valid values are 00 through 15. The default is 15.

PARMID=01 or xx Specifies the set of C/I options to be used for all jobs read in through this reader. The value must be the same as the PARMID associated with a CIPARM statement. The default is 01.

6.7 Disk reader (DR) job submission

```
*X DR M=JOBBATCH B=4 H
IAT6306 JOB (JOB25013) IS DR      , CALLED BY VAINI
*I X D=ISDRVR
IAT7100 (DR      ) *I X D=ISDRVR
IAT8475 (IQDX) - ISDRVR   MXCT=000002 USE=000000 MOD=YES
IAT6202 JOB DRJOB1   FROM 8312 INTRP BY JOB (JOB25014)
IAT6202 JOB DRJOB2   FROM 8312 INTRP BY JOB (JOB25014)
IAT6202 JOB DRJOB3   FROM 8312 INTRP BY JOB (JOB25014)
IAT6202 JOB DRJOB4   FROM 8312 INTRP BY JOB (JOB25014)
IAT6201 JOB DRDR -04 (JOB25014),04 JOBS,00008 CDS,P=15,HOLD
IAT6202 JOB DRJOB5   FROM 8312 INTRP BY JOB (JOB25015)
IAT6202 JOB DRJOB6   FROM 8312 INTRP BY JOB (JOB25015)
IAT6202 JOB DRJOB7   FROM 8312 INTRP BY JOB (JOB25015)
IAT6202 JOB DRJOB8   FROM 8312 INTRP BY JOB (JOB25015)
IAT6201 JOB DRDR -04 (JOB25015),04 JOBS,00008 CDS,P=15,HOLD
IAT6202 JOB DRJOB9   FROM 8312 INTRP BY JOB (JOB25016)
IAT6202 JOB DRJOB10  FROM 8312 INTRP BY JOB (JOB25016)
IAT6201 JOB DRDR -04 (JOB25016),02 JOBS,00008 CDS,P=15,HOLD
IAT7450 JOB DR       (JOB25013) PURGED
*F J=25014 R
IAT8080 JOB DRDR -04 (JOB25014) RELEASED
IAT6100 (JOB25014) JOB DRJOB1   (JOB25017), PRTY=01, ID=*UNKNOWN
IAT6100 (JOB25014) JOB DRJOB2   (JOB25018), PRTY=01, ID=*UNKNOWN
IAT6100 (JOB25014) JOB DRJOB3   (JOB25019), PRTY=01, ID=*UNKNOWN
IAT6100 (JOB25014) JOB DRJOB4   (JOB25020), PRTY=01, ID=*UNKNOWN
IAT7450 JOB DRDR -04 (JOB25014) PURGED
```

Disk reader job submission

The operator calls the disk reader (DR) and specifies to read the job streams from member JOBBATCH.

The default batch size specified on the `*X` command is 4. There are 10 jobs in the specified member. Therefore, 3 DRDR jobs are created, the two first with 4 jobs and the third with 2 the jobs. The `H` on the `*X DR M=JOBBATCH B=4 H` command specifies the JES3 control-card processor to be put in the hold state.

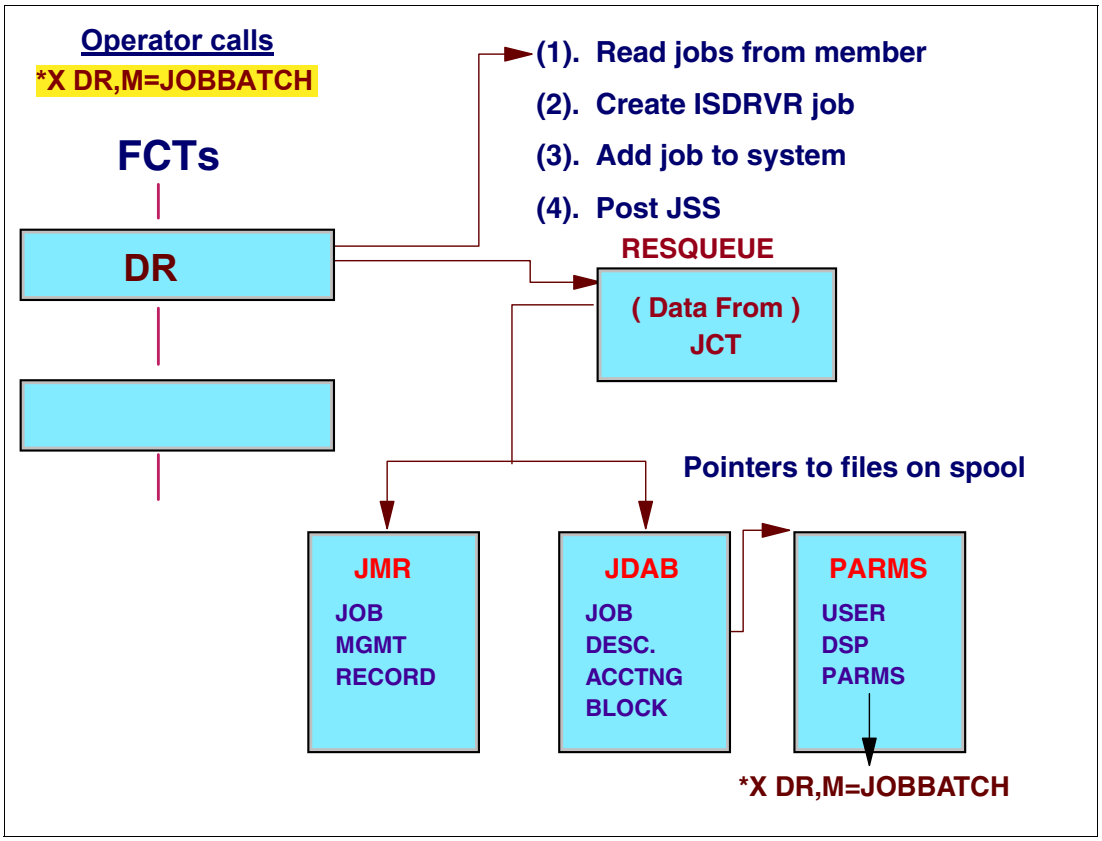
Release a called job

The `*F J=25014 R` command releases the first DRDR job. When the DRDR job is scheduled, it begins to add the individual jobs from the input batch JDS into the system.

Using JES3 command statements

The first statement in the JOBBATCH disk reader data set member is a JES3 command statement--`//**I X D=ISDRVR` . JES3 command statements are placed before the first JOB statement in the input stream. JES3 treats any JES3 command statements that follow the JOB statement as comment statements.

6.8 Disk reader processing DSP



Disk reader phase processing

The operator calls the DR DSP using the command **`*X DR,M=jobbatch`**. The member contains a stream of jobs is called jobbatch.

The reader phase reads jobs and places the jobs on JES3 spool in batches for later processing. For each reader batch, an input service job (ISDRVR) is created. Jobs submitted from RJP remote stations are processed as if they come from a local card reader.

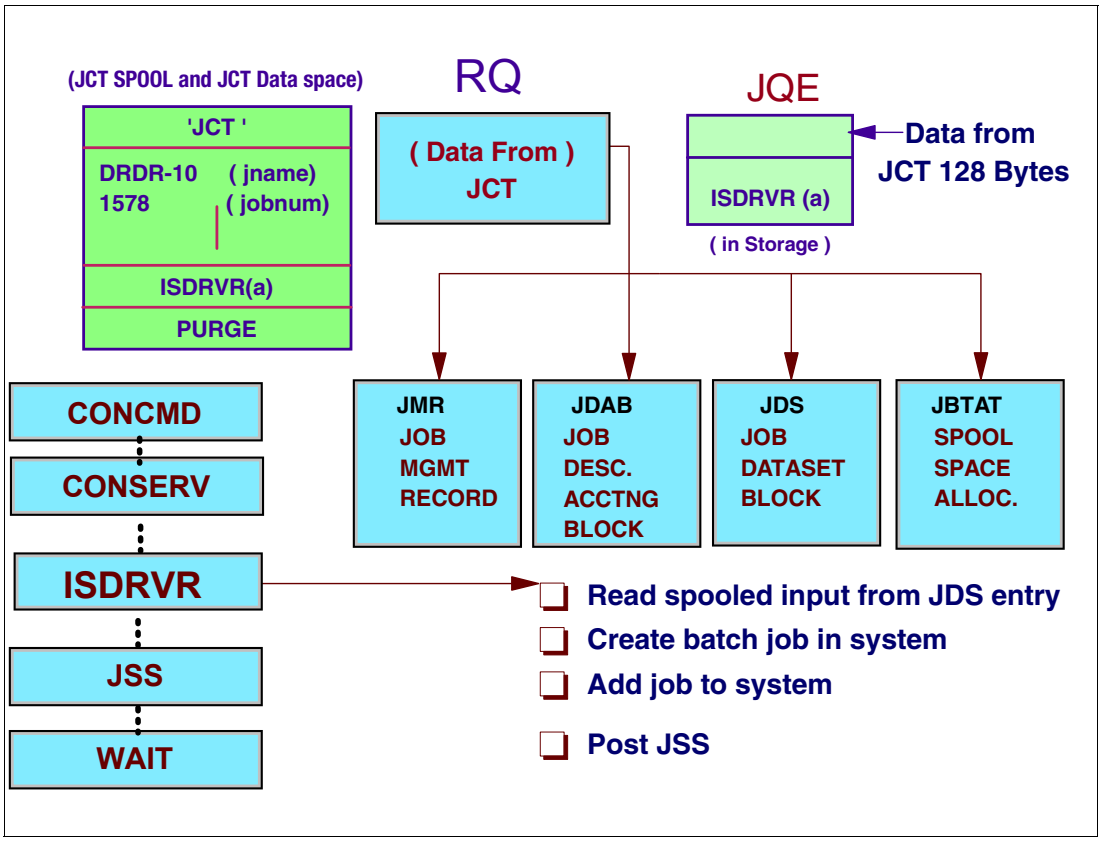
Disk reader data set

The DR module (IATISDR) opens the disk reader data set, then loads and calls the reader logic module (IATISRL). The IATISRL module reads cards until EOF is reached. It creates job batches and creates a JES3 input service job for each batch. Whenever IATISRL module needs to read cards, it calls the IATISDR module to perform I/O to the disk reader data set.

The disk reader uses the basic partitioned access method (BPAM) to read one block at a time from a partitioned data set. The data control block (DCB) for the partitioned data set (PDS) is located in module IATISCB.

When the DR DSP is terminating, the disk reader data set is closed, then an exit back to JSS is made. The OPEN/CLOSE is done with a JES3 general subtask through the IATXCSF macro. The BPAM FIND/READ is done also with a JES3 general subtask. This prevents the implied wait of DR data set I/O operations from waiting the JES3 nucleus task.

6.9 ISDRVR job processing



Reader phase job processing

The control statement processing function creates jobs and places them in the JES3 job queue. This phase is started in one of four ways:

- When input enters the system through an external reader device (a *CALL to CR, DR, or TR), the reader phase creates an ISDRVR job and adds it to the queue. Then the job segment scheduler schedules the job.
- When a demand select job (MOUNT, LOGON, or started task) is to be processed, main service (IATMSGC) creates and attaches a special ISDRVR FCT. This ISDRVR DSP takes a special "fast path" through input services and eventually, via JSERV, issues a response back to the initiator.
- When input enters the system through an MVS CLOSE or ENDREQ request on a data set allocated to the internal reader, data management services (IATDMJA) creates an INTRDR job and adds it to the JES3 job queue.
- When networking JOB/SYSOUT receive module (IATNTJS) receives a job stream from remote nodes, it creates an ISDRVR job to process the networking stream locally.

ISDRVR DSP processing

When the control statement processing phase of input service begins under the ISDRVR DSP (or the INTRDR DSP for internal readers), input service driver module (IATISDV) performs the following major functions:

- For each input job, get the buffers for and initialize the JDS and JDAB

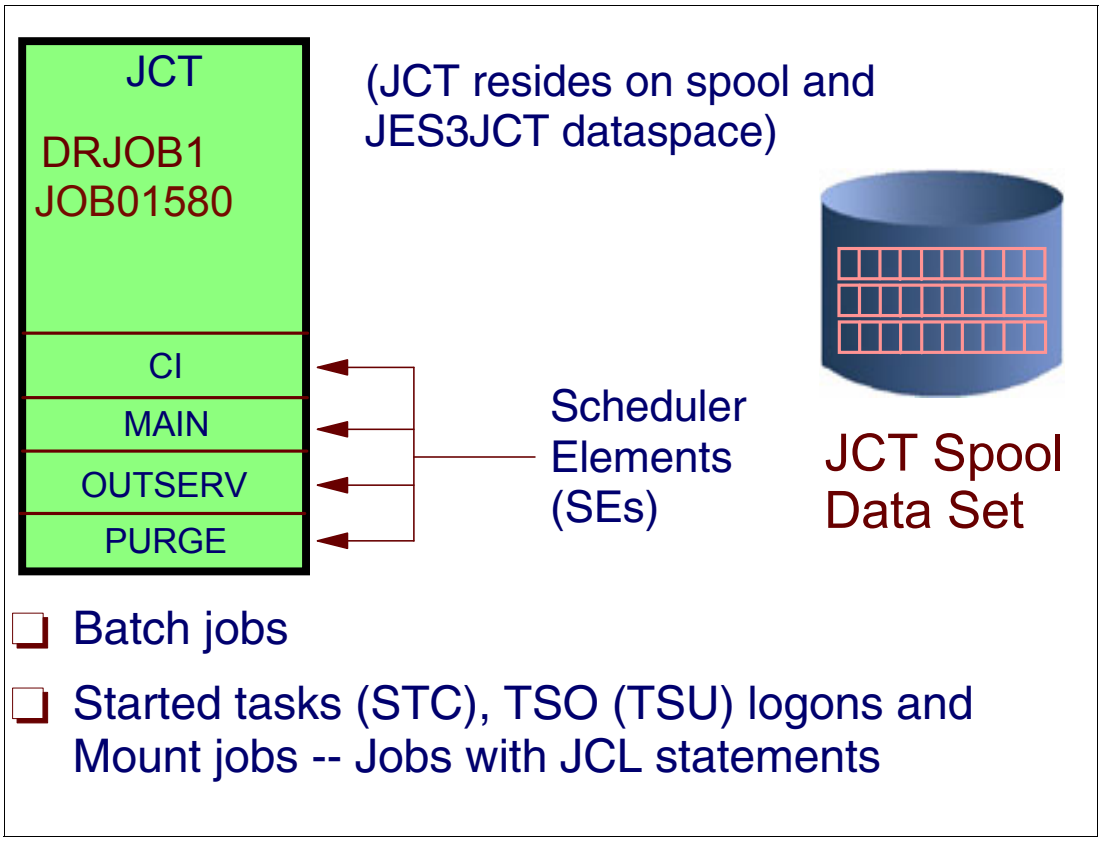
- Read the spooled job input for each job in the job stream and pass to the input service logic driver (IATISLG), one-at-a-time, the JDS entries for the multi-record files that are the input to the ISDRVR DSP. There is an input job associated with each JDS entry.

Module IATISLG

The IATISLG reads the multi-record file passed to it by IATISDV containing the records for the jobs. For each record, IATISLG must determine if it is a JECL or a JCL statement, interfaces the proper routine to process the statement, and places each statement in the appropriate data sets. When the EOF of the input multi record file is reached, the JES3 job control blocks needed for the life of the new job are finalized and added to the spool, and the JCT is added to the JES3 job queue. The new job can now be scheduled by JSS as JSS is posted that a new job has been added to the JCT spool data set.

Input service provides user exits that allow installation-written routines to define the scheduler elements of a standard job, to examine and modify incoming JCL and JECL, and to examine JES3 control blocks before the job is added to the JES3 job queue.

6.10 JES3 standard job created by ISDRVR DSP



Creation of a standard job

The ISDRVR function in a JES3 system creates all the jobs in the system and places them on the JCT data set and posts the job segment scheduler (JSS) to attempt to schedule the first scheduler element (SE) for the newly created job.

If a job contains no //*PROCESS control statements, the control statement processing phase defines the job as requiring the standard sequence of scheduler elements (SEs). The standard sequence is:

- Converter/interpreter (CI)
- Main device scheduler and generalized main scheduling (MAIN)
- Output service (OUTSERV)
- Purge (PURGE)

Job with JES3 control statements

If a job contains one or more //*PROCESS statements, the control statement processing phase defines the job as requiring the sequence of scheduler elements named on the //*PROCESS statements. The job may also contaon other JECL statements.

6.11 Input service driver job control

```
ISDRVR   IATYDSD PRTY=4,PABLE=YES,CSECT=IATISDT,
         DRVR=IATISDV,REENT=YES,MAXCT=2,
         JOBVAL=IATISJV

*I X,D=ISDRVR
 IAT8475 (IQDX) - ISDRVR   MXCT=000002 USE=000000 MOD=YES

*F X,D=ISDRVR,MC=20
 IAT8484 (MODX) - ISDRVR   OLDMXCT=000002 NEWMXCT=20

*I X,D=ISDRVR
 IAT8475 (IQDX) - ISDRVR   MXCT=000020 USE=000000 MOD=YES

*I A,D=ISDRVR
 IAT8520 NO JOBS ACTIVE ON ISDRVR
 IAT8593 INQUIRY ON ACTIVE JOBS COMPLETE,  0 JOBS DISPLAYED
```

Controlling input service jobs

The ISDRVR DSP reads batches of jobs from a spool data set and constructs a separate JCT entry for each job. An input service driver job is represented by a JCT containing two scheduler elements: one for the ISDRVR DSP and one for the PURGE DSP.

This segment of input service begins as each batch job produced by the reader phase is read from spool. Under the ISDRVR DSP, the JES3 job control blocks needed for the life of the job are built and the defaults in the control blocks are modified with the information retrieved from the job's control statements. The control blocks are written to spool and later used by JES3 functions in determining processing requirements for the job and its output data.

*I X,D=

This command displays maximum counts for the specified in the global address space.

*F X,D=

This command allows the operator to change the current DSP count.

*I A,D=

This command displays active ISDRVR jobs.

6.12 Jobs can enter system in hold status

- JES3 hold status conditions
  - OP - Operator hold status
  - DJ - DJC operator hold status
  - N - DJC network hold status
  - ER - Operator hold because of an error in the job
  - PR - Priority hold status
  - SP - Spool hold status
  - ARM - Automatic restart management hold status

```
*I Q H
IAT8674 JOB DRDR -10 (JOB27764) P=15 HOLD=(OP) ISDRVR
IAT8674 JOB DRDR -01 (JOB27765) P=15 HOLD=(OP) ISDRVR
IAT8674 JOB JOBA     (JOB02266) P=01 CL=A       HOLD=(OP) CI
IAT8674 JOB JOBB     (JOB02267) P=01 CL=A       HOLD=(OP) CI
IAT8674 JOB JOBC     (JOB02268) P=01 CL=A       HOLD=(OP) CI
IAT8674 JOB JOBD     (JOB02269) P=01 CL=A       HOLD=(OP) CI
IAT8674 JOB VAINIGRS (JOB24427) P=01 CL=A     HOLD=(OP) MAIN
IAT8595 INQUIRY ON JOB QUEUE STATUS COMPLETE,      7 JOBS DISPLAYED
```

Job hold status

When a job enters the system, there are a number of circumstances that can affect when a job is selected to run. For example, the job could be placed into operator hold, the job's resources may not be available, the job's class and group may not be enabled, or all initiators may be in use.

In response to an **`*I Q,H`** command, a job in priority hold only displays HOLD=PR if other hold types also apply. If PR is the only hold type that applies, IAT8674 is not issued for that job. If you need to see which jobs are in priority hold, use the **`*I Q,H,PR`** command.

In response to an **`*I,Q,H`** command, a job in priority hold only displays HOLD=PR if other hold types also apply. If PR is the only hold type that applies, IAT8674 is not issued for that job. If you need to see which jobs are in priority hold, use the **`*I,Q,H,PR`** command.

6.13 JES3 job control statements

- //*DATASET statement
- //*ENDDATASET statement
- //*ENDPROCESS statement
- //*FORMAT PR statement
- //*FORMAT PU statement
- //*MAIN statement
- //*NET statement
- //*NETACCT statement
- //*OPERATOR statement
- //**PAUSE statement
- //*PROCESS statement
- //*ROUTE XEQ statement

JES3 job control statements

Users code JES3 control statements with JCL statements to control the input and output processing of jobs. This segment of input service begins as each batch job produced by the reader phase is read from spool. Under the ISDRVR DSP (or the INTRDR DSP for internal readers), the JES3 job control blocks needed for the life of the job are built and the defaults in the control blocks are modified with the information retrieved from the job's control statements. The control blocks are written to spool and later used by JES3 functions in determining processing requirements for the job and its output data.

//*DATASET Use the //*DATASET statement to identify the beginning of an in-stream data set, which can contain JCL statements or data. (The //*ENDDATASET statement ends the in-stream data set.) The data set can be used as input to a dynamic support program (DSP), such as OUTSERV.

//*FORMAT Use the //*FORMAT PR statement to specify to JES3 processing instructions for sysout data sets that are printed. These instructions permit special processing of sysout data sets. The //*FORMAT PR statement applies only to sysout data sets printed by JES3. The statement is ignored for data sets sent to a TSO/E userid or processed by an external writer.

//*MAIN Use the //*MAIN statement to define the processor requirements for the current job. Many of the parameters are used to override parameters on the JES3 STANDARDS initialization statement.

//*NET Use the //*NET statement to define the dependencies between jobs in a dependent job control (DJC) network. JES3 sets up a network of dependent jobs and executes them in a specific order. (Once set up, the

structure of a DJC network cannot be changed unless all of the jobs in the network are resubmitted.) Jobs belonging to a DJC network cannot be registered with the automatic restart manager (ARM).

//*NETACCT Use the //*NETACCT statement to specify accounting information that JES3 is to transmit with a job to another node in the network.

//*OPERATOR Use the //*OPERATOR statement to issue a message to the operator. Columns 1 through 80 are written on the operator console and in the job's hard-copy log when JES3 reads in the job. **//**operator must issue a ***S** command or a remote work station with console level 15 must send a start message.

//*PROCESS Use the //*PROCESS statement to control how JES3 processes a job. A job that contains //*PROCESS statements receives only the JES3 processing specified on the //*PROCESS statements plus certain required processing. Specifically, the //*PROCESS statement calls a dynamic support program (DSP) in the DSP dictionary. JES3 must be able to process the called DSP.

//*ROUTE XEQ Use the //*ROUTE XEQ statement to send the following input stream to a network node where the job is then executed. JES3 stops transmitting input stream records when it finds one of the following:

- The second JOB statement after the //*ROUTE XEQ statement.
- The input stream runs out of card images.

All output from the job is assumed to print/punch at the originating node unless otherwise specified on a DEST parameter. The //*ROUTE XEQ statement must be given 80 character records.

Note: The //*ENDDATASET statement is required to end the data set statements. The //*ENDPROCESS statement is not a required statement.

See *z/OS MVS JCL Reference*, SA22-7597 for a complete description of these control statements.

Control statement considerations

If the job contains a //*MAIN control statement with the UPDATE parameter specified, the control statement processing phase adds the DISABLE SE (before the MAIN SE) and the ENABLE SE (after the MAIN SE) to the job's processing.

If the job contains one or more //*PROCESS statements, the job is defined as requiring the sequence of scheduler elements named on the //*PROCESS statements.

6.14 Facilities for debugging JCL

- JCLTEST Facility
  - // EXEC PGM=JCLTEST
- JSTTEST Facility
  - // EXEC PGM=JSTTEST
- //*PROCESS CBPRNT
  - //*PROCESS CI
  - DEBUG = ALL

```
>>──blank──┬─DEBUG=──┬─cbname──────────┬──┬──────────────────┬──┬──────────────────┬──><
           │         │  <─,────┐       │  │                  │  │                  │
           │         └─────cbname──┘   │  └─,CLASS=class─────┘  │                  │
           └─CLASS=class,DEBUG=──┬─cbname──────────┬──────────────┘                  │
                                 │  <─,────┐       │                                 │
                                 └─────cbname──┘   │                                 │
```

Debugging JCL

JCLTEST and JSTTEST are ways to diagnose errors that occur during CI or MDS processing.

JCLTEST facility

The JCLTEST facility generates a listing of interpreted JCL that has been processed by the MVS converter interpreter. You can use this listing to verify JCL results before allowing further processing of the job.

If you specify the PGM=JCLTEST parameter on an EXEC statement, JES3 stops processing the job when it completes converter/interpreter processing; the job is not scheduled for execution. The JCL and any applicable diagnostic messages are then printed. If you want to use JCLTEST for a deferred-restart job, you must specify PGM=JCLTEST on the EXEC statement located after the one for the step names on the RESTART parameter of the JOB statement.

The output from the JCLTEST facility is a listing of interpreted JCL. The JCL below runs the JCLTEST facility.

```
//        EXEC        PGM=JCLTEST
```

JSTTEST facility

The JSTTEST facility allows you to obtain summary information that describes the resources required by a job in order to execute. If you specify the PGM=JSTTEST parameter on an EXEC statement, JES3 uses the job's Job Summary Table (JST) to produce a summary of the devices that should be allocated to your job. JES3 then stops processing the job when it

completes converter/interpreter processing; the job is not scheduled for execution. The JCL and JSTTEST output are then printed. The JCL to run the facility is illustrated as follows:

```
//stepname  EXEC  PGM=JSTTEST
```

The JSTTEST facility uses the information in the job summary table (JST) to obtain information for the messages that describe the allocation decisions made during CI. Sets of messages are written to the JESMSGLG data set. The first set of messages describes the job step of the job.

CBPRNT DSP

For errors during CI processing:

- Run the failing job with a //*PROCESS CI statement followed by a DEBUG=ALL parameter statement.
- Run the failing job with a //*PROCESS CBPRNT statement followed by a DEBUG=ALL parameter statement.
- For SETUP problems, rerun the job with a // EXEC PGM=JSTTEST statement.
- Run the failing job with a //*PROCESS OUTSERV statement followed by a DEBUG=ALL parameter statement.

For MDS errors:

- If improper JCL handling is suspected, rerun the failing job with // EXEC PGM=JSTTEST.
- For job-related problems, run the DC DSP with the SNP option.
- Run the failing job with a //*PROCESS CI statement followed by:

```
DEBUG=ALL
//*PROCESS CBPRNT
//*PROCESS MAIN
//*PROCESS CBPRNT
//*PROCESS OUTSERV
```

The CBPRNT DSP prints JES3 and MVS control blocks. The blocks are written in the CBPRNT data set and printed at job termination. CBPRNT provides for the printing of the following control blocks:

- FRP
- JDAB
- JDS
- JST
- JVT
- MOSE
- CI SWA control blocks
- OSE
- PARM buffer
- RESQ (RQ)
- TAT (JBTAT)

6.15 Job using CBPRNT and DEBUG=ALL

```
//ROGERSZ JOB
(POK,999),'ROGERS',MSGCLASS=A,NOTIFY=ROGERS,
//      USER=ROGERS
//*PROCESS  CI
DEBUG=ALL
//*PROCESS  CBPRNT
//*PROCESS  OUTSERV
//PRINT    EXEC PGM=IEBGENER
//SYSPRINT  DD  SYSOUT=*
//SYSIN     DD  DUMMY
//SYSUT2    DD  SYSOUT=(Z,INTRDR)
//SYSUT1  DD DISP=SHR,DSN=ROGERS.JCL.VERS5(IEBGENER)
/*
//
```

CNPRNT and DEBUG JCL

This job will create two data sets for debugging purposes, as follows:

- A CBPRNT SYSOUT data set
- A DEBUG SYSOUT data set

DEBUG=cbname

DEBUG=cbname | ALL

Identifies the one or more control blocks to be printed. You must separate the control blocks by a comma if you specify more than one control block. The DEBUG or CLASS parameter must start on a new line following the //*PROCESS CI statement and a blank must precede the first parameter you specify. The syntax for using DEBUG and CLASS statements for the C/I Debug Facility follows:

//*PROCESS CI - followed by (on a separate statement) and preceded by a blank, as follows:

- DEBUG=cbname,cbname
- CLASS=class,DEBUG=cbname

cbname is the converter/interpreter debug facility specifcation that will print the following:

LOC Locate table (LVS) entries as they are built and the Locate Response Area (LRS).

JST Job summary table (JST) entries as they are built.

JVT Job volume table (JVT) entries as they are built.

COMP MVS control blocks as they are read and the compatibility interface records as they are referenced. Control blocks are the JCT, ACT, SCT, SIOT, and JFCB. Compatibility interface records are the JES3 job level, step level, and DD level records (JBL, STP, DDL).

SWB Scheduler work blocks after processing is complete.

PCAT Pass/catalog entries as they are built.

IJS Intermediate job summary table (IJS) entries as they are built.

CKPT Checkpointed buffers for any of the specified options. This provides the final contents of the control blocks.

ALL All the previously listed control blocks.

CLASS specification

Allows the user to assign a JES3 message class to the DEBUG data set for a job. The class must be a single alphabetic (A through Z) or a numeric (0 through 9) character. It can either precede or follow the DEBUG=keyword and must be separated by a comma.

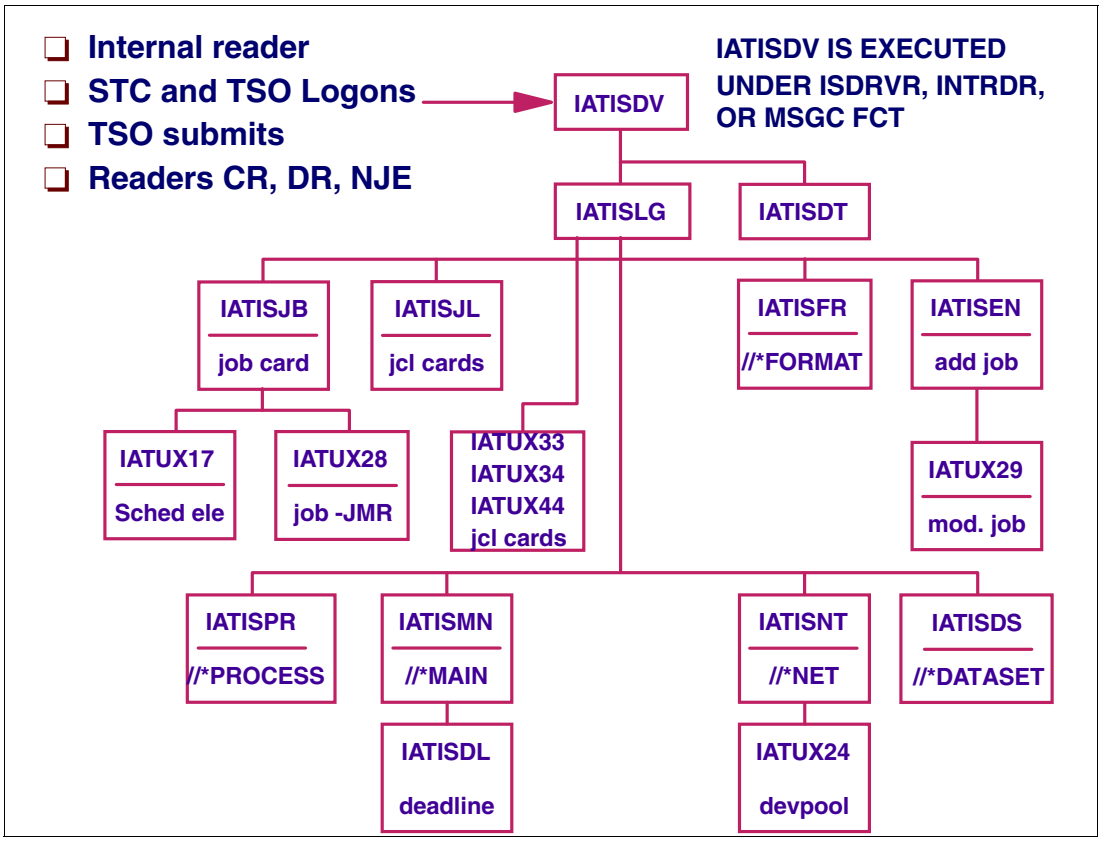
If you do not specify the message class for the converter/interpreter debug facility, JES3 uses the message class specified by the DBGCLASS parameter on the STANDARDS initialization statement.

Specifying DEBUG and CLASS

```
VALID specifications:
   DEBUG=ALL
   DEBUG=ALL,CLASS=M
   DEBUG=IJS,JST,JVT,CLASS=C
   CLASS=4,DEBUG=ALL

Not valid specifications:
   DEBUG=ANY
   CLASS=*,DEBUG=ALL
   DEBUG=(IJS,JST),CLASS=E
```

6.16 Input service modules and exits



Input service modules and exits

This figure shows the input service module flow and when the input service user exits are entered. The scheduler element the this processing takes place under is ISDRVR or INTRDR. All modules are loaded and entered as they are needed.

Input service provides user exits that allow installation-written routines to define the scheduler elements of a standard job, to examine and modify incoming JCL or JES3 control statements, and to examine JES3 control blocks before the job is given to the MVS converter/interpreter.

- IATISDV -- Input Service Driver
- IATISLG -- Input Service Logic Driver
- IATISDT -- Input Service Data Csect
- IATISJB -- Input Service Job Card Processor
- IATISJL -- Input Service Jcl Processor
- IATISDS -- //*DATASET Control Card Processor
- IATISPR -- //*PROCESS and //*ENDPROCESS Statement Processor
- IATISMN -- Input Service //*MAIN Statement Processor
- IATISFR -- Input Service //*FORMAT Statement Processor
- IATISDL -- Input Service Deadline Processing Routines
- IATISNT -- Input Service //*NET Processing Routine
- IATISEN -- Input Service End-of-Task Routine

6.17 Input service exits

The 7 Input Service Exits

- UX17 - Installation specified scheduler elements
- UX28 - Examine job card
- UX33 - Examine EXEC and //*
- UX34 - Examine JCL DD statements
- UX44 - Examine // OUTPUT statement
- UX24 - Examine //*NET parameters
- UX29 - Examine ALL job control blocks

Input service exits

Input service provides user exits that allow installation-written routines to define the scheduler elements of a standard job, to examine and modify incoming JCL or JES3 control statements, and to examine JES3 control blocks before the job is given to the MVS converter/interpreter. The input service exits are:

UX17 This exit is only called for normal jobs, it is not called for demand select jobs (e.g. TSO logons and started tasks). Basically, it allows the installation to specify the default scheduler elements for the standard job. For example if the installation could insert an "archive scheduler segment" between OUTSERV and PURGE to save SYSOUT data sets that are appropriately marked (e.g. USER1 or USER2 bits turned on). The exit is limited to supplying a maximum of 10 SEs without modification of JES3 data areas.

Installation exit IATUX17 allows an installation to modify or set the scheduler elements to an installation standard set.

UX28 This exit is called for all the card images that comprise the MVS // JOB card. The exit will NOT see the imbedded comments on the job card.

This has been the standard exit to do accounting information verification for batch jobs. Depending on how the installation decides whether the user is allowed to use the account code, this function is better performed in the C/I FSS to avoid bottlenecks and local lock contention in the JES3 Global address space. However, this is the best exit to extract the accounting information and store it in the SMF26 record if it is imperative to have the information for jobs that may fail prior to the convertor phase or not execute on this complex.

UX28 is the only place where the job card for started tasks may be altered so that the results are visible. One can easily place accounting information, MSGCLASS, MSGLEVEL, etc. on the job card.

UX44 This exit will see all MVS JCL statements other than // JOB, // EXEC or // DD. It will see statements like // OUTPUT, // PROC and // PEND. Furthermore, the exit will see any of the comments imbedded in these statements. It will also see statements that are "unknown".

UX34 This exit will see all of the // DD statement cards and any imbedded comments statements.

UX33 This exit will see all JES3 JECL statements except //*ENDDATASET (contrary to documentation, the exit will see //*DATASET). The exit will see all stand alone comment statements. UX33 sees all the EXEC statements and any embedded comments.

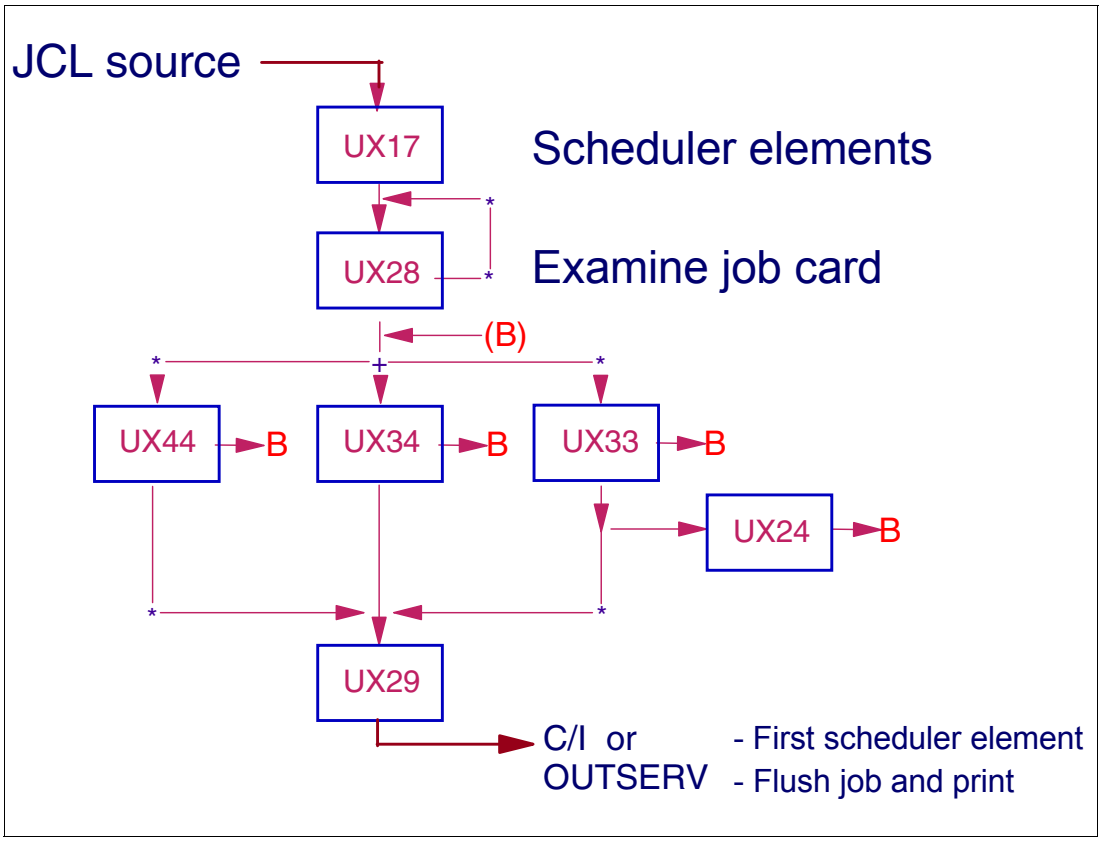
This exit is the place to add default // OUTPUT statements to started tasks. As an aside, don't add JES3 JECL cards to demand select jobs, JES3 will fail the job should this be detected.

UX24 This exit is called only if a //*NET statement is being processed that contains a DEVPOOL= keyword.

UX29 This exit is called once at the end of input service. It will see most jobs that have been failed and flushed.

At this point all of the information that can be detected about the job is available to the installation. This exit can be used to override anything that the user may have specified on JECL statements. The installation can also override JES3 defaults such as staging area limits or ring check.

6.18 Input service exit flow



Input service exit flow

The input service exits are entered in the order shown in the visual.

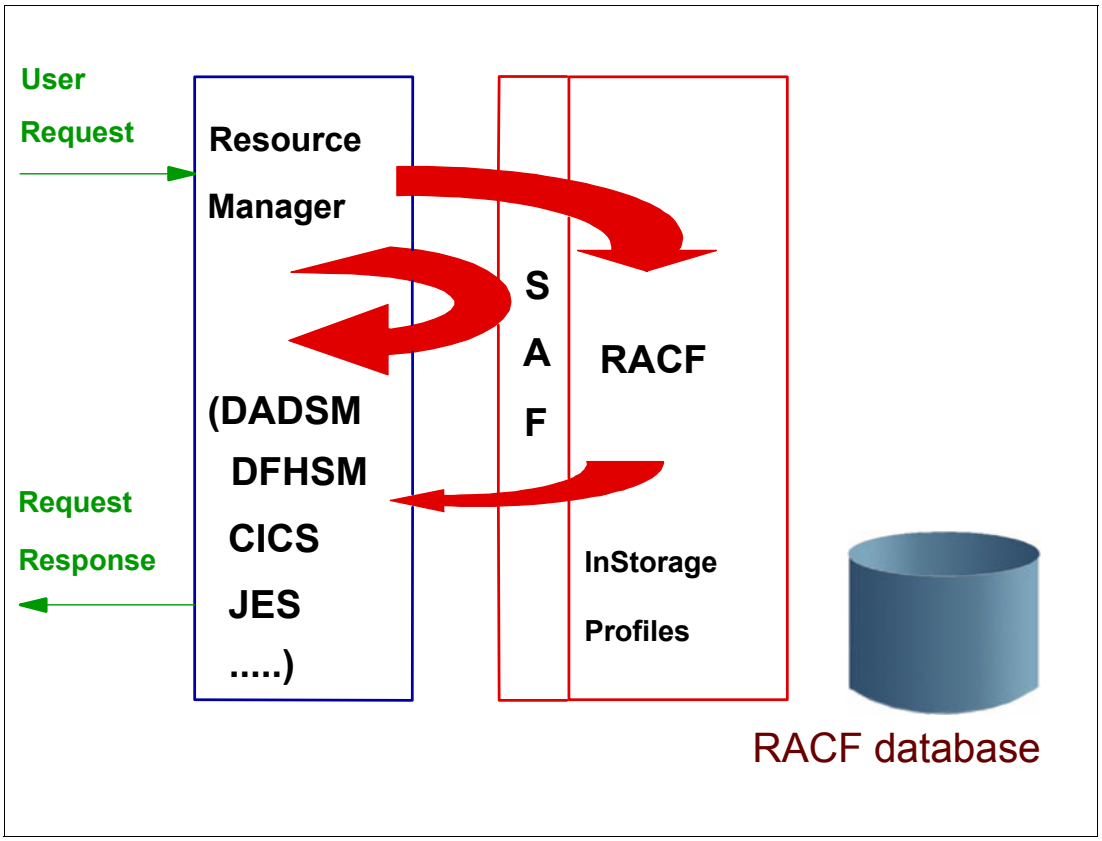
All of the exits in this phase of processing run only in the JES3 global's address space under IATNUC's task.

Exit IATUX29

The above exits are called until all statements comprising the job are processed or until an error occurs that causes the job to be flushed. This is the last exit entered and JES3 spool control blocks can be modified in this exit. After examining all control blocks and job characteristics, this exit can cause the job to be flushed from the system.

If the job is to be flushed, the CI and MAIN scheduler elements are marked complete and the job prints and purges from the system.

6.19 Security request overview



Security request overview

JES3 accesses RACF services through the MVS System Authorization Facility (SAF). Through SAF, JES3 passes security information about jobs and resources to RACF, which evaluates the security information and returns the results of that evaluation to JES3. JES3 then enforces the results of the security check, such as permitting or denying access to a data set or allowing a job to process. If RACF is inactive or not installed, SAF returns default security information to JES3. JES3 may or may not perform alternative security processing depending on how you tailor your system through JES3 initialization statements or installation exits.

System Authorization Facility (SAF)

The System Authorization Facility (SAF) provides a system interface that conditionally directs control to the Resource Access Control Facility (RACF), if RACF is present, and/or a user-supplied processing routine when receiving a request from a resource manager. Another external security product may be substituted for RACF. SAF does not require any other program product as a prerequisite, but overall system security functions are greatly enhanced and complemented by the concurrent use of an external security product. The key element in SAF is the MVS router.

MVS router

SAF provides an installation with centralized control over system security processing by using a system service called the MVS router. The MVS router provides a focal point and a common system interface for all products providing resource control. The resource managing components and subsystems call the MVS router as part of certain decision-making functions

in their processing, such as access control checking and authorization-related checking. These functions are called "control points". This single SAF interface encourages the use of common control functions shared across products and across systems.

The router is always present whether or not an external security product is present. If an external security product is available in the system, the router passes control to the external security product. Before it calls the external security product, the router calls an optional, user-supplied security processing exit if one has been installed.

MVS router exit

The MVS router provides an optional installation exit that is invoked whether or not RACF or another external security product is installed and active on the system. If an external security product is not available, the router exit acts as an installation-written security processing (or routing) routine. If an external security product is available, the exit acts as an external security product preprocessing exit.

Resource managers

Resource managers are responsible for calling SAF to determine whether to allow a user access to the system or to a resource. The resource manager is responsible for enforcing the decision made by SAF or RACF. Resource managers include:

- DADSM for data set access authority
- DFHSM for data set allocation authority
- CICS® for CICS sign-on and transaction authorization
- JES3 for user identification and verification

Based on the original user's request, the resource manager formulates a request and passes it to SAF. Depending on the nature of the request, SAF may respond directly or may pass the request to RACF. In either case, the user receives a response from the resource manager after the resource manager considers the response from SAF.

How JES and RACF work together

JES requests RACF services by issuing the RACROUTE macro. The MVS system authorization facility (SAF) handles the RACROUTE macro invocation. If RACF is installed, SAF passes the security information specified by JES on the RACROUTE macro invocation to RACF. RACF evaluates the security information and returns the results of that evaluation to JES. JES then enforces the results of the security check, such as permitting or denying access to a data set, or allowing a job to execute.

6.20 JES3 job validation objectives

- Provide early validation for jobs
  - USERID - GROUP - SECLABEL - PASSWORD
    - From Job card
- Propagation of user's security to job
  - Internal batch jobs only
- Validation based on active resource classes
  - Port of Entry (POE) - JESINPUT
  - Job name control - JESJOBS
  - Surrogate user support - SURROGAT
  - Is job submitted by NJE? - NODES
- IATXSEC macro
  - Setup parameters and invoke JES3 security processing service routine IATGRSC.

Job validation objectives

Jobs entering the JES3 complex are verified at job input time. Job validation occurs during the input phase of processing and consists of the following checks when an applicable resource class is activate.

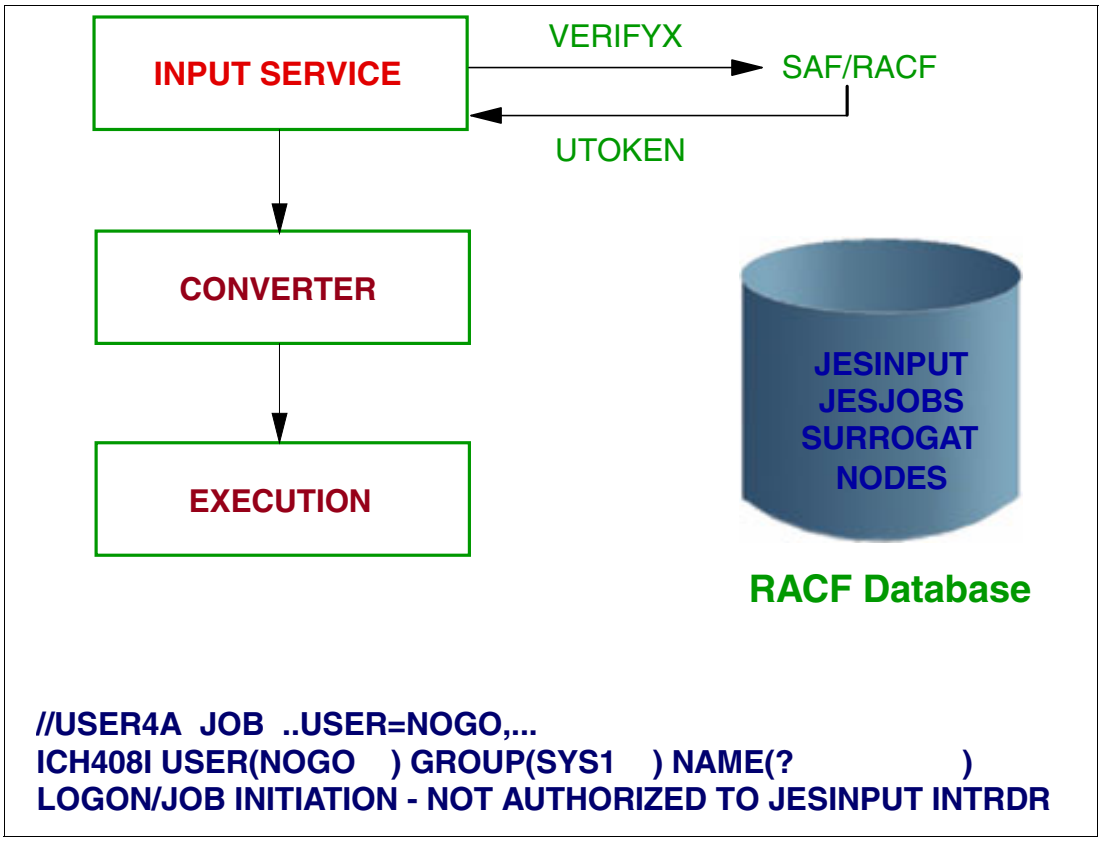
- USERID - GROUP - SECLABEL - PASSWORD -- From Job card
- Propagation of user's security to job
- Port of Entry (POE) - JESINPUT check
- Job name control - JESJOBS check
- Surrogate user support - SURROGAT check

A **UTOKEN** is returned after job validation.

- UTOKEN. The RACF user security token. A UTOKEN is an encapsulation or representation of the security characteristics of a user. RACF assigns a UTOKEN to each user in the system.
- STOKEN. A UTOKEN associated with a user who has submitted work.
- RTOKEN. The RACF resource security token. An RTOKEN is an encapsulation or representation of the security characteristics of a resource. Resource managers, for example JES, can assign RTOKENs to the resources they manage; for example, JES spool files.

IATXSEC macro This macro is used to setup the necessary parameters and invoke the JES3 security processing service routine IATGRSC.

6.21 JES3 security verification at input



JES3 security verification at input

JES3 accesses RACF services through the MVS System Authorization Facility (SAF). Through SAF, JES3 passes security information about jobs and resources to RACF, which evaluates the security information and returns the results of that evaluation to JES3. JES3 then enforces the results of the security check, such as permitting or denying access to a data set or allowing a job to process. If RACF is inactive or not installed, SAF returns default security information to JES3. JES3 may or may not perform alternative security processing depending on how you tailor your system through JES3 initialization statements or installation exits.

JES3 DSPs use the IATXSEC macro service to setup the necessary parameters and invoke the JES3 security processing service routine IATGRSC. IATGRSC, JES3 common security processing, routes the requests to the security product using the RACROURE macro. When the security product finishes its processing, control is returned to the caller with the appropriate information and return code.

Input service verification

JES3 input service (IATISEN) determines if jobs can enter the system by issuing an IATXSEC VERIFYX request to invoke RACROUTE VERIFYX processing. The RACROUTE REQUEST=VERIFYX macro verifies a user and builds a UTOKEN based on the information passed in the parameter list, and also handles the propagation of submitter ID.

SAF either rejects or accepts the job:

- If the job is rejected, an error token is returned. RACF messages are written to the job's JESMSGLG data set and the job is flagged to be flushed from the system. The JESMSGLG data set is the only output printed.
- If the job is accepted, SAF propagates the current RACF userid, and, if used, the security label (SECLABEL) if this security information is not specified on the job statement.

SAF also returns a UTOKEN.

JES3 RACF resource classes

JES3 uses the RACF resource classes listed below. To enable RACF protection, the resource classes must be made active and profiles defined to grant users access to the resources:

JESINPUT

JESINPUT RACF class permits conditional access for commands or jobs that are entered into the system from a JES input device. After creating at least one profile to allow access, activate the JESINPUT class:

```
SETROPTS CLASSACT(JESINPUT)
```

JESJOBS

JESJOBS RACF class permits an installation to control which job names users can submit or cancel. For example, you can permit certain job names to be entered only by certain userids or groups. This class is also used by TSO to cancel jobs by job name. After creating at least one profile to allow access, activate the JESJOBS class:

```
SETROPTS CLASSACT(JESJOBS)
```

SURROGAT

SURROGAT RACF class allows an installation to establish surrogate userids. An user,user1, can submit jobs on behalf of another user, user2, without having to specify user2's password. Jobs submitted in this way by user1 execute with the authority of user2. To activate the SURROGAT class, specify:

```
SETROPTS CLASSACT(SURROGAT)
```

NODES

User verification for NJE jobs normally is done at the execution node. Submitter information is propagated from trusted nodes. To provide security for Network Job Entry (NJE), activate the NODES class (for inbound work) or the WRITER class (for outbound work), and define the profiles needed to enforce your installation's NJE security policy. As with other RACF-protected resources, you must gather information from your JES system programmer to define profiles.

Note: The RACROUTE REQUEST=VERIFYX request is not issued for started tasks, TSO logons, and mounts. MVS will do the required security checks.

6.22 Job validation and propagation

- All jobs are validated at input time
- Implemented using
  - IATXSEC VERIFYX
    - RACROUTE request VERIFYX
- Propagation of submittor's:
  - USERID
  - SECLABEL
- Propagation only done for
  - INTRDR submitted jobs
  - NJE jobs from "TRUSTED" nodes

Job validation and propagation

All jobs are validated at input time as JES3 requests SAF services by using the IATXSEC macro. At JES3 input service, this request is a IATXSEC VERIFYX. This request is issued by JES3 for security authorization checking for job entry.

When RACF is active, RACF ensures that the job password, userid, group ID, and security label are valid before allowing userid, groupid, and security label are valid before allowing the job to be processed.

User propagation

For each previously-validated RACF user who is submitting a batch job to JES through a JES internal reader, SAF propagates the following security information to the batch job:

- If USER is not specified on the JOB statement, the current RACF userID is used.
- If PASSWORD is not specified on the JOB statement, the current user password is not required if the submitter propagates.
- If SECLABEL is not specified on the JOB statement, the submitter's current security label is used.

INTRDR propagation

SAF propagates the current RACF userid from each (already validated) RACF user who submits a batch job either by using the INTRDR or the TSO SUBMIT command. Thus, jobs executed within the same JES complex from which they are submitted, are automatically

identified with the user and the user's RACF profile. The other user information that can be propagated is the SECLABEL.

NJE jobs

With NJE jobs, the submitter information used depends on whether the submitting node is trusted. If the submitting node is trusted, the submitter information is either used as passed or translated through NODES profiles. This information is subject to reverification during any submit check that may be performed. This is consistent with local jobs.

Tip: Security classification of users and data allows installations to impose additional access controls on sensitive resources. Each user and each resource can have a security classification in its profile. You can choose among the following:

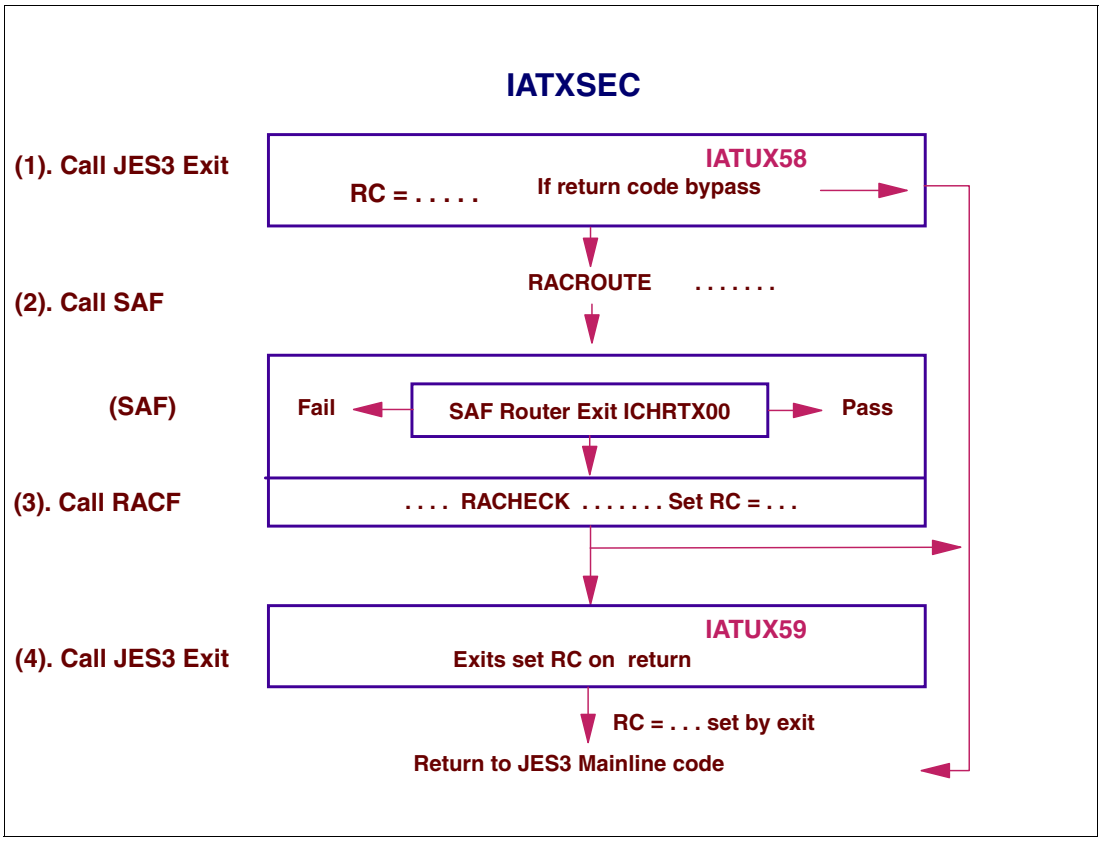
- Security levels, security categories, or both
- You can use security labels, which are a combination of security levels and security categories, and are easier to maintain

A security level (SECLEVEL) is an installation-defined name that corresponds to a numerical security level (the higher the number, the higher the security level).

A security category (CATEGORY) is an installation-defined name that corresponds to a department or an area within an organization in which the users have similar security requirements.

A security label (SECLABEL) is an installation-defined name that corresponds to a security level and zero or more security categories.

6.23 JES3 security call and exits



JES3 security call and exits

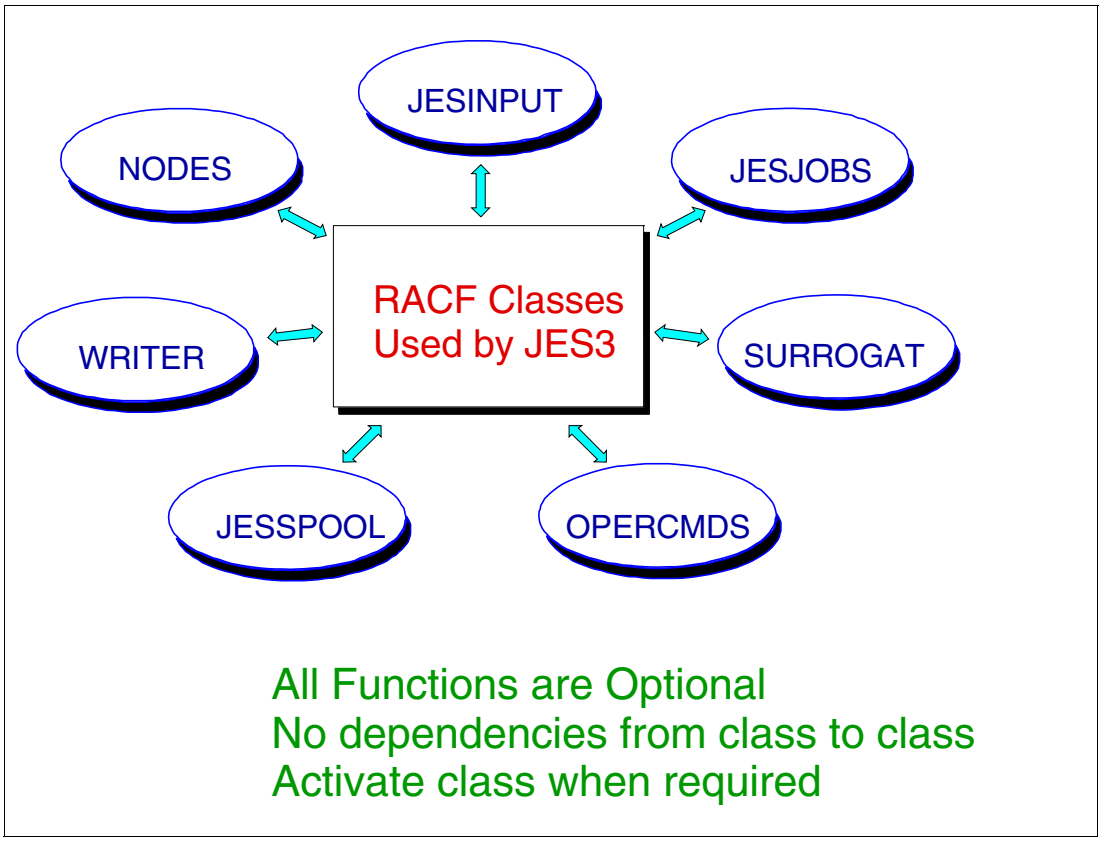
JES3 calls SAF/RACF for all security decisions and auditing. If no decision is made, any existing default checking is used. Access to the SAF/RACF services is provided by the IATXSEC macro service.

This single macro calls the JES3 common security processing routine, IATGRSC, which calls the user exit IATUX58 before the SAF/RACF calls. The exit can modify security checks or make security decisions for JES3. The security decisions that can be controlled include restricting access to SYSIN/SYSOUT data sets, controlling the use of operator commands, and controlling network jobs and data.

If IATUX58 does not make the security decision, IATGRSC calls SAF by issuing the RACROUTE macro. After the SAF call JES3 user exit IATUX59 is invoked.

IATUX59 can again modify security checks or to make security decisions for JES3.

6.24 RACF classes used to protect JES3 resources



RACF classes used to protect JES3 resources

To activate RACF protection, have your security administrator enable the RACF class for each type of resource that you want to protect. A RACF class is a collection of RACF-defined entities, such as data sets or devices, that have similar characteristics. In addition to activating RACF classes, the security administrator will typically also need to define RACF profiles that grant users access to these resources. The RACF resource classes used by JES3 are:

JESINPUT JESINPUT RACF class permits conditional access for commands or jobs that are entered into the system from a JES3 input devices. After creating at least one profile to allow access, activate the JESINPUT class:

```
SETROPTS CLASSACT(JESINPUT)
```

JESJOBS JESJOBS RACF class permits an installation to control who can submit jobs by job name. For example, you can permit certain job names to be entered only by certain user IDs or groups. This class is also used by TSO to control cancellation of jobs by job name. After creating at least one profile to allow access, activate the JESJOBS class:

```
SETROPTS CLASSACT(JESJOBS)
```

SURROGAT SURROGAT RACF class allows an installation to establish surrogate userids. User1 can submit jobs on behalf of another user, user2, without having to specify user2's password. Jobs submitted in this way

by user1 execute with the authority of user2. To activate the SURROGAT class, specify:

```
SETROPTS CLASSACT(SURROGAT)
```

NODES

NODES RACF class controls whether jobs and SYSOUT can enter the system from other nodes. Also, RACF can control whether the jobs entering the system from other nodes need user ID and password verification checking. The NODES class is also used to translate userids, groups, and SECLABELs. To activate the NODES class, specify:

```
SETROPTS CLASSACT(NODES) RACLIST(NODES)
```

WRITER

WRITER RACF class controls where output can be sent. You can restrict or authorize the use of writers for local printers and punches, RJE devices, and NJE transmissions. For NJE, RACF verifies the security of jobs and SYSOUT transmissions to ensure that the user is authorized to send data to another node in a network. After creating at least one profile to allow access, activate the WRITER class:

```
SETROPTS CLASSACT(WRITER)
```

JESSPOOL

JESSPOOL RACF class permits an installation to protect the data that resides on the JES spool. It prevents unauthorized users from reading, modifying, printing, deleting, or copying a job's data. After creating at least one profile to allow access, activate the JESSPOOL class:

```
SETROPTS CLASSACT(JESSPOOL)
```

OPERCMDS

OPERCMDS RACF class permits an installation to authorize a userid to a command or group of commands by creating a RACF user profile for the console and placing the console's or operator's userid in the access list of the OPERCMDS profile. Profiles can be defined in the OPERCMDS class to authorize all operator commands. To activate the OPERCMDS class, specify:

```
SETROPTS CLASSACT(OPERCMDS)   RACLIST(OPERCMDS)
```

CONSOLE

CONSOLE RACF class permits an installation to specify which consoles operators can use to enter certain commands. After creating at least one profile to allow access, activate the CONSOLE class:

```
SETROPTS CLASSACT(CONSOLE)
```

6.25 Writing user exits

- Creating User Exits
  - Using standard JES3 exits - (IATUXxx)
  - Creating user DSPs
  - Writing function in user memory
- What is Needed ?
  - New function
  - Standard exit decisions
    - Whats available in Exits

Writing user exits

Installation exits enable an installation to tailor JES3 without having to modify JES3 code. By not modifying JES3 code, you reduce the amount of work necessary to install JES3 maintenance or new JES3 function. Thus, installation exit routines can be very useful in helping you develop an operating system tailored to your needs.

IBM supplies a module for each JES3 installation exit. All exits begin with IATUXxx. (Some exits perform a default function, but most are dummy modules). You may, of course, write your own installation exit routine in place of any provide by IBM. For each exit, either the IBM-supplied routine or an installation exit routine must reside in the JES3 module library, or JES3 issues warning messages IAT3020 and IAT3102.

JES3 is a general job entry subsystem of MVS and sometimes cannot satisfy all installation-specific needs at a given installation. If you modify JES3 code to accomplish your specific functions, you then are susceptibly to the migration and maintenance implications that result from installing new versions of JES3. JES3 exits allow you to modify JES3 processing without directly affecting JES3 code. In this way you keep your modifications independent of JES3 code, making migration to new JES3 versions easier and making maintenance less troublesome.

All JES3 installation exits and JES3 modules, data areas, etc. (with very few exceptions) reside in virtual storage above 16 megabytes and execute in 31-bit addressing mode. Coding techniques assume 31-bit addressing mode to be the rule and not the exception. When certain segments of code need to refer to data below 16 megabytes, this book will clearly state so.

6.26 JES3 exit register conventions

- Register Conventions for modules they call exits

R0, R1	Parameter information
R2-R9	DSP scratch registers
R10	DSP base register
R11	FCT address
R12	TVT address
R13	CSECT address
R14	Return address
R15	Entry point or return code

JES3 exit register conventions

Register usage for installation exits is defined for each exit in In general, registers 11, 12, 14, and 15 on entry are as defined below but other input registers are also provided. Also, most installation exits use the same register conventions.

JES3 stores the following data in the registers before passing control to a DSP:

- Registers 0 through 10 are undefined.
- Register 11 contains the address of the function control table (FCT) for the DSP.
- Register 12 contains the address of the JES3 transfer vector table (TVT).
- Register 13 is the base register of the DSP's data CSECT, if one is defined for the DSP.
- Register 14 is the address to which a called routine must return.
- Register 15 is the entry point address of the called program.

JES3 DSP register usage

JES3 DSPs use the following register conventions that all installation exit code for JES3 should also use.

Exception: JES3 installation exits that run outside of the JES3 address space may follow other register conventions (for example, IATUX26 and IATUX57). In these cases, check the installation exit for register conventions

The JES3 register notation follows:

- Registers 0, 1, and 2 should be reserved for passing parameters when invoking JES3 services.
- Registers 3 through 9 are used as work registers.
- Register 10 is the base register for all DSPs. Register 9 is the recommended second base register if one is required.
- Register 11 contains the address of the function control table (FCT) for the DSP. The multifunction monitor depends on the correct setting of this register at all times to maintain control of the system. A user-written DSP must never alter this register.
- Register 12 contains the address of the JES3 transfer vector table (TVT). A user-written DSP must never alter this register.
- Register 13 is the base register of the DSP's work area or data CSECT.
- Register 14 is the address to which a called routine must return.
- Register 15 is the address of the called program. When control is returned to the calling program, register 15 contains the return code.

6.27 *I A command

Displaying the status of active jobs - *I A

```
IAT8499 NO JOBS ACTIVE ON SC64
IAT8524 JOB ZFS      (JOB32615) ZFS      ON SC70     098562.90 MIN
IAT8524 JOB OAM      (JOB32612) OAM      ON SC70     098562.90 MIN
IAT8524 JOB OPTSO    (JOB32620) OPTSO    ON SC70     098562.90 MIN
IAT8524 JOB ASCHINT  (JOB32621) ASCHINT  ON SC70     098562.90 MIN
IAT8524 JOB VTAM44   (JOB32622) VTAM44   ON SC70     098562.90 MIN
```

```
>>──┬─*INQUIRY─┬──,A──┬──────────────────────────────────────────┬──────────────><
    └─*I───────┘      │─,C=class─────────────────────────────────│
                      │─,D=─┬─dspname──────────────────────────┬─│
                      │     │─ALL──────────────────────────────│ │
                      │     │─DEADLINE─────────────────────────│ │
                      │     │─DLINE────────────────────────────│ │
                      │     └─INTRDR─┬──────┬──────────────────┘ │
                      │              └─,R───┘                    │
                      │─,G=group─────────────────────────────────│
                      │─,J=jobno─────────────────────────────────│
                      │──┬─,SRVCLASS=─┬─srvclass────────────────│
                      │  └─,SRVCL=────┘                          │
                      └─,main─┬──────────────────────────────────┬│
                              │─,C=class─────────────────────────│
                              │─,G=group─────────────────────────│
                              └──┬─,SRVCLASS=─┬─srvclass─────────┘
                                 └─,SRVCL=────┘
```

Using the *I A command

You can use the **`*I A`** command to display the following active functions:

- Status of jobs active on a particular DSP
- Names of jobs waiting to be transmitted to other nodes via BSC/NJE, and the transmission paths for the jobs
- Status of JES3 jobs active on each main
- Status of jobs in execution for the specified job class.
- Status of jobs in execution for the specified job class group.
- Status of jobs in execution for the specified service class.
- Status of active jobs in the deadline queue
- Names of jobs that are currently being transmitted to other nodes
- Names of the logical senders handling the transmissions

Note: This display does not include the names of jobs that were put in action as a result of using the **`*F NJE,NAME=,HOLD`** command. Use the **`*I Q,H`** command to display these jobs. Use the **`*I A,D=NJESND`** command only for BSC/NJE communication.

6.28 *I A command examples

```
*I A,D=CI
IAT8522 JOB JOB200      (JOB33518) ACTIVE  ON CI        0000.59 MIN
IAT8522 JOB JOB11       (JOB33604) ACTIVE  ON CIFSS1    0000.12 MIN
IAT8522 JOB JOB2        (JOB33606) ACTIVE  ON CIFSS1    0000.09 MIN
*I A,D=NJESND
IAT8522 JOB jobno jobname ACTIVE ON NJESND TIME NAVAIL   SCHED FOR nodename
IAT8522 JOB jobno jobname ACTIVE ON NJESND lsender  time
*I A,SC65
IAT8524 JOB RACF        (JOB32766)             ON SY65     004502.93 MIN
IAT8524 JOB SYSLOG      (JOB32765)             ON SY65     004502.93 MIN
IAT8524 JOB JES3CI      (JOB33379) CIFSS2  ON SY65     004364.71 MIN
IAT8524 JOB GTFJES3     (JOB33401) GTF            ON SY65     004218.21 MIN
IAT8524 JOB JOB23       (JOB33727) STEP1   ON SY65     000001.03 MIN
IAT8524 JOB JOB11       (JOB33718) STEP1   ON SY65     000000.50 MIN
IAT8524 JOB JOB26       (JOB33734) STEP1   ON SY65     000000.01 MIN
IAT8593 INQUIRY ON ACTIVE JOBS COMPLETE,    7 JOBS DISPLAYED
*I A,SC65,C=A
IAT8524 JOB JOB12       (JOB33827) STEP1   ON SY1       000001.30 MI
IAT8524 JOB JOB30       (JOB33834)             ON SY1       000000.40 MI
IAT8524 JOB JOB33       (JOB33846)                ON SY1       000000.09 MI
IAT8593 INQUIRY ON ACTIVE JOBS COMPLETE,    3 JOBS DISPLAYED
```

Command examples

For **`*I A,D=CI`**, JES3 displays the following message for each job active in CI. If a job is being processed by a C/I FSS address space, the C/I FSS name is displayed.

The **`*I A,D=NJESND`** command displays the names of all jobs waiting to be transmitted and all jobs currently being transmitted to other nodes in the network. JES3 displays the following message for each job waiting to be transmitted:

```
IAT8522 JOB jobno jobname ACTIVE ON NJESND TIME NAVAIL SCHED FOR nodename
```

JES3 displays the following message for each job currently being transmitted:

```
IAT8522 JOB jobno jobname ACTIVE ON NJESND lsender  time
```

To display the status of all jobs active on system SY65, Enter **`*I A,SY65`** and JES3 displays the messages shown in the visual.

To display the status of jobs in class A that are active on SY65, Enter **`*I A,SY65,C=A`** and JES3 displays the messages shown in the visual.

6.29 *I B command

Displaying the status of active jobs - *I A

```
IAT8499 NO JOBS ACTIVE ON SC64
IAT8524 JOB ZFS      (JOB32615) ZFS      ON SC70     098562.90 MIN
IAT8524 JOB OAM      (JOB32612) OAM      ON SC70     098562.90 MIN
IAT8524 JOB OPTSO    (JOB32620) OPTSO    ON SC70     098562.90 MIN
IAT8524 JOB ASCHINT  (JOB32621) ASCHINT  ON SC70     098562.90 MIN
IAT8524 JOB VTAM44   (JOB32622) VTAM44   ON SC70     098562.90 MIN
```

```
>>──┬─*INQUIRY─┬──,A──┬──────────────────────────────────────────────────┬──────────────><
    └─*I───────┘      │ ┌──────────────────────────────────────────────┐ │
                      │ │,C=class──────────────────────────────────────│ │
                      │ │,D=──┬─dspname──────────────────────────────┬─│ │
                      │ │     ├─ALL──────────────────────────────────┤ │ │
                      │ │     ├─DEADLINE─────────────────────────────┤ │ │
                      │ │     ├─DLINE────────────────────────────────┤ │ │
                      │ │     └─INTRDR──┬──────┬─────────────────────┘ │ │
                      │ │               └──,R──┘                       │ │
                      │ │,G=group──────────────────────────────────────│ │
                      │ │,J=jobno──────────────────────────────────────│ │
                      │ │──┬─,SRVCLASS=──┬─srvclass──────────────────│ │
                      │ │  └─,SRVCL=─────┘                            │ │
                      │ └─,main──┬──────────────────────────────────┬──┘ │
                      │          │ ┌──────────────────────────────┐ │    │
                      │          │ │,C=class──────────────────────│ │    │
                      │          │ │,G=group──────────────────────│ │    │
                      │          │ │──┬─,SRVCLASS=──┬─srvclass────┬─│ │    │
                      │          │ │  └─,SRVCL=─────┘             │ │ │    │
                      │          │ └──────────────────────────────┘ │ │    │
                      └──────────┴──────────────────────────────────┴─┴────┘
```

*I B command

Use the **`*I B`** command to display the following backlog of jobs at each JES3 function:

- The number of jobs backlogged for each JES3 function (DSP).
- The number of jobs backlogged for a job class.
- The number of jobs backlogged for a job class group
- The number of jobs backlogged for a service class.
- The number of jobs backlogged for a terminal group.
- The number of jobs backlogged for a main or all mains.

7

Converter/interpreter processing

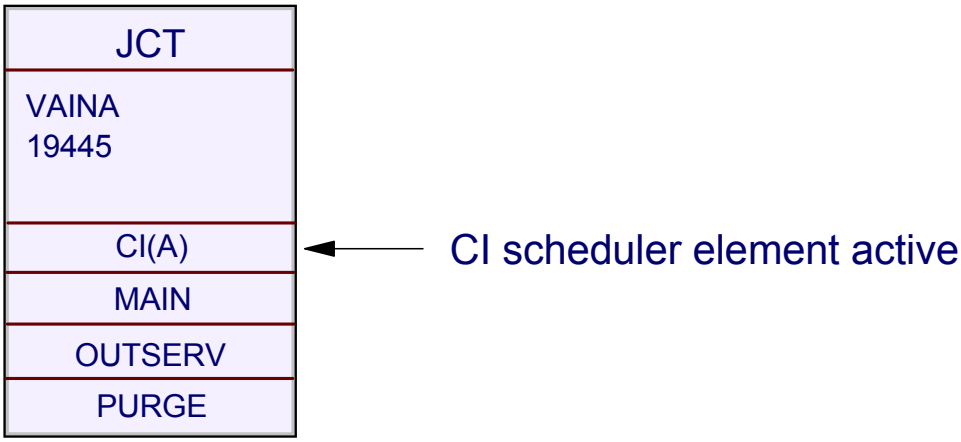
The JES3 converter/interpreter (C/I) service controls the conversion of JCL statements to converter/interpreter text and then into SWA control blocks. This service comprises primarily the JES3 CI and POSTSCAN dynamic support programs, the C/I subtasks under which the MVS C/I routines run, and the initiator support routines.

The Converter/Interpreter processing of a job control language (JCL) statement includes the stage of converting the JCL statement to converter / interpreter (C/I) text, a form of data that the job entry subsystem (JES) and the job scheduler function of MVS both recognize. The converter takes the job's JCL, merges it with JCL from a procedure library, and converts the composite JCL into C/I text. The converter scans each JCL statement for syntax errors and issues appropriate error messages. The converter also resolves symbolic parameters and assigns default values. The C/I text is further interpreted to build the necessary control blocks needed before the job can be scheduled for execution. The output of the interpretation is stored the scheduler work area (SWA).

The three JES3 principal phases of C/I service are:

- **Converter/Interpreter phase**: Uses the MVS C/I routines to convert and interpret JCL into scheduler control blocks. At this time, the scheduler control blocks are created in the scheduler work area (SWA).
- **Prescan phase**: Extracts job requirements from the scheduler control blocks for use in the postscan phase. At the end of the prescan phase, the scheduler control blocks are moved from the SWA to JES3 spool.
- **Postscan phase**: Locates data sets and gathers information for use in JES3 device setup.

7.1 Job active at CI scheduler element



- JSS calls C/I Sheduler IATIICS to schedule a job for CI:
  - Global CI or CIFSS
    - IATUX46 - Select Processors Eligible for C/I Processing
    - IATUX49 - Override Address Space Selected for C/I Processing
  - CLASS
    - CIBATCH=JOB|CLASS|ANY - Limit CI processing eligibility
    - CIDEMAND=JOB|CLASS|ANY - Limit CI processing eligibility
  - STANDARDS CIBATCH= and CIDEMAND=

Job active at CI scheduler element

When JES3 has a job that can be scheduled for C/I processing, it determines whether a CI DSP in the JES3 global address space or a C/I FSS address space is available for the job. You can select the main where the job will be scheduled for C/I service as well as whether the job is eligible for C/I processing in the JES3 global address space by implementing installation exit IATUX46 (Select Processors Eligible for C/I Processing). If the initialization stream defines C/I FSS address spaces, you can override JES3's selection of an address space by implementing installation exit IATUX49 (Override the Address Space Selected for C/I Processing). JES3 schedules the job to an available CI DSP based on the installation exit routines' responses.

JSS scheduling of CI

JES3's decision as to where to schedule C/I processing are also influenced by the CIDEMAND (for demand select jobs) and CIBATCH (for batch jobs) initialization parameters on the CLASS and/or STANDARDS initialization statements. These parameters indicate that jobs of a certain class or all jobs may have C/I processing performed only where the job's class is enabled or where the job is eligible to run.

Exit IATUX46

When JSS is ready to schedule a job for converter/interpreter, C/I) processing it calls the converter/interpreter (C/I) scheduling module IATIICS as a subroutine. IATIICS calls the select processors eligible for C/I processing exit IATUX46 when C/I FSS address spaces are defined. Exit IATUX46 allows to override JES3's decision and provide your own list of mains

that are eligible to do the converter/interpreter, prescan, and locate phases of a job's C/I processing.

If your installation does not define any C/I FSS address spaces, JES3 marks exit IATUX46 routine as a dummy at initialization and never calls it, because without any C/I FSSs, all C/I processing will occur on the global main.

The converter/interpreter (C/I) scheduling module IATIICS calls t exit IATUX49 routine after an address space has been selected for a job's C/I processing. IATUX49 can accept or reject the address space selected. If the CI address space selection is accepted, a RESQUEUE and a CI FCT are initialized and attached to the proper chains. The job is now under the control of a CI/POSTSCAN/CICLENUP DSP driver module IATIIDR.

(JES3 C/I subtask (IATIIST) calls installation exit routine IATUX41 (Determine the Disposition of a Job that Exceeds the Job JCL Limit) if the number of JCL statements in a job exceeds the job JCL statement limit -- MAXJOBST parameter on the STANDARDS statement. If so, exit IATUX41 can allow JES3 to continue processing. Otherwise JES3 cancels the job from the system with print.)

CLASS initialization statement

The CLASS initialization statement defines the characteristics of JES3 job classes which include:

- CIBATCH= indicates whether or not batch jobs of this class must have CI processing limited to certain processors. If specified for a class, this will override the value specified on the CIBATCH parameter on the STANDARDS statement. If not specified, JES3 will use the value specified on or defaulted to on the STANDARDS statement.
  - JOB - indicates CI processing must be performed on a system on which the job is eligible to run.

    Note: Only the JES3 //*MAIN SYSTEM= JECL statement (NOT the job's scheduling environment) is considered for CI scheduling purposes when determining where the job is eligible to run.

  - CLASS - indicates CI processing must be performed on a system on which the job's JOB CLASS is enabled.
  - ANY - indicates CI processing may be performed on any processor regardless of job or class eligibility.
- CIDEMAND= indicates whether or not demand select jobs of this class must have CI processing limited to certain processors. If specified for a class, this will override the value specified on the CIDEMAND parameter on the STANDARDS statement. If not specified, JES3 will use the value specified on or defaulted to on the STANDARDS statement.
- JOB - indicates CI processing must be performed on a system on which the job is eligible to run.
- CLASS - indicates CI processing must be performed on a system on which the job's JOB CLASS is enabled.
- ANY - indicates CI processing may be performed on any processor regardless of job or class eligibility.

STANDARDS initialization statement

The STANDARDS initialization statement to specify default values for information not provided on other JES3 initialization statements. The CIBATCH= and CIDEMAND= parameter definitions are the same as on the CLASS statement.

7.2 Interpreter service

- Translate JCL into - scheduler work area (SWA)
- Determine "setup" requirements
  - Prescan phase
    - Devices - Volumes - Data sets
- "Setup" requirements
  - Postscan Phase
    - Locates for cataloged data sets
    - Create JST and JVT control blocks (MDS)
  - Determine minimum devices required (HWS)

Interpreter service

The converter/interpreter (C/I) is the first scheduler element for every standard job. During C/I processing, the JCL is converted into scheduler work area (SWA). The SWA is stored on the JES3 spool. The JES3 CI process extracts jobs' I/O resource requirements from the SWA control blocks if pre-execution setup is to occur. C/I routines provide input to the main device scheduling (MDS) routines by determining jobs' devices, volumes, and data sets requirements.

Prescan phase

The primary function of the prescan phase is to extract each job's resource requirements from the SWA control blocks created in the interpreter phase. On entry to the prescan phase, JES3 examines the SETUP parameter on the JES3 STANDARDS initialization statement:

- If you specify SETUP=NONE (that is, you do not want JES3 to perform pre-execution setup for the complex), JES3 does not build extract resource requirements.
- If you specify any value other that NONE on the SETUP= keyword, JES3 determines the job's resource requirements.

Installation exits IATUX04 (Examine the Job Information), IATUX05 (Examine the Step Information), and IATUX06 (Examine the DD Statement Information) allow you to examine or change job, step, and DD information, respectively. You can use these exits to examine or change the information before processing begins.

If the job is being processed in a C/I FSS, JES3 determines whether the job is eligible to have catalog searches performed on the main where the C/I FSS is executing. If catalog searches

can be performed on that main, locate processing occurs in the C/I FSS and control then returns to the JES3 global address space for the remainder of postscan processing.

Postscan phase

The functions of the postscan phase are to resolve cataloged data set references and prepare the job for the main device scheduler (MDS).

Cataloged data set resolution

If SMS is active, JES3 calls it to search the system catalogs and collect scheduling information for the job. SMS searches the system catalogs to obtain information about a job's data sets when the job's JCL does not specify unit information.

If SMS fails to find a data set name, the postscan phase calls installation exit IATUX07 (Examine/Substitute Unit Type and Volume Serial Information). Through this installation exit, you can examine the available data set information and, if necessary, supply the unit and volume information.

If a data set has been migrated (or is eligible to be migrated) by the Hierarchical Storage Manager (HSM), the LOCATE request for that data set causes JES3 to associate the data set with a set of volumes to which it can be recalled. HSM limits the choice of volumes eligible for recall during LOCATE processing in accordance with its space management algorithms. JES3 processing continues as though the data set is recalled to all eligible volumes. However, the actual recall does not occur until job execution when MVS issues a LOCATE request. At that time, HSM determines which volume is the best choice to recall the data set to and then recalls the data set to that volume. HSM derives an SMS storage class and an SMS storage group for SMS-managed data sets.

In addition, whenever the response of a LOCATE request has been received, you can use installation exit routine IATUX11 (Inhibit Printing of the LOCATE Request/Response) to inhibit printing of the LOCATE request/response in the JESYSMSG data set.

JST and JVT

The JST(IATYJST mapping) control block describes a job’s unit, volume, and data set requirements. The JVT (IATYJVT mapping) contains lists of volumes and is used to pre-allocate volumes for a job. The JVT used with the JST. Each JVT entry is associated with a JST entry. The first JVT entry associated with a JST entry is copied into the JST entry.

High watermark setup (HWS)

High watermark setup, as defined on the HWSNAME initialization statement, reduces the number of devices JES3 reserves for a job. To determine how many devices of a particular type to reserve for a job, JES3 considers the needs of each of the job's steps. In this way, JES3 determines which step needs the greatest number of devices of that type; JES3 then reserves that many devices of that type for the job. JES3 repeats this process for each device type the job needs. As a result, high watermark setup can cause premounting of all mountable volumes. Volume unloading and remounting may occur for both private and public volumes, even when RETAIN® has been specified on the applicable DD statement.

7.3 Interpreter parameters

- Options for converter/interpreter parameters
  - CIPARM initialization statement

    CIPARM,PARM=(00000300051231E00011A),
    REGION=999M,AUTH=ALL,COMMAND=IGNORE
    CIPARM,PARM=(00014390051231E00011V),PARMID=02,
    REGION=9999K,AUTH=INFO,COMMAND=EXECUTE
  - Account number & programmer name & SWA location
  - Job step time limit - 003000 - 143900
  - Job step region size - 512K
  - MSGLEVEL - 1
  - MSGCLASS - A - V
  - **REGION=** max 9999K | 2047M
  - **AUTH=** ALL | INFO | CONS | SYS
  - **COMMAND=** DISPLAY | EXECUTE | IGNORE | VERIFY

Interpreter parameters

The CIPARM initialization statement specifies the options to be used by the MVS converter/interpreter (C/I). These options are used as system defaults applied to certain JCL statement parameters and other options for jobs scheduled on any main.

The default CIPARM is:

```
CIPARM,PARM=(00000300025631E00011A),PARMID=01,AUTH=ALL,COMMAND=IGNORE
```

PARMID parameter

Specifies a 2-byte identifier associated with this option list. This parameter provides the facility to have a variety of C/I option lists. The operator may select the option list to be used by specifying the identifier on the *CALL, CR, DR, or TR command.

Specify the PARMID parameter to distinguish option lists if you include more than one CIPARM statement in the initialization stream.

PARM parameter

The PARM parameter defines a 21-character parameter string that the MVS Converter/Interpreter uses when processing jobs. The parameter string format:
bpp***mmmmssccc***r***l***aaaa***efh***

b	An account number or programmer name is required
pp	Default job priority. This field is ignored by JES3
mmmmss	Maximum length of time each job step may execute

```
ccc       Job-step region default
l         Processing options for BLP parameter of the DD statements
e         MSGLEVEL default
f         Allocation MSGLEVEL default
h         MSGCLASS default
```

Two fields have a fixed value always set by JES3: positions 13 (r) and 15-18 (aaaa).

AUTH parameter

The AUTH parameter specifies which commands will be accepted through COMMAND JCL statements in the job stream. The groups includes:

- SYS -- system commands
- IO -- input/output commands
- CONS -- console commands
- INFO -- information commands (such as display)
- ALL -- all operator command types.

COMMAND parameter

The COMMAND parameter specifies the disposition of commands entered through COMMAND JCL statements in the job stream as follows:

- DISPLAY -- The command is displayed and scheduled for execution.
- EXECUTE -- The command is scheduled for execution.
- IGNORE -- The command is ignored (that is, interpreted as a “no operation”). IGNORE is the default.
- VERIFY -- Specifies that the system displays the command, asks the operator whether the command should be executed, and if the operator replies “YES”, schedules the command for execution.

REGION parameter

The REGION parameter specifies the job-step region default size.

- nnnn -- specifies the 1- to 4-digit number of units used for the default region size for a job step. This value overrides the region size in the PARM option list, and is otherwise processed the same way. If you do not specify this parameter, JES3 obtains the region size from the PARM option list.
- x -- Indicates the unit of measure as kilobytes (K) or megabytes (M)
  - For nnnnK, the maximum allowable value is 9999K
  - For nnnnM, the maximum allowable value is 2047K.

PRTY parameter on STANDARD statement

The PRTY parameter on the STANDARDS statement overrides any value specified in columns two and three (pp) of the option list.

7.4 C/I parameters and managing proclibs

- Callable Input Reader DSPs - DR
  - *X DR,M=member,PARMID=02
- INTRDR job streams
  - STANDARDS statement

```
STANDARDS,INTPROC=xx|ST,INTPMID=ID|01
                ,STCPROC=xx|ST,STCPMID=ID|01
                ,TSOPROC=xx|ST,TSOPMID=ID|01
```

- JDABPMID - User exit IATUX29
- Procedure libraries
  - IATPLBxx DD-statement in JES3 start procedure
  - IATPLBxx DYNALLOC initialization statement
  - //*MAIN PROC=ST|xx

C/I parameters -- operator called reader jobs

The PARMID= specifies a 2-byte identifier associated with an CIPARM option list. This parameter provides the facility to have a variety of C/I option lists. The operator may select the option list to be used by specifying the identifier on the DR or CR command.

To define and name an options list, code a CIPARM initialization statement. Use the PARMID= parameter on this statement to name the list. You must code a separate CIPARM statement for each options list.

Each time the operator calls a disk reader, a card reader, or a tape reader, the operator can select one of the options lists that you defined. Thereafter, each time the MVS C/I routines process a job read from that particular reader, the MVS C/I routines use the selected options list.

C/I parameters - internal reader jobs, started tasks, and TSO logons

To specify which options list the MVS C/I routines should use for internal reader jobs, started tasks, and TSO LOGON jobs, use the INTPMID, STCPMID, and TSOPMID parameters on the STANDARDS initialization statement.

The INTPMID= parameter on the STANDARDS statement specifies the 2-byte identifier (ID) of the converter/interpreter options list for jobs entered using the internal reader. The ID must match the ID specified by the PARMID parameter on a CIPARM initialization statement. If no CIPARM statements are included in the initialization stream, the default value is used.

User exit IATUX29

IATUX29 exit is entered at the completion of input service processing. Module IATISEN provides access to the job control table (JCT), job description accounting block (JDAB), and job management record (JMR) for the job. This exit could be used to move the accounting information to the SMF record. You can examine and modify the job accounting information in the JDAB, the CIPARM list ID (JDABPMID), the procedure library ID (JDABPROC), and output limits an output options for jobs that are started tasks.

PROCLIBs

During the CI processing the converter takes the job's JCL, merges it with JCL from a procedure library, and converts the composite JCL into C/I text.

The procedure library concatenations JES3 passes to the MVS converter are defined on the STANDARDS initialization statement:

- The INTPROC parameter specifies the appropriate IATPLBxx procedure library to be searched by the MVS converter when resolving procedure references for jobs submitted using the internal reader.

  Note: Jobs entering from a disk reader or NJERDR use the standard procedure library as defined on the IATPLBST.

- The STCPROC parameter specifies the appropriate IATPLBxx procedure library used for started task jobs.
- The TSOPROC parameter specifies the appropriate IATPLBxx procedure library to be used for TSO LOGON jobs.

All procedure libraries must be defined by an IATPLBxx DD statement in the JES3 start procedure or by a DYNALLOC initialization statement.

An individual job can override the procedure library ID using the PROC parameter on the //*MAIN JES3 control statement:

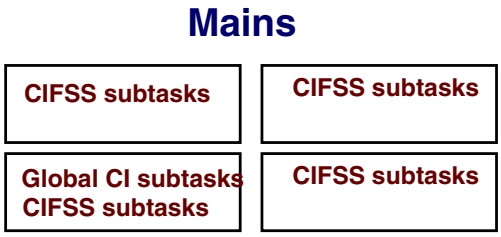
- PROC=xx names the procedure library that the system is to search for cataloged procedures called by EXEC statements in the job. If a procedure cannot be found in the named library, JES3 abnormally terminates the job.

  If this parameter is omitted, the default depends on the source of the job. If the job is submitted as a batch job, the default is ST. If the job is submitted from an internal reader, the default can be another procedure library, as specified by the installation on the STANDARDS initialization statement (the INTPROC, STCPROC, or TSOPROC parameters).

Note: Converter/Interpreter functional subsystems (C/I FSS) and the PROCLIB update function will obtain unit and volume information for the procedure libraries from the catalog. For these functions, JES3 ignores unit and volume information that you specify in the JES3 start-up procedure or on a DYNALLOC initialization statement.

7.5 CI initialization definitions

- How many subtasks ?
  - Batch jobs
  - Demand select jobs
- Where are subtasks ?
  - JES3 address space
    - STANDARDS,.....,CICNT=(maxbatchs,maxdemsel)
  - C/I FSS address spaces
    - FSSDEF,TYPE=CI,DSPCNT=(maxbatchs,maxdemsel)



CI initialization definitions

During JES3 initialization, JES3 attaches an installation-defined number of converter/interpreter (C/I) tasks. The C/I tasks are subtasks to the JES3 nucleus task. When a C/I subtask processes a job, the scheduler work area (SWA) in the address space in which the service is invoked (global or functional subsystem) provides temporary storage for the job's converted JCL statements. The SWA storage remains allocated until the C/I subtask finishes processing the job. When several C/I subtasks run concurrently, the SWAs of all the jobs are in-storage simultaneously. The initially by a C/I subtask created SWA is always located above 16-megabytes.

CI subtasks

The STANDARDS statement CICNT parameter specifies the maximum number of CI DSPs that can operate in the JES3 global address space at any time. The first subparameter (maxbatch) specifies the maximum number of CI DSPs that process batch jobs. The second subparameter (maxdemsel) specifies the maximum number of CI DSPs that process demand select jobs (that is, started tasks and TSO LOGONs). The sum of the two subparameters cannot exceed 255. CI DSPs defined to process batch jobs cannot be used to process demand select jobs, and vice versa.

CIFSS subtasks

The CI DSPs execute in the JES3 global address space. The characteristics of C/I FSS functional subsystems, which operates in their own address spaces, are defined with the FSSDEF,TYPE=CI statement. DSPCNT parameter specifies the maximum number of CI DSPs that can operate in the C/I FSS address space at any time.

CI scheduling rules

When a job is ready for converter/interpreter service (represented by the C/I scheduler element) the job segment scheduler (module IATGRJS) calls IATIICS (using the IATXSCH macro) to schedule the job for C/I. JSS does the following in selecting where to run CI:

1. First, IATIICS checks for an available CI DSP first in the JES3 global address space. If C/I FSSs were defined at initialization, IATIICS scans the C/I FSS tables to see if any available CI DSPs are in any of the C/I FSS address spaces. It then sets up a main mask to indicate which main processors have C/I FSSs with available CI DSPs. If no CI DSPs are available, IATIICS checks to see if the job is a demand select job used to start a C/I FSS address space. If so, the job is scheduled to use the C/I subtask reserved for starting C/I FSS address spaces. Otherwise, IATIICS returns to JSS (IATGRJS) indicating that no CI DSP is available. JSS will try again when a CI DSP is available.
2. IATIICS searches the main processor control tables (MPCs) to see if there are any eligible processors for locate processing. Locate processing is done only on mains that are eligible to run the job and only those mains that are connected and on-line. If no mains are available, IATIICS returns to JSS, indicating that no main processor is available. (JSS will try again when a processor is available.)
3. If the job uses a procedure library, IATIICS searches the procedure library tables to find the procedure library that the job uses. If the procedure library is being updated or is disabled IATIICS returns the job to JSS, indicating that no procedure library was available. (JSS will try again when a processor is available.)
4. If all the above conditions are met and C/I FSSs are defined, IATIICS calls user exit IATUX46. IATUX46 selects the eligible processors for C/I (using the main mask set up earlier), and indicates whether the job can be scheduled for C/I in the JES3 global address space. (IATIICS does not call IATUX46 if it is a dummy exit or if the special demand select job that starts a C/I FSS address space is being scheduled.)
5. If the user exit eliminates all possible choices for CI, IATIICS returns to JSS, indicating the user exit rejected the job. If the main mask was modified by IATUX46, IATIICS searches the C/I FSS tables to eliminate as scheduling choices, any FSSs not included in the main mask.
6. Once an address space is selected, IATIICS calls user exit IATUX49. IATUX49 accepts or rejects the address space selected for a given job. (IATIICS does not call IATUX49 if it is a dummy exit or if the job reserved for starting C/I FSS address spaces is scheduled to use the C/I subtask.)
7. If IATUX49 accepts the choice, IATIICS does processing based on the address space selected, JES3 global or C/I FSS. If the JES3 global was selected, IATIICS initializes a RESQUEUE and a CI FCT and puts them on the proper chains.

   If a C/I FSS is selected, IATIICS initializes the RESQUEUE, locks the control block FDBs in the RESQUEUE (used in the C/I FSS address space) so that they cannot be updated while they are in the FSS address space. IATIICS places the FSID into the job's JCT (to identify where the job is, should there be a JES3 restart), and sends the job to the FSS address space by issuing the IATXCIO macro.
8. If the job cannot be scheduled for converter/interpreter service, (for example, all FSSs terminated or user exit IATUX49 rejected all choices) IATIICS cleans up and returns control to the job segment scheduler.

Note: In order for CI to be scheduled on a main processor, the main processor must not have been previously ruled out due to the CIBATCH or CIDEMAND parameter.

7.6 Where does CI processing take place

- STANDARDS initialization statement parameters
  - CIBATCH= ANY | CLASS | JOB
  - CIDEMAND= ANY | CLASS | JOB
    - JOB - indicates CI processing must be performed on a system on which the job is eligible to run
    - CLASS - indicates CI processing must be performed on a system on which the job's JOB CLASS is enabled
    - ANY - indicates CI processing may be performed on any processor regardless of job or class eligibility - **Default**
- //*MAIN JECL
  - Affects CI scheduling determination
- Controlling number of JCL statements
  - MAXJOBST=0 | nnn
  - MAXASST=0 | nnn
  - MAXINDD=maxjobst | nnn

CI scheduling

CIBATCH= indicates whether batch jobs must have CI processing limited to certain processors. This will apply to all batch jobs unless overridden by the CIBATCH parameter on the CLASS statement pertaining to a job's JOB CLASS.

CIDEMAND= indicates whether demand select jobs must have CI processing limited to certain processors. This will apply to all demand select jobs unless overridden by the CIDEMAND parameter on the CLASS statement pertaining to a job's JOB CLASS.

- **JOB** - indicates CI processing must be performed on a system on which the job is eligible to run.

  Note: For batch jobs only, the JES3 //*MAIN SYSTEM= JECL statement (NOT the job's scheduling environment) is considered for CI scheduling purposes when determining where the job is eligible to run.

- **CLASS** - indicates CI processing must be performed on a system on which the job's JOB CLASS is enabled.
- **ANY** - indicates CI processing may be performed on any processor regardless of job or class eligibility. This is the default.

Additional STANDARDS statements for CI

To prevent a job with many JCL statements from dominating the SWA, use the job JCL statement limit. If a job contains more JCL statements than the job JCL statement limit allows

and you have not provided an exit routine for installation exit IATUX41, JES3 cancels the job. If you provide the exit routine, JES3 allows the exit routine to decide whether to cancel the job or to let the job continue.

To select the job JCL statement limit, determine the number of JCL statements that are in the largest job you want to run at your installation. Specify that number on the MAXJOBST= parameter of the STANDARDS initialization statement. The next time you do a hot start with refresh, warm start, or cold start using the initialization stream containing that STANDARDS statement, JES3 uses the job JCL statement limit you specified.

JCL statement number control - exit IATUX41

If the number of JCL statements in a job exceeds the job JCL statement limit, the JES3 calls installation exit routine IATUX41 to see if it should cancel the job. If so, JES3 cancels the job from the system with print. (Operator messages refer to this type of cancellation as the job being "express canceled".) If the job is not to be canceled and there are no JCL errors, JES3 links to the MVS interpreter to create the scheduler control blocks from the converter/interpreter text.

You can override the effects of this limit on a particular job by writing an exit routine for installation exit IATUX41. This installation exits lets you decide whether a job that exceeds the job JCL statement limit should continue processing. To find out how to write an exit routine for installation exit IATUX41, see z/OS JES3 Customization, SA22-7542.

The JCL control parameters are:

MAXJOBST= Specifies the maximum number of JCL statements that a batch job can include. JES3 control statements are not counted. This limit applies no matter where in the complex the job's C/I processing takes place. For a discussion about how to select the job JCL limit. This parameter is used to prevent jobs JCL from dominating the amount of SWA being created.

nnn - Specifies the total number of JCL statements that a single job may contain and still be processed by a converter/interpreter (CI) DSP. nnn may be any decimal integer from 0 to 99999999, inclusive. Specifying 0 indicates that you do not want to limit the total number of JCL statements in a single job.

MAXASST= Specifies the maximum number of JCL statements for batch jobs that may be processed concurrently by all CI DSPs in the JES3 global address space. The value must be an integer between 0 and 99999999, inclusive. A value of 0 means no JCL statement limit applies; JES3 does not check how many JCL statements are being processed.

MAXINDD= Specifies the maximum number of JCL SYSIN DD statements that a batch job can include. Only DD* and DD DATA DD statements are included in this count. Other types of DD statements are not counted.

nnn - Specifies the number of JCL SYSIN DD statements that a single job may contain and still be processed. nnn may be any decimal integer from 0 to 99999999, inclusive. Specifying 0 indicates that you do not want to limit the number of DD statements in a single job.

> **Note:** The parameter default will be whatever value was specified for the MAXJOBST parameter, or the default value for MAXJOBST if MAXJOBST was not specified.

7.7 Initialization Parameters for CI

- STANDARDS initialization statement
  - CICNT=(2,1) — Number of CI DSPs on global JES3
    - CICNT=(0,0) — No CI DSPs on global JES3 address space
  - PSTCNT=(2,1) — Number of POSTSCAN DSPs
    - The combined maximum number of POSTSCAN DSPs for jobs and STCs/logons that are processing in the C/I DSP that can process at any time

CI initialization parameters

The STANDARDS initialization statement to specify default values for C/I processing not provided on other JES3 initialization statements.

PSTCNT= (maxbatch,maxdemsel)

Specifies the maximum number of POSTSCAN DSPs that can operate in the JES3 global address space at any one time. The first subparameter (maxbatch) indicates the maximum number of POSTSCAN DSPs that can process batch jobs. The second subparameter (maxdemsel) indicates the maximum number of POSTSCAN DSPs that process demand select jobs (that is, started tasks and TSO LOGONs). A POSTSCAN DSP defined as processing batch jobs cannot be used to process demand select jobs, and vice versa.

The total of both numbers specified for this parameter must be between 1 and 32,767 (inclusive).

STANDARDS statement parameter for CI

The STANDARDS initialization statement has the following parameters:

CICNT Specifies the maximum number of CI DSPs that can operate in the JES3 global address space at any time. The first subparameter (maxbatch) specifies the maximum number of CI DSPs that process batch jobs. The second subparameter (maxdemsel) specifies the maximum number of CI DSPs that process demand select jobs (that is, started tasks and TSO LOGONs).

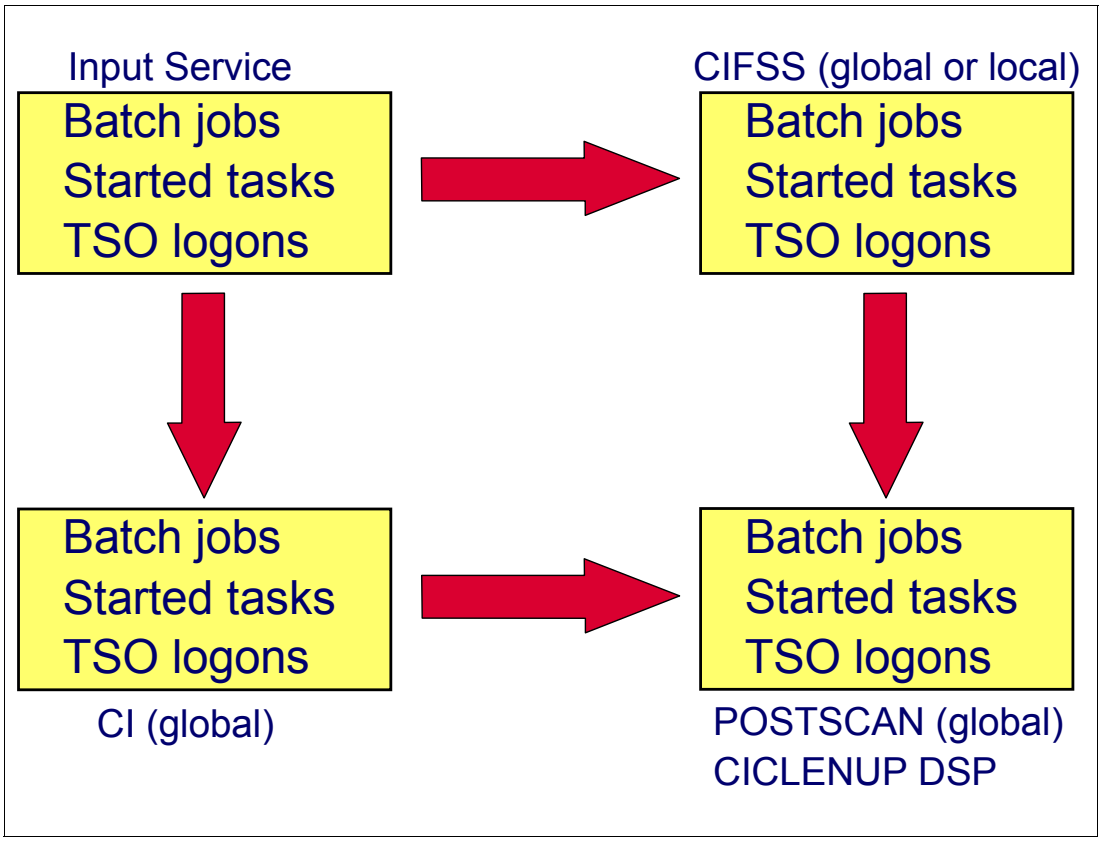
DEFAULT: (2,1)

PSTCNT This parameter specifies the number of active POSTSCAN FCTs in the JES3 global address space.

Postscan processing takes place in the global address space for all jobs that go through their JCL conversion/interpretation and PRESCAN in an FSS address space and for job that have been rescheduled for postscan.

DEFAULT: (2,1)

7.8 Jobs entering CI processing



Jobs processing in CI

The CI, POSTSCAN, and CICLENUP driver module (IATIIDR) provides for the logical flow of jobs through C/I processing, global locate processing, job summary table creation for MDS. IATIIDR handles the postscan phase by loading and calling the postscan module (IATIIPN).

Converter/interpreter seven processing modes

The “mode” of a DSP determines the type of processing performed for a job in C/I service. Mode is dependent on the type of DSP (CI, POSTSCAN, or CICLENUP) and the environment where the DSP is running (JES3 global or C/I FSS address space).

CI DSP modes

There are three modes for the CI DSP as follows:

1. JES3 C/I mode (CI DSP) -- The DSP runs in the JES3 global address space and processes a job through MVS C/I, prescan and postscan.
2. FSS C/I mode (CI DSP) -- The DSP runs in a C/I FSS address space and processes a job through MVS C/I and prescan. The job must be returned to the JES3 global address space for postscan processing.
3. C/I FSS demand select mode (CI DSP) -- The DSP runs in the JES3 global address space and processes a demand select job used to start a C/I FSS address space through MVS C/I, prescan and postscan. MVS C/I is performed using the special C/I FSS demand select subtask.

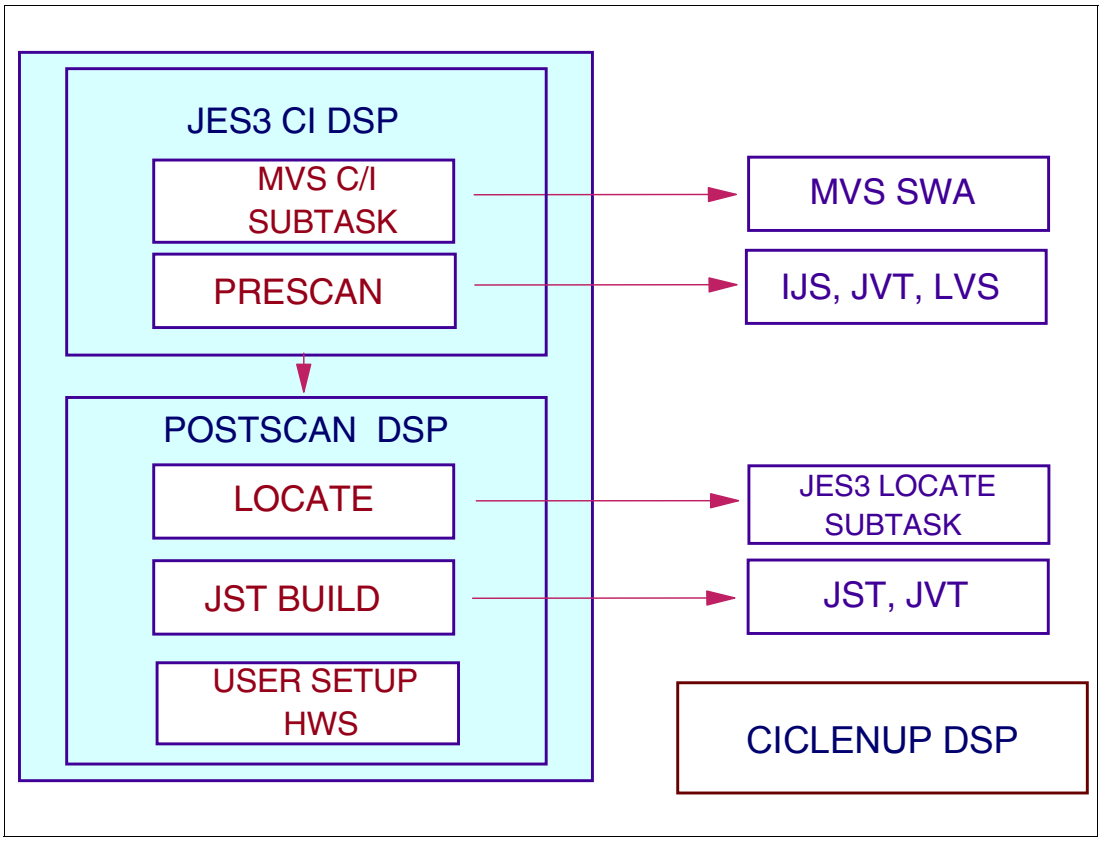
There are two modes for the POSTSCAN DSP as follows:

4. Postscan mode (POSTSCAN DSP) -- The DSP runs in the JES3 global address space and processes a job (that has successfully completed MVS C/I and prescan in a C/I FSS address space) through postscan.
5. Reschedule mode (POSTSCAN DSP) -- The DSP runs in the JES3 global address space and processes a job (that was rescheduled after MVS C/I and prescan) through postscan. A dependent job control (DJC) job is rescheduled if its predecessor jobs have not completed.
   - Locate processing is performed on a processor that is eligible to run the job. If an eligible processor is not available, the job is rescheduled.
   - The JES3 global address space may terminate before a job, (that has completed MVS C/I and prescan in a C/I FSS address space) can be scheduled for postscan. The job is scheduled for postscan after JES3 is restarted.

There are two modes for the CICLENUP DSP as follows:

6. Cleanup/cancel mode (CICLENUP DSP) -- The DSP runs in the JES3 global address space and performs cleanup processing for a job that either has failed C/I in a C/I FSS address space or was cancelled.
7. Cleanup/reschedule mode (CICLENUP DSP) -- The DSP runs in the JES3 global address space and performs cleanup processing for a job that is being rescheduled for postscan by CIDRVR. The CIDRVR DSP reschedules a job for postscan if it is a member of a DJC network whose predecessor jobs have not completed, or if no eligible processors are available for locate processing. JSS will schedule the job for postscan when the job is eligible.

7.9 Converter/Interpreter service phases



JES3 job interpretation

When JES3 has a job that can be scheduled for C/I processing, it determines whether a CI DSP in the JES3 global address space or a C/I FSS address space is available for the job. At the end of interpretation, the storage management subsystem (SMS) scans the SWA control blocks to determine the required catalogs for new data sets. Catalogs for jobs can be SMS-managed or JES3-managed. SMS uses the list of SMS-managed volumes to determine which mains are eligible for catalog searches. A list of catalogs that are not SMS-managed is returned to JES3 and becomes part of JES3's setup requirements. When the C/I subtask finishes its work, the job enters the prescan phase of C/I service.

Converter/interpreter phases

The CI, POSTSCAN, and CICLENUP DSPs perform the actual converter/interpreter processing. The converter/interpreter service comprises primarily the JES3 CI and POSTSCAN DSPs, the C/I subtasks under which the MVS C/I routines run, and the initiator support routines. You can define one or more C/I functional subsystem (FSS) address spaces to perform the converter/interpreter, prescan, and locate (catalog search) portion of the postscan phase for some or all jobs. A C/I FSS can operate on the global or any local main. All of the C/I FSSs are controlled from within the JES3 global by the CIDRVR DSP.

MVS C/I subtask

The CI DSP for processing includes setting up the MVS C/I subtask. For example, the IDD (the DSP's data CSECT) is initialized with the procedure library and C/I parameter information to be used by the C/I subtask from the job's JDAB (job data and accounting block). The C/I subtask phase can be performed in the JES3 global or a C/I FSS address spaces and

consists of calls to the MVS converter to convert the job's JCL into converter/interpreter text, and to the MVS interpreter, to convert the converter/interpreter text into SWA control blocks. Most of this processing is done under a CI subtask to isolate the JES3 nucleus task from implied WAITs and abends. All of the C/I FSSs are controlled from within the JES3 global by the CIDRVR DSP.

Prescan phase

The primary function of the prescan phase is to extract each job's resource requirements from the SWA control blocks created in the interpreter phase. On entry to the prescan phase, JES3 examines the SETUP parameter on the JES3 STANDARDS initialization statement:

- If you specify SETUP=NONE (that is, you do not want JES3 to perform pre-execution setup for the complex), JES3 does not build extract resource requirements.
- If you specify any value other that NONE on the SETUP= keyword, JES3 determines the job's resource requirements.

During the prescan phase, the JCL for the job is examined for PGM=JCLTEST or PGM=JSTTEST. If PGM=JCLTEST is found on an EXEC statement, the JCL is interpreted and the job is then express canceled on completion of the CI DSP. If PGM=JSTTEST is found on an EXEC statement, the job is processed through the prescan and postscan phases, a printed format of the job summary table (JST) is printed on the JESYSMSG data set, and the job is then canceled-with-print on completion of the CI DSP. Installation exits IATUX04, IATUX05, and IATUX06 allow you to examine or change job, step, and DD information, respectively. You can use these exits to examine or change the information before processing begins. For more information on JCLTEST and JSTTEST, see *z/OS JES3 Diagnosis*, GA22-7547.

If the job is being processed in a C/I FSS, JES3 determines whether the job is eligible to have catalog searches performed on the main where the C/I FSS is executing. If catalog searches can be performed on that main, locate processing occurs in the C/I FSS and control then returns to the JES3 global address space for the remainder of postscan processing.

Prescan phase control blocks

The prescan phase uses the SWA control blocks to build tables used in postscan processing. Prescan builds the intermediate job summary table (IJS), the locate request table (LVS), and the job volume table (JVT); these specify required resources for the job. Prescan also writes SWA control blocks to spool for later use during the job select for execution process.

IJS Intermediate job summary table (IJS) contains an entry for each DD statement that requires a JES3-managed or SMS-managed device An IJS entry is also created when volume and unit information is not specified, which means that the catalog must be searched to determine whether the device is JES3-managed.

JVT Job volume table (JVT) contains an entry for each volume (needed by the job) that requires a JES3-managed device. Each JVT entry is associated with the IJS entry for that DD statement.

LVS The locate request table (LVS) contains an entry for each data set that requires a catalog search to obtain unit and volume information. The data set is assumed to require a volume on a JES3-managed device until the catalog is searched. Each LVS entry is associated with the IJS entry for that data set.

Postscan phase

The functions of the postscan phase are to resolve cataloged data set references and prepare the job for the main device scheduler (MDS). The POSTSCAN DSP runs in the JES3 global address only. It performs postscan processing for a job that has already completed MVS C/I and prescan. For example, the CIDRVR schedules a job for postscan after it

successfully completes MVS C/I and prescan in a C/I FSS address space. JSS schedules a job for postscan when the job is a DJC job and all of its predecessor jobs have completed.

LOCATE processing

Also called cataloged data set resolution. If SMS is active, JES3 calls it to search the system catalogs and collect scheduling information for the job. SMS searches the system catalogs to obtain information about a job's data sets when the job's JCL does not specify unit information. If SMS fails to find a data set name, the postscan phase calls installation exit IATUX07 (Examine/Substitute Unit Type and Volume Serial Information). Through this installation exit, you can examine the available data set information and, if necessary, supply the unit and volume information. If a data set has been migrated (or is eligible to be migrated) by the Hierarchical Storage Manager (HSM), the LOCATE request for that data set causes JES3 to associate the data set with a set of volumes to which it can be recalled. HSM limits the choice of volumes eligible for recall during LOCATE processing in accordance with its space management algorithms. JES3 processing continues as though the data set is recalled to all eligible volumes. However, the actual recall does not occur until job execution when MVS issues a LOCATE request. At that time, HSM determines which volume is the best choice to recall the data set to and then recalls the data set to that volume. HSM derives an SMS storage class and an SMS storage group for SMS-managed data sets.

JST build

Although the JES3 main device scheduler performs volume fetching and setup, JES3 must determine the type of setup used for each job. You specify the type of job setup by coding the SETUP parameter on the JES3 STANDARDS initialization statement or the end user can override your specification using the SETUP parameter on the //*MAIN statement in a job's JCL. The IJS and JVT are used to create the job summary table (JST). The JST contains a summary of the job's data set, volume, and unit requirements. The JST is built so that all of the job's resources are allocated before the job executes. (This is known as "job setup".)

User setup

User setup override processing is invoked if the user specifies setup and fetch overrides on the //*MAIN statement. If so, the JST is modified to reflect those requirements.

HWS setup

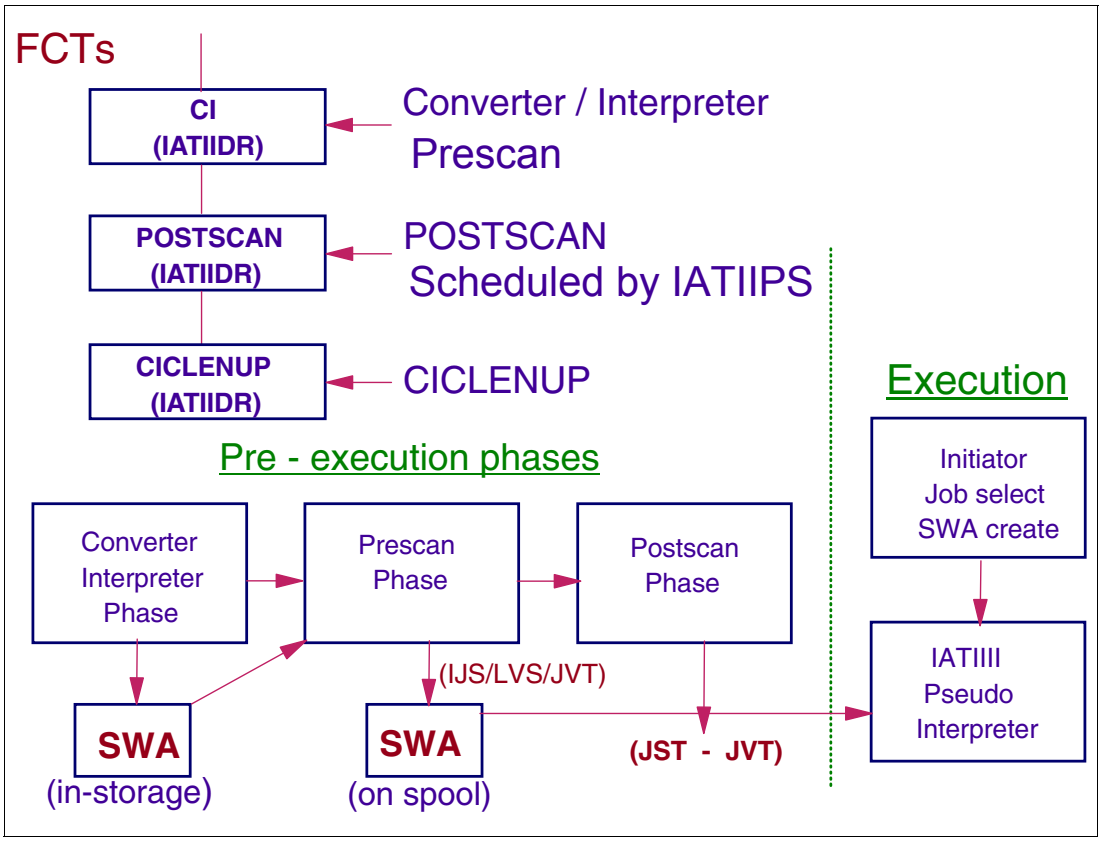
High-watermark setup processing (HWS) is used if the installation specifies the high-watermark setup algorithm. The JST is modified so that the minimum number of devices required to run the job are set up.

CICLENUP DSP

The CICLENUP DSP runs in the JES3 global address space only. It performs cleanup processing such as updating control blocks and issuing error messages. A separate DSP is used for cleanup processing because it may require I/O to read and checkpoint the job's control blocks. It is set up by the CIDRVR when one of the following occurs:

- A job fails C/I in a C/I FSS address space
- A job is cancelled while it is in a C/I FSS address space, or while it is waiting for the CIDRVR to schedule it for postscan
- A job must be rescheduled for postscan because there are no eligible processors to perform locate processing. A processor is "eligible" if it is connected, on-line and in the job's main mask.
- A job is a member of a DJC network and its predecessor jobs have not completed

7.10 CI global processing overview



CI processing overview

A CI DSP exists for one main reason: to gather information concerning the job's device, volume, and data set requirements prior to MVS execution of the job. This information is necessary for pre-execution setup, one of the major facilities of JES3. The information is taken from DD statements in the job's JCL. The CI DSP serves as an interface between JES3 and the JES3 CI subtasks that invoke the MVS converter/interpreter. The interface subtask (IATIIST) processing for a job includes:

- To fail the job if JCL errors are detected by the MVS converter/interpreter.
- To fail any job that contains more JCL statements than the limit allows.
- To delay processing of any job that would temporarily cause the C/I subtasks to process more JCL statements than the system limit allows.
- To set up security environment for converter and open procedure libraries
- To call the MVS converter and MVS interpreter
- To copy the SWA control blocks that are produced by the MVS converter/interpreter onto spool.

Converter Interpreter processing begins in module IATIIDR under the CI FCT, continues under the POSTSCAN FCT, and is completed under the CICLENUP FCT.

Pre-execution phases

These DSPs driver module processing includes several functional phases:

Converter and interpreter phases

The beginning phase includes setting up the CI DSP for processing. For example, the DSP's data CSECT (IATYIDD) is initialized with information from the job's JDAB (job data and accounting block). This phase is performed in both the JES3 global and C/I FSS address spaces.

The interpreter phase includes converting and interpreting a job's JCL. This phase consists of calls to the MVS converter to convert the job's JCL into C/I text, and to the MVS interpreter, to convert the C/I text into SWA control blocks. Most of this phase is done under a C/I subtask to isolate the JES3 nucleus task from implied WAITs and abends. If the converter detects any JCL syntax or logic errors, the job is printed and purged (and the MAIN scheduler element is bypassed). This phase is performed in both the JES3 global and C/I FSS address spaces.

Prescan phase

The prescan phase includes scanning the SWA control blocks created by the MVS interpreter and creates the following spool-resident control blocks:

- Intermediate job summary table (IJS) - The IJS contains an entry for each DD statement that requires a JES3-managed or SMS-managed device. An IJS entry is also created when volume and unit information is not specified, which means that the catalog must be searched to determine whether the device is JES3-managed.
- Job volume table (JVT) - The JVT contains an entry for each volume (needed by the job) that requires a JES3-managed device. Each JVT entry is associated with the IJS entry for that DD statement.
- Locate request table (LVS) - The LVS contains an entry for each data set that requires a catalog search to obtain unit and volume information. The data set is assumed to require a volume on a JES3-managed device until the catalog is searched. Each LVS entry is associated with the IJS entry for that data set.

  These control blocks are used later, during the postscan phase, to create other control blocks used by JES3 setup processing. At the end of the prescan phase, the SWA control blocks are written to spool. They will be relocated in the user's address space when the job executes.

  This phase is performed in the JES3 global and C/I FSS address spaces. If the job is executing in a C/I FSS address space, it is returned to the JES3 global address space for the postscan phase.

Postscan phase

This phase runs only in the JES3 global address space since it may require locate processing or catalog setup (catalog setup does not occur in the C/I FSS address space). The postscan phase performs the following functions:

- Locate processing - During locate processing the master and user catalogs are searched to obtain unit and volume information for each of the cataloged data sets referenced by the job. If the job's JCL specified JOBCAT or STEPCAT DD statements, the main device scheduler is used to set up the private catalogs before issuing locates. These catalogs are searched before any other catalogs.
- Next, the LOCATE FCT is posted to handle the locate request. The LOCATE FCT passes the request to the locate subtask. The locate subtask issues the actual MVS locate (SVC 26), and creates a response that contains the volume and unit information requested. When locate processing completes, the LOCATE DSP posts the waiting DSP. Then, the unit and volume information is used to determine if the data set will be managed by JES3. If so, additional JVT entries and, possibly, IJS entries are created.

- Job summary table creation - The IJS and JVT are used to create the job summary table (JST). The JST contains a summary of the job's data set, volume, and unit requirements. The JST is built so that all of the job's resources are allocated before the job executes. (This is known as “job setup”.)
- User setup/fetch override processing - The user can specify setup and fetch overrides on the //*MAIN statement. If so, the JST is modified to reflect those requirements.
- High watermark setup processing - If the user or the installation specifies the high-watermark setup algorithm, the JST is modified so that the minimum number of devices required to run the job are set up.
- JSTTEST processing - If the user specifies PGM=JSTTEST on an EXEC statement, a summary of the job's resource requirements is printed in the job's JESMSG data set.
- **Ending phase**

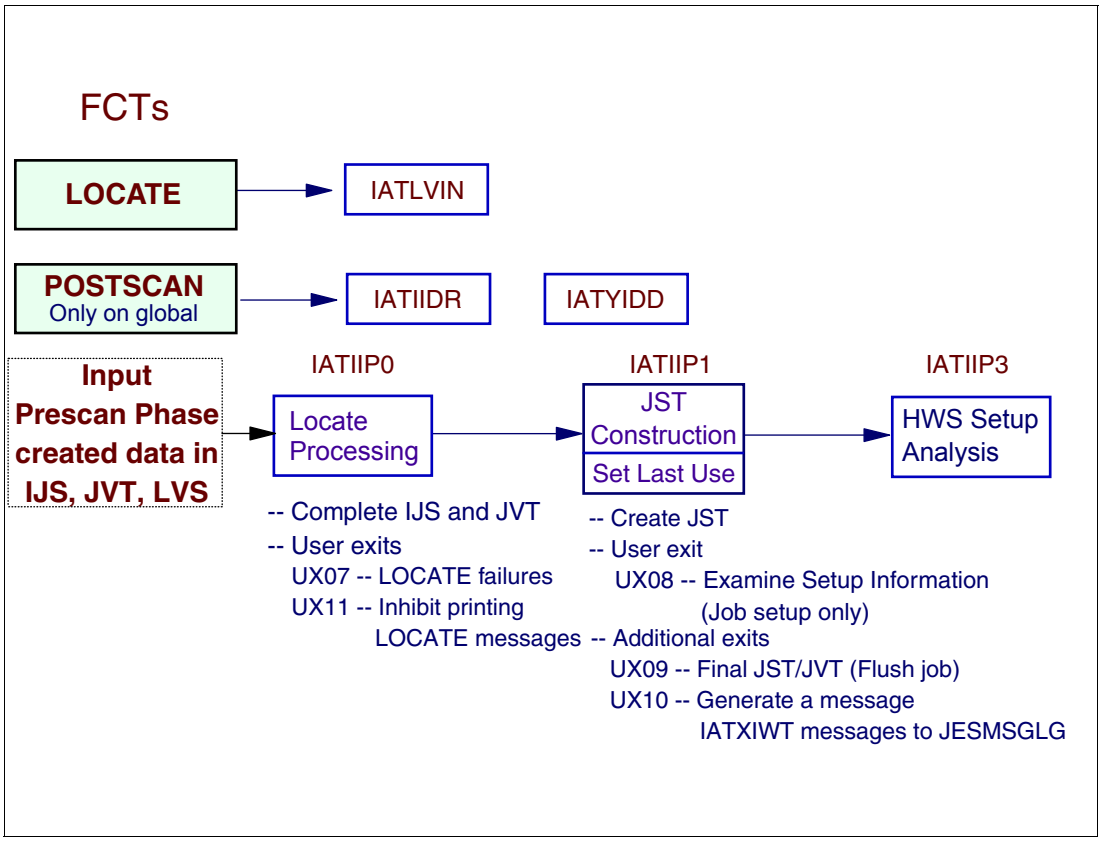
  The ending phase involves cleaning up and returning the job to the appropriate FCT for further processing. If the job is executing in a C/I FSS address space, it is returned to the JES3 global address space for postscan, or for cancel processing if the job failed C/I. If the job is in the JES3 global address space, it is returned to the job segment scheduler (JSS) so that the next scheduler element can be scheduled. This phase is performed in the JES3 global and C/I FSS address spaces.

IATIIPS - POSTSCAN scheduler module

IATIIPS is called by JSS and the CI Driver FCTs to schedule a job for POSTSCAN. The CI driver calls IATIIPS when a job (which has completed MVS CI and PRESCAN in an FSS address space) is ready for POSTSCAN. A RQ is supplied.

When IATIIPS gets control, it first checks for an available batch or demand select POSTSCAN DSP. If there is an available POSTSCAN DSP it will be used. Next, the job's main mask is used to find an IPLed and online main processor for CI LOCATE processor and the job is setup for POSTSCAN DSP. A POSTSCAN FCT is initialized and the RQ chained and the FCT is attached to the FCT chain.

7.11 POSTSCAN phase



What was done in prescan phase

The prescan phase begun by determining any debug facilities being used and setting proper switches. IATIIDR loaded module IATIIPRE which contains C/I DSP PRESCAN Processor IATIIPR, Converter/Interpreter Compatibility IATIICM and SWA Block Spooling Routine IATIISP. Next, IATIIDR called the prescan module IATIIPR to create Intermediate Job Setup (IJS) entries, locate request data (LVS) entries, and Job Volume Table (JVT) entries and then spools the SWA control blocks for main initiation. Checkpointing of the IJS, LVS and JVT control blocks was then done to provide reschedule data from DJC or specialized reschedule jobs. The reschedule path reads these control blocks in for use in the postscan phase.

POSTSCAN phase

The functions of the postscan phase are to resolve cataloged data set references and prepare the job for the main device scheduler (MDS). Cataloged Data Set Resolution is performed by the JES3 Locate Function Driver Module (IATLVIN). The C/I Service Preparation for the Main Device Scheduler is done by the CI/POSTSCAN/CICLENUP Driver Module (IATIIDR).

The job segment scheduler calls the postscan scheduler module (IATIIPS) whenever a job is rescheduled for postscan processing. In CI FSS address space, the CIDRVR postscan scheduling routine (IATIIFS) calls IATIIPS whenever a job, which has completed MVS C/I and prescan processing in an FSS address space, is ready for POSTSCAN FCT processing.

The postscan scheduler module (IATIIPS) checks for an available batch or demand select POSTSCAN DSP for the job. Next, the job's main mask is used to find an IPLed and online

main processor for locate processor, a POSTSCAN FCT is initialized, the job is added on the FCT's RQ chain, and the FCT is attached to the FCT chain.

POSTSCAN DSP

The POSTSCAN DSP runs in the JES3 global address only. It performs postscan processing for a job that has already completed MVS C/I and prescan. For example, the CIDRVR schedules a job for postscan after it successfully completes MVS C/I and prescan in a C/I FSS address space. JSS schedules a job for postscan when the job is a DJC job and all of its predecessor jobs have completed.

LOCATE FCT

Locates are done during postscan processing under the LOCATE FCT. The locate function driver module (IATLVIN) receives requests for locate services to obtain information from the catalog about one or more datasets. To request locate services, a JES3 FCT chains a locate entry table (LET) anchored off of the locate data area (LDA). The LET contains the spool address of the locate request table (LVS). The LVS specifies the datasets that require locate processing. The global LOCATE FCT is responsible for determining where locates will be performed and sending the request to the correct main processor. The locate FCT on each processor is responsible for passing the request to a locate subtask to process. The locate subtask issues the actual MVS locates (SVC 26), and creates a response that contains the volume and unit information requested. After processing each request, the locate subtask creates a locate response (LRS) containing the information from the catalog.

During the prescan phase, the Intermediate job summary table (IJS), job volume table (JVT), and locate request data (LVS) were constructed. The postscan phase is entered to resolve catalog data set references, complete construction of the job summary table (JST) from the IJS. If locate processing is required (an LVS exists), IATIIP0 issues locates to resolve all catalog data sets references.

POSTSCAN JST Build IATIIP1 builds the job summary table (JST) from the intermediate job summary table (IJS) and the job volume table (JVT). If job setup was specified, user exit IATUX08 is called whenever a job specifies or defaults to the job setup algorithm to examine the unit counts determined by IATIIP1. The last use of each volume, device, and data set is flagged in the JST. After examining the unit counts, IATUX08 either fails the job, continues with job setup, as specified, or sends the job through high-watermark setup to allocate the minimum number of devices for the job.

POSTSCAN High Watermark Setup Processing IATIIP3 is called during the postscan phase of CI, to perform high watermark setup processing. This involves determining the minimum number of units required to run the job being processed. The JST entries are updated according to the type of high watermark setup (e.g. tape hws).

In addition, whenever the response of a LOCATE request has been received, you can use installation exit routine IATUX11 (Inhibit Printing of the LOCATE Request/Response) to inhibit printing of the LOCATE request/response in the JESYSMSG data set.

7.12 Converter/interpreter exits

1. UX03 - Modify/Examine internal text
2. UX04 - Examine job information (SWA)
3. UX05 - Examine step information (SWA)
4. UX06 - Examine DD information (SWA)
5. UX07 - Substitute volume/unit type if catalog dataset not found
6. UX08 - Examine setup information
7. UX09 - Examine final job status, JST and JVT
8. UX10 - Generate a message from user exits
9. UX11 - Inhibit printing of LOCATE requests/responses
10. UX26 - Examine MVS scheduler control blocks
11. UX41 - For jobs that exceed the job JCL limit
12. UX46 - Select processors eligible for CI processing (CIFSS)
13. UX49 - Override Address Space selected for CI processing

Converter/interpreter exits

The converter interpreter exits are as follows:

UX03 This is the internal text exit called during the convertor phase of C/I. The internal text for each JCL statement (some number of card images) is passed to this exit, one statement at a time.

The installation has the opportunity of changing the internal text to meet its need. This exit is prior to UX04, UX05, and UX06 and you can do the following in this exit:

- Control dispatching priority and performance groups
- Force VIO for certain DD statements
- Change non-existent UNIT= values to something supported in the installation, thus avoiding JCL changes.

UX04 This exit is called to allow the installation access to information obtained from the SWA related to the JOB (IATYJBL).

This is the recommended exit to verify accounting information. The exit is running in the C/I FSS address space (hopefully) and any delays in processing due to I/O or local lock contention will not adversely affect JES3 global's performance.

UX05 This exit is called to allow the installation access to information obtained from the SWA related to a job step (IATYSTP).

UX06 This exit is called to allow the installation access to information obtained from the SWA related to a DD statement within a step (IATYDDL). DUMMY, SYSIN and SYSOUT DD statements are NOT passed to the exit.

UX07 This exit is called whenever a LOCATE is unsuccessful. The installation has the option of providing the necessary information in order for the job to continue processing. If the exit does not provide information, the job will be failed. For jobs that fail C/I processing, user exit UX08 will not be called. User exit UX09 will be called in all cases.

This exit may use IATXIWT to write to the JESMSGLG data set. Should it use this service, user exit UX10 will get control.

UX08 This exit is called to allow the installation to determine what level of high water mark setup should be performed. The exit is only called when job setup is specified or defaulted. This exit or UX09 is most useful in placing a job in a particular job class based upon its physical resource requirements (e.g., tape drive).

This exit may use IATXIWT to write to the JESMSGLG data set. Should it use this service, user exit UX10 will get control.

UX09 This exit is called after highwatermark setup has been performed, or if the job has been failed earlier in the C/I or MDS phase.

This exit may use IATXIWT to write to the JESMSGLG data set. Should it use this service, user exit UX10 will get control.

UX10 User exits UX04, UX05 and UX06 are not able to write to the jobs JESMSGLG directly, this must be done using IATXIWT. If this service is used, exit UX10 is invoked to turn the message number (on the IATXIWT call) into text that will be written to the JESMSGLG data set. This exit is invoked under IATNUC or IATNUCF task.

UX11 This exit is called regardless as to whether the locate was successful or not. It has the option of having the locate responses printed or not printed in the JESMSGLG data set.

This exit may use IATXIWT to write to the JESMSGLG data set. Should it use this service, user exit UX10 will get control.

UX26 This exit is called repetitively as the SWA control blocks are read from spool. Any WTO's issued from this exit will appear in the JESMSGLG data set belonging to the initiator and not the job being placed into execution. The security environment for the job has not been established.

This is the last JES3 exit that can affect where SWA will be located for the job's execution phase.

UX41 This exit is after convertor processing has occurred and is called only if the job exceeds the installation number of allowed JCL statements (not card images). The exit can allow the job to proceed or fail the job at this time.

This doesn't prevent the user from submitting a job stream that has 200,000 JCL statements and causing virtual storage failures. The output of the convertor is kept in storage that is below the 16M line. You will have to use an SMF exit to prevent this from occurring.

UX46 This exit is called to allow the installation to influence whether a job will go through C/I on the global processor or on one of the processors having a C/I FSS address space. It has a partner exit in this process, exit 49.

UX49 This exit is called after an address space has been selected by JES3 for the C/I process based upon information from the previous user exit and the job's main mask. The installation can accept or reject JES3's decision. The exit will be iteratively called as long as JES3 has alternatives. If the exit does not find a suitable address space for the C/I, the job will be failed.

7.13 C/I FSS address space

- Optional function
- CI processing offloaded
  - Converter/Interpreter
  - Prescan
    - Catalog search (locate) portion of the postscan phase when necessary catalogs can be accessed
- Postscan remains on global
- Mini-JES3 address space
  - Complete JSAM function
  - FSS nucleus subset of IATNUC

C/I FSS address space

The primary advantage of using the optional C/I FSS address spaces is to off-load much of the overhead caused by converter/interpreter processing from the global address space.

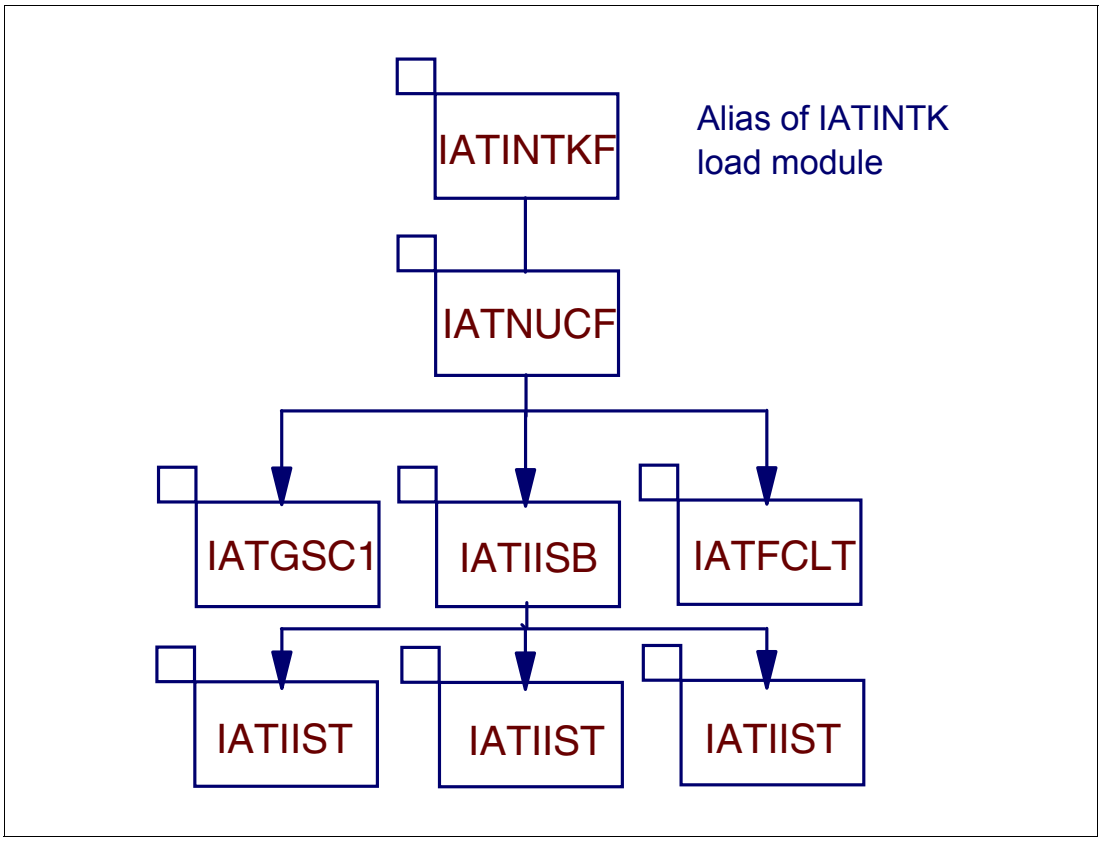
The C/I FSS address space can be on any MVS system in the complex. The scheduling of the CI scheduler element is still done by JSS in the global address space.

The postscan function for a job are done in the JES3 global address space. The CI DSPs that run in a C/I FSS address space process jobs through the converter/interpreter and prescan phases of C/I service. JES3 can also perform the catalog search (LOCATE) portion of the postscan phase in a C/I FSS if the main where the CI FSS runs has access to the necessary catalogs.

The CI FSS address space has certain JES3 global address space functions available in the CI FSS. These functions are:

- A modified JES3 nucleus, called IATNUCF
- A modified resident FCT chain
- JSAM I/O
- Trace facility via IATXTRC- Makes an entry in a JES3 trace table
- Storage management queue via IATXSQE - is used to add or delete an entry from the storage management queue
- ABEND/FAILSOFT capability

7.14 C/I FSS TCB structure



C/I FSS TCB Structure

In the FSS address space, the first task is IATINTKF. This name appears on the EXEC statement in the procedure used to start a C/I FSS address space. IATINTKF is an alias of IATINTK. An alias is required because the JES3 and FSS tasks have unlike program properties. For example, JES3 is non-cancellable, so IATINTK is defined as non-cancellable in the MVS program properties table. However, an FSS address space can be cancelled so another entry point is defined (IATINTKF).

In the JES3 global address space, IATINTK attaches the JES3 nucleus task, IATNUC. In the C/I FSS address space, IATINTKF attaches the FSS nucleus task IATNUCF. IATNUCF contains a subset of the routines contained in IATNUC.

During C/I subtask initialization, IATNUC or IATNUCF attaches module IATIISB, the master C/I subtask. IATIISB's function is to attach the C/I subtasks that interface with the MVS converter and MVS interpreter. The C/I subtask resides in module IATIIST. These subtasks are divided into three types, depending on the type of jobs they process.

In the C/I FSS address space, IATNUCF attaches a listen subtask (IATFCLT) as a result of an FSS CONNECT. IATFCLT receives order and post FSI requests from the JES3 global and, using the proper FSI interface routine, passes them to the FSS or FSA for processing.

The figure shows three CI subtasks (IATIIST).

7.15 C/I initialization statement - FSSDEF

Initialization Statement

```
FSSDEF,TYPE=CI
        ,FSSNAME=fssname
   ------------- Optional parameters -----------------
        ,PNAME=procname
        ,SYSTEM={sysname|sysnamei,..}
        ,TERM={YES|NO}
   ------------- CI parameters only -------------------
        ,START={YES|NO}
        ,DSPCNT={(nnn,nnn)|(nnn,mmm)|(,mmm)
        ,MAXASST=nnnnnnnn
```

C/IFSS Procedure

```
//JES3CI  PROC JES=JES3
//JES3CI  EXEC PGM=IATINTKF,DPRTY=(15,14)
//* ----------------------------------------------------*
//*   JES3 CI FSS PROCEDURE:  JES3CI
//*      CHKPT:   SYS1.JES3CKPT
//*               SYS1.JES3CKP2
//*      JES3OUT: SYSOUT A
//*      DUMPS:   SYSOUT A
//* ----------------------------------------------------*
//CHKPNT   DD DISP=SHR,DSN=SYS1.&JES.CKPT
//CHKPNT2  DD DISP=SHR,DSN=SYS1.&JES.CKP2
//JES3OUT  DD SYSOUT=A
//JES3SNAP DD SYSOUT=A
//SYSABEND DD SYSOUT=A
//JESABEND DD SYSOUT=A
```

START= How to start - JES3 or operator
DSPCNT= How many subtasks

BUFFER,PAGES=(global,local,cifss)

FSSDEF statement

Use the FSSDEF statement to define the characteristics of a functional subsystem (FSS) which operates in its own address space. Use a FSSDEF statement to define one or more C/I FSSs.

FSSNAME Specifies the name of a WTR FSS for which this printer is to be associated.

PNAME Specifies a member of the procedure library for started task jobs, which contains a cataloged procedure for starting the FSS. The member must be in the procedure library defined by the STCPROC parameter of the STANDARDS statement, or in procedure library IATPLBST, if the STCPROC parameter is omitted.

TERM Identifies whether or not the FSS terminates if the JES3 global terminates as the result of an *RETURN or *DUMP operator command.

SYSTEM Specifies the JES3 main on which the FSS is to operate. The name(s) must be the same as specified on the NAME parameter of the MAINPROC statement for the main.

DSPCNT Specifies the maximum number of CI DSPs that can operate in the C/I FSS address space at any time. The first subparameter (nnn) specifies the maximum number of CI DSPs that process batch jobs. The second subparameter (nnn) specifies the maximum number of CI DSPs that process demand select jobs (that is, started tasks and TSO LOGONs).

START Specifies whether or not JES3 should start the FSS automatically when the main on which the FSS is to run is connected to the global. This parameter

applies only to C/I FSSs. If specified for an output writer FSS, this parameter is ignored.

MAXASST Specifies the maximum number of JCL statements that can be processed concurrently by all CI DSPs in the C/I FSS address space. The value must be an integer between 0 and 99999999, inclusive. A value of 0 means no JCL statement limit applies; JES3 does not check how many JCL statements are being processed.

CI/FSS procedure

f C/I FSS address spaces are used, a cataloged procedure for starting the C/I FSS address spaces must be defined. Include the procedure for starting the a C/I FSS address space in an appropriate procedure library. A sample C/I FSS start procedures are in SYS1.SIATSAMP library in the members IATJ3CI and JES3CI. Always specify the same checkpoint data set(s) as specified in the JES3 start procedure and use DISP=SHR.

MAXASST Specifies the maximum number of JCL statements that may be processed concurrently for all CI DSPs in both the JES3 address space and C/I FSS address spaces.

The MAXASST keyword on the FSSDEF statement defines the number for an C/I FSS address space.

Demand select jobs are excluded from the limit function.

DEFAULT: 0

BUFFER statement

The converter/interpreter DSP requires JSAM I/O. The CI FSS supports JSAM I/O in the C/I FSS address space.

The BUFFER statement specifies of the number of JSAM buffers in the C/I FSS address space.

global Specifies number of pages for JES3 global address space.

local Specifies number of pages for JES3 local address space.

cifss Specifies the number of pages for a CI FSS address space.

Defining number of CI and POSTSCAN DSPs

The combined maximum number of CI and POSTSCAN DSPs in all address spaces equals the maximum number of jobs that C/I service can process at any time. Increasing the number of CI DSPs in any address space allows C/I service to process more jobs concurrently in that address space and in the JES3 complex. However, a CI DSP in a C/I FSS address space that has finished processing a job is considered "in use" until a POSTSCAN DSP becomes available. The number of POSTSCAN DSPs, therefore, can limit the number of CI DSPs in a C/I FSS address space that are actively processing jobs.

7.16 Starting a C/I FSS address space

- How does the FSS start?
  - Definition in INISH deck
    - START=YES - JES3 starts
    - START=NO - Operator starts
- Operator START Command
  - *F F,FSS=cifss,ST = Y
  - An internal MVS START command is issued by JES3

    S JES3CI.cifss,,,(JES3,10,IATINFC)

```
*I F FSS=cifss
IAT8701 FSSNAME  TYP  SYSTEM   PROCNAME JOBID      STAT  T S MD     RC
IAT8701                                                  DSP/DEV  MAXASST
IAT8702 cifss         C/I    SC65      JES3CI      JOB25072 ACTV      N Y
IAT8702                                                   003,001   00000000
```

Starting a CIFSS address space

When an FSSDEF statement for CI is processed during JES3 initialization, module IATINI1 loads all modules necessary to provide this function. A CIDRVR FCT is chained into the FCT chain to provide communication with a C/I FSS address space. (The CIDRVR DSP controls all of the C/I FSSs from within the JES3 global address space.)

Following JSS first pass processing, the CIDRVR FCT in module IATIICD will get control to start the C/I FSS address spaces.

This is done in module IATIIFS by issuing the IATXFSS TYPE=START,PARM=(R0),... macro request. The IATXFSS macro passes control to JES3 global FSS/FSA services (IATGRFS) at the entry point corresponding to the TYPE= function specified.

The PARM=(R0) on the macro allows specification of a parameter string. Module IATGRFS gets control when this macro is issued and builds and issues the start command.

START command for FSS

The **START** command has the following format:

```
S PROCNAME.FSSNAME,,,(SSID,FSSID,PARM)
Where:
   SSID is the name of the subsystem (JES3 for example)
   FSSID is functional subsystem ID in zoned decimal format
   PARM is the parameter data to be passed to the FSS
```

IATGRFS sends (JSERV) the start command to the system specified on the FSSDEF statement.The **START** command will be issued internally (MGCRE macro) with a program token.

> **Restriction:** The cataloged procedure for starting an FSS should not use any JCL symbols. The JES3 generated internal start command does not support any overriding keyword parameters.

Command to modify DSPC count

A C/I FSS is ended by JES3 in response to one of the following:

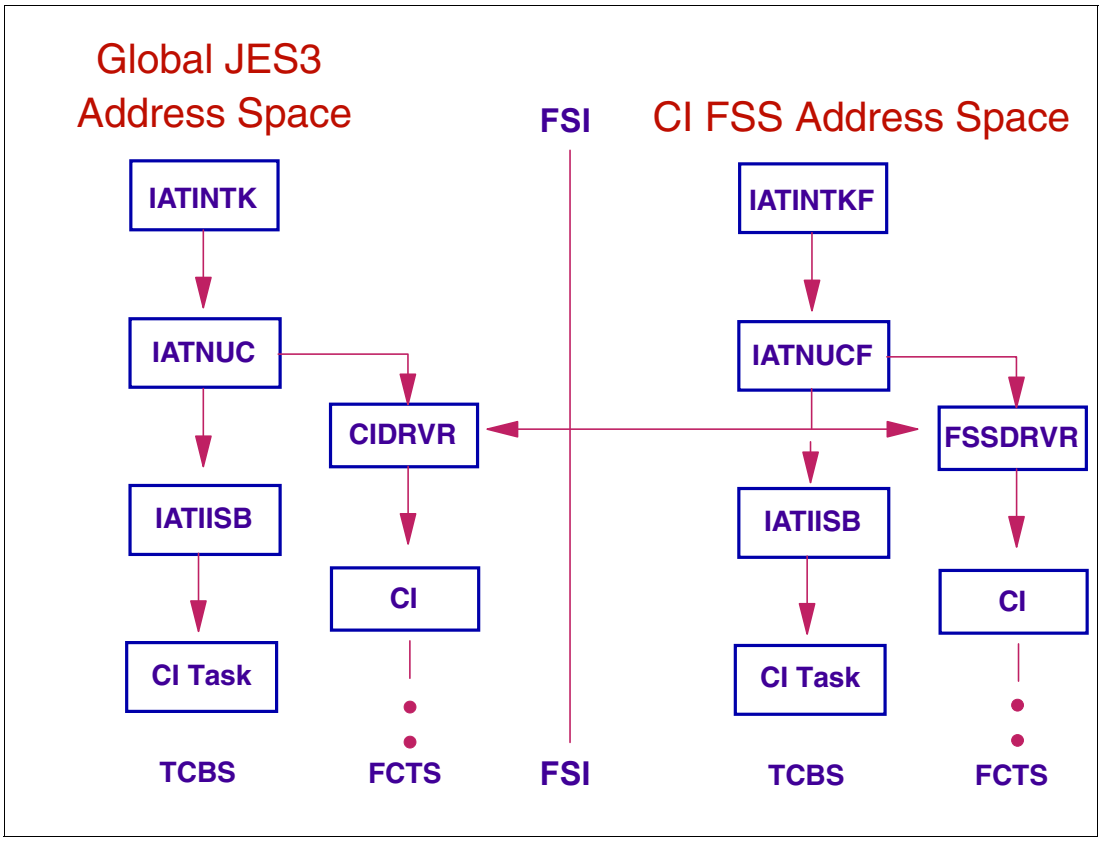
- A **`*F F,FSS=fssname,DSPC=(mmm,nnn)`** command is entered for a C/I FSS, changing both DSP count values to zero.
- An MVS CANCEL command is entered that specifies a C/I FSS name.

Command to display FSS information

The **`*I,F`** command to displays:

- The attributes and current status of the FSS
- Information for all FSSs of a particular type (WTR, C/I), whether active or inactive
- Maximum counts and status information for C/I service DSPs running in a C/I FSS.

7.17 C/I FSS communication



C/I FSS address space communication

The C/I FSS address spaces are controlled from the JES3 address space by a single CIDRVR FCT entry.

- The CI driver DSP module is IATIICD. IATIICD initializes the CI driver FCT. It gives control to the proper routine when the CI driver is posted. IATIICD also contains the CI driver's console appendage, JESTAE exit, and JESTAE retry routine.

  IATIICD controls C/I processing in either a C/I FSS or JES3 global address space. This includes the following:

  - Starting the C/I FSSs and controlling the various phases of a C/I FSS's initialization
  - Processing operator commands related to the C/I FSSs
  - Processing CANCEL and FAIL commands for jobs in the C/I FSS address spaces
  - Cleaning up after a C/I FSS terminates
  - Scheduling jobs for postscan (see “The POSTSCAN DSP”) after they have successfully completed MVS C/I and prescan in a C/I FSS address space
  - Scheduling jobs for cleanup processing if the job fails MVS C/I in an FSS address space, or is cancelled
  - Causing the C/I FSSs to enable and disable procedure libraries when a job updates one or more procedure library data sets.
- All operator commands related to CI processing are handled by the JES3 address space.
- All C/I FSS address spaces must be defined via the FSSDEF initialization statement.

In the C/I FSS address space, module IATINTK has an entry point IATINTKF. IATINTK attaches the C/I FSS address space nucleus module IATNUCF.

FSSDRVR DSP

The FSSDRVR DSP (module IATIIFC) controls a single FSS address space from within that address space. This includes the following:

- Controlling the various phases of a C/I FSS's initialization
- Setting up a CI FCT for each job that is sent to the C/I FSS address space by JSS
- Returning jobs that have completed C/I processing to the CIDRVR in the JES3 global address space
- Enabling and disabling procedure libraries, if requested by the CIDRVR
- Cancelling and failing jobs, if requested by the CIDRVR
- Processing operator `*F` commands, if requested by the CIDRVR.

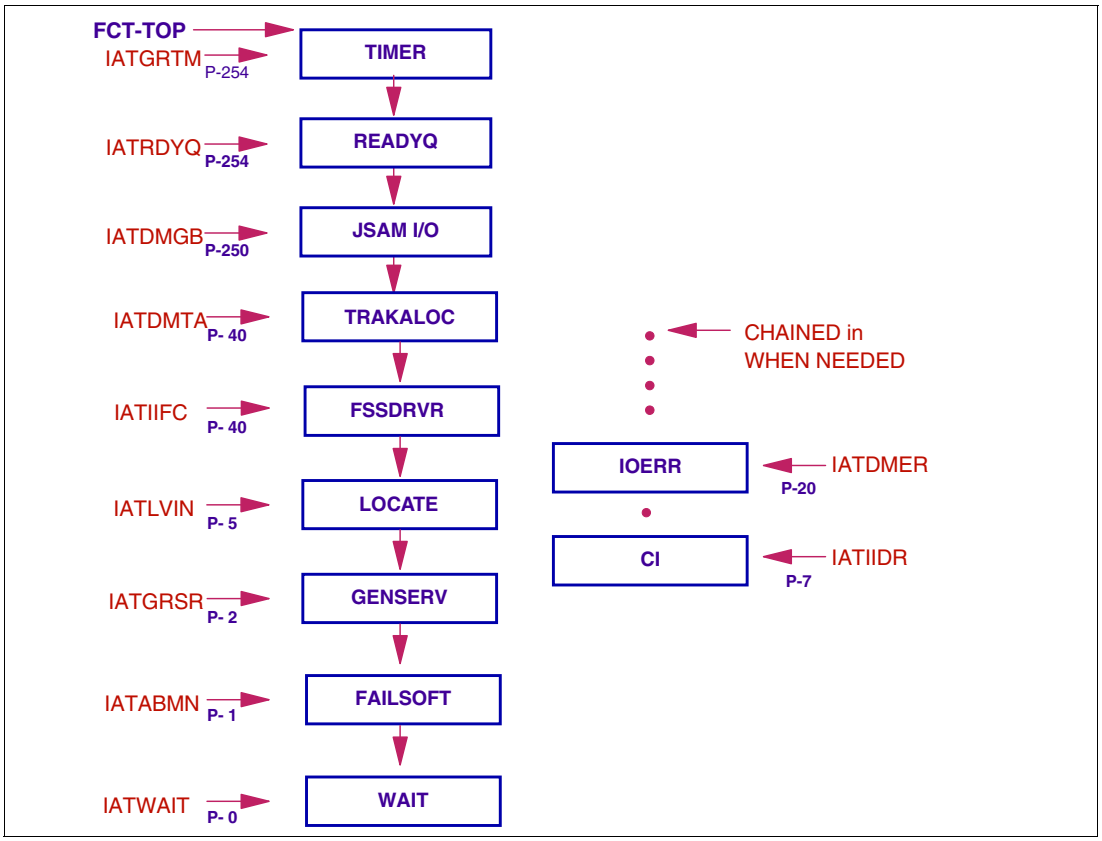
A C/I FCT executing in the FSS address space has also the IATIIDR as the driver module.

Functional subsystem interface

JES and the FSS/FSA communicate through the functional subsystem interface (FSI). The FSI is a one-level interface which provides two way communication. The FSI consists of a set of macro-invoked service routines provided by both JES and the FSS/FSA. These service routines are:

- JES routines that reside in the FSS address space
- SSI routines that JES provides
- FSS/FSA-supplied routines.
  - Functional subsystem (FSS) is an address space uniquely identified as performing a specific function related to the JES.
  - Functional subsystem application (FSA) is the functional application program managed by the functional subsystem.

7.18 C/I FSS FCTs



CIFSS FCTs

This figure shows the resident FCT chain in the CI FSS address space.

When a job is scheduled to this address space for converter processing, a CI FCT is added to the FCT chain. These are the only FCTs necessary for processing the converter interpreter.

FSSDRVR FCT

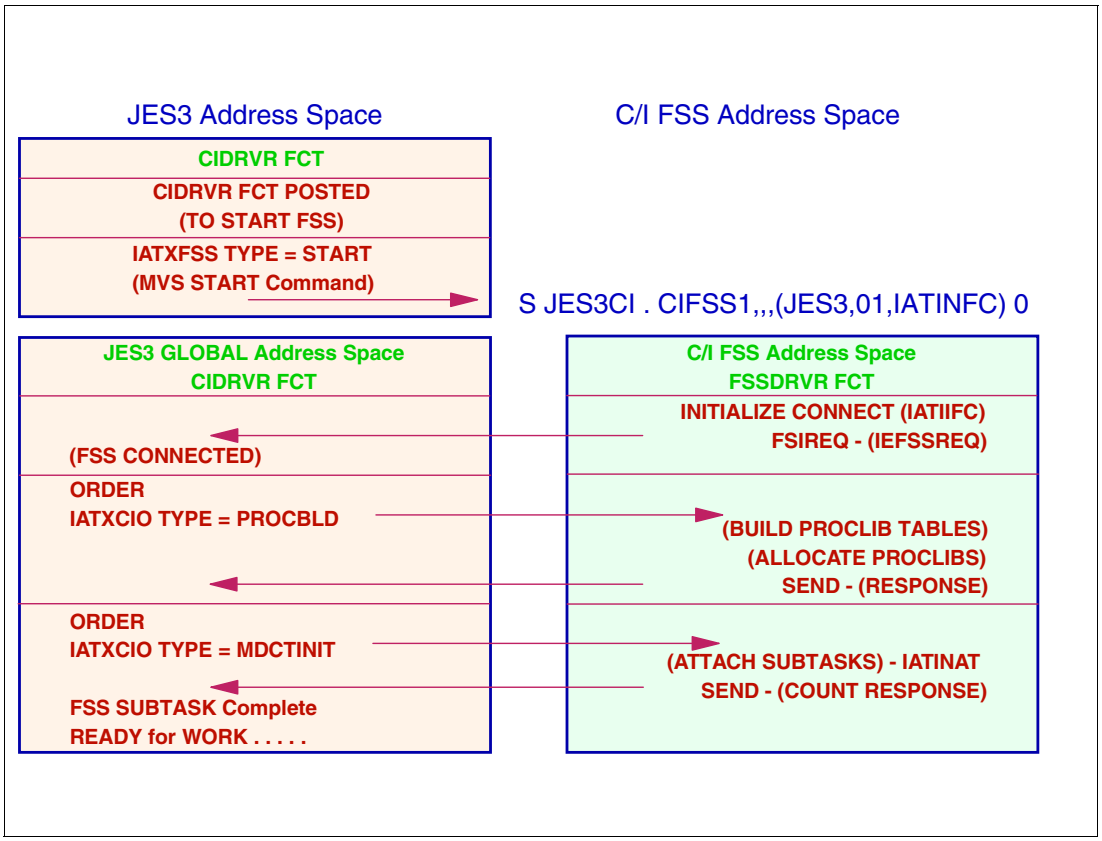
The FSSDRVR, module IATIIFC, controls a single FSS address space from within that address space. It controls the following:

- Initializing C/I FSS's (various phases)
- Setting up the C/I FCT
- Returning jobs to CIDRVR in the JES3 global address space
- Enabling and disabling the procedure libraries
- Processing modify commands

FSSDRVR processing

When an ORDER is sent by the JES3 global, FSSDRVR is posted.

7.19 C/I FSS FSI CONNECT to global



CIFSS FSI CONNECT to global

The functional subsystem interface (FSI) allows communication between JES and your functional subsystem (FSS) and functional subsystem application (FSA). A functional subsystem (FSS) is a collection of programs residing in an address space separate from JES that communicates with JES to provide a JES-related function. An FSS extends the scope of JES processing.

Note: C/I FSS address spaces do not use FSAs.

JES and the FSS/FSA communicate through the functional subsystem interface (FSI). The FSI is a one-level interface which provides two way communication. The FSI consists of a set of macro-invoked service routines provided by both JES and the FSS/FSA. These service routines are:

- JES routines that reside in the FSS address space
- SSI routines that JES provides
- FSS/FSA-supplied routines.

The FSS/FSA and JES use the FSIREQ macro to invoke functional subsystem interface (FSI) services. The FSIREQ macro allows JES to issue orders to the FSS/FSA and the FSS/FSA to issue requests to JES.

FSI communication services

The functions of the individual FSI communication services are:

- FSI CONNECT -- The FSS and FSA invoke the FSI CONNECT service to establish the functional subsystem interface to JES. FSI CONNECT processing tells JES that the FSS/FSA is started. It also identifies to the FSI the addresses of FSS/FSA routines that are to receive control when JES issues the FSIREQ macro and the addresses of JES routines that are to receive control when the FSS/FSA issues the FSIREQ macro.
- FSI DISCONNECT -- The FSS and FSA invoke the FSI DISCONNECT service to terminate connection with JES.
- FSI ORDER -- JES invokes the FSI ORDER service to issue orders to the FSS/FSA. When an operator issues a JES command that requires the participation of an FSS/FSA, JES converts that command into an order. An order represents a unit of work known to both JES and the FSS/FSA. The FSS/FSA performs the actions associated with the order and then responds to JES with the required information.

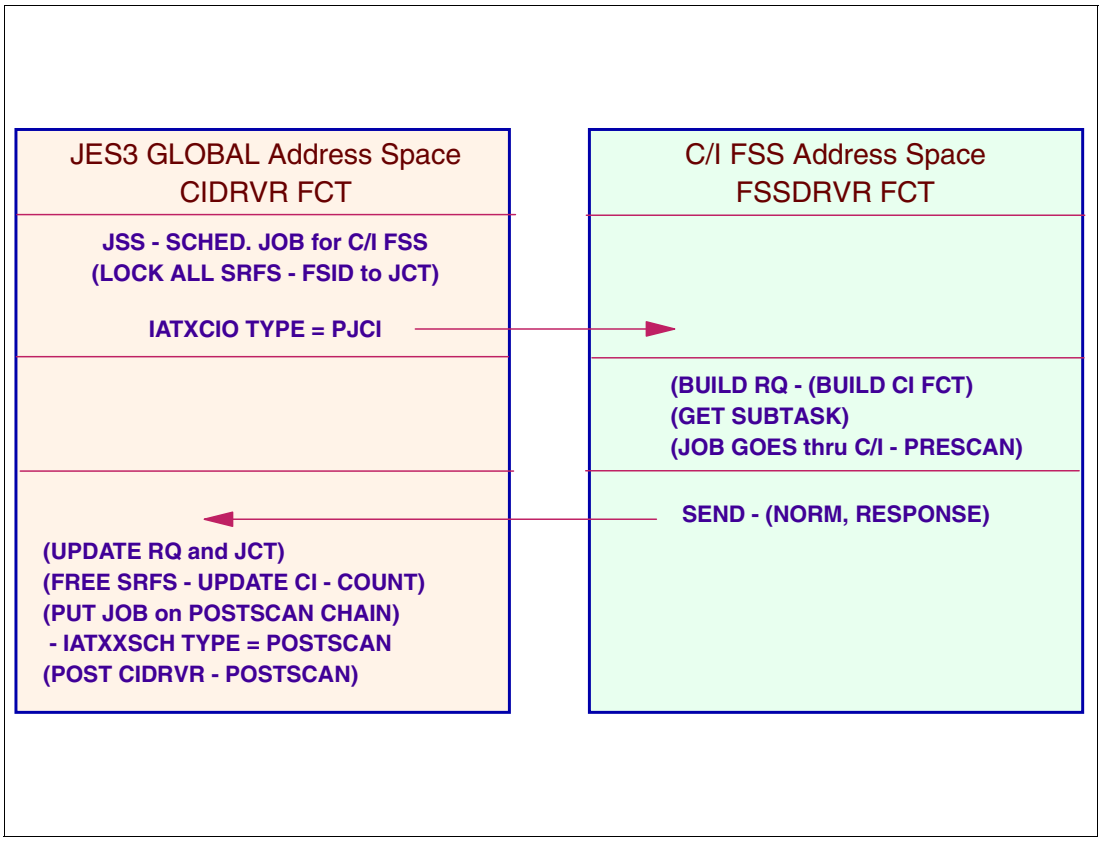
JES3 starts the FSS address space

Whenever an FSS is eligible to be started the CIDRVR “driver IATIICD” is posted. The driver invokes a service routine to start the FSS. The service routine causes a START command to be issued on the main processor where the FSS is to run. However, before the C/I FSS can process jobs, start up and initialization must be performed. Module IATINFC receives control after the JES3-related initialization is complete. C/I FSS Initialization module IATINFC handles C/I FSS specific initialization including creating tables and loading modules used during C/I processing. Then, control is passed to the FSSDRVR FCT (module IATIIFC) to issue the FSS CONNECT that informs the CIDRVR that the FSS has completed the first phases of initialization.

CIDRVR FCT on global

After the CIDRVR receives the FSS CONNECT, it sends a PROCLIB (TYPE=BUILD) ORDER to begin the C/I FSS's procedure library initialization. The FSSDRVR creates the procedure library and dynamically allocates the procedure libraries. Once procedure library initialization completes successfully, the FSSDRVR sends a PROCLIB (TYPE=BUILD) response to the CIDRVR. When the CIDRVR receives the PROCLIB response, it checks if the CI DSP counts are still non-zero and, if so, sends a MODIFY COUNT (TYPE=INIT) ORDER to start the FSS's C/I subtask initialization. If successful, the FSSDRVR sends a MODIFY COUNT (TYPE=INIT) response to the CIDRVR and the C/I FSS is considered ready for work.

7.20 Schedule a Job for C/I FSS



Schedule a job for C/I FSS

The job segment scheduler (JSS) on the global is responsible for scheduling jobs for C/I.

A JES3 macro, (IATXCIO TYPE=), is used to send orders to the C/I FSS address space. All calls through IATXCIO enter module IATIIOR where the JSERV macro is used to send order staging area to the C/I FSS address space. An IATXCIO TYPE=PJCI request specifies that the C/I FSS address space should process the job.

In the C/I FSS address space, the CIDRVR builds a RQ and CI FCT and obtains a CI subtask. The selected job then goes through converter interpreter processing.

POSTSCAN on global

The job is then returned to the global where it is updated and the job is now placed onto the POSTSCAN queue to continue CI processing.

7.21 Operator control of CI subtasks

- Display CI subtask counts
  - *I X,D = CI

    IAT8475 (IQDX) - CI MXCT - (00001,00002) USE - (00000,00000) MOD - YES
    IAT8475 (IQDX) - CI IS NOT IN HOLD
    IAT8487 (IQDX) - ADDR SPACE JCL LIMIT 3000 JOB JCL LIMIT 50000

  - *I F TYPE=CI|FSS=cifss

    IAT8701 FSSNAME TYP SYSTEM PROCNAME JOBID STAT T S MD RC
    IAT8701 DSP/DEV MAXASST
    IAT8702 cifss C/I SC64 JES3CI NONE INAC N Y
    IAT8702 003,001 00000000
- Modify number of CI's for FSS
  - *F F,FSS = CIFSS1,DSPC = (3,5)
- Modify number of CI's - Global
  - *F X,D = CI,MC = (3,5)

Operator control of CI subtasks

The **`*I X,D=CI`** command displays the current number of CI subtasks. The messages text shows:

- Whether they can be modified, MOD=YES
- Whether CI is in hold
- The MAXASST JCL limit

During JES3 initialization, the system programmer specifies the number of copies of the C/I DSPs and POSTSCAN DSPs to be used by C/I service. The JES3 STANDARDS initialization statement and the FSSDEF initialization statement define the number of C/I DSPs and POSTSCAN DSPs that are to be used by batch jobs and started task jobs and TSO/E LOGON jobs. The number of copies of the C/I DSPs and POSTSCAN DSPs can be modified with the **`*F X`** command.

These counts can be modified for CI FSS address spaces and for the number of subtasks on the global.

7.22 Display CI processing status of a job

- Status for a job active for CI scheduler element
  - *I J = 10

IAT8674 ..... TEST P-00 CL-A **CI(CF)** FSSNAME-CIFSS1

- Status Displayed - **CI(status)**
  - CS - Job waiting catalog mount
  - CF - Job active in CI FSS
  - A - Job active in JES3 global CI
  - PS - Job active in postscan DSP
  - LC - Job active in postscan - locates
  - WP - Job waiting postscan DSP
  - * - Job terminated by failsoft in CI
  - R - Job rescheduled for postscan DSP

Display CI processing status

The **`*I J=`** command displays the status of a job, which scheduler element is the current one. When the current scheduler element is CI, the status is shown as follows:

```
CI(status)
```

CI(Status)

The C/I scheduler element is ready to be, or is being, processed for the job. If the element is not being processed, only 'CI' appears in the message. Otherwise, status indicates where the job is processing in C/I processing. Inquiries on the POSTSCAN DSP also display 'CI' since the POSTSCAN DSP runs under the C/I scheduler element.

CI(Status) values

The CI status displayed may have following values:

CS The job is in catalog setup (JOBCAT or STEPCAT DDs)

CF The job is active in C/I and PRESCAN processing in a functional subsystem (FSS) address space. In this case, the FSS name of the CI FSS that is processing the job is also displayed.

A The job is active in C/I and PRESCAN processing in the JES3 address space.

PS The job is active in POSTSCAN processing in the JES3 address space.

LC LOCATE processing is being performed for the job.

WP The job is waiting to be scheduled for POSTSCAN processing. The name of the FSS that the job is associated with is also displayed.

***** The C/I scheduler element was ended by JES3 FAILSOFT processing.

R The CI DSP had specified rescheduling because a main was not available to do LOCATE processing, or the job is in DJC hold status.

Other message text and response

No specific action is required if this message is issued in response to a command. If it is issued during restart processing, cancel the indicated job when restart processing is complete. The only commands that JES3 can accept for the job are **`*I`** and **`*F jobno,C`**.

FSS=fssname fssname identifies functional subsystem (FSS) name.

MPNAME=mpname mpname identifies the main processor name.

7.23 Change CI status

- Disable DSP activity
  - *F X,D = CI,HOLD
  - *F X,D = DISABLE, HOLD
    - Function specified in IATGRPT
    - dspname IATYDSD HABLE = YES,.....
    - DISABLE and CI - HABLE = YES
- Stop C/I FSS address space
  - *F F,FFS=cifss,DSPC=(0,0)
  - CANCEL fssname - MVS command
  - *RETURN,FSS=(cifss,..)|ALL|NONE - on global

Change CI status

The CI and DISABLE scheduler elements status for scheduling can be changed with the **`*F X`** operator command. HOLD or RELEASE parameters specify that C/I or DISABLE DSP activity should be stopped (HOLD) or resumed (RELEASE).

The **`*F X=`** command can also change the MAXASST parameter and the MAXJOBST parameter for the C/I DSP.

- MAXASST=xxxxxxxx -- Specifies the maximum number of JCL statements that can be processed simultaneously in the JES3 global address space.
- MAXJOBST=yyyyyyyy -- Specifies the maximum number of JCL statements to be allowed in a single job.

DSP dictionary HABLE parameter

The **`*F X,HOLD`** or **`*F X,RELEASE`** command is enabled for DSPs by specifying the HABLE parameter in the DSP dictionary entry for the DSPs. HABLE is specified as follows:

HABLE=YES Allows the DSP's work to be held and released by the **`*F,X,D=dsp,HOLD`** and **`*F,X,D=dsp,RELEASE`** commands. For DSPs that have an FCT, the command takes affect immediately upon being issued. For DSPs having no FCT, the command prevents the job segment scheduler (JSS) from scheduling the DSP.

HABLE=NO Disallows the DSP's work to be held and released by the **`*F,X,D=dsp,HOLD`** and **`*F,X,D=dsp,RELEASE`** commands.

The default settings for DISABLE and CI is HABLE=YES. The default for the POSTSCAN DSP is HABLE=NO.

Stopping an C/I FSS address space

Several attributes assigned to a functional subsystem (FSS) can be modified, allowing an installation to balance its workload and prevent bottlenecks. For example, if your current workload is heavy, you can have JES3 start a C/I FSS, which has a non-zero DSP counts, by modifying the start value (ST) for that FSS from NO to YES.

JES3 stops an active C/I FSS address space when the maximum number of C/I DSPs is set zero (0).

```
*F F,FSS=cifss,DSPC=(0,0)
```

Note: The start value (ST) for that FSS is set to NO when the C/I FSS address space is terminated.

The MVS **CANCEL** command ends an active job, started task, or time-sharing user immediately.

```
CANCEL [jobname.]cifss [,A=asid] - End a started task
```

The identifier *cifss* is the same that was specified on the FSS START command (the *ffsname* on the FFSDEF initialization statement).

Stopping the global processor -*RETURN

Issue the *RETURN command to end JES3.

```
*RETURN,FSS=(cifss,..)|ALL|NONE - on global
```

The FSS=fssname or (fssname,fssname...) or ALL or NONE parameter specifies that a specific functional subsystem (fssname), several functional subsystems (fssname,fssname...), all functional subsystems (ALL), or no functional subsystems (NONE) are to be ended also. (This command is valid only when entered from the global.)

If FSS= is not specified, the status of the functional subsystems depends upon the value defined in the TERM= parameter of the FSSDEF initialization statement.

7.24 INQUIRY/MODIFY POSTSCAN

- POSTSCAN FCTs
  - *I X,D = POSTSCAN
  - *F X,D = POSTSCAN,MC = (2,4)
- How many POSTSCAN FCTs
  - Default is (2,1)
  - Standards,.....,POSTSCAN = (2,1),...
- Display active POSTSCAN jobs
  - *I A,D=POSTSCAN
- Display jobs waiting for POSTSCAN
  - *I Q,D=POSTSCAN

INQUIRY/MODIFY POSTSCAN

The number of POSTSCAN FCTs can be controlled via operator command. You can display the current number, **`*I X,D=POSTSCAN`**.

You can modify the current counts, **`*F X,D=POSTSCAN,MC=(x,x)`**.

POSTSCAN FCT counts are specified on the STANDARDS statement.

You can display the current jobs that are active under the POSTSCAN FCTs as follows:

```
*I A,D=POSTSCAN
```

You can also display the jobs waiting for the POSTSCAN FCT;

```
*I Q,D=POSTSCAN
```

7.25 JES3 proclib update facility

- Hold jobs using effected proclibs until library updates are complete
- Specified on JECL
  - //*MAIN UPDATE =
    - Specify proclib data set names
- Input service
  - Builds JCT scheduler elements
  - CI-DISABLE-MAIN-ENABLE-OUTSERV-PURGE
  - Records proclib data set names into SRF
    - PDB - FDB in JCTUPDTE field

JES3 proclib update facility

The JES3 proclib update facility allows users to update one or more procedure library data sets using the UPDATE keyword on the //*MAIN statement. Input Service creates a JCT for the proclib update job with two additional scheduler elements: DISABLE and ENABLE.

Procedure libraries

The installation can define one or more procedure libraries to JES3 by specifying the data sets and DD-names in the JES3 start procedure, or on the DYNALLOC JES3 initialization statement. The procedure libraries are allocated during JES3 initialization and also by the C/I FSSs, since the MVS converter also runs in a C/I FSS address space.

UPDATE parameter

An UPDATE parameter on the //*MAIN statement causes all jobs using the identified procedure library data sets and any concatenated data sets to be held until the procedure library update is complete.

Input service validates the data set names specified on the UPDATE keyword on the //*MAIN statement and creates a procedure library data set block (PDB) for them. Input Service also creates a JCT for the job with two additional scheduler elements: DISABLE and ENABLE. These scheduler elements appear before and after the MAIN scheduler element, respectively.

To prevent new jobs from updating the procedure library, change the DISABLE DSP maximum use count to 0 or issue the **`*F,X,D=DISABLE,HOLD`** command. To resume updating, increase the DSP maximum use count or issue the **`*F,X,D=DISABLE,RELEASE`** command.

Proclib update considerations

Note: If you use the proclib update facility to move a procedure library to another volume, and the procedure libraries are allocated through the JCL statements in the JES3 cataloged start procedure, The JES3 local address spaces must be restarted in order to:

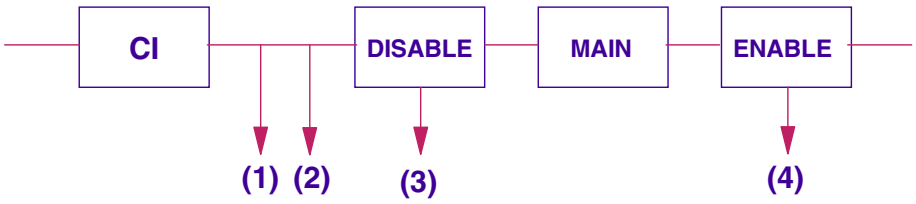
1. Reallocate the procedure library on the new volume. This is necessary if a JES3 local processor is a DSI candidate. Before a DSI is performed, all locals, which are DSI candidates must be restarted in order to pick up the change.
2. Vary offline the volume containing the old procedure library. During proclib update processing, the JES3 global and C/I FSS address spaces unallocate the procedure libraries being updated. However, the JES3 local address spaces do not unallocate the proclibs during proclib processing. This causes the VARY command for the proclib to fail when you attempt to vary the proclib offline to one of the local processors.

The JES3 local address spaces do not have to be restarted if the procedure libraries are defined via the DYNALLOC statements in the initialization stream.

Attention: If a job that updates a procedure library is in a JES-managed job class group, and it is updating the procedure library used to start initiators, make sure that there is at least one initiator started before allowing the job to enter the system. Otherwise, a deadlock will occur; the procedure library used to start the initiator is disabled, the job is waiting for an initiator, and the initiator is waiting for the procedure library to be enabled. If this situation arises, the updating job must be cancelled and resubmitted or a **`*RESTART,SETUP,jobno,CI`** command can be used to enable the procedure libraries (and let the initiator start) and restart the job through C/I processing.

7.26 UPDATE job scheduler elements

- Jobs scheduler elements



- (1) - JSS Scheduling - IATIIPC
  - "HOLD" all jobs using proclibs with data set name
- (2) - NO CI Jobs Using PROCLIB
  - DISABLE DSP to execute for update job

UPDATE job scheduler elements

After the job completes C/I, JSS determines if the DISABLE scheduler element can be scheduled. JSS reads the PDBs and searches the procedure library tables for any other jobs updating the same data sets as this job. If there are none, JSS marks each procedure library needed as “held for update” and sets up a DISABLE DSP to handle the disabling of the procedure libraries. Since the procedure library is held for updating, no other job that use the procedure library are scheduled for C/I.

When a procedure library update job is ready for DISABLE processing, the job segment scheduler, module IATGRJS, calls Proclib Disable Processing and Scheduling Module IATIIPC to determine if the procedure libraries can be updated. If so, IATGRJS schedules the DISABLE scheduler element to disable the procedure libraries) in the JES3 global address space that an update job will be using.

7.27 Update job processing

- (3) - IATIIDS - IATIIUN
  - UNALLOC ALL PROCLIBs for data set name
  - All FSSes receive disable order
  - Wait for all disables to occur
- (4) - IATIIEN
  - Allocate, concatenate (dynal),
  - (OPEN,BLDL,CLOSE)
  - All FSSes receive enable order
  - Release "HOLD"

Update job processing

For each procedure library to be disabled, the DISABLE DSP (IATIIDS) unallocates the procedure library in the JES3 global address space. If there are C/I FSS defined, the DISABLE DSP posts the CIDRVR DSP to have the CI FSSs disable the procedure library. In turn, the CIDRVR DSP directs the C/I FSSs to disable the procedure library. When all C/I FSSs have disabled the procedure library, the CIDRVR DSP posts the DISABLE DSP.

ENABLE DSP

After the job completes Main Service, JSS schedules the ENABLE scheduler element by setting up an ENABLE DSP (IATIIEN). For each procedure library that is being updated by this job, the ENABLE DSP checks to see if it is the last job updating data sets in this procedure library. If so, the procedure library is dynamically allocated and opened in the JES3 global address space.

If the ENABLE DSP is successful, the CIDRVR DSP is posted to have the C/I FSSs enable the procedure library. The CIDRVR DSP directs the C/I FSSs to enable the procedure library. When all C/I FSSs have enabled the procedure library, the CIDRVR DSP posts the ENABLE DSP.

The ENABLE DSP resets the “held for update” flag and posts JSS so that jobs that use the procedure library can be scheduled for C/I.

7.28 PROCLIB operator command

```
IAT4890 PROCLIB IATPLBST HAS BEEN DISABLED FOR UPDATE BY JOB VAINPROC (JOB25093)
IAT2000 JOB VAINPROC (JOB25093) SELECTED SC65     GRP=A
IEF403I VAINPROC - STARTED - TIME=11.43.31
*I PROCLIB,ID=ST
IAT8952 PROCLIB IATPLBST IS UNALLOCATED AND DISABLED
IAT8954 IATPLBST, DSN=SYS1.TEST1.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=SYS1.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=CPAC.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=SYS1.IBM.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=SYS1.PROCLIB.OLD00 BEING UPDATED BY JOB VAINPROC (JOB25093)
IEF404I VAINPROC - ENDED - TIME=11.44.23
IAT4885 PROCLIB HAS BEEN ENABLED FOR DDNAME=IATPLBST
*I PROCLIB
IAT8952 PROCLIB IATPLBST IS ENABLED
IAT8954 IATPLBST, DSN=SYS1.TEST1.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=SYS1.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=CPAC.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=SYS1.IBM.PROCLIB NOT BEING UPDATED
IAT8954 IATPLBST, DSN=SYS1.PROCLIB.OLD00 NOT BEING UPDATED
```

INQUIRY PROCLIB Operator Command

To determine why a job is delayed in C/I service, you can display the status of a procedure library. The *INQUIRY,PROCLIB command displays whether all procedure library concatenations are enabled or disabled, and the job number of the job that is updating a particular data set. The procedure library status can be displayed with the operator command:

```
*I PROCLIB
```

The messages show which proclibs are being updated.

In the second command:

```
*I PROCLIB,ID=ST
```

ID=ST specifies the procedure library id. If omitted, the status of all procedure libraries will be displayed. Use the two digit suffix of the procedure library when entering a value for procid on the ID= keyword.

8

Main device scheduling (MDS)

JES3 provides a device management facility called the main device scheduler (MDS) that can wholly or partially support the MVS allocation process. The purpose of MDS is to satisfy job resource requirements (the devices, volumes, and data sets needed) before and during job execution, thus allowing execution to proceed without allocation delays. MDS also allows controlled multisystem access to commonly accessible data sets in the JES3 complex environment.

You must choose whether to use the JES3 main device scheduler or use MVS (which controls the job execution) for the entire allocation process as each step begins execution. If you choose MDS, you must then decide whether utilization of MDS is to be partial (set up some jobs, some resources) or total (set up all jobs, all resources).

You need to be aware that if MDS is used to provide resource management, MDS also takes into account the availability of a job's scheduling environment during resource allocation. Scheduling environments and resources are defined by the installation in the Workload Manager (WLM) policy. A scheduling environment is a list of resource names and their required states that are used to ensure that units of work are sent to systems that have the appropriate resources to handle them.

Scheduling environments and resource names reside in the WLM service definition and apply across the entire sysplex. Resource states may have a different setting in each system in the sysplex and are, therefore, system-oriented.

Each element in a scheduling environment consists of the name of a resource and a required state of either ON or OFF, as follows:

- If the required state is ON, then the resource state must be set to ON on an MVS image for the requirement to be satisfied.
- If the required state is OFF, then the resource state must be set to OFF on an MVS image for the requirement to be satisfied.

The SCHENV parameter on a JOB JCL statement specifies the name of the WLM scheduling environment to associate with the job.

8.1 Main device scheduling features

- **MDS OPTIONAL** - SETUP=NONE on STANDARDS statement
  - MVS allocates step at a time a job's I/O resources
  - MVS allocation only for the system executing the job
- **JES3 device management - Main Device Scheduler (MDS)**
  - Pre-execution I/O resource set up - Early resource release
    - Devices - Volumes - Data sets
  - MDS considers devices, volumes and data sets for a total job
    - JES3 assumes that all job steps will execute
  - MDS considers devices, volumes and data sets for all systems
  - MDS requests and verifies mounting of initial volumes
  - Useful on single or multisystem system configurations
  - Does not replace MVS allocation

Main device scheduling features

Main device scheduling (MDS) functions are optional and may be bypassed entirely or be applied only for a subset of resources.

MVS allocation

In systems that do not use MDS a job's I/O resource requirements are not known until the job is selected for execution, and a system initiator begins the step allocation process. At each job step, MVS allocation attempts to satisfy the requirements for the step, in contention with every other job step currently executing on the same processor. If the requirements cannot be met, MVS allocation gives the operator the option of canceling the job or allowing it to wait for resources. Thus, in a system that does not use MDS, there may be jobs executing and other jobs in execution waiting for resources.

The jobs waiting in MVS allocation hold critical resources (a system initiator, an address space, data sets, and possibly devices). Holding these resources longer than necessary makes it very difficult for the system programmer to determine how many initiators should be started to keep the system fully used, because at any given time, an unknown number of initiators may be waiting. MDS offers a solution to this problem.

Main device scheduler (MDS) allocation

With MDS, the resources (data sets, devices, and volumes) that a job requires are already set up when the job is passed to MVS for execution. There should never be an idle initiator caused by a job waiting for these resources. Setup occurs while a job is in the JES3 address space, and the only system resource used while the job is waiting is the JES3 queueing

space. MDS helps the system make maximum use of devices and allows jobs to run in a minimum amount of time when they are passed to the system for execution. Also, MDS cooperates with the workload management (WLM) component of MVS to ensure that the scheduling environments for jobs are honored.

MDS requests and verifies the mounting of the initial volumes a job requires on each device before the job can be selected for execution (unless deferred volume mounting is specified in the JCL).

Pre-execution setup

Because MDS setup occurs before job execution, JES3 cannot react to processing dependencies that can occur between different jobs and between different steps in the same job. This limitation is particularly important when considering the cataloging and passing of data sets. JES3 cannot determine whether any conditional job steps are skipped as a result of condition code processing. JES3 assumes that all job steps will execute. JES3 also counts the number of I/O devices needed by each step.

The JES3 main device scheduler controls the volume fetching, allocation, mounting, and deallocation of I/O devices associated with job execution on all processors in the complex. MDS allocates I/O resources among competing jobs and release resources as soon as possible after they have been used.

Scheduling environments

If a job is assigned a scheduling environment (for example, the SCHENV= parameter was specified on the JOB statement), the availability of the job's scheduling environment is taken into consideration when determining which systems have the resources for a job. If a job's scheduling environment is not available on a particular system, the job will not run on that system even if the other resources required by the job (e.g. DASD volumes, SMS storage groups) are available on that system. If the scheduling environment later becomes available on a system, the job will be able to run on that system provided that the resources required by the job are also available on that system.

MDS processing is divided into several stages:

- Volume fetch
- System select
- Allocation
- Volume verification
- System verify
- Breakdown

MDS is useful on a global only system or a multi-image complex. It does not replace MVS allocation, but works together with MVS allocation to manage control system resource such as devices, volumes, and data sets.

8.2 MDS benefits

- MVS initiators not waiting for I/O resources
  - No allocation recovery messages
- All tape units available to all systems all time
  - MVS supports automatic tape switching (ATS)
- Data set integrity
  - JES3 - All data sets available before MVS execution
  - MVS - Data set awareness processing - job initiation time
- Device fencing (pooling)
- Operators job made easier

MDS benefits

With MDS, the I/O resources (data sets, devices, and volumes) that a job requires are already set up when the job is passed to MVS for execution. Since I/O resources are available for a job when JES3 passes it for MVS execution, MVS allocation can satisfy the I/O requirements for the steps of the job. The job's I/O requirements are met and MVS allocation does not need to communicate with the operator and issue allocation recovery messages. JES3 allows all devices, including assignable devices (for example tapes), be online to all MVS systems in the JES3 complex. This provides a way to for better tape utilization because JES3 manages the allocation to tape units prior to execution of the using jobs.

Note: MVS supports automatic tape switching (ATS). An automatically switchable tape device can be online to some or all systems that are participating tape sharing within a sysplex. Automatically switchable tapes relieves the operators from keeping track of the online and offline status of tape devices, but MVS allocation may still not be able to meet a job's tape requirements and may have to issue allocation recovery messages. If the **`VARY AUTOSWITCH`** command is issued for a tape device that is online or managed by JES3, the system alerts you to the error.

Data set awareness

In a JES3 complex both MVS and JES3 provide data set awareness. Data set awareness ensures that:

- While one job is reading a data set another job does not change that data set

- Only one job at a time uses a sequentially accessed device that is attached to one or more processors

Data set awareness is provided by JES3 in a single or multi-system complex. MDS provides the capability to fence devices for jobs or for a class of jobs. MDS management of resources provides an easier operations for the operator.

MVS data set awareness

Before allocating data sets to a job, MVS enforces data set integrity for MVS-managed data sets. To ensure data set integrity, MVS does not allocate a data set to a job if:

- The job requests non-shared access to an allocated data set
- The job requests shared access to a data set that is allocated as non-shared

MVS does not begin the allocation process until the integrity of all the data sets in the job has been enforced. Once integrity has been established, MVS then begins allocating the resources needed for the job, one step at a time. JCL DD-statement DISP= parameter determines a data set's integrity requirements.

MVS provides integrity for data sets within a single MVS system or across multiple MVS systems in a sysplex.

JES3 data set awareness

JES3 enforces data set awareness for:

- Data sets that are requested via DD statements that require JES3-managed devices
- Data sets that are requested dynamically that require JES3-managed devices
- Data sets that are SMS-managed, unless SMSSETUP=NO is specified on the SETPARAM statement.

If a job requests a data set via a DD statement, JES3 enforces data set awareness before scheduling the job for execution. For dynamically requested data sets, JES3 enforces data set awareness at the time of the request. To ensure data set awareness, JES3 denies a job's request for a data set if:

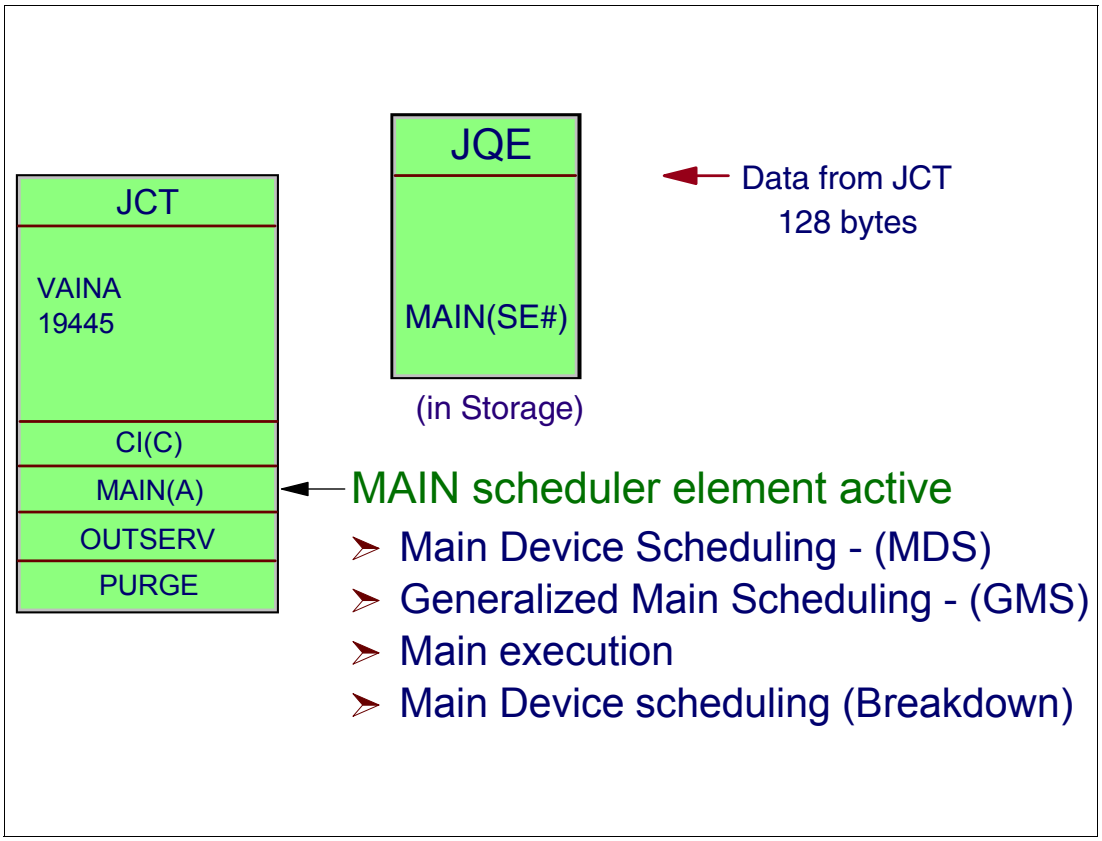
- The request is for non-shared access to an allocated data set
- The request is for shared access to a data set that is allocated non-shared

If a job's allocation request is denied by JES3 and the requested was made using a DD statement, JES3 does not schedule the job for execution at that time. The job must wait until all of the resources it needs are available. If the data set was dynamically requested, JES3 will not let MVS allocate the data set to the job at that time. The job, however, can continue to execute. JES3 also enforces data set awareness for data sets on sequentially accessed devices managed by JES3. Examples of such devices are tape drives. To ensure data set awareness, JES3 allows only one job at a time to use such a device anywhere in the complex.

Device fencing

Device fencing (sometimes called device pooling) for job-class groups can be defined by specifying either the DEVPOOL parameter or the device dedication subparameters in the EXRESC parameter of the GROUP statement. The difference between the two methods of dedication is that devices dedicated via the EXRESC parameter are dedicated when the group is allocated on the processor specified in the EXRESC parameter and DEVPOOL-requested dedication is accomplished when the GROUP is enabled on any processor. The operator use several commands to control the JES3 MDS processes. In general, in a JES3 complex, operators control complex and its systems through JES3 commands - not the individual MVS systems with MVS commands.

8.3 Job active in main scheduler element



Job active in main scheduler element

The main scheduler element represents the second phase of job processing. Converter/interpreter is the first phase. The converter/interpreter routines construct a job summary table (JST) that lists required data sets and devices, and a job volume table (JVT) that describes the volumes the main device scheduling routines will fetch and allocate. Volumes that are mounted will be verified.

Standard jobs

For standard jobs, main service processing begins when the job segment scheduler reaches the MAIN scheduler element in the job's JCT entry. The MAIN scheduler element represents two DSPs: setup (SETUP) and generalized main scheduling (MAIN). Both of these DSPs are resident and each has a permanent entry on the FCT chain, so the job segment scheduler need not construct the FCT entries.

Main device scheduling (MDS)

The work performed by the SETUP and MAIN DSPs is crucial to JES3 processing. The goals of the SETUP and MAIN DSPs are effective resource utilization and maximum JES3 complex throughput. The processing sequences are:

Initial setup (MDS) processing to prepare I/O resources. The initial setup processing is performed in phases. During the volume fetch phase, messages are sent to tape or DASD library consoles to indicate that a volume is to be fetched.

The allocation phase may be started automatically after the fetch phase, or manually when the operator indicates that the required volumes have been fetched to the work area. If volumes used are always readily available, automatic allocation could be more meaningful because no operator action would be necessary to cause allocation to begin. During the allocation phase, operator messages are issued for pre-mounting of volumes.

During the verification processing phase, checking is performed to ensure that the proper volumes are being mounted on the proper devices. This work is carried on asynchronously as devices become ready. The SETUP DSP maintains “verify counts” relative to individual jobs. The verify count is the number of volumes yet to be mounted for a job. When the verify phase is accomplished, the job will pass from initial setup processing to generalized main scheduling -- at this point, the job can be passed to MVS for execution.

Generalized main scheduling (GMS)

Generalized main scheduling selects and passes a job to an initiator. The job is ready for processing by the MAIN DSP. The main processing of jobs waiting to be selected by MVS is controlled by the installation's tailoring of JES3. When triggered by an MVS initiator's request for a job, the MAIN DSP chooses one jobs to give to the initiator. The selection of one job, in preference to another, is influenced by variables such as:

- The type of work being processed during a shift (test, production, online, batch, etc.)
- Eligibility relationships between jobs and processors based on:
  - Job classes
  - The number of active initiators for the jobs
  - Whether processors are online or offline
  - I/O rates of jobs in execution
  - Virtual storage requirements relative to the working set size of address spaces given to jobs
- Job priorities, to the extent that the installation wishes to honor priorities

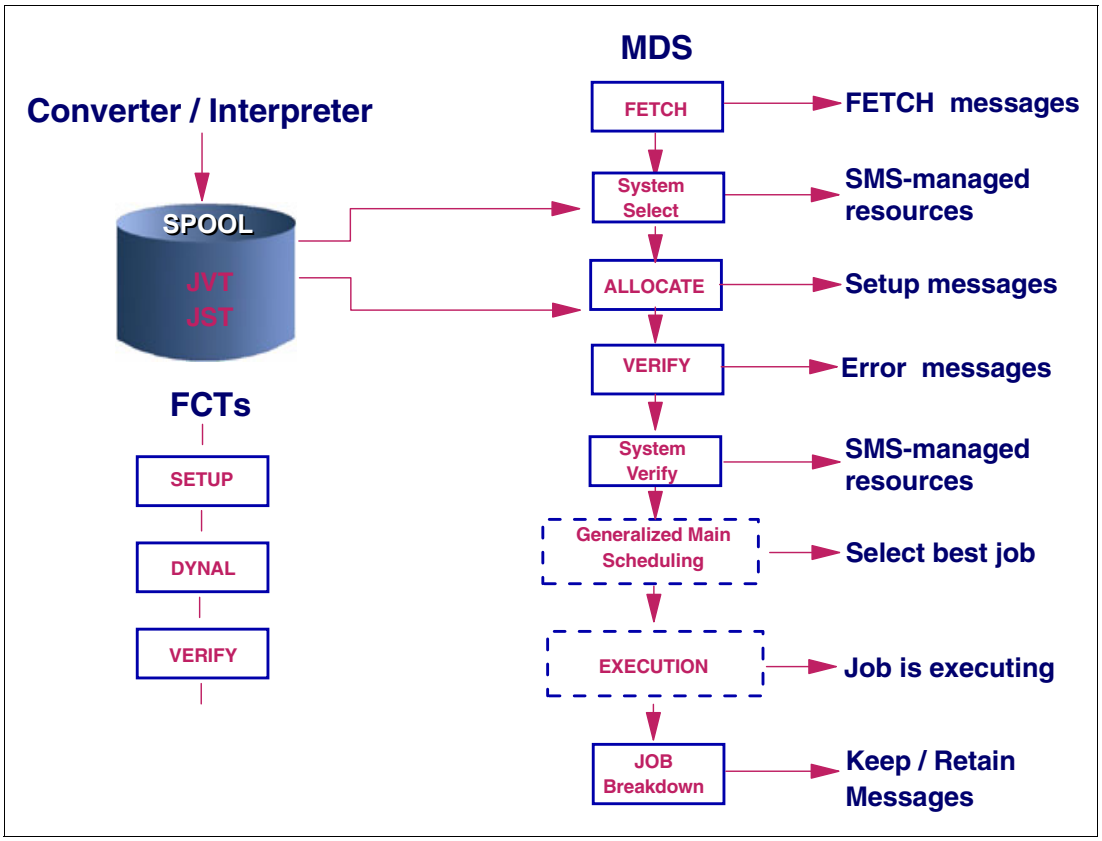
After considering these factors, the MAIN DSP picks our a job, sends information about the job to the requesting initiator, and indicates that the selected job is now “on main”. The MAIN DSP may skip jobs on a given selection pass. Having received the job, the initiator schedules it through all its steps, with JES3 being involved only for items such as:

- Step to step transition
- Opening of SYSIN/SYSOUT data sets
- Dynamic allocation and deallocation of data sets on JES3-managed or SMS-managed devices
- Requests for spool space

MDS breakdown

Breakdown processing to relinquish I/O resources. When a job completes execution under MVS, it is returned to the SETUP DSP for device breakdown processing. At the end of each job step, that is the last to use a device, volume, or data set, the resources are returned to the SETUP DSP for early resource release. Many, if not all, of the job's resources may already have been returned; but in most cases, the devices required for execution of the last step of the job must be returned to JES3 (along with data sets and volumes). Breakdown processing consists of updating control blocks by removing entries or reducing use counts and issuing appropriate “keep” or “retain” operator messages. The purpose, of course, is to make the resources available for use by other jobs that may require them. At this point the job segment scheduler marks the MAIN scheduler element complete and schedules the OUTSERV scheduler element.

8.4 MDS processing phases



MDS processing phases

Each MDS phase has its own queue for jobs waiting to be processed by the phase. The `*I,S` command, if none of the optional parameters is specified, displays the status of all mains in the complex and a summary of the MDS queues.

MDS FCTs

The FCTs that are involved with the MDS job processing are:

SETUP Main-line processing for volume fetch, job setup, high watermark setup, and explicit setup functions.

VERIFY When a job's "soft" allocation is successful, MDS allocation issues mount messages for those volumes that need to be mounted on a device. In addition, MDS allocation sends "arm" requests to the VERIFY FCT (IATLVVR) on the selected processor to prepare the device for volume mounting.

DYNAL Main-line processing for dynamic allocation-fast path.

Fetch phase

Volume fetch, the first phase of MDS, is performed for all jobs entering MDS. This phase determines the volumes required by the job and, if necessary, instructs the operator to get the volumes from the library. This phase also eliminates those mains on which the job cannot run. During fetch processing, JES3 builds volume entries and issues messages for volumes that have no entries in the Resident Volume Allocation Table (SETVOL - IATYVLM) which contains the volume serial number for each reference to a device managed by MDS. Volume fetch

messages are selected optionally by specifying FETCH=YES on the JES3 SETPARAM initialization statement. When you select the fetch option, JES3 issues volume fetch messages to indicate which volumes are required for specific jobs to execute. Volumes already mounted require no fetch processing, and volumes that have been fetched but not mounted get action-coded messages. Device types other than tape or disk do not require operator action. If you do not specify the volume fetch option, jobs go directly into the system select stage of MDS if the job requires SMS-managed resources, or into allocation if the job does not require SMS-managed resources.

System select phase

The system select phase of MDS is performed when a job requires one or more resources managed by the storage management subsystem (SMS). If the job does not require any SMS-managed resources, the job proceeds directly to MDS allocation. JES3 is not aware of the availability or connectivity of SMS-managed resources. If a job requires SMS-managed resources, JES3 requests SMS to determine the availability of those resources and to determine which mains have access to those resources. If JES3 determines that one or more mains have access to all of the required resources, the job proceeds into the allocation phase. If no mains have access to all of the required resources, JES3 invokes user exit IATUX61 (Cancel Jobs Going on the MDS Error Queue) to determine whether the job should be placed on the MDS error queue or canceled. If you do not implement IATUX61, MDS places the job on the MDS error queue where an operator must either restart the job or cancel it using a *RESTART,SETUP or *CANCEL,SETUP command respectively.

Allocate phase

This phase of MDS uses allocation algorithms to provide required devices for jobs. When allocation is successful, JES3 issues mount request messages for all required volumes except:

- Deferred mount requests - where no mount is as yet requested but the device that the volume is to be mounted on is allocated to the user.
- Permanently resident volumes - where the mount is unnecessary
- Multi-volume mount - where only the first volume of a multi-volume data set is mounted; secondary volumes are not mounted until required.

The ALLOCATE= keyword on the JES3 SETPARAM initialization statement controls how jobs are processed during MDS allocation. If ALLOCATE=AUTO (default) is specified, MDS sends incoming jobs directly into allocation unless a job requires SMS-managed resources. If a job requires SMS-managed resources, the job is first sent to system select before proceeding into the allocation phase. If ALLOCATE=MANUAL is specified, the operator must issue the ***S S** command for each job requiring volumes to be fetched before the job can go through MDS allocation. Jobs that require volumes to be fetched are kept in the MDS WAITVOL queue. At the start of the MDS allocation phase, a job is selected from the ALLOCATE queue. MDS examines the job's resource requirements and attempts to allocate the required devices, volumes, and data sets. For SMS-managed data set resource, only the data set is allocated. JES3 is not aware of SMS-managed volumes and devices. When MDS initially tries to set up a job, it records the total device, volume, and data set requirements for the job. When allocation fails because needed devices, volumes, or data sets are unavailable, the job is queued for another attempt. However, MDS sends jobs back to the system select phase of MDS if all of the following conditions occur:

1. A job requires both SMS-managed resources and MDS-managed resources.
2. The list of eligible mains determined by the system select phase do not have access to both the MDS-managed resources and the SMS-managed resources.
3. One or more mains not in the original list of mains has access to all of the required resources, and the SMS-managed resources are temporarily unavailable to those mains.

Note: The generalized main scheduling (GMS) resources (job class groups and job classes) must be available before MDS resource allocation is attempted.

Verify phase

JES3 issues messages that instruct you to mount a job's required volumes. You can implement installation exit IATUX62 (Verify a Mount Request) to validate, accept, or override JES3's decision about whether a volume has been successfully mounted. JES3 invokes IATUX62 after the volume verification phase of MDS. The VERIFY function automatically obtains the volume serial number, label status, and other information for MDS after you mount the job's required volumes. You can install installation exit IATUX25 (Examine/Modify Volume Serial Number) to validate any nonstandard labels used in the installation. When all volumes are properly mounted, the job is ready for execution. However, if the job requires SMS-managed resources, it proceeds to the system verify phase of MDS before execution. Device types other than tape or disk do not require operator action.

System verify phase

The system verify phase of MDS is performed when a job requires SMS-managed resources. This phase of MDS ensures that all of the SMS-managed resources required by a job are still available before execution. For example, if a job spends too much time in MDS allocation or a long period of time elapses between a mount request and the actual mounting of a required volume, one or more of the SMS-managed resources could become unavailable. If the status of SMS-managed resources required by the job has not changed, that is, all of the required SMS-managed resources are still available and are accessible by the eligible main(s), the job can proceed into execution. However, if SMS-managed resources required by a job are no longer available, JES3 generates a new list of systems where SMS managed resources are available. JES3 then checks this list against the following:

- All jobs to see if they can access the SMS and JES3 resources
- Batch jobs to see if the class and group are available on the systems where SMS managed resources are available.
- Batch jobs with associated scheduling environments to see if there are any systems where the scheduling environment and the SMS managed resources are available.

If there are no systems where all of the above resources are available, the job is sent to the breakdown phase where MDS deallocates resources held by the job. MDS then sends the job back to the system select phase where MDS retries allocation.

Breakdown phase

JES3 automatically performs the breakdown phase of MDS when a job no longer needs a resource such as a data set, volume, or device. The resource is then available for use by other jobs. MDS does not deallocate SMS-managed resources other than data sets because JES3 is not aware of SMS-managed devices and volumes.

JES3 issues messages that indicate whether volumes should be retained for use by other jobs or demounted. The RETAIN and KEEP messages issued by MVS allocation apply only to the resources used within one job, while the RETAIN and KEEP messages issued by MDS consider volume usage by all jobs currently in the system that use JES3-managed or jointly-managed devices. In the event that both MVS and JES3 issue KEEP or RETAIN messages regarding a specific volume, the JES3 messages take priority.

8.5 SETPARAM statement

- Parameters for MDS and DYNAL DSP for allocation, mounting, and deallocation of devices for jobs

```
SETPARAM,ADDRSORT=YES,ALLOCATE=AUTO
         ,DAFETCH=97
         ,DEFERCT=NO
         ,DSN=0
         ,FETCH=YES
         ,MDSLOG=97
         ,REMOUNT=1
         ,TAFETCH=(97,97)
         ,ALWIO=10
         ,MAXIO=25
         ,PRJESMSG=NONE
         ,SMSSETUP=YES
         ,MTJESMSG=(ALLOC,BREAKDWN)
```

SETPARAM statement

Use the SETPARAM initialization statement to specify parameters that the JES3 main device scheduler (MDS) and the DYNAL DSP uses in allocation, mounting, and deallocation of devices for jobs run on all mains. The SETNAME and DEVICE statements are used with the SETPARAM statements. SETNAME and DEVICE identify the devices to be managed by MDS. SETPARAM also indicates how MDS is to manage devices.

Note: For the DAFETCH, MDSLOG and TAFETCH parameters, the default of 97 is the routing code equivalent of JES3 destination class S1.

ADDRSORT= Specifies the order in which JES3 MDS allocates devices.

NO - Indicates that devices within a device type are to be allocated in the same order as the DEVICE statements are placed in the initialization stream.

YES - Indicates that devices within a device type are to be allocated by the order of their device numbers, that is, 188, 189, 18A.

ALLOCATE= Specifies whether automatic allocation of a job is to immediately follow MDS volume fetch. This parameter is ignored for jobs that reference only premounted volumes. The FETCH parameter specified may override the ALLOCATE parameter.

MANUAL - Indicates that all jobs are to be suspended following volume fetch until the operator causes them to continue. Note that

ALLOCATE=MANUAL is ignored if FETCH=NO is indicated; ALLOCATE=AUTO is assumed instead.

AUTO - Specifies that MDS will automatically attempt allocation of resources for all eligible jobs. If a job requires SMS-managed resources and you specify ALLOCATE=AUTO, MDS sends the job through the system select phase before allocation to determine which mains have access to the required SMS-managed resources. Note that ALLOCATE=AUTO is assumed (ALLOCATE=MANUAL is ignored) if FETCH=NO is specified.

DAFETCH= Specifies the routing information for direct-access volume fetch messages. msgdest Specifies a SETUP-related console destination class. Direct-access volume fetch messages are issued with the routing code equivalent of this destination class.

NONE - Indicates that volume fetch messages are not to be issued.

97 - Indicates that volume fetch messages are to be issued with routing code 97; messages also are recorded on the hard-copy log. Note that this parameter is ignored if FETCH=NO is also specified. 97 is the routing code equivalent of JES3 destination class S1.

nnn - Specifies an MVS routing code from 1 to 28, or 41 to 128. Routing codes 29 through 40 are reserved for IBM's use and will be ignored if specified.

DEFERCT= Specifies whether jobs requiring deferred mounts (whether explicitly requested through JCL, or implicitly requested because of using tape library devices) should be included in the CLASS/SELECT SDEPTH counts. The default is DEFERCT=NO.

DSN= Specifies the number of characters (0 to 44) of the data set name to be included in MDS volume fetch, mount, and breakdown messages. This parameter is used for message formatting. If DSN=0 is specified, then the data set name is omitted from these MDS messages.

FETCH= Indicates whether MDS is to issue volume fetch messages. Note that the FETCH parameter can override the ALLOCATE parameter.

NO - Specifies that MDS is not to issue volume fetch messages. If FETCH=NO is specified, ALLOCATE=MANUAL is overridden (and ALLOCATE=AUTO assumed); MDS automatically attempts to set up jobs.

YES - Indicates that MDS is to issue volume fetch messages.

MDSLOG= Specifies the routing information for all non-action messages (that is, job LOGON and error messages).

msgdest - Specifies a SETUP-related console destination. Non-action messages are issued with the routing code of the destination class.

97 - Indicates that non-action messages are issued with routing code 97.

nnn - Specifies an MVS routing code from 1 to 28, or 41 to 128. Routing codes 29 through 40 are reserved for IBM's use and is ignored if specified.

MTJESMSG= Specifies whether you want FETCH, ALLOCATION, and BREAKDOWN messages for mountable devices to appear in the JESMSGLG data set.

FETCH - Specifies that you want fetch messages for mountable devices written into the JESMSGLG data set.

ALLOC - Specifies that you want allocation messages for mountable devices written into the JESMSGLG data set.

BREAKDWN - Specifies that you want breakdown messages for mountable devices written into the JESMSGLG data set.

ALL - Specifies that you want fetch, allocation, and breakdown messages for mountable devices written into the JESMSGLG data set.

NONE - Specifies that you do not want fetch, allocation, or breakdown messages for mountable devices written into the JESMSGLG data set.

> **Note:** When you use the default (ALLOC and BREAKDWN), allocation and breakdown messages for mountable devices are written into the JESMSGLG data set.

FETCH - Specifies that you want fetch messages for permanently resident or reserved DASD written into the JESMSGLG data set.

ALLOC - Specifies that you want allocation messages for permanently resident or reserved DASD written into the JESMSGLG data set. If this value is specified, an allocation message will be written for all non-mountable requests in addition to permanently resident DASD.

ALL - Specifies that you want both fetch and allocation messages for permanently resident or reserved DASD and allocation messages for all other devices written into the JESMSGLG data set.

NONE - Specifies that you do not want fetch or allocation messages for permanently resident or reserved DASD written into the JESMSGLG data set.

REMOUNT= Specifies the number of times that an operator can retry to correct volume mount errors for a job before the devices for the job are released and allocation is restarted. The value of nnn specifies the number of retries allowed, from 0 to 255. For example, if REMOUNT=1 is specified, the operator can make two attempts to mount the volume--the original mount request and one retry request.

SDEPZERO= Indicates whether jobs that require a tape mount, but are in a CLASS, are defined as SDEPTH=0, should wait on the MDS allocate queue, the default, or be sent to the MDS error queue.

SMSSETUP= Specifies whether JES3 manages SMS data sets.

NO - Indicates that SMS data sets are not to be managed by JES3.

YES - Indicates that SMS data sets are to be managed by JES3.

If you specify an incorrect subparameter or do not specify the SMSSETUP= parameter, MVS determines whether JES3 manages SMS data sets or not.

TAFETCH= Specifies the routing information for tape volume fetch messages. The first operand specifies the routing information for specific (nonscratch) volume requests. The second operand specifies the routing information for scratch volume requests.

msgdest - Specifies a SETUP-related console destination class. Tape volume (scratch or nonscratch) fetch messages are issued with the routing code equivalent of this destination class.

NONE - indicates that volume fetch messages are not to be issued.

97 - Indicates that volume fetch messages are to be issued with routing code 97; messages also are recorded on the hard-copy log. Note that this parameter is ignored if FETCH=NO is also specified.

nnn - Specifies an MVS routing code from 1 to 28, or 41 to 128.

ALWIO= The ALWIO parameter specifies the current number of asynchronous I/O requests which can be processed concurrently. This value must be a number from 1 to the value specified in the MAXIO parameter. The value specified in the ALWIO parameter must be less than or equal to the value specified in the MAXIO parameter.

This parameter can be displayed through the **`*I S,ALWIO=nn`** command, and modified through the **`*F S,ALWIO=nn`** operator command.

MAXIO= The MAXIO parameter specifies the maximum number of asynchronous I/O requests that can be processed concurrently. Storage is obtained for the number of requests specified here. Note that an increase of one in this parameter results in a 76-byte increase in storage used. This parameter can only be changed when performing a warm start or cold start. The value specified in the MAXIO parameter may be a number from 1 to 99. The default value is 25. The value specified in the MAXIO parameter must be greater than or equal to the value specified in the ALWIO parameter.

SETPARAM example

In the following example, volume fetch messages are issued with the routing code equivalent of the destination classes specified:

S7 Nonscratch tape volume fetch messages.

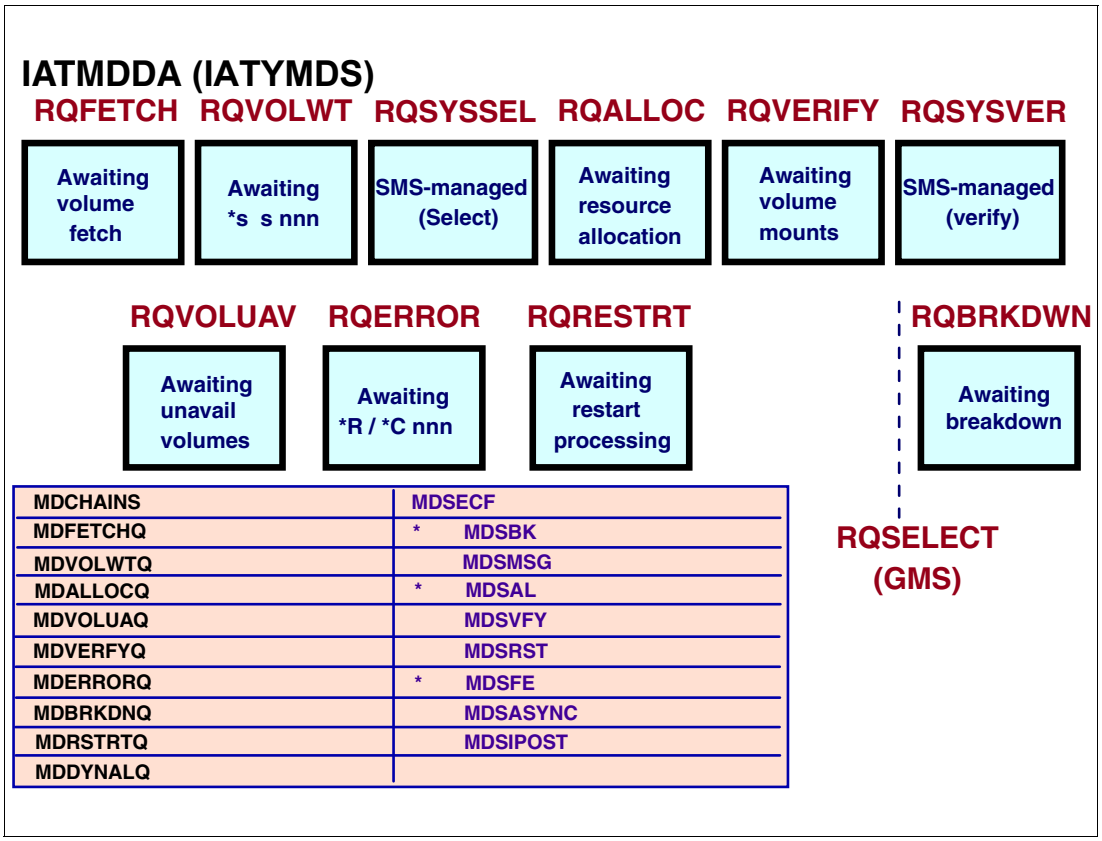
S10 Scratch tape fetch messages.

S9 Direct-access volume fetch messages.

Also, MDS messages would identify the first 15 characters of the data set names. All nonaction messages would go to console destination S1. If necessary, one retry to mount any volume would be allowed. Allocation would occur automatically following volume fetch. Allocation order for devices would be by the order of their device numbers, as follows:

```
SETPARAM,FETCH=YES,TAFETCH=(S7,S10),DAFETCH=S9,DSN=15
```

8.6 MDS processing queues



MDS processing queues

The MDS processing queues can be displayed by using the operator commands shown in the figure under each queue.

The SETUP DSP's data CSECT, IATMDDA mapped by IATYMDS macro, contains the MDS chain pointers.

```
MDCHAINS DS    0F                START OF MDS CHAINS
MDFETCHQ DC    F'0'              START OF FETCH CHAIN
MDVOLWTQ DC    F'0'              START OF VOLUME WAIT CHAIN
MDALLOCQ DC    F'0'              START OF ALLOCATE CHAIN
MDVOLUAQ DC    F'0'              START OF UNAVAILABLE CHAIN
MDVERFYQ DC    F'0'              START OF VERIFY CHAIN
MDERRORQ DC    F'0'              START OF ERROR CHAIN
MDBRKDNQ DC    F'0'              START OF BREAKDOWN CHAIN
MDRSTRTQ DC    F'0'              START OF RESTART CHAIN
MDYNALOQ DC    F'0'              START OF DYNAMIC ALLOC CHAIN
```

SETUP FCT await flags

The SETUP FCT waits on the MDSECF flag byte for posts. The SETUP FCT posts are:

```
MDSECF   EQU   MDSECFAR,1        MDS ECF
MDSBK    EQU   X'80'             BREAKDOWN POST BIT
MDSMSG   EQU   X'40'             MESSAGE POST BIT
MDSAL    EQU   X'20'             ALLOC POST BIT(SEE MDSALECF)
MDSVFY   EQU   X'10'             VERIFY POST BIT
MDSRST   EQU   X'08'             RESTART POST BIT
```

```
MDSFE    EQU   X'04'               FETCH POST BIT
MDSASYNC EQU   X'02'               ASYNCRONOUS PROCESSING REQ'D
MDSIPOST EQU   X'01'               INTERNAL PROCESSING REQ'D
```

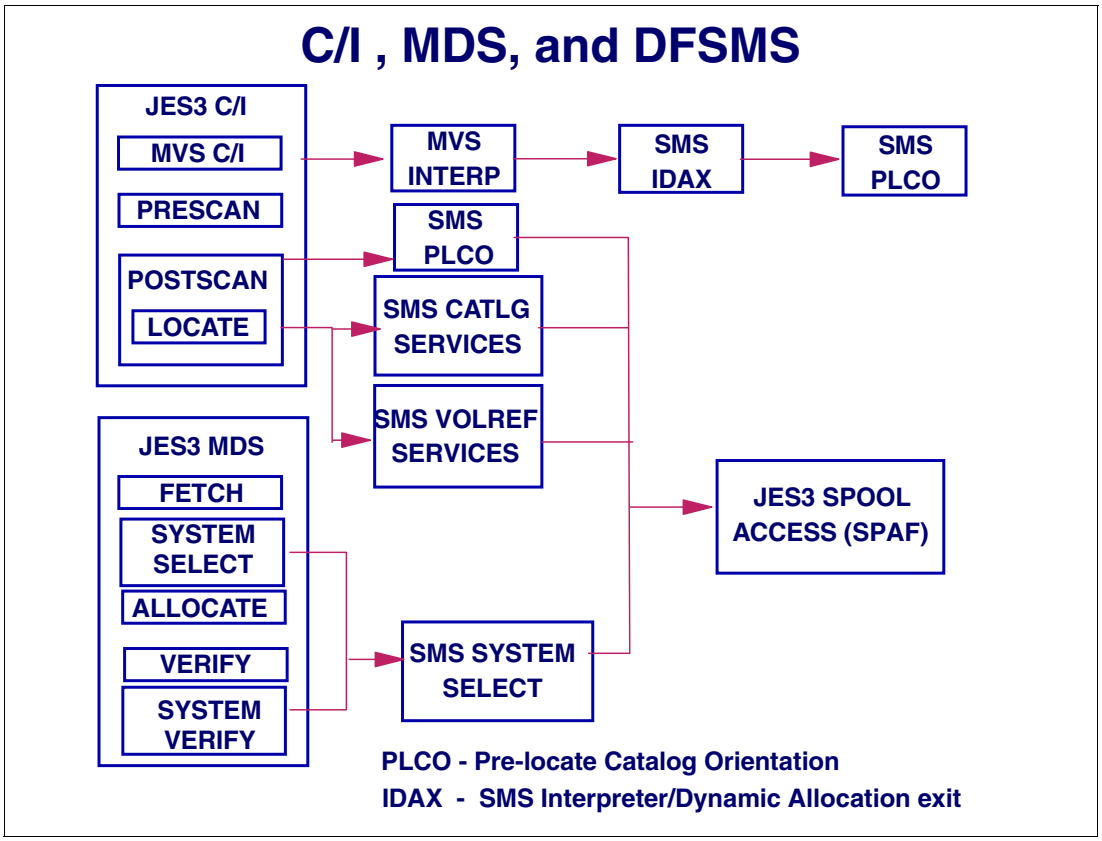
SETUP FCT posted

In general, the SETUP FCT is posted whenever there is a change of the MDS managed resources, the status of the jobs running under the MAIN scheduler element. Examples of the events that cause SETUP to be posted:

- JSS adds new jobs
- SETUP queues a job to new subchain
- Main processor comes online or goes offline
- GMS job class group, job class, or scheduling environment status change
- WLM policy change
- SMS resources, storage groups and volumes, change status
- Jobs status is changed by operator (hold, release, cancel, class change)
- Device availability status is changed
- Jobs in MVS execution request dynamic allocation/unallocation
- Jobs in MVS execution terminate

The MDS processing queues can be displayed with the *INQUIRY S the operator command.

8.7 JES3/DFSMS communication



JES3/DFSMS communication

JES3 SMS support provides complex-wide data set awareness for DFSMS-managed data sets through subsystem interface communication with DFSMS, Main processor and DFSMS resource availability are determined for scheduling jobs into execution using these interface calls.

JES3 and DFSMS communicate required to:

- Make sure that catalog locates are done on a processor with access to the required catalogs.
- Make sure that jobs requiring DFSMS resources execute on processors to which the resources are accessible and available.
- Provide complex-wide data set awareness for all DFSMS managed requests (even for new, non-specific requests).
- Remove JES3 awareness of units and volumes for DFSMS managed data sets. (One of the DFSMS objectives is to remove user awareness of the physical storage.)

System Select is a phase of MDS processing and calls DFSMS System Select to access the JES3 spool through the SPAF interface. DFSMS System Select is invoked prior to MDS allocation to determine the availability and processor connectivity of a job's DFSMS managed resources. This frees JES3 from having to know about the status of DFSMS resources, and prevents MDS from allocating and mounting devices before the status of DFSMS resources is known. If all the resources are available, the job continues into MDS; otherwise, the job waits until the required resources become available.

System Verify is a phase of MDS processing and calls DFSMS System Select to access the JES3 spool through the SPAF interface. DFSMS System Select is also invoked after MDS verify to determine whether the status of DFSMS resources has changed since the last call to DFSMS System Select was made. If the status has changed and there is now a conflict in the systems required to access the DFSMS resources, the job is failed. If the status of the job and scheduling information has not changed, the job is placed on the Generalized Main Scheduling (GMS) queue. If the information from the SPAF file has changed, the job is recycled through MDS.

The **IDAX** is invoked by the MVS Interpreter at the end of interpretation to analyze the JCL and to construct SWBs for required DFSMS information. When the C/I is running under JES3, it indicates to the IDAX that the caller wants system scheduling information. IDAX collects scheduling information for new data sets to be allocated by this job and saves it in the scheduling information SPAF file. The Scheduler JCL Facility (SJF) is used to scan the SWA blocks, and IDAX creates SWBs for new DFSMS managed data sets. The Automatic Class Selection (ACS) routines and the installation ACS user exits are invoked at this point. The data class and storage class is selected at this time for new DFSMS managed data sets. IDAX catalog processing determines all of the catalogs required by a job and divides them into two categories: DFSMS managed user catalogs, and JES3 managed user catalogs. Information about the DFSMS managed user catalogs is written to the job's SPAF file for later use in MDS.

In addition, if the job has DFSMS managed user catalogs, **PLCO** is invoked to determine the processors that have access to the required catalogs. This information is returned to JES3 and used to determine where locates should be scheduled. Information about the non-DFSMS managed user catalogs is also returned to JES3. JES3 then adds those user catalogs that require JES3 managed devices to the job's setup requirements.

DFSMS PLCO is invoked through an SSI Call from IDAX. There are two calls to DFSMS PLCO; from DFSMS IDAX and from POSTSCAN. The PLCO is invoked by the DFSMS IDAX during MVS interpretation to determine the availability and processor connectivity of all DFSMS managed catalogs required by the job. DFSMS returns to JES3 a list of processors that can currently access all the DFSMS managed catalogs required by the job. This list is used by JES3 to determine which main processors can be used for locate processing. If one or more catalogs is unavailable, the job is rescheduled for locate processing when it becomes available.

From POSTSCAN, DFSMS PLCO is invoked prior to JES3 locate processing for jobs that have been rescheduled to determine whether the SMS-managed catalogs required by the job are available. Access to the SPAF files is through the resource status token passed from the DFSMS PLCO call. The main processors eligible for locate processing are determined and the job's main mask is updated. DFSMS PLCO is performed for the first time for all jobs during MVS interpretation.

DFSMS Catalog Services is invoked during locate processing, instead of SVC 26, for all existing data sets when DFSMS is active. Locates are required for all existing data sets to determine whether they are DFSMS managed, even if VOL=SER= is present on DD statement. If VOL=REF= is present on a DD statement, the DFSMS VOLREF Service is invoked.

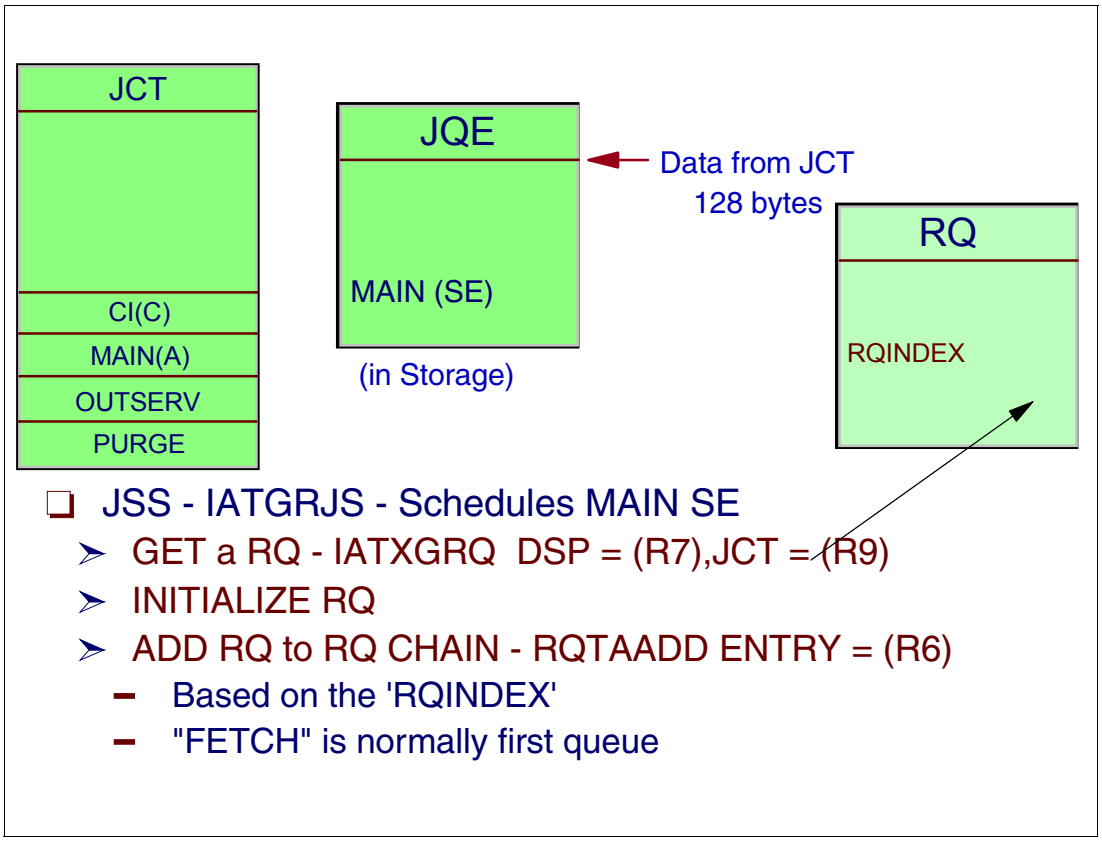
Note: JES3 locate processing calls DFSMS Catalog Services at the end of locate processing for cleanup and at this time writes the scheduling information, collected by DFSMS Catalog and VOLREF Services, and writes it to the spool through SPAF. DFSMS does not have to be active on local processors for locates to take place there.

DFSMS VOLREF Services are invoked during locate processing, instead of SVC 26, for each data set that contains a volume reference to a cataloged data set. DFSMS VOLREF Services determines whether the data set referenced by a VOL=REF= parameter is DFSMS managed. Note that VOL=REF= now maps to the same storage group for an DFSMS managed data set, not necessarily to the same volume. DFSMS VOLREF Services also collects information about the job's resource requirements.

SPAF is used by DFSMS to access resource information on spool. For DFSMS, this includes the DFSMS scheduling information spool data set and the DFSMS job information spool data set. The DFSMS scheduling information spool data set is a spool-resident data set that contains scheduling information for all the new and old DFSMS managed data sets referenced in a job, as well as scheduling information for the catalogs required by the job. The DFSMS job information spool data set is a spool-resident data set that contains job related information used during locate processing to determine to which storage group a migrated DFSMS managed data set may be recalled. The SPAF is invoked by SSI and uses USAM to read from and write to the data set specified by caller. SPAF offers read only access through the USAM interfaces.

Note: The systems in the JES3 complex must be defined in the DFSMS control data set for proper JES3 DFSM operation.

8.8 JSS scheduling of MAIN SE



JSS scheduling of MAIN SE

The JES3 job segment scheduler (IAGRJS) schedules a job's MAIN scheduler element by obtaining an RQ, appropriately setting RQINDEX, and calling RQTAADD to add the RQ to the RQINDEX chain. The RQTAADD macro posts the appropriate DSP.

IATXGRQ macro service obtains an RQ from a RESQUEUE cellpool. The type of RQ is based on DSP dictionary entry provided. If a JCT address is also provided, the RQ is initialized with data from the JCT.

RQINDEX parameter

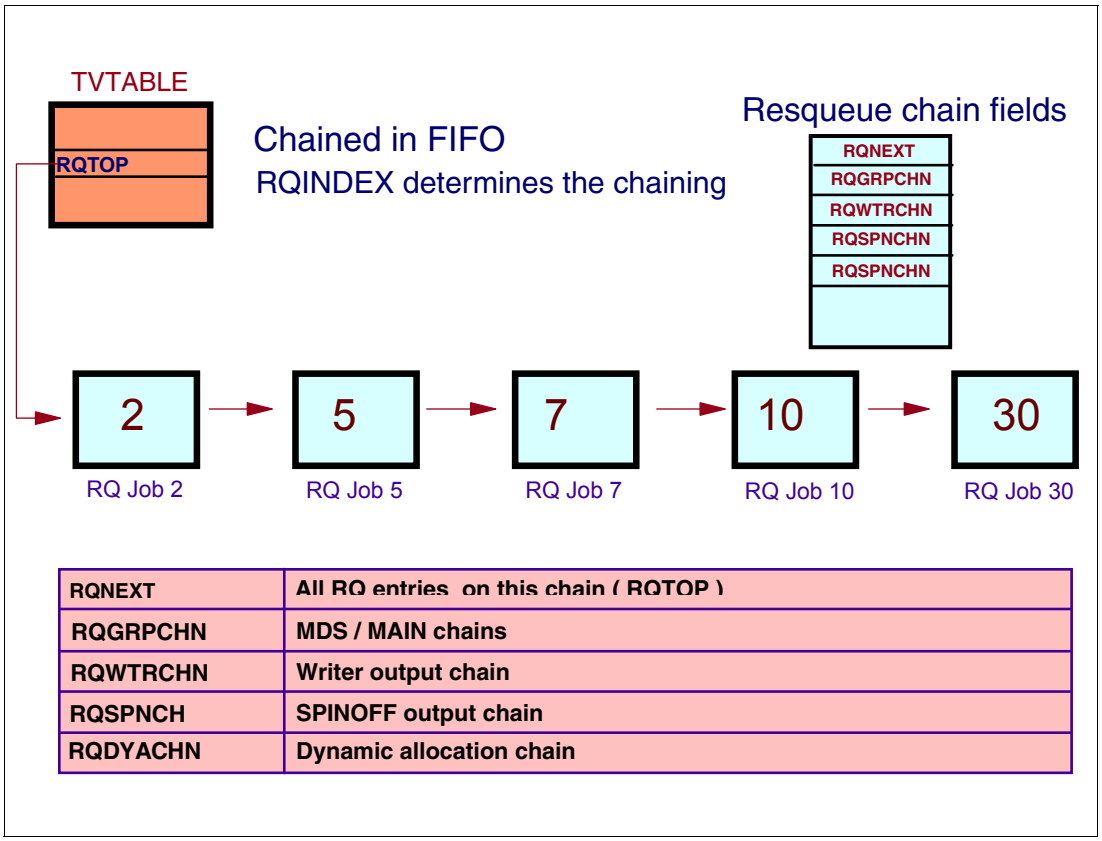
This parameter specifies the chain where the entry is placed. If "name" is coded, the name specified must be one of the terms defined for field RQINDEX in the IATYRSQ macro. If omitted, the index in the entry is used.

The INDEX parameter on the RQTAADD macro specifies the chain where the entry is placed. These fields are defined in the IATYRSQ macro. During the time a job exists in the system, the job will exist on many of these resqueue indexes. The RQINDEX value is then an indicator of where the job is currently processing. The RQ index values used for processing of jobs through both MDS, MAIN, and OUTSERV scheduler elements are:

```
RQNOSUB   NO SUBCHAIN EXISTS
RQFSSCI   ACTIVE IN CI IN AN FSS ADDRESS SPACE
RQPSCBAT  AWAITING POSTSCAN (BATCH)
RQPSCDSL  AWAITING POSTSCAN (DEMSEL)
RQFETCH   AWAITING VOLUME FETCH
```

```
RQVOLWT   AWAITING START SETUP
RQSYSSEL  AWAITING/ACTIVE IN MDS SYSTEM SELECT PROCESSING
RQALLOC   AWAITING RESOURCE ALLOCATION
RQVOLUAV  AWAITING UNAVAILABLE VOL(S)
RQVERIFY  AWAITING VOLUME MOUNTS
RQSYSVER  AWAITING/ACTIVE IN MDS SYSTEM VERIFY PROCESSING
RQERROR   ERROR DURING MDS PROCESSING
RQSELECT  AWAITING SELECTION ON MAIN
RQONMAIN  SCHEDULED ON MAIN
RQWTR     AWAITING WTR OUTPUT (ASP)
RQTERM    AWAITING MAIN TERMINATION (ASP)
RQBRKDWN  AWAITING BREAKDOWN
RQRESTRT  AWAITING MDS RESTART PROC.
RQDONE    MAIN AND MDS PROC. COMPLETE
RQOUTPT   AWAITING OUTPUT SERVICE
RQOUTQUE  AWAITING OUTPUT SERVICE WTR
RQOSWAIT  AWAITING RSVD SERVICES
RQCMPLT   OUTPUT SERVICE COMPLETE
RQDEMSEL  AWAITING SELECTION ON MAIN (DEMAND SELECT JOB)
RQEFWAIT  ENDING FUNCTION RQ WAITING FOR I/O COMPLETION
RQEFBAD   ENDING FUNCTION RQ NOT PROCESSED
```

8.9 RESQUEUE chaining



RESQUEUE chaining

When the job segment scheduler (JSS) is dispatched, it examines JQEs to find an eligible job so it can schedule a DSP. Part of the processing is construction of an FCT entry when the DSP being scheduled is not represented by a resident FCT entry. In addition to constructing and enqueuing an FCT entry (if necessary), the job segment scheduler *always* builds a resident queue element (RQ or RESQUEUE). The RQ is built after the job is scheduled and it thus becomes the basis for DSP processing.

The RQ is a large control block containing status flags, job information fields, and queuing pointers. RQs last only for the life of a scheduler element. Pointers in the RQ allow JES3 DSPs working on behalf of a job to find all the other job-related information kept by JES3.

The RQ represents a scheduling chain within a given DSP. It contains a summary of the information the DSP needs to accomplish its function, plus additional information. The JSS builds the RQ from information in the JCT entry. Pointers from the JCT entry to the job's other single-record files (SRFs) are extracted and placed into the RQ. The RQ contains spool information for the job that is executing and holds even more information about the job than the JCT entry.

MAIN scheduler element processing

During the MAIN SE processing a job may be active on several DSPs and several DSPs can be active with one job at the same time. To facilitate this, an RQ control block includes several chain fields:

- RQNEXT -- All RQs are chained together through this field.

- RQGRPCHN -- CI/MDS/MAIN/OUTSERV/JSS subchain.
- RQWTRCHN -- WRITER OUTPUT subchain.
- RQSPNCH -- SPINOFF OUTPUT subchain.
- RQDYACHN -- DYNAMIC ALLOCATION subchain

Anchor chain pointers

The TVT table contains the following RQ chain anchors:

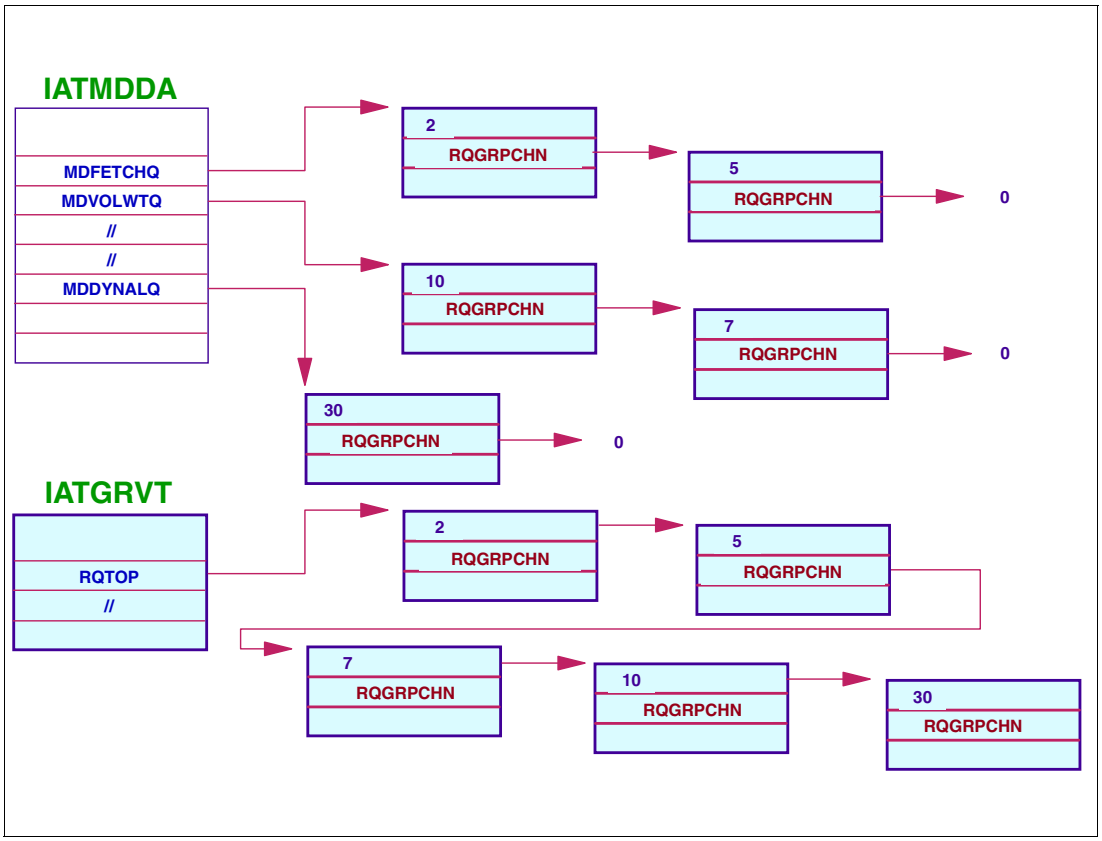
- RQTOP -- Start of RQ chain. All RQs are on this chain
- SPORQTOP -- Start of SPINOFF RQ chain
- OSSWAIT -- OUTPUT SERVICE WAIT chain
- OSSRQTOP -- Start of RQ OUTPUT chain
- Some functions have their RQ chain anchors in the DSP's data csect. These data csects are pointed from the TVT table. For example MDS data csect (IATMDDA) is pointed from MDSPARM field in TVT.

RQ chaining

RQs are chained to proper queues using macro services. IATGRRQ module contains the routines to service the following macros:

- The RQTAADD macro adds an RQ to the RESQUEUE chain and subchain.
- The RQTAPUT macro moves an from one chain to another or changes the priority within a chain.
- The RQTADEL macro deletes an RQ from a RESQUEUE chain and subchain.

8.10 Job chains in MDS processing queues



Job chains in MDS processing queues

The SETUP DSP's data CSECT (IATMDDA) is pointed from MDSPARM field in TVT. The IATMDDA control block contains commonly used data areas and flags referenced by all MDS modules. The MDS RQ subchain anchors are in the IATMDDA data CSECT:

```
MDFETCHQ        START OF FETCH CHAIN
MDVOLWTQ        START OF VOLUME WAIT CHAIN
MDALLOCQ        START OF ALLOCATE CHAIN
MDVOLUAQ        START OF UNAVAILABLE CHAIN
MDVERFYQ        START OF VERIFY CHAIN
MDERRORQ        START OF ERROR CHAIN
MDBRKDNQ        START OF BREAKDOWN CHAIN
MDRSTRTQ        START OF RESTART CHAIN
MDYNALOQ        START OF DYNAMIC ALLOC CHAIN
```

Active job in MAIN SE

When a job is active under the MAIN scheduler element in SETUP DSP, the placement of a job (represented by an RQ) on a MDS resqueue subchain is based on the RQINDEX value that represents the current MDS processing phase. In a very busy system, there are many jobs in each of these chains as shown in the visual as an example. MDS processing under the SETUP FCT searches each chain to schedule the job to the next RQINDEX value when it is then placed at the end of the next RQINDEX chain.

RQTOP chain

All RQ entries exist on a separate chain entry called RQTOP that exists in the TVTABLE control block (module IATGRVT).

RQINDEX chain pointers in IATMDDDA

The MDS RQINDEX values are:

```
RQFETCH        AWAITING VOLUME FETCH
RQVOLWT        AWAITING START SETUP
RQALLOC        AWAITING RESOURCE ALLOCATION
RQVOLUAV       AWAITING UNAVAILABLE VOL(S)
RQVERIFY       AWAITING VOLUME MOUNTS
RQERROR        ERROR DURING MDS PROCESSING
RQSELECT       AWAITING SELECTION ON MAIN
RQBRKDWN       AWAITING BREAKDOWN
RQRESTRT       AWAITING MDS RESTART PROC.
```

8.11 A job's MAIN scheduler element status

*I,J | *I,P | *I,Q - - IAT8674 JOB jobname (jobid) text

FETCH	-- Awaiting volume Fetch	MDS	*I,S,F
WAITVOL	-- Awaiting start setup	MDS	*I,S,W
SYSTEM SELECT	-- Awaiting start setup	MDS	*I,S,W
ALLOCATE	-- Awaiting resource allocation	MDS	*I,S,A
VOL UNAVAIL	-- Awaiting unavailable resources	MDS	*I,S,U
VERIFY	-- Awaiting volume mounts	MDS	*I,S,V
SYSTEM VERIFY	-- AWAITING SMS VERIFICATION	MDS	*I,S,SV
MDS ERROR	-- Awaiting operator decision	MDS	*I,S,E
GMS SELECT	-- Awaiting main selection	GMS	*I,B *I,G *I,J
DEMAND SELECT	-- Awaiting main selection "SystemTasks"	GMS	*I,J
EXECUTING	-- Executing on main processor	MAIN	*I,A
BREAKDOWN	-- Awaiting breakdown	MDS	*I,S,B
MDS RESTART	-- Awaiting MDS restart processing	MDS	*I,S,R
MAIN COMPLETE	-- Main scheduler element complete		*I,J
DSP ABEND	-- Main SE terminated by failsoft		*I,J
OUTSERV WAIT	-- AWAITING RESCHEDULE		*I,J
I/O WAIT	-- ENDING FUNCTION WAITING FOR I/O		*I,J
ENDFUNC ERROR	-- RQ WAS NOT PROCESSED		*I,J

A job's MAIN scheduler element status

The following commands **`*I J`**, **`*I P`**, and **`*I Q`** produce the IAT8674 message the indicates current status of jobs for that command.

*I J= command

On a job inquiry command, **`*I J=`**, when the MAIN scheduler element is the active one, the status shown in the MAIN scheduler element is ready to be processed or is being processed for the job. If no DSP is running yet, only MAIN appears in the message. Otherwise, the status indicates where the job is in relationship to the functions of main service processing.

```
FETCH          The job is waiting for volume fetch processing.
WAITVOL        The job is waiting for *S setup processing.
SYSTEM SELECT  The job is on the system select queue.
ALLOCATE       The job is waiting for resources to be allocated.
VOL UNAVAIL    The job is waiting for an unavailable volume.
VERIFY         The job is waiting for volumes to be mounted.
SYSTEM VERIFY  The job is on the system verify queue.
MDS ERROR      The job is waiting for an operator decision. (MDS error queue)
GMS SELECT     The job is waiting to be selected for processing on the main.
EXECUTING      The job is in execution.
BREAKDOWN      The job is waiting for its resources to be deallocated (MDS breakdown).
MDS RESTART    The job is waiting for MDS restart processing.
MAIN COMPLETE  The job is complete on main.
OUTSERV WAIT   The job is in the process of being rescheduled.
DEMAND SELECT  The job is a demand-select job that is waiting to be selected for main.
I/O WAIT       The ending function RESQUEUE is waiting for I/O to complete.
```

```
ENDFUNC ERROR   Error - the ending function RESQUEUE was not processed.
DSP ABEND       The MAIN scheduler element ended by JES3 failsoft processing.
```

*I S command

The ***I S** command to displays the status of jobs currently in setup or the status of volumes and data sets controlled by MDS.

```
*I S
IAT5619 ALLOCATION QUEUE    = 0000001  BREAKDOWN QUEUE = 0000000
IAT5619 SYSTEM SELECT QUEUE = 0000000  ERROR QUEUE     = 0000000
IAT5619 SYSTEM VERIFY QUEUE = 0000000  FETCH QUEUE     = 0000000
IAT5619 UNAVAILABLE QUEUE   = 0000000  RESTART QUEUE   = 0000000
IAT5619 WAIT VOLUME QUEUE   = 0000000  VERIFY QUEUE    = 0000000
IAT5619 ALLOCATION TYPE = AUTO
IAT5619 CURRENT SETUP DEPTH - ALL PROCESSORS = 0000000
IAT5619 MAIN NAME    STATUS      SDEPTH     DASD        TAPE
IAT5619 SC64      ONLINE NOTIPLD 020,000 05997,00000 00064,00000
IAT5619 SC70      ONLINE NOTIPLD 020,000 05997,00000 00064,00000
IAT5619 SC65      ONLINE    IPLD 020,000 05997,00000 00064,00000
```

*I B command

The ***I B** command to displays the number of jobs backlogged for each JES3 function (DSP), for a job class, for a job class group, for a service class, for a terminal group, or for a main or all mains.

```
*I B
IAT8688 FUNCTION  ACTIVE    WAITING
IAT8688 CI        00000000  00000004
IAT8688 INTRDR    00000002  00000000
IAT8688 ISDRVR    00000000  00000002
IAT8688 MAIN      00000063  00000004
IAT8688 MONITOR   00000001  00000000
IAT8688 NJECONS   00000001  00000000
IAT8688 NJERDR    00000001  00000000
IAT8688 OUTSERV   00001412  00000000
IAT8688 SNARJP    00000001  00000000
IAT8688 TCP       00000001  00000000
IAT8688 WTR       00000001  00000000
IAT8619 INQUIRY ON BACKLOG COMPLETE
```

*I G command

Use the ***I G** command to obtain the status of GMS components of JES3 and to display the name of the spool partition assigned for a specific main or all mains. A main's spool partition contains the spool data for each job that runs on that main unless other partitions were specifically assigned for the job's job class, SYSOUT data, or in the job's //*MAIN control statement.

Display the status of class A on all systems.

```
*I,G,ALL,C,A
IAT8934 CLASS - A        - STATUS=ON  - GRP=JES3TEST - SY2
IAT8934 CLASS - A        - STATUS=ON  - GRP=JES3TEST - SY1
IAT8934 CLASS - A        - STATUS=ON  - GRP=JES3TEST - SY3
```

8.12 MDS operator commands

- MDS operator commands
  - MDS queues and initialization parameters
    - *I S
  - MDS queues
    - *I S,options
    - *F S,options
  - Commands for jobs in MDS queues
    - *S S,jobno
    - *R S,jobno
    - *C S,jobno

MDS operator commands

The operator has many commands to determine the MDS options and to look at each of the MDS processing queues. The initialization options may be displayed and then modified.

*I S command

The `*I S` command to display the status of jobs currently in setup, counts of jobs in various MDS queues, the status of main processors, or the status of volumes and data sets controlled by MDS.

*F S command

The `*F S` command allows to:

- Make a volume unavailable for JES3 setup processing
- Make a volume available for JES3 setup processing
- Keep a real direct access volume mounted on a designated device. This command is also valid for devices containing SMS-managed volumes
- Unload a volume mounted on a designated device
- Specify automatic or manual allocation of JES3-managed devices. This command overrides the specification established by the ALLOCATE parameter on the SETPARAM initialization statement.
- Specify whether or not to include jobs that require only deferred mounts in the setup depth counts (SDEPTH).

- Determine if volumes required and presumed mounted for a designated job have been mounted. Use this command if all of the volumes have been mounted, but the expected responses, via JES3 volume verification, have not been recognized.
- Change the current number of asynchronous I/O requests that can be processed simultaneously.
- Specify whether scratch and specific tape requests and scratch requests of different media types are separated during high watermark processing.
- Specify whether or not all initial verify response messages are written to the hardcopy message log (MLOG).

*S, *R, *C setup commands

For jobs active in the MDS processing queues, the operator may issue start, restart, or cancel to the job.

The **`*S S`** command -- If manual allocation was specified during JES3 initialization or with the **`*F,S,AL=M`** command, the **`*S S`** command to allows a job to proceed to allocation processing. The **`*S S`** command is not required when automatic allocation was specified during JES3 initialization or by the **`*F S,AL=A`** command.

The **`*R S`** command to returns a job to the allocation stage (after volume fetch). The *RESTART,S command is used when a volume or device requested or needed by the job is unavailable. Generally, the **`*R S`** command can be used to return any job in the main device scheduling (MDS) processing to the MDS allocate queue. If a job is MVS restarting and the **`*R S`** command is issued to restart that job, the job becomes eligible to run on any main rather than only on the main where it was originally selected.

The **`*C S`** command to cancels a job currently being processed by main device scheduling (MDS).

8.13 MDS commands

```
*I S
 IAT5619 ALLOCATION QUEUE    = 0000001  BREAKDOWN QUEUE = 0000000
 IAT5619 SYSTEM SELECT QUEUE = 0000000  ERROR QUEUE     = 0000000
 IAT5619 SYSTEM VERIFY QUEUE = 0000000  FETCH QUEUE     = 0000000
 IAT5619 UNAVAILABLE QUEUE   = 0000000  RESTART QUEUE   = 0000000
 IAT5619 WAIT VOLUME QUEUE   = 0000000  VERIFY QUEUE    = 0000000
 IAT5619 ALLOCATION TYPE = AUTO
 IAT5619 CURRENT SETUP DEPTH - ALL PROCESSORS = 0000000
 IAT5619 MAIN NAME    STATUS      SDEPTH     DASD        TAPE
 IAT5619 SC64      ONLINE NOTIPLD 020,000 07801,00000 00112,00000
 IAT5619 SC70      ONLINE    IPLD 020,000 07801,00000 00112,00000
 IAT5619 SC65      ONLINE    IPLD 020,000 07801,00000 00112,00000
 IAT5619 DEFERCT=NO
 IAT5619 SDEPZERO=WAIT
```

MDS commands

The **`*I S`** command to display the status of jobs currently in setup, counts of jobs in various MDS queues, the status of main processors, or the status of volumes and data sets controlled by MDS.

The **`*I S`** command can also be used to display information for main processors, but only if SETUP is active in the complex. IBM suggests using the **`*I MAIN=`** command for this purpose instead of **`*I S`** as it does not depend on SETUP and displays more comprehensive information relevant to main processors.

DEFERCT=YES | NO

The current value of the DEFERCT option. This indicates whether or not JES3 is to include jobs that require only deferred mounts in the CLASS/SELECT SDEPTH counts.

SDEPZERO=WAIT | ERROR

The current value of the SDEPZERO parameter. This indicates whether jobs that require a tape mount, but are of a class that does not permit them, for example SDEPTH=0, should wait, for example, for a class or SDEPTH change, or be treated as if in error.

8.14 JES3 device concepts

- Types of devices
  - 1. Global "support" or JES3 devices -- **"JUNIT"**
    - Satisfy JES3 functions, such as DSP functions
  - 2. "Execution" devices -- **"XUNIT"**
    - JES3 allocates devices to MVS for user job execution
    - Permanently resident or reserved execution devices that are defined to JES3 and MVS are called **jointly-managed**
  - 3. Global "support" and "execution" devices
    - These are called **shared devices**

Global Devices -- "JUNIT"	Execution Devices -- "XUNIT"
Card Readers / Punches	DASD
Tapes	Tapes
Printers	Printers
Main Processors	
Remote Terminal Devices	

JES3 device concepts

Before JES3 can manage the use of an I/O device, the device must be attached to a main processor and it must be defined in the JES3 initialization stream. Devices are defined to JES3 with a DEVICE initialization statement. The DEVICE statement is used to define a device that JES3 can use as follows:

- To satisfy its own functions (JES3 device or global device).
- To satisfy the needs of a job (execution device).
- As a JES3 device or as an execution device (shared device).

Global devices

JES3 devices must be attached to the global processor. The only exception is a device driven by an output writer functional subsystem (FSS), which you may define as a JES3 device and attach to the global processor or a local processor. The JUNIT keyword is used to define the device that can be used by JES3 functions such as DSPs.

Execution devices

The DEVICE statement specifies device characteristics and tells JES3 how to use that device. The XUNIT parameter specifies the characteristics of a device attached to one or more mains (LPARs). If a device is shared between two or more mains, all four subparameters are specified for each main to which the device is attached. If a device is shared between channels of the same main, the SYSGEN primary address should be the only subparameter indicated.

Global and execution devices

Devices can be defined as both a global used device and an execution device. If JES3 functions are using the device, it is not available as an execution device and visa versa.

Defining devices

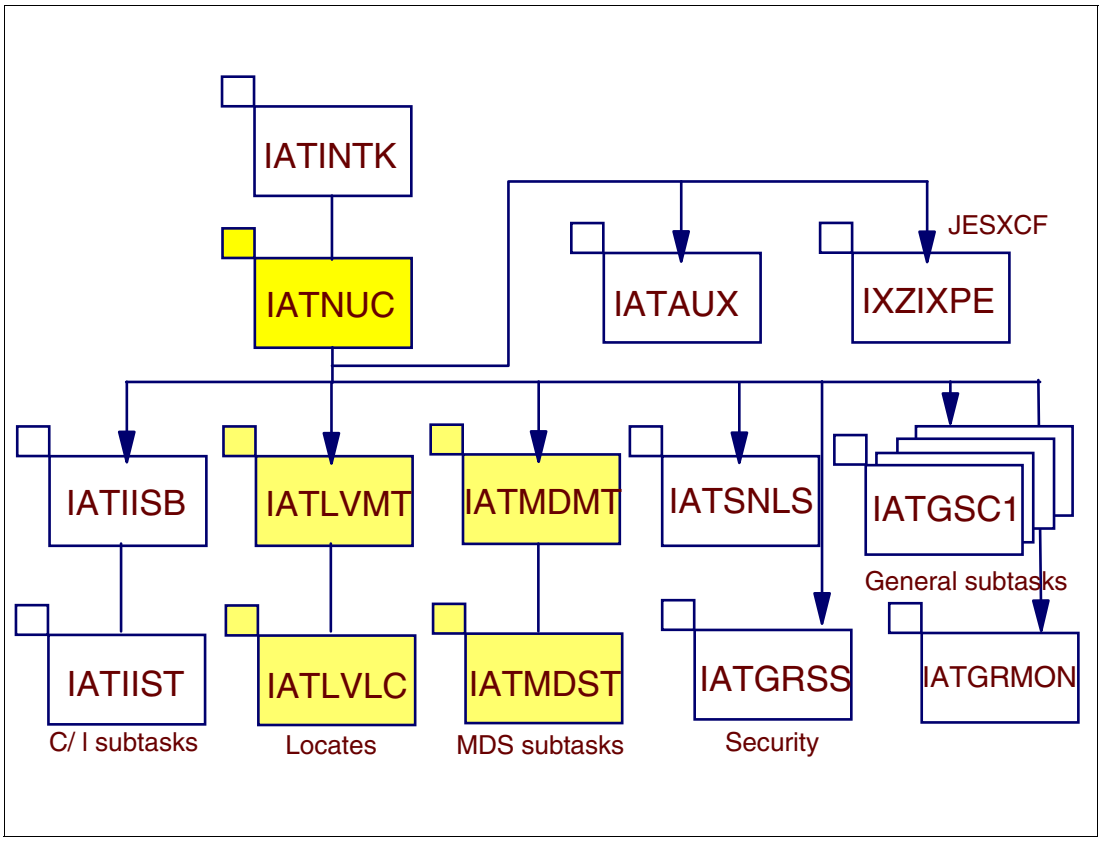
The ways to define a device to a JES3 system are:

- You can specify that JES3 is to use a device to satisfy JES3 functions, such as DSP requests, input processing, and output processing. This is called a JES3 device (aka support unit).
- You can specify that JES3 may allocate a device to MVS to execute user jobs. This is called an execution device.
- An execution device that is defined to JES3 and MVS as permanently resident or reserved is called a jointly-managed device.
- You can specify that JES3 can use the device as either a JES3 device or as an execution device. This is called a shared device.

JES3 devices must be attached to the global processor. The only exception is a device driven by an output writer functional subsystem (FSS), which you may define as a JES3 device and attach to the global processor or a local processor.

You can define as JES3 devices: main processors, network lines, printers, card punches, card readers, remote terminals, and tape drives.

8.15 JES3 task structure for MDS



JES3 MDS tasks

The MDSSRS FCT, like SETUP, resides permanently in the JES3 global address space along with an MDS master subtask attached by IATMDAT and several MDS subtasks attached by IATMDMT. The number of MDS subtasks changes, corresponding to the amount of work there is to do. The FCT is responsible for setting up and passing work to available subtasks. Each subtask is represented by an MDS control table used for communication between the FCT and the subtask.

Each LPAR has a locate master task and multiple locate subtasks. The locate master task is responsible for attaching new locate subtasks. The locate subtasks are responsible for performing the actual locates and each subtask is represented by a locate control table (LCT). The locate FCT maintains the locate master task and subtasks on that LPAR and is responsible for the following activities:

- Initializing new subtasks
- Terminating subtasks that are no longer needed
- Terminating subtasks for jobs that have been cancelled
- Cleaning up subtasks that have abended

The locate FCT on each LPAR is responsible for determining when to attach and detach locate subtasks. If the average number of waiting requests is greater than the number of locate subtasks attached, the locate FCT attaches more subtasks if a job requires one. The number that can be attached is one half of the difference between the average number of waiting requests and the number of subtasks attached.

8.16 JES3 MDS initialization statements

- MDS initialization statements such as:
  - DEVICE
  - SETACC
  - SETNAME
  - SETPARAM
  - SETRES
  - DYNALDSN
- MDS builds MDS-related tables, loading the resident MDS processing modules
- Attachs the MDS master task which attaches the MDS subtasks

MDS initialization statements

Main device scheduling initialization includes processing MDS-related initialization statements such as DEVICE, SETACC, SETNAME, SETPARAM, SETRES and DYNALDSN, building MDS-related tables, loading the resident MDS processing modules, and calling IATMDAT to attach the MDS master task which attaches the MDS subtasks.

DEVICE statement (Tape and DASD)

The DEVICE statement is used to define I/O devices to JES3. To define an execution device, specify the XTYPE and XUNIT parameters on the DEVICE statement. When any execution device is initialized, JES3 varies the device online or offline to MVS and JES3 based on the XUNIT parameter specification. You can attach execution devices to the global processor or to any local processor.

SETACC statement

Use the SETACC initialization statement to identify those mains that normally have access to a permanently resident direct-access volume. The SETACC statement identifies the location of a volume on the uninitialized mains in a JES3 complex. SETACC prevents JES3 from setting up a job that needs the mounted volume until the main is initialized. When all mains are initialized or the volume is found, the SETACC definition is no longer used and normal JES3 management of the volume and device occurs. The devices on which the volumes reside are defined on a DEVICE statement with an XTYPE parameter and PR subparameter.

SETNAME statement

Use the SETNAME initialization statement to specify all user-assigned names and device type names associated with MDS-managed devices.

SETPARAM statement

Use the SETPARAM initialization statement to specify parameters that the JES3 main device scheduler (MDS) and the DYNAL DSP uses in allocation, mounting, and deallocation of devices for jobs run on all mains. The SETNAME and DEVICE statements are used with the SETPARAM statements. SETNAME and DEVICE identify the devices to be managed by MDS. SETPARAM also indicates how MDS is to manage devices.

SETRES statement

The SETRES statement identifies frequently used direct-access volumes which are not permanently resident. The SETRES statement specifies volumes which may be on devices at main initialization time. When a specified volume is found to be present on an MDS-managed, removable, direct-access device during main initialization, the volume is considered mounted by MDS, without a MOUNT command being necessary.

DYNALDSN statement

Use the DYNALDSN statement to specify which data sets on permanently resident or reserved DASD volumes require data set awareness protection when the data set is dynamically allocated.

8.17 DEVICE initialization statement

DEVICE	**,JNAME =** name	JES3 device name
	,DTYPE =	JES3 device type
	,JUNIT = (denum,main,msgdest,ON/OFF)	JES3 device
	,XUNIT = (name.type,RM|PR)	Execution device
	,XTYPE = (denum,main,msgdest,ON/OFF)	Execution device

Types of DEVICE definitions for MDS:

- Define processor status
- Define I/O devices

Examples: JES3 device DTYPE		**Execution XTYPE**	
SYSMAIN	**RMTxxxx**	**CA**	**TA**
PRTxxxx	**TAyxxxx**	**GR**	**UR**
PUNxxxx	**username**	**DA**	
RDRxxxx	**NJELINE**		

DEVICE initialization statement

To define a JES3 device, specify the DTYPE, JNAME, and JUNIT parameters on the DEVICE statement. If applicable, you should also specify the DGROUP parameter and any printer or punch parameters. One DEVICE statement must be coded for each I/O device that JES3 uses or manages. Define devices as follows:

- To define a device as a shared device, specify the DTYPE, JNAME, and JUNIT parameters on the DEVICE statement for that device.
- To define an execution device, specify only the XTYPE and XUNIT parameters on the DEVICE statement for that device.
- To define a device as a shared device, specify the DTYPE, JNAME, JUNIT, XTYPE, and XUNIT parameters on the DEVICE statement for that device.

Execution devices

To define an execution device, specify the XTYPE and XUNIT parameters on the DEVICE statement. When any execution device is initialized, JES3 varies the device online or offline to MVS as well as to JES3 based on the XUNIT parameter specification. You can attach execution devices to the global processor or to any local processor.

DTYPE parameter

The DTYPE parameter specifies a device type being defined to JES3:

SYSMAIN This type DEVICE statement defines the initial status for a main.

NJELINE	This type of DEVICE statement defines a BSC line or a CTC connection to another node in a NJE network.
PRTxxxxx	Identifies a locally-attached printer. Specify PRTAFP1 for AFP1 printers. An AFP1 printer can be either channel attached or non-channel attached.
PUNxxxx	Identifies a locally-attached punch.
RDRxxxx	Identifies a locally-attached reader.
RMTxxxx	Identifies a remote terminal (described by an RJPTERM or RJPWS statement). For SNA RJP printers, specify RMTPRINT. For SNA RJP punches, specify RMTPUNCH.
TAxxxxx	Identifies a tape device.
username	Indicates a device type which is associated with a user DSP.

XTYPE parameter

The XTYPE=(name,devtype) parameter on a DEVICE statement specifies the characteristics of the JES3-managed or jointly managed device as it is used by jobs in execution. It must precede the XUNIT parameter, which is required if XTYPE is specified.

The *name* subparameter of the XTYPE parameter specifies a 1- to 8-character name that defines a device that can be referenced. It should match the name specified in the XTYPE parameter on a SETNAME initialization statement.

The values of *devtype* subparameter of the XTYPE parameter are:

- CA -- Specifies that the device is cartridge tape.
- TA -- Specifies that the device is reel tape.
- GR -- Specifies that the device is graphic.
- UR -- Specifies that the device is unit record.
- DA -- Specifies that the device is direct access. DASD devices can have attribute *removable volumes* (RM) whose mounting is to be controlled by MDS or *MVS permanently resident volumes* (PR).

Note: Devices within a specific XTYPE should have compatible characteristics. For a SNA-attached AFP printer, XTYPE is not a valid parameter.

XUNIT parameter

The XUNIT=(/devnums,main,msgdest,ON|OFF,....) parameter on a DEVICE statement specifies the characteristics of a device attached to one or more mains. If a device is shared between two or more mains, all four subparameters are to be specified for each main to which the device is attached unless the *ALL main name is used. complex. When *ALL is used, no other group of devnum,main,msgdest,OFF|ON can be used on the XUNIT parameter of this DEVICE initialization statement, and the values specified for devnum,main,msgdest,OFF|ON are the same for all mains.

Types of DEVICE definitions

The DEVICE initialization statements is used to define:

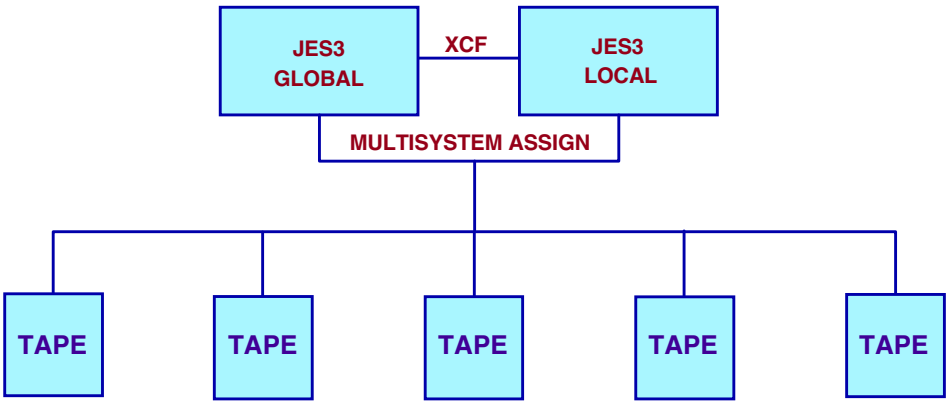
- Processor Status -- This form of the *optional* DEVICE statement defines the initial status of mains in a JES3 complex. If omitted, the processor in question is initialized online to every processor in the complex. For more information on defining mains. For example:

```
DEVICE,DTYPE=SYSMAIN,JNAME=SC64,JUNIT=(,SC64,,OFF,,SC70,,ON,,SC65,,ON)
DEVICE,DTYPE=SYSMAIN,JNAME=SC64,JUNIT=(,*ALL)
```

- I/O Devices -- This form of the DEVICE statement defines a device that JES3 can use to satisfy its own functions (JES3 device), to satisfy the needs of a job (execution device), or as a JES3 device or as an execution device (shared device).

8.18 Defining tape devices

- Tapes are serially reuseable -- not sharable
- JES3 provides the seialization that makes them "sharable"
  - Tapes can be online all time for all systems in JES3 complex
  - MDS keeps complex wide track of tape drives allocation



Defining tape devices

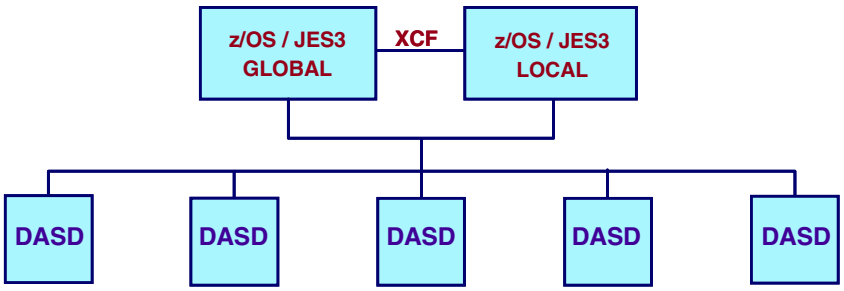
At any one time, a single tape device can be used only by one processor. The processor assignment hardware feature prevents a tape drive from being used by another host processors in shared configuration. If no assign commands have been issued to a tape drive, any attached host processor may use that drive. Drive assign and unassign commands are issued by the MVS operating system when a tape drive is brought online or is taken offline. In non-JES3 multisystem complex operators must coordinate to which system tape drives are online.

A tape drive assigned to more than one processor is called multisystem assigned. JES3 uses multisystem assign when it brings tape drives online. Thus all tape drives can be simultaneously online to all mains in the JES3 complex. MDS keeps complex wide track of the defined tape drives' allocation status and lets only those jobs go to execution that have all tape allocation requested satisfied. If a job's all allocated tape drives have connectivity to several mains in the JES3 complex, the job can be sent to execution to any of these processors.

Note: The MVS automatic tape switching (ATS) implements the tape sharing within a sysplex without using multisystem assign. When a tape device is defined as automatically switchable, the device must be in a offline state. The VARY device,AUTOSWITCH,ON operator command marks a tape device autoswitchable. The device (UCB) is set online, but it is not assigned. The device will be assigned during MVS allocation for the system where a job requests tape drives. When the job unallocates the tape drives, MVS unassigns the devices.

8.19 Defining DASD devices

- DASD devices are sharable
- Complex wide data sets integrity by JES3 and MVS
  - JES3 integrity -- Data set name and volume serial
  - MVS (GRS) integrity -- Data set name only
- JES3 manages requests for SMS-managed data sets
  - Unless SMSSETUP=NO set on SETPARAM statement
  - JES3 is not aware of units for SMS-managed data sets
  - SMS managed devices do not require DEVICE statements



Defining DASD devices

DASD devices are sharable. Multiple jobs may access the same device while executing.

Data set awareness

The JES3 main device scheduler fetch processing scans jobs' JST entries and builds a SETVOL table entry (IATYSET) for each of the volumes required by the job. For SMS-managed data sets, JES3 is not concerned with and does not know about the volumes where the data sets reside. However, the fetch routine still creates a dummy SETVOL table entry since MDS is responsible for complex wide data set awareness. The fetch routine creates the SETVOL entry using a dummy volser. This allows the SMS-managed data sets to be distributed over a number of SETVOL entries. If a volume is not mounted, the fetch routine issues a fetch message to the operator. The SETVOL table entries point to the SETDSN entries (IATYDS) for the data sets on the volume.

JES3 enforces data set awareness for:

- Data sets that are requested through DD statements that require JES3-managed devices
- Data sets that are requested dynamically that require JES3-managed devices
- JES3-managed sequentially accessed devices that are attached to one or more processors
- Data sets that are SMS-managed, unless SMSSETUP-NO is specified on the SETPARAM statement.
  - JES3 is not aware of units for SMS-managed data sets:

```
*I S DE=VAINI.SMS.MANAGED.DATA
IAT5674 DATASET IS SMS MANAGED
IAT5610  AL=0000001 DSN=VAINI.SMS.MANAGED.DATA
IAT5612   JOB VAINI    (JOB24889) AL=YES S-VAINI.SMS.MANAGED.DATA
*I S DE=VAINI.NONSMS.DATA
IAT5606 SER=VAINIJ DA J=0000022 AL=0000022        ONADEV
IAT5610  AL=0000001 DSN=VAINI.NONSMS.DATA
IAT5612   JOB VAINI    (JOB24889) AL=YES S-VAINIJ S-VAINI.NONSMS.DATA
```

The JES3 data set awareness processing is based on the SETVOL/SETDSN table structure and the serialization is *by data set name on a volume*. Since MVS and JES3 data set awareness processing are using different convention for serialization, JES3 may allow a job, allocating a data set with a specific volume request, to go to MVS execution when a data set name is already serialized in the MVS terms.

MVS serialization

In a sysplex MVS serializes sysplex wide (ENQ scope SYSTEMS) the access to data sets. During a job's first step initialization MVS allocation serializes access to all data sets referenced by the job. If the a data set is allocated with DISP=SHR, the serialization is for shared access, otherwise (DISP=OLD/MOD/NEW) the access is exclusive. The MVS data set serialization is *by data set name* only.

SMS-managed data sets

JES3 manages all requests for SMS-managed data sets unless SMSSETUP=NO is specified on the SETPARAM initialization statement. However, JES3 is not aware of specific units for SMS-managed data sets. JES3 treats requests for these data sets as single requests and sends them through dynamic allocation - fast path. All dynamic allocation requests for SMS-managed data sets that require more than one SIOT, such as GDG-all requests, will be sent through MDS for processing instead of dynamic allocation - fast path.

8.20 Generic and esoteric I/O device names

Generic Device Type	3490				3380				3390			
Esoteric Names (Defined by the Installation)					SYSDA		SYSDA		SYSDA		3390A	
Device Numbers	131	132	133	134	181	182	183	184	191	192	193	194	195
Subgeneric Groups	1				2	3	2		4		5		

Generic and esoteric I/O device names

You use the Hardware Configuration Definition (HCD) program to add, delete, or change JES3 defined devices. These can be JES3 global devices (JUNIT), execution devices (XUNIT), or shared devices (both JUNIT and XUNIT). When making these changes, be careful not to introduce subgeneric splits, define devices to JES3 but not to MVS, or define devices as one device type to JES3 but a different device type to MVS.

If you make a mistake, JES3 will tolerate the error and initialize (if there are no other errors with higher impact); however, you should correct the error and perform a hot start with refresh at your earliest convenience.

An eligible device table (EDT) is an installation-defined and named representation of the I/O devices that are eligible for allocation. Using HCD, you define EDT information in an IODF. At IPL or dynamic configuration, information in the IODF and UIMs is used to build the EDT.

Esoteric device groups

The execution devices defined to JES3 should be grouped using esoteric names. An esoteric (or esoteric device group) is an installation-defined and named grouping of I/O devices of usually the same device group. EDTs define the esoteric and generic relationship of these devices. The name you assign to an esoteric can be used in the JCL DD statement. The job then allocates a device from that group instead of a specific device number or generic device group.

Generic names are provided by the unit information modules (UIMs) included in hardware configuration definition (HCD). Generic (or generic device type) is an MVS-defined grouping of devices with similar characteristics. Every generic has a generic name that is used for device allocation in the JCL DD statement. MVS interprets this name as “take any device in that group”.

To request allocation of a device from an esoteric device group, specify the esoteric name on the UNIT= parameter of a JCL DD statement. (In JCL terminology, an esoteric name is called a group name.)

Subgeneric groups

MVS allocation divides generic device types into subgeneric groups. The subgeneric groups allows MVS allocation to serialize a subset of units within a generic name. For example, using the figure, if 3390A is requested, MVS allocation needs to serialize only subgeneric group 5 rather than all 3390 devices. As a result, more than one allocation can process the same generic device type, as long as the allocations require different subgeneric groups within that generic.

The guidelines by which MVS determines subgeneric groups are:

- If an esoteric name (for example, 3390A) includes only a subset of the units in a generic name, that subset is a *subgeneric group*. (For 3390A in the visual, units 193, 194, and 195 belong to a subgeneric group.)
- If an esoteric (for example, SYSDA) includes different generic device types, the units in each generic name belong to different subgeneric groups. (For SYSDA in the visual, units 181, 183, and 184 belong to one subgeneric group; units 191 and 192 belong to another subgeneric group; and units 193, 194, and 195 belong to a third group.)
- The units not contained in the intersection of a generic group and its new subgeneric groups constitute a subgeneric group. (For SYSDA and 3380 in the visual, unit 182 comprises a new subgeneric group.)

If one unit of a subgeneric group is defined to JES3, all units of the subgeneric group must be defined to JES3. (For subgeneric group 2 in the visual, if unit 183 is assigned to JES3, units 181 and 184 must also be assigned to JES3.) Thus, a subgeneric group cannot have system-managed devices mixed with JES3-managed and jointly managed devices.

8.21 Grouping I/O devices

```
* 3490 -- Tape
DEVICE,DTYPE=TA33490,JNAME=TAPE,JUNIT=(131,*ALL,S1,OFF),NUMDEV=4,
XTYPE=(D13490,CA),XUNIT=(131,*ALL,S1,ON)
SETNAME,XTYPE=D13490,NAMES=(3480)
* 3380 SYSDA -- DASD
DEVICE,XTYPE=(D23380,DA,PR),XUNIT=(181,*ALL,S1,ON)
DEVICE,XTYPE=(D23380,DA,PR),XUNIT=(183,*ALL,S1,ON),NUMDEV=2
* 3380 -- DASD
DEVICE,XTYPE=(D33380,DA,PR),XUNIT=(182,*ALL,S1,ON)
* 3390 SYSDA -- DASD
DEVICE,XTYPE=(D43390,DA,PR),XUNIT=(191,*ALL,S1,ON),NUMDEV=2
* 3390 SYSDA 3390A -- DASD
DEVICE,XTYPE=(D53390,DA,PR),XUNIT=(193,*ALL,S1,ON),NUMDEV=3
SETNAME,XTYPE=D23380,NAMES=(3380,SYSDA)
SETNAME,XTYPE=D33380,NAMES=(3380)
SETNAME,XTYPE=D43390,NAMES=(3390,SYSDA)
SETNAME,XTYPE=D53390,NAMES=(3390,SYSDA,3390A)
```

Generic Device Type	3490				3380				3390				
Esoteric Names					SYSDA				SYSDA				
											3390A		
Device Numbers	131	132	133	134	181	182	183	184	191	192	193	194	195
Subgeneric Groups	1				2	3	2		4		5		

Grouping I/O devices

When the devices assigned to an esoteric or generic name are to be managed by JES3, one or more of the subgeneric groups that constitute that name must be defined to JES3 via DEVICE initialization statements. For example, SYSDA in the visual is composed of subgeneric groups 2, 4, and 5. For SYSDA devices to be managed by JES3, at least one of the subgeneric groups, 2, 4, and 5 must be defined to JES3. For SYSDA devices in subgeneric group 4 (191 and 192), or all the devices in subgeneric group 2 (181, 183, and 184), all the devices must be defined to SYSDA on the NAMES parameter of the SETNAME initialization statement. Thus, an esoteric or generic name may comprise JES3-managed, jointly managed, and system-managed devices. Since a subgeneric group can belong to different generic or esoteric names, it is possible that the subgeneric group could be managed both by JES3 and MVS. For example, 3390A in the visual might be MVS-managed, whereas subgeneric group 5 might be JES3-managed. If a device is being managed both by JES3 and by MVS, MVS does not allocate to any device that does not have a permanently resident volume (jointly managed). Sample JES3 DEVICE statements for each subgeneric:

```
* 3490 -- Tape
DEVICE,DTYPE=TA33490,JNAME=TAPE,JUNIT=(131,*ALL,S1,OFF),NUMDEV=4
       XTYPE=(D13490,CA),XUNIT=(131,*ALL,S1,ON)
SETNAME,XTYPE=D13490,NAMES=(3480)
* 3380 SYSDA -- DASD
DEVICE,XTYPE=(D23380,DA,PR),XUNIT=(181,*ALL,S1,ON)
DEVICE,XTYPE=(D23380,DA,PR),XUNIT=(183,*ALL,S1,ON),NUMDEV=2
* 3380 -- DASD
DEVICE,XTYPE=(D33380,DA,PR),XUNIT=(182,*ALL,S1,ON)
* 3390 SYSDA -- DASD
```

```
DEVICE,XTYPE=(D43390,DA,PR),XUNIT=(191,*ALL,S1,ON),NUMDEV=2
* 3390 SYSDA 3390A -- DASD
DEVICE,XTYPE=(D53390,DA,PR),XUNIT=(193,*ALL,S1,ON),NUMDEV=3
SETNAME,XTYPE=D23380,NAMES=(3380,SYSDA)
SETNAME,XTYPE=D33380,NAMES=(3380)
SETNAME,XTYPE=D43390,NAMES=(3390,SYSDA)
SETNAME,XTYPE=D53390,NAMES=(3390,SYSDA,3390A)
```

XTYPE parameter

In the sample, the two first characters of the XTYPE name are used to refer to subgeneric groups. For example, the XTYPE name for tape units is **D1**3490, where the two first characters (D1) are used to reference subgeneric group 1 on the visual and for the rest of name is used the generic device type.

The XTYPE=(name,type,PM|PR) parameter of the DEVICE statement specifies the characteristics of the JES3-managed or jointly managed device as it is used by jobs in execution. It must precede the XUNIT parameter, which is required if XTYPE is specified.

Where:

name - specifies a name that defines a device that can be referenced. It should match the name specified in the XTYPE parameter on a SETNAME initialization statement.

type:

- **CA** Specifies that the device is cartridge tape.
- **TA** Specifies that the device is reel tape.
- **GR** Specifies that the device is graphic.
- **DA** Specifies that the device is direct access.
- **UR** Specifies that the device is unit record.

RM - Specifies that the device will contain removable volumes whose mounting is to be controlled by MDS.

PR - Specifies that the device will contain MVS permanently resident volumes.

SETNAME initialization statement

The SETNAME initialization statement specifies all user-assigned names and device type names associated with MDS-managed devices. The NAMES parameter identifies all names (user-assigned and device type) used to refer to the device defined in the associated DEVICE statement. NAMES= indicates what names are used to specify devices in the UNIT parameter of DD statements.

8.22 Using *ALL in XUNIT and JUNIT definitions

- XUNIT= (/devnum,*ALL,msgdest,ON|OFF)
- JUNIT= (/devnum,*ALL,msgdest,ON|OFF)
  - *ALL defines a device to all main processors
  - Definition within XUNIT and JUNIT parameters
  - No change to DEVICE statements when MAINPROCs are added
  - DEVICE,DTYPE=SYSMAIN can specify
    - JUNIT=(NONE,*ALL,,ON)
    - Can be omitted to get this definition as a default
  - MAINPROC is still required

Using *ALL in XUNIT and JUNIT parameter

The XUNIT= and JUNIT= parameter of the DEVICE statement has the same syntax:

`(devnum,main,msgdest,OFF|ON)`

main Specifies a processor which can use this device as a JES3 global device when that main is global. This name must be the same name as that defined in the NAME parameter on a MAINPROC initialization statement. The device must be attached to this main at the address specified by /devnum or devnum. Alternatively, a main name of *ALL can be used. Using *ALL indicates that all processors in the complex are eligible to use this device as a JES3 global device when the processor is the global. When *ALL is used, other group of *devnum,main,msgdest,OFF|ON* cannot be used on the JUNIT parameter of this DEVICE initialization statement, and the values specified for *devnum,main,msgdest,OFF|ON* are the same for all mains.

Specifying *ALL

Using *ALL on a device is particularly useful if you will be adding MAINPROCs later, as you do not need to add the new main name to any DEVICE statement that uses *ALL. The following rules apply when using *ALL:

- Mixing *ALL and system names within the same JUNIT or XUNIT is not allowed.
- *ALL must appear only once within the same JUNIT or XUNIT.
- If *ALL is used on the JUNIT and the device has both a JUNIT and an XUNIT, *ALL also must be used on the XUNIT. If *ALL is used on the XUNIT, the JUNIT can list processors, but the use of *ALL on the JUNIT is suggested.

8.23 Defining a range of devices - NUMDEV parameter

- **NUMDEV=nnnn** parameter on DEVICE statement
  - JNAME parameter specifies a prefix
    - A four digit device number based on the JUNIT and NUMDEV is concatenated with the prefix to form a omplete JNAME.
    - If the JUNIT is three digit, it is left padded with 0
    - Prefix cannot exceed four characters
    - NUMDEV=1 is not the same as omitting NUMDEV
  - NUMDEV cannot be 0
  - NUMDEV cannot exceed 65,535 or cause a device number range to exceed FFFF
  - NUMDEV requires JUNIT with JNAME and/or XUNIT

Defining a range of devices

NUMDEV=nnnn parameter on DEVICE statement specifies the number of devices to be defined by this DEVICE statement, starting with the specified JUNIT. For example, if a DEVICE statement defines a JUNIT of 140 and NUMDEV=32, the statement defines 32 devices with device numbers of 140 through 15F. The NUMDEV parameter requires at least one JUNIT with a device number (that is, a device number other than NONE).

When NUMDEV is used, the JNAME parameter specifies a prefix rather than a complete JNAME. A four digit device number is built based on the JUNIT and NUMDEV and concatenated with the prefix to form a complete JNAME. For example, if the JUNIT is 140, the specified JNAME is LINE, and NUMDEV=32, the statement will define 32 JNAMEs, LINE0140 through LINE015F.

If the JUNIT combines mains with different device numbers (for example, JUNIT=(140,SY1,,ON,8C0,SY2,,ON), the first specification is used to build the JNAMEs (140 is this case).

If the JUNIT combines mains with device numbers and NONE (for example, JUNIT=(NONE,SY1,,ON,8C0,SY2,,ON), the first group with an actual device number is used to build the JNAMEs (8C0 in this case).

Note: The NUMDEV parameter requires at least one JUNIT or XUNIT with a device number. For example, NUMDEV is not valid on a DTYPE=SYSMAIN or a VTAM-attached FSS printer.

Referenced JNAME

If the device is a JES3 global device, the JNAME parameter specifies a prefix rather than a complete JNAME. A four digit device number is built based on the JUNIT and NUMDEV and concatenated with the prefix to form a complete JNAME. For example, if the JUNIT=3A0, the specified JNAME is TAPE, and NUMDEV=32, the statement will define 32 JNAMEs of TAPE03A0 through TAPE03BF. Because of converting JNAME this way, omitting NUMDEV for JES3 global devices is not the same as specifying NUMDEV=1.

If the JUNIT combines mains with different device numbers, for example, JUNIT=(3A0,SY1,,ON,9C0,SY2,,ON), the first specification is used to build the JNAMEs (3A0 in this case).

If the JUNIT combines mains with device numbers and NONE, for example, JUNIT=(NONE,SY1,,ON,9C0,SY2,,ON), the first group with an actual device number is used to build the JNAMEs (9C0 in this case).

Since the JUNIT part of the JNAME is padded with 0's if it is a three digit device, be careful to use the correct JNAME if you reference a JNAME on any other initialization statement (for example, FSSDEF or SYSOUT).

NUMDEV rules

When using NUMDEV for a global device, the JNAME specifies a prefix. JES3 combines the prefix with each JUNIT in the requested range as a four digit device number to construct JNAMEs for all devices defined by this statement. In the following example, the JNAMEs for the printers defined by the statement are AFPP0803, AFPP0804, AFPP0805, AFPP0806, and AFPP0807. The prefix cannot exceed four characters.

```
DEVICE,DTYPE=PRTAFP1,JNAME=AFPP,FSSNAME=FSSAFP1,
JUNIT=(803,SY1,S1,OFF,803,SY2,S2,ON),XTYPE=(PRTAFP1,UR),
XUNIT=(803,SY1,S1,OFF,803,SY2,S2,OFF),WS=(D),
DGROUP=ARM1,PM=(LINE,PAGE),MODE=FSS,NUMDEV=5
```

NUMDEV cannot be zero, and cannot cause the range to exceed FFFF. If NUMDEV is 1, the JNAME specifies a prefix just as it does for larger values.

If you define an FSS managed printer without using the FSSNAME= parameter, the FSS name is the same as the constructed JNAME. If you use the FSSNAME= parameter, the FSS name is the specified FSSNAME.

If a JNAME of a device defined using the NUMDEV statement is referenced anywhere else in the initialization stream (for example, on the DEST= keyword of a SYSOUT statetement) make sure that the referenced name is the same as the constructed name.

At least one XUNIT or channel attached JUNIT is required. For example, NUMDEV cannot be used on a DEVICE statement for a 3820 printer. JNAME cannot be specified if JUNIT is not specified.

If the device is defined with different device numbers on different processors, each processor gets a range of the same number of devices starting with the specified unit. The unit used to build the JNAME is the first channel attached unit that appears in the JUNIT parameter.

8.24 JES3 device tables

Device Type	Control Block	Location
"Support" - dtype	SUPUNITS	JES3 Private
"Execution" - xtype	SETUNITS	CSA
All Devices	SYSUNITS	CSA

- JES3 device allocation
  - "Support" devices via GETUNIT macro
    - JSS when Scheduling DSP
    - DSPs via their FCT
  - "Execution" devices via MDS

JES3 device tables

JES3 MDS device allocation uses five tables: SYSUNITS, SETUNITS, SETNAME, SETDSN, and SETVOL.

JES3 GETUNIT macro

Module IATGRGU services requests for JES3 global device allocation (GETUNIT) and unallocation (PUTUNIT). GETUNIT/PUTUNIT service uses the SYSUNITS table which provides information on the current status of global devices. JSS issues the macro when scheduling JES3 DSPs that require devices during their execution.

SUPUNITS table

The SUPUNITS table (IATYSUP) is created for JES3 devices (DTYPE, JNAME, and JUNIT parameters on the DEVICE statement) and provides information on the current JES3 status of the global devices.

The SUPUNITS table entries identify the devices that are allocated to the JES3 global. These devices are used by JES3's support services (i.e. ‘readers, printers, tape units, RJP lines and networking lines).

The SUPUNITs table is initialized only on the global main and contains entries for each defined JES3 device that names the current global processor in the JUNIT specification.

SYSUNITS table

The SYSUNITS table (IATYSYS) contains JES3 managed device allocation status for the entire JES3 complex. SYSUNITS table entries include a pointer to the SETUNITS entry for each main processor, SETVOL entry address of current volume, masks of mains to which the device is attached and online, and the current allocation status.

Separate entries exist for the same device when it is shared by two or more mains.

SETUNIT tables

The SETUNITS tables (IATYSET) contains an entry for each JES3 managed devices attached to a main. Each defined main has its own SETUNITS table. The SETUNIT table contains information representing device status on a particular main processor. The status information includes the RQ address of the job using the device.

The SETUNITS table is generated to contain control information for all devices attached to a main that can be set up by MDS from the global main. The data for the table originates from the XUNIT and XTYPE parameters of the DEVICE initialization statement.

SETVOL and SETDSN tables

The SETVOL (IATYVLM) table is generated to maintain information regarding all known volumes requirements for jobs in the system and to track the status of currently mounted volumes. The source of data includes DD statements, automatic verification (by JES3 VERIFY), and operator commands.

The information kept in the SETVOL table entry includes pointers to the SETDSN table.

SETDSN table is created to represent all data sets allocated to volumes. It is used in conjunction with the SETVOL table to ascertain when a volume is no longer in use.

SETDSN table entries (IATYDSN) are used by MDS to control the serialization and shareability of a particular data set.

SETNAMES table

The SETNAMES table (IATYNAM) correlates the device types specified by the JCL UNIT parameter and the devices satisfying that designation.

The SETNAMES table is generated from parameters on the SETNAME statements. The data is used to identify a name than can be used in the UNIT parameter of a DD statement for a device represented in a DEVICE initialization statement.

Note: A JES3 formatted dump includes SUPUNITS, SUPUNITS, SUPUNITS, SETVOL, SETDSN and SETNAMES information.

8.25 SETNAME initialization statement

- **SETNAME,XTYPE**=devicetype,**NAMES=**(name,...)
- **SETNAME,XTYPE**=devicetype,**POOLNAMS=**(pool,...)
  - XTYPE= indicates a name which matches the name in the XTYPE parameter on a DEVICE statement
  - NAMES= identifies all names (generic&esoteric) used to refer to the device defined in the associated DEVICE statement
  - POOLNAMS= specifies name(s) that dedicated only for devices used to job class groups or dependent job control jobs
- **JES3 manages request**
  - If unitname via UNIT = appears on SETNAME

```
SETNAME,XTYPE =D3390,NAMES = (3390,SYSDA,.....
```

```
//MYDD   DD   DSN = MYDSN,UNIT = SYSDA,DISP = OLD,
//                VOL = SER = MYDISK
```

SETNAME Initialization Statement

The JES3 MDS algorithm uses the information specified on the SETNAME statement when searching for the proper device to be allocated for a DD request.

If a DD request is to be handled by JES3, the value (excluding the specific device number) specified in the UNIT parameter of the DD statement must also be specified in the NAMES parameter of the SETNAME statement. If the request specifies a device number in the UNIT parameter, the device number must also be specified in the XUNIT parameter of the DEVICE statement.

SETNAME statement parameters

```
SETNAME,XTYPE=devicetype,NAMES=(name,...)
```

The following applies to SETNAME parameters:

- The XTYPE parameter indicates the name which matches the first operand in the XTYPE parameter on a DEVICE statement. The XTYPE parameter associates this statement with a *specific* device definition indicated on a DEVICE statement. Devices within a specific XTYPE should have compatible characteristics.
- The NAMES parameter identifies all names (user-assigned and device type) used to refer to the device defined in the associated DEVICE statement. The NAMES parameter indicates what names are used to specify devices in the UNIT parameter of any DD statements. Note that device type names must be included if allocation of cataloged data sets that reside on these device types is done by JES3 even though the devices are not directly referred to in the UNIT parameter of a DD statement.

- POOLNAMS parameter specifies the name(s) that may be used only for dedicating devices to job class groups or dependent job control jobs. The pool names specified may be used in the DEVPOOL parameter of the GROUP statement and/or the //*NET statement. These names may not be used in the UNIT parameter of the DD statement.

Note: The MVS device generic names must be included data sets are to be allocated using cataloged information. This even though direct reference to the devices using the generic names are not made in the UNIT parameter of allocating DD statements.

There must be a SETNAME statement for the IBM generated esoteric names SYSALLDA, SYS3480R, and SYS348XR in the initialization deck if these values are to be coded or defaulted for allocations which are to be JES3 manages.

If a dynamic allocation is made for a time-sharing user and a UNIT parameter is not included in the DD statement, the UNIT parameter is obtained from the time-sharing user attribute data set (UADS). If the UADS does not contain a UNIT parameter, or if the user is not a time-sharing user, an MVS default of SYSALLDA (that is, all direct access devices) is assumed.

It is recommended that a default UNIT parameter value be defined in the UADS or RACF user profile's TSO segment, in the ALLOCxx parmlib member, and that all dynamic allocations specify a UNIT parameter. Otherwise, if JES3 is to manage the dynamic allocations that do not have a UNIT parameter, the MVS default UNIT value (SYSALLDA) must be included in SETNAMES for all DA device types.

JCL example

The JCL DD statement in the figure requests UNIT=SYSDA. This UNIT=SYSDA must then appear on a SETNAME statement in the initialization stream. The SETNAME statements that have this UNIT= have a corresponding XTYPE that will indicate which devices are eligible to satisfy the DD request.

In other words, if a DD request is to be handled by JES3, the value (excluding the specific device number) specified in the UNIT parameter of the DD statement must also be specified in the NAMES parameter of the SETNAME statement. If the request specifies a device number in the UNIT parameter, the device number must also be specified in the XUNIT parameter of the DEVICE statement.

8.26 HWSNAME initialization statement

- **HWSNAME,TYPE=** (*devname,altnam....*)
  - *devnames* and *altnames* specified on HWSNAME statement must be defined on the SETNAME statement
  - *devname* identifies a device type for high watermark setup
  - *devname* - any generic or esoteric name associated with the specified unit name(s).
  - *altnam* specifies a list of generic or esoteric device names
    - *altnam* lists alternate units to be used in device selection
    - Order of the *altnam* names is the allocation attempt order
    - When a device is selected, *altnam* search ends

```
HWSNAME,TYPE=(SYS3480R,SYS348XR,3490)
//STPA EXEC PGM=IEFBR14
//DDA1 DD DSN=A,UNIT=SYS3480R,VOL=SER=V1,DISP=OLD
//STPB EXEC PGM=IEFBR14
//DDB1 DD DSN=B,UNIT=3490,VOL=SER=V2,DISP=OLD
The same tape drive allocated to DDA1 step STPA and DDB1 step STPB
```

HWSNAME initialization statement

The HWSNAME statement is used to define, to JES3, all names by which users can reference a given device type to enable high watermark setup (HWS) processing. The statement identifies the characteristics of each device name for the specific JES3 complex. This statement can be used to define which device names are subsets of other device names. In general, the fewer the number of alternate names, the more restrictive the device name being defined. This ensures that initial allocation for devices that are reused from step to step is the most restrictive device. It also ensures that attempts to override passed or cataloged unit names are processed correctly. Non-HWS users are encouraged to supply HWSNAME information to take advantage of this function. The HWSNAME statement follows:

```
HWSNAME,TYPE=(devname,altnam....)
```

TYPE= specifies the name(s) of a device type that is valid for high watermark setup.

devname Specifies any user-supplied or IBM-supplied group name associated with the specified unit name(s). The identifies a device type valid for high watermark setup.

altnam Specifies a list of valid user-supplied or IBM-supplied device names. These are alternate units to be used in device selection. The order of these names is the order in which allocation is attempted; when a device is selected, no search for a later alternate is made.

altnam specification

Special care must be taken when specifying alternate names. The alternate device must be compatible, for MVS allocation purposes, with the device specified by the devname subparameter. Thus, you may not specify a 3390 DASD as an alternate name for a 3330 DASD, and you may not specify 3330 as an alternate name for 3390. Similarly, you may not specify a 2400-series tape drive as an alternate for a 3400-series tape drive. You may, however, specify a 3400-series tape drive as an alternate for a 2400-series tape drive.

HWS benefits

High watermark setup, as defined on the HWSNAME initialization statement, reduces the number of devices JES3 reserves for a job. To determine how many devices of a particular type to reserve for a job, JES3 considers the needs of each of the job's steps. In this way, JES3 determines which step needs the greatest number of devices of that type; JES3 then reserves that many devices of that type for the job. JES3 repeats this process for each device type the job needs. As a result, high watermark setup can cause premounting of all mountable volumes. Volume unloading and remounting may occur for both private and public volumes, even when RETAIN has been specified on the applicable DD statement.

When high watermark setup is used, as in job setup, devices, volumes, and data sets are returned to JES3 for use by other jobs as soon as the DD statement is deallocated in the last step using the resources. When it is advantageous to use fewer devices for a job, high watermark setup is preferable to job setup.

STANDARDS statement specification

High watermark setup is used when SETUP=THWS (for tapes only), SETUP=DHWS (for disks only), or SETUP=HWS (for tapes, disks, graphics, and unit-record devices). If the SETUP parameter is not specified on a //*MAIN statement, then high watermark setup is used only when SETUP=THWS, SETUP=DHWS, or SETUP=HWS is specified on the STANDARDS initialization statement or when the specified setup is overridden by installation exit IATUX08.

Cataloged data sets may reference devices that are valid for use during HWS. You define these devices using the HWSNAME initialization statement. If a cataloged data set references a device that has been removed as a HWS device (that is, the device has been removed from the HWSNAME initialization statement), that data set is no longer eligible for HWS processing.

The STANDARDS statement specifications are:

```
SETUP=JOB |NONE | DHWS | THWS | HWS
```

8.27 MDS initialization parameters

```
STANDARDS,....
        ,SETUP = NONE | JOB | DHWS | THWS | HWS,..
          -  SETUP= Specifies the type of setup processing
          -  Provides default for SETUP parameter on  //*MAIN JECL
          -  SETUP = NONE   -   No Main Device Scheduling

SETPARAM,....
        ,FETCH = NO | YES
        ,TAFETCH = (S1,S1)
        ,ALLOCATE = MANUAL | AUTO
        ,ADDRSORT= YES | NO
        ,MTJESMSG= FETCH | ALLOC |  BREAKDWN | ALL | NONE
        ,PRJESMSG= FETCH | ALLOC | ALL | NONE
        ,REMOUNT= nnn
```

MDS initialization parameters

The STANDARDS initialization statement specifies default values for information not provided on other JES3 initialization statements. It also provides standards to be applied to all jobs entering the system.

The SETPARAM initialization statement specifies parameters that the JES3 main device scheduler (MDS) and the DYNAL DSP uses in allocation, mounting, and deallocation of devices for jobs run on all mains. The SETNAME and DEVICE statements are used with the SETPARAM statements. SETNAME and DEVICE identify the devices to be managed by MDS. SETPARAM also indicates how MDS is to manage devices.

SETUP parameter of the STANDARDS initialization statement

The SETUP= parameter of the STANDARDS initialization statement indicates the system standard for allocation of devices identified by the NAME parameter of SETNAME statement(s). The SETUP= parameter specifies the type of setup processing, such as job setup, tape high watermark setup, or disk high watermark setup. This parameter also provides the default for the SETUP parameter on the //*MAIN JES3 statement and also, SETUP=NONE overrides the SETUP parameter on the //*MAIN statement.

NONE Specifies that no pre-execution setup is to occur. All devices are allocated, mounted, and deallocated by MVS; the SETPARAM statement is ignored. The SETUNITS and SETNAME tables are created even though MDS processing does not occur.

JOB Specifies that pre-execution setup is to occur by job for those devices (indicated in the UNIT parameter of DD statements for the job) which are identified in the

SETNAME statement. All devices for the job are allocated to the job from first step to last. This type of setup improves job turnaround time at the expense of overall device usage efficiency.

DHWS Specifies that high watermark setup is to occur for only MVS direct-access units identified in the SETNAME statement. The direct-access units must also be specified on the TYPE parameter of the HWSNAME statement. MDS will attempt to allocate the minimum number of direct-access devices for a job. Other devices are allocated based on the amount required for the entire job.

THWS Specifies that high watermark setup is to occur for only tape units identified in the SETNAME statement. The tape units must also be specified on the TYPE parameter of the HWSNAME statement. MDS will attempt to allocate the minimum number of tape devices for a job. Direct-access devices are allocated based on the amount for the entire job.

HWS Specifies that high watermark setup is to occur for all devices required for a job running on an MVS processor which are indicated in the SETNAME statement. The devices must also be specified on the TYPE parameter of the HWSNAME statement. MDS will attempt to allocate the minimum number of devices required for the job to run.

SETPARAM initialization statement

The SETPARAM initialization statement specifies parameters that the JES3 main device scheduler (MDS) and the DYNAL DSP uses in allocation, mounting, and deallocation of devices for jobs run on all mains. SETPARAM also indicates how MDS is to manage devices.

If SETUP=NONE is specified on the STANDARDS statement to indicate no MDS, then SETPARAM is ignored and all devices are allocated, mounted, and deallocated by MVS.

FETCH and TAFETCH parameters

FETCH=YES|NO indicates whether or not MDS is to issue volume fetch messages.

NO Specifies that MDS is not to issue volume fetch messages. If FETCH=NO is specified, ALLOCATE=MANUAL will be overridden (and ALLOCATE=AUTO assumed); MDS will automatically attempt to set up jobs.

YES Indicates that MDS is to issue volume fetch messages

TAFETCH specifies the routing information for tape volume fetch messages. The first operand specifies the routing information for specific (nonscratch) volume requests. The second operand specifies the routing information for scratch volume requests. S1 indicates that volume fetch messages are to be issued with routing code 97; messages also are recorded on the hard-copy log. Note that this parameter is ignored if FETCH=NO is also specified. 97 is the routing code equivalent of JES3 Dest Class S1.

ALLOCATE parameter

The ALLOCATE=MANUAL|AUTO keyword specifies whether or not automatic allocation of a job is to immediately follow MDS volume fetch. This parameter is ignored for jobs that reference only premounted volumes. The FETCH parameter specified may override the ALLOCATE parameter.

MANUAL Indicates that all jobs are to be suspended following volume fetch until the operator causes them to continue. Note that ALLOCATE=MANUAL is ignored if FETCH=NO is indicated; ALLOCATE=AUTO is assumed instead.

AUTO Specifies that MDS will automatically attempt allocation of resources for all eligible jobs. If a job requires SMS-managed resources and you specify ALLOCATE=AUTO, MDS sends the job through the system select phase before allocation to determine which mains have access to the required

SMS-managed resources. Note that ALLOCATE=AUTO is assumed (ALLOCATE=MANUAL is ignored) if FETCH=NO is specified.

ADDRSORT parameter

ADDRSORT specifies the order in which JES3 MDS allocates devices.

NO Indicates that devices within a device type are to be allocated in the same order as the DEVICE statements are placed in the initialization stream.

YES Indicates that devices within a device type are to be allocated by the order of their device numbers, that is, 188, 189, 18A.

MTJESMSG parameter

MTJESMSG specifies whether you want FETCH, ALLOCATION, and BREAKDOWN messages for mountable devices to appear in the JESMSGLG data set.

FETCH Specifies that you want fetch messages for mountable devices written into the JESMSGLG data set.

ALLOC Specifies that you want allocation messages for mountable devices written into the JESMSGLG data set.

BREAKDWN Specifies that you want breakdown messages for mountable devices written into the JESMSGLG data set.

ALL Specifies that you want fetch, allocation, and breakdown messages for mountable devices written into the JESMSGLG data set.

NONE Specifies that you do not want fetch, allocation, or breakdown messages for mountable devices written into the JESMSGLG data set.

PRJESMSG parameter

PRJESMSG specifies whether you want FETCH and ALLOCATION messages for permanently resident or reserved DASD to appear in the JESMSGLG data set.

FETCH Specifies that you want fetch messages for permanently resident or reserved DASD written into the JESMSGLG data set.

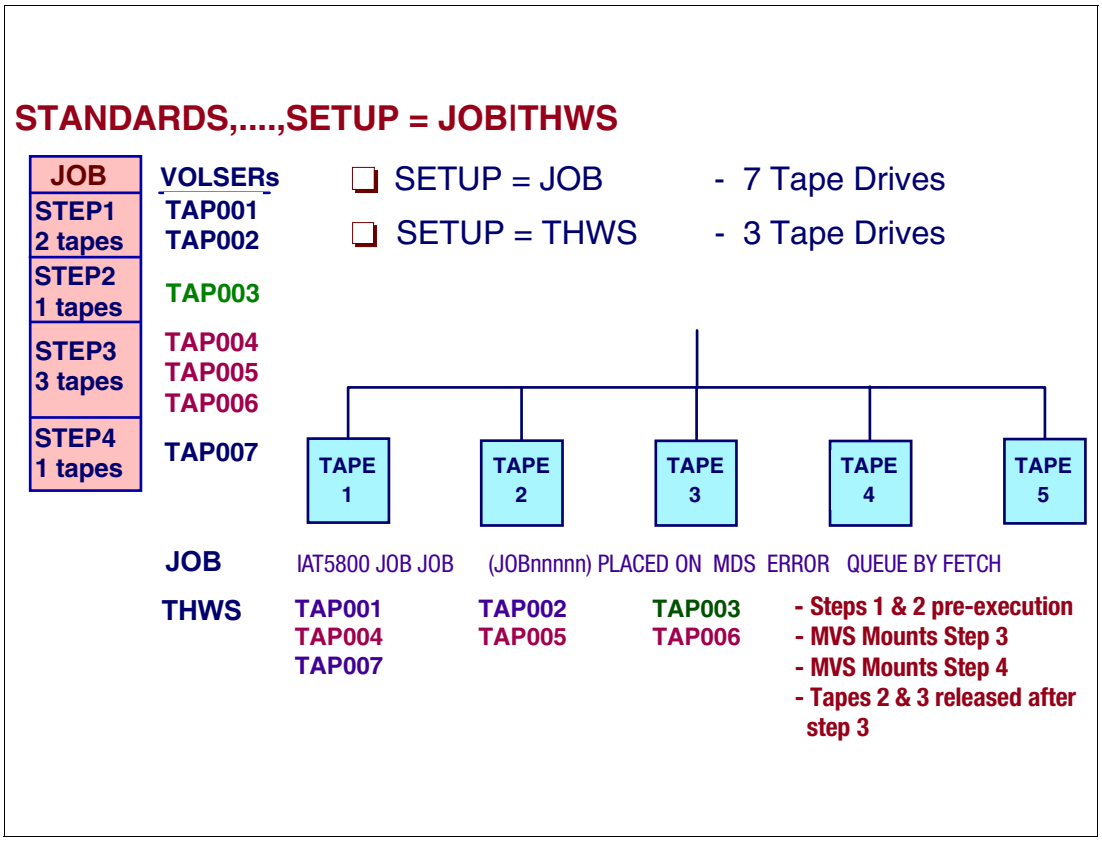
ALLOC Specifies that you want allocation messages for permanently resident or reserved DASD written into the JESMSGLG data set. If this value is specified, an allocation message will be written for all non-mountable requests in addition to permanently resident DASD.

ALL Specifies that you want both fetch and allocation messages for permanently resident or reserved DASD and allocation messages for all other devices written into the JESMSGLG data set.

REMOUNT parameter

REMOUNT parameter specifies the number of times that an operator can retry to correct volume mount errors for a job before the devices for the job are released and allocation is restarted. The value of nnn specifies the number of retries allowed, from 0 to 255. For example, if REMOUNT=1 is specified, the operator can make two attempts to mount the volume--the original mount request and one retry request.

8.28 MDS setup options - JOB versus THWS



MDS setup options - JOB versus THWS

This sample setup job has 7 tapes requests in 4 job steps. Using SETUP=JOB on the STANDARDS statement would cause JES3 to assign 7 tape drives prior to execution for the job. Since there are only 5 tape units, the MDS places the job into the error queue.

Using SETUP=THWS, the same job would only need 3 tape units to execute the job. This is because the maximum number of units in any of the steps is 3. What is also true, is that the unit types must all be the same in this example.

In the sample the tape mounts are as follows:

- Prior to execution under the initiator, JES3 MDS would select 3 taoe units and issue mount messages to the operator to mount the first 3 tapes needed by the job. So, volsers TAP001, TAP002, and TAP003 are mounted.
- When step 1 completes, the first two tape units are now free but still belong to the job. The operator receives message to remove the tape volumes.
- When step 2 completes, the third tape unit is now free and TAP003 is removed by the operator.
- When step 3 starts, MVS allocation issues mount messages to the operator to mount volsers TAP004, TAP005, and TAP006. JES3 releases two tape units for other jobs to use.
- When step 4 starts, MVS allocation issues a mount message to the last tape unit owned by the job for volser TAP007. When step 4 completes, the last tape unit is free for other jobs to use.

When the job ends, the release of the still allocated tape units is done by MDS breakdown.

8.29 MDS allocation mode

- SETPARAM,ALLOCATE = MANUAL|AUTO
  - MANUAL mode
    - Allows for tapes to arrive from "Manual Tape Library"
    - "*S S jobno" command required to allow a job to proceed to allocation processing.
  - AUTO mode
    - MDS automatically attempts allocation
- *F,S,AL=A | M command
  - Change allocation mode
- *F S, VU= T-|D-volser | VA = T-|D-volser
  - Make tape (T) or disk (D) volume unavailable or available

MDS allocation mode

If manual allocation was specified during JES3 initialization or with the **`*F S,AL=M`** command, use the **`*S S`** command to allow a job to proceed to allocation processing. The **`*S S`** command is not required when automatic allocation was specified during JES3 initialization or by the **`*F S,AL=A`** command.

If ALLOCATE=AUTO (parameter default) is specified on the SETPARAM initialization statement or if you enter the **`*F S,AL=A`** command (automatic allocation), the job will proceed to the allocate queue. If you specify automatic allocation and SMS resources are required, the job will proceed to the system select queue. If ALLOCATE=MANUAL is specified on the SETPARAM initialization statement or if you enter the **`*F S,AL=M`** command, the job is put into the WAITVOL queue, from which it is released by entering the **`*S S`** command when the volumes are available (at which time the volume-request messages are issued).

Use the **`*F S VU=|VA=`** commands to:

- Make a volume unavailable for JES3 setup processing
- Make a volume available for JES3 setup processing

```
Tape volumes R67891 and R67892 are to be made unavailable:
*F,S,VU=(T-R67891,T-R67892)

Tape volumes R67891 and R67892 are to be made available:
*F,S,VA=(T-R67891,T-R67892)
```

8.30 Tape fetch processing

- SETPARAM,FETCH = YES,TAFETCH = (S1,S1),DSN = 44

USES

IAT5110 JOB JOB1 (JOB00004) GET T 123456 ,SL data.set.name

UNAV

- Volume not available

*F S,VU = T - 123456

- All jobs requiring volume go to MDS UNAVAILABLE queue
- Volume available

*F S,VA = T - 123456

Tape fetch controls

The MDS fetch routine runs under the SETUP FCT and is the first phase of setup processing. Volume fetch messages are selected optionally by specifying FETCH=YES on the JES3 SETPARAM initialization statement. When you select the fetch option, JES3 issues volume fetch messages to indicate which volumes are required for specific jobs to execute. JES3 sends fetch messages to the console specified by the TAFETCH (for tape volumes) and DAFETCH (for direct-access volumes) parameters on the SETPARAM initialization statement. Volumes already mounted require no fetch processing, and volumes that have been fetched but not mounted get action-coded messages.

The message indicates the action required, volume type, volume serial number, tape label type, and data set name for the volume referenced by the specified job.

GET The volume that will be required by this job, but is currently not in use.

UNAV The volume has been made unavailable by the operator.

USES The volume that was previously fetched and has not yet been returned to the library.

If a volume is not found, it can be placed in an unavailable status. Use the `*F S` command to:

- Make a volume unavailable for JES3 setup processing
- Make a volume available for JES3 setup processing

The commands in the visual specify either tape (T) or disk (D) volumes that are to be made unavailable. The designated volumes remain unavailable until a `*F S,VA=` command is issued.

8.31 Volume mounting

- MDS requests and verifies initial volume mounting
  - On each device before a job requires
    - Unless deferred volume mounting is specified
    - Multi-volume mount - only the first volume is mounted; secondary volumes are not mounted until required
- VERIFY function obtains volume information
  - IATUX25 - validate any nonstandard labels
  - REMOUNT - number retries to correct mount errors
    - Remounts fail - Devices released and allocation restarted
- When all volumes are properly mounted, the job is ready for execution

Volume mounting

For main device scheduler requests, MDS verifies the mounting of the initial volumes a job requires on each device before the job can be selected for execution (unless deferred volume mounting is specified in the JCL).

MDS VERIFY processing

If verify encounters any correctable mount errors (and the operator remount count for the job is not yet reached) MDS re-issues mount messages for the correct volumes required for job execution.

Operator REMOUNT processing

The SETPARAM REMOUNT parameter specifies the number of times that an operator can retry to correct volume mount errors for a job before the devices for the job are released and allocation is restarted. The value of nnn specifies the number of retries allowed, from 0 to 255. For example, if REMOUNT=1 is specified, the operator can make two attempts to mount the volume--the original mount request and one retry request.

8.32 MDS mount messages and commands

- Mount messages
  - IAT5200 JOB JOB1 JOB00004 (JOB1) IN SETUP ON MAIN = SY1
    - The indicated job has allocated devices on the specified main
  - *IAT5210 JOB JOB1(JOB00004) MOUNT T-123456 ON 180 ,SL,NORING
    - This message indicates that a volume is required and/or that a device has been allocated
- Display outstanding MOUNT messages - MVS D R command
  - D R,L,CN=(ALL) - All messages
  - D R,I,KEY=MOUNT,JOB=jobname - A single job
  - D R,KEY - Messages with KEY names
- MDS detected mount error
  - IAT5310 *dev description* -- IAT5310 180 NO RESPONSE
  - *S J = *jobno*,V - Force MDS to re-verify
    - If same response - IAT5310 *dev description*
    - *V *dev*,OFF,*main*
    - *R S,*jobno* - Places job on allocate queue

MDS mount messages

Message IAT5200 indicates a job has allocated devices on the specified main.

Message IAT5210 indicates that a volume is required and/or that a device has been allocated. The ddname rather than the job name may appear in the JESMSGLG.

Operator commands fir outstanding mounts

After MDS has issued mount messages for jobs and those mounts have not occurred, these jobs can be displayed by issuing one of the following commands to determine which jobs are waiting for mounts:

```
D R,L - All messages
D R,L,Key=mount
D R,KEY - Messages with KEY names
D R,I,KEY=MOUNT,JOB=jobname - A single job
```

Mount failures

If MDS has detected an error on a device during the mounting of a volume previously requested in message IAT5210, message IAT5310 is issued.

By issuing the **`*F S,J=jobno,V`** command, MDS tries again to verify what the operator mounted. If this fails, the tape unit should be varied offline on all systems that share the device.

8.33 Device mount status

***I ,D,D= command -- IAT8572 message**

- **VERIFIED** - the volume has completed MDS verification
- **MOUNTED** - the volume is mounted on the indicated device
- **NOT RDY** - the device is not ready
- **NO RESP** - no response has been received from MDS verify on the local
- **I/O ERR** - a permanent I/O error has occurred on the indicated device
- **VOLID ERR** - an error was encountered reading the volume label
- **ALLOCATED** - the device is allocated
- **DUP VOL** - the indicated volume serial number is in use on another device
- **OFFLINE** - the device is offline but contains a permanently resident volume
- **INIT COMP** - the initial MDS verifies are complete ((that is), restart is complete)
- **NOT OPR** - the device is not operational
- **EXPD ERR** - the expiration date has not yet been reached
- **LOAD CHK** - load check error has occurred.
- **TIMEOUT** - an execute channel program has timed-out

Device mount status

During JES3 starts, volumes that are mounted on the DASD devices are verified for their current status. This JES3 verify status is shown in the visual and the mount status indicates the results of the last attempt to perform volume verification.

***I,D** command response displays the verify status in the IAT8572 message and indicates the results of the last volume verification. It may be one of the following:

VERIFIED	The volume has completed MDS verification.
MOUNTED	The volume is mounted on the indicated device.
NOT RDY	The device is not ready.
NO RESP	No response has been received from MDS verify on the local.
I/O ERR	A permanent I/O error has occurred on the indicated device.
VOLID ERR	An error was encountered reading the volume label.
ALLOCATED	The device is allocated.
DUP VOL	The indicated volume serial number is in use on another device.
OFFLINE	The device is offline but contains a permanently resident volume.
INIT COMP	The initial MDS verifies are complete ((that is), restart is complete).
NOT OPR	The device is not operational.
EXPD ERR	The expiration date has not yet been reached.
LOAD CHK	Aload check error has occurred.
TIMEOUT	An execute channel program has timed-out.
NO DEVICE	There is no unit control block (UCB) for the indicated device.
GRP	The device is dedicated to a job class group.
NET	The device is dedicated to a DJC job network.
name	The name of the job class group or DJC network ID to which the device is currently dedicated.

8.34 Inquiry command for devices

```
*I D,D=3E30
 IAT8572 3E30 (AV ) SC64     NW3E30,JES=P,OS=P
 IAT8572 3E30 (AV ) SC70     NW3E30,JES=P,OS=P            MOUNTED
 IAT8572 3E30 (AV ) SC65     NW3E30,JES=P,OS=P            MOUNTED
 IAT8500 INQUIRY ON DEVICES COMPLETE

*I D,V=NW3E30
 IAT8572 3E30 (AV ) SC64     NW3E30,JES=P,OS=P
 IAT8572 3E30 (AV ) SC70     NW3E30,JES=P,OS=P            MOUNTED
 IAT8572 3E30 (AV ) SC65     NW3E30,JES=P,OS=P            MOUNTED
 IAT8500 INQUIRY ON DEVICES COMPLETE
*I D,D=3E30,S
  IAT8572 3E30 (AV ) SC65     NW3E30,JES=P,OS=P           MOUNTED

*I D,D=SC65
 IAT8562 SC65                IS THE GLOBAL
 IAT8500 INQUIRY ON DEVICES COMPLETE
*I D,D=SC70
 IAT8562 SC70                IS A LOCAL
 IAT8500 INQUIRY ON DEVICES COMPLETE
```

Inquiry device command

The **`*I D,D=`** command allows inquiry messages to be displayed for all devices defined in the in the initialization statements using the DEVICE statement.

The first command on the visual uses a know device address. The second command uses a volume serial number as possibly the device address is not know by the operator.

The **`SHORT`** or **`S`** option of **`*I D D=`** display gives the status of a device on a specified main.

The fourth and fifth commands are used to display the role (global or local) of a main.

Device status

The device status in the visual (AV) indicates the device is available. Other possible device status conditions are as follows:

- AC - the device is in use by a setup job. (XUNIT status)
- ACO - the device is in use but has been varied offline to JES3. (XUNIT status)
- AC - the device is in use but is pending offline
- RS - the device is reserved by a setup job above a setup barrier. (XUNIT status)
- RSO - the device is reserved but has been varied offline to JES3. (XUNIT status)
- AV - the device is online and not in use. (XUNIT status)
- AVU - the device is online and not in use, but it is in the process of being unloaded.
- OFF - the device is offline to JES3. (XUNIT status)
- OFN - the device is offline because there are no paths available
- ACN - the device is allocated and is offline because there are no paths available

8.35 MDS volume and data set control

- In use "managed" volume on "managed" devices
  - Every volume has mount and use attributes
    - Mount attribute controls demounting
    - Use attribute helps determines volume allocation
  - USE count for each volume
  - Recorded in SETUNITS, SYSUNITS, and SETVOL tables
    - Pointer to data set entries
- Data set are managed by volume and data set
  - One data set name entry for each data set on a volume
  - Using SETVOL and SETDSN tables
- Resident data sets -- RESDSN initialization statement

Volume use and mount attributes

Volume fetch, the first phase of MDS, is performed for all jobs entering MDS. This phase determines the volumes required by the job and, if necessary, instructs the operator to get the volumes from the library. This phase also eliminates those mains on which the job cannot run.

During fetch processing, JES3 builds volume entries and issues messages for volumes that have no entries in the SETVOL table which contains the volume serial number for each reference to a device managed by MDS.

As the fetch routine scans the JST entries, it builds a SETVOL table entry for each of the volumes required by the job. For SMS-managed data sets, JES3 is not concerned with and does not know about the volumes where the data sets reside. However, the fetch routine still creates a dummy SETVOL table entry since MDS is responsible for complex wide data set awareness. The fetch routine creates the SETVOL entry using a dummy volser. This allows the SMS-managed data sets to be distributed over a number of SETVOL entries. If a volume is not mounted, the fetch routine issues a fetch message to the operator.

Volume fetch messages are selected optionally by specifying FETCH=YES on the JES3 SETPARAM initialization statement. When you select the fetch option, JES3 issues volume fetch messages to indicate which volumes are required for specific jobs to execute. JES3 sends fetch messages to the console specified by the TAFETCH (for tape volumes) and DAFETCH (for direct-access volumes) parameters on the SETPARAM initialization statement. Volumes already mounted require no fetch processing, and volumes that have been fetched but not mounted get action-coded messages.

Device types other than tape or disk do not require operator action. If you do not specify the volume fetch option, jobs go directly into the system select stage of MDS if the job requires SMS-managed resources, or into allocation if the job does not require SMS-managed resources.

MDS allocation

The allocation phase of MDS uses allocation algorithms to provide required devices. When allocation is successful, JES3 issues mount request messages for all required volumes except:

- Deferred mount requests - where no mount is as yet requested but the device that the volume is to be mounted on is allocated to the user.
- Permanently resident volumes - where the mount is unnecessary
- Multi-volume mount - where only the first volume of a multi-volume data set is mounted; secondary volumes are not mounted until required.

During the volume verification JES3 issues messages that instruct you to mount a job's required volumes.

SETDSN and SETVOL tables

The SETDSN and SETVOL tables contain the data set names and volume serial numbers, respectively, for each reference to a device managed by MDS. The data set name entry table maps the SETDSN entry used by MDS to control the serialization and sharability of a particular data set.

Identifying resident data sets

During JES3 initialization you can identify data sets that are resident on all systems. Thereafter, when the names of these data sets appear as catalog references on a DD statement for a job, JES3 will bypass catalog searches (also called LOCATE processing) during the postscan phase of C/I service. Thus, you improve JES3 performance. To specify the names of the resident data sets, use the RESDSN initialization statement.

JES3 does not perform data set awareness processing for resident data sets. Therefore, do not specify SMS-managed data sets as resident data sets unless the data set can be accessed from only one processor or unless the user provides data set awareness. MVS provides data set integrity when a resident data set can be accessed from only one processor.

8.36 Managing JES3 device online/offline status

- Vary devices on and offline
  - *V (*devnum,/devnum,..*),OFFLINE,*main*
  - *V *devnum*,ON,*main* - execution device on *main*
  - *V *devnum*,ON - varies device online for global JES3 use
- Vary device offline to all systems
  - *V *devnum*,OFF,ALL
- Vary main processor on and offline
  - *V *main*,ON and *F V,*main*,OFFLINE
    - OFF stops jobs from being scheduled to the system
    - ON enables job scheduled to the system

Managing JES3 device online/offline status

The online/offline status of a device controls whether the device is eligible to be allocated or not.

The ***V** command to makes JES3, and JES3-managed devices available or unavailable for JES3 scheduling. These devices include RJP lines, devices at BSC RJP or SNA RJP workstations, logical senders used by JES3 networking, and mains.

Vary tape devices 180 and 181 offline on SY1:

```
*V,(180,181),OFFLINE,SY1
```

Vary local main processor SY1 offline:

```
*V,SY1,OFF
```

Use the ***V** command to vary SMS-managed devices online or offline to any processor in the JES3 complex.

When varying a JES3-managed execution device online or offline to JES3 on a main, JES3 attempts also to vary the device online or offline to MVS. JES3 varies a device offline only if the device is not allocated. If you want the MVS status to differ from the JES3 status, you then must enter the MVS **VARY** command to change the MVS online/offline status of the device (after JES3 initialization or a JES3 ***V**). You should not, however, vary an online JES3-managed device offline to MVS.

Device in use

If you enter a **`*V xxx,OFFLINE`** command for a device that is currently in use (except for a tape drive that is in use on another system), JES3 will mark the device pending offline to JES3 and will not vary the device offline to MVS until it is unallocated.

> **Note:** When varying a 3480 or 3490 tape device online or offline to JES3 as a JES3-managed device or a JES3 device, JES3 varies the device online or offline to MVS. JES3 also performs an ASSIGN (online) or UNASSIGN (offline). The device remains online and assigned to MVS only if it is online as a JES3 device or as a JES3-managed device. A device path must be online before the operator or JES3 can vary the corresponding device online to JES3 or MVS.

8.37 Data awareness in a JES3 complex

- Both MVS and JES3 provide data set awareness
  - JES3 data set awareness
    - Shared use - through JCL DISP = SHR
    - Exclusive use - through JCL DISP = OLD/MOD/NEW
    - Only for data sets on JES3 managed devices and SMS managed data sets (unless SMSSETUP-NO set)
  - Bypassing JES3 Integrity Checking
    - DYNALDSN initialization statement
    - RESDSN initialization statement
  - MVS data set integrity
    - Multiple Users - READ only - DISP = SHR
    - Single User - UPDATE data set - DISP = OLD/MOD/NEW
    - Only by data set name

Data set awareness

In a JES3 complex both MVS and JES3 provide data set awareness. Data set awareness specified in the JCL DISP parameter ensures that:

- While one job is reading a data set another job does not change that data set.
- While one job is updating a data set no another job can read that data set.
- Two ways to establish the extent of VSAM data set sharing are the data set disposition specified in the JCL and the share options specified (SHAREOPTIONS) in the access method services DEFINE or ALTER command.
- Only one job at a time uses a sequentially accessed (serially reuseable) device that is attached to one or more processors

JES3 data set awareness processing

JES3 provides complex-wide data set awareness for managed data sets. MDS data set awareness is responsible for controlling access to data sets within a JES3 complex. This prevents jobs running in different processors in the complex simultaneously from updating a serially reusable resource (existing data set with DISP=OLD/MOD/NEW). To provide data set awareness for non-SMS managed data, MDS has to know the volumes associated with the data set. Non-DFSMS managed, new non-specific DASD data sets, do not have an associated volser and thus MDS cannot provide data awareness for them.

MVS allocation

MVS and MDS allocation consider a job's resource requirements at different levels. MVS allocation considers job requirements one step at a time for the processor executing the job; MDS considers the resource requirements for all the steps in a job for all processors in the loosely-coupled complex. These two approaches lead to the following differences between MDS and MVS allocation:

- MVS allocation: In systems that do not use JES3, jobs are presented to MVS based on criteria such as job class, priority, or WLM work classification. In these systems, a job's resource requirements are not known until the job entry subsystem selects the job for execution, and a system initiator begins the step allocation process. At each job step, MVS allocation attempts to satisfy the requirements for the step, in contention with every other job step currently executing on the same processor or in the sysplex. If the requirements cannot be met, MVS allocation gives the operator the option of canceling the job or allowing it to wait for resources. Thus, in a system that does not use MDS, there may be jobs executing and other jobs waiting for resources. The jobs waiting in MVS allocation hold critical resources (a system initiator, an address space, data sets, and possibly devices).
- Main Device Scheduler (MDS) allocation: With MDS, the resources (data sets, devices, and volumes) that a job requires are already set up when the job is passed to MVS for execution. Setup occurs while a job is in the JES3 address space, and the only system resource used while the job is waiting is the JES3 queueing space. MDS helps the system make maximum use of devices and allows jobs to run in a minimum amount of time once they are passed to the system for execution. Also, MDS cooperates with the workload management (WLM) component of MVS to ensure that the scheduling environments for jobs are honored.

MVS data set integrity

Before allocating data sets to a job, MVS enforces data set integrity for MVS-managed data sets. To ensure data set integrity, MVS does not allocate a data set to a job if:

- The job requests non-shared access to an allocated data set (JCL DISP OLD/MOD/NEW)
- The job requests shared access to a data set that is allocated as non-shared

MVS does not begin the allocation process until the integrity of all the data sets in the job has been enforced. When integrity has been established, MVS then begins allocating the resources needed for the job, one step at a time. MVS provides integrity for data sets within a single MVS system or across multiple MVS systems in a sysplex.

JES3 data set awareness

JES3 enforces data set awareness for:

- Data sets that are requested through DD statements that require JES3-managed devices
- Data sets that are requested dynamically that require JES3-managed devices
- Data sets that are SMS-managed, unless SMSSETUP-NO is specified on the SETPARAM statement.
- JES3-managed sequentially accessed devices that are attached to one or more processors

If a job requests a data set through a DD statement, JES3 enforces data set awareness before scheduling the job for execution. For dynamically requested data sets, JES3 enforces data set awareness at the time of the request.

To ensure data set integrity, JES3 denies a job's request for a data set if:

- The request is for non-shared access to an allocated data set
- The request is for shared access to a data set that is allocated non-shared

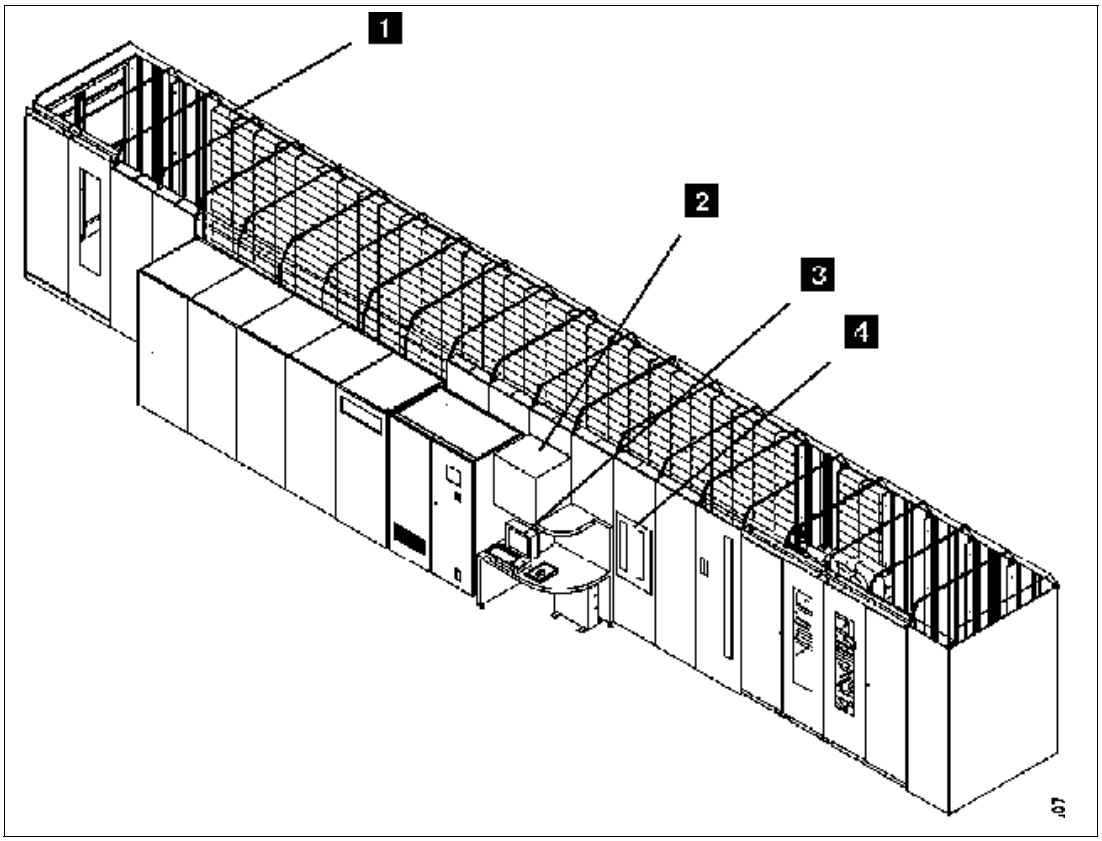
What happens to a job that is denied an allocation request by JES3? This depends on how the job requested the data set.

- If it was requested using a DD statement, JES3 does not schedule the job for execution at that time. The job must wait until all of the resources it needs are available.
- If the data set was dynamically requested, MVS will not allocate the data set to the job at that time. The job, however, can continue to execute.

Bypassing JES3 awareness checking

To bypass JES3 data set awareness checking for dynamically allocated data sets or permanently-resident volumes mounted on JES3-managed devices, code the data set names on a JES3 DYNALDSN initialization statement. If you bypass JES3 setup for permanently resident data sets, you also bypass JES3 data set awareness checking for those data sets. This happens for data sets specified on a RESDSN initialization statement. Although you bypass JES3 data set awareness checking, MVS data set integrity checking remains in effect.

8.38 IBM 3495 automated tape library data server



Automated tape library data server - ATLDS

The automated tape library dataserver (ATLDS) and its supporting software streamlines and automates the roles of the storage administrator, tape operator, and the tape librarian, and uses the concepts of SMS to manage the tape volumes within the library.

An automated tape library dataserver (ATLDS) consists of tape drives, tape cartridges, a tape cartridge storage area, input and output stations for inserting and removing cartridges, and a mechanism for moving tape cartridges among these areas. The volumes within an automated tape library are known as library-resident tape volumes. Tape volumes can also be located on shelves outside the automated tape library. These volumes are known as shelf-resident tape volumes.

Tape cartridges are stored and retrieved by an automated cartridge accessor. The cartridges are placed in an input station by the tape library operator. The cartridge accessor then scans the external volume label on the cartridge, carries the cartridge to the appropriate storage location, and places it into the library. When a volume mount is requested, the cartridge accessor retrieves the cartridge from the storage location, carries it to the requested drive, and mounts the cartridge in the drive. Upon completion of the tape operation, the tape cartridge is unloaded, the accessor retrieves it from the drive, and returns it to a storage location in the library.

However, the tape library operator can continue library operation during periods when the cartridge accessor is not operational. During this time the operator responds to commands displayed on the manual mode console. This is known as manual mode operation.

JES3 IBM tape library data servers

JES3 support for the 3494/3495 tape library data servers provided with DFSMS enables a JES3 installation to manage tape drives within an IBM 3494/3495 tape library data server. (The following text references the IBM 3495 tape library data server.

JES3 supports the definition and usage of one or more IBM 3494/3495 tape library data servers within a JES3 complex. The tape drives within a given library are subsetted into library device groups (LDGs) based upon the device type. JES3 uses specific esoteric names to direct tape allocations to devices within the correct library when the tape resides in a library, and to drives outside of a library when the tape is not resident within a library.

The esoteric names for devices comprising a LDG are conveyed to JES3 through the SETNAME initialization statement. These esoteric names, in conjunction with the SETNAME statement, allow JES3 to associate the devices to a device group, the device groups to a library, and the libraries to a JES3 complex.

3495 tape data sets allocation

DFSMS directs new data sets to the IBM 3495 Tape Library Dataserver using the data class, storage class, and storage group information specified in JCL or provided by Automatic Class Selection (ACS). Storage class identifies the data set as DFSMS managed mountable. Data class is used to determine the required cartridge and device type. Storage group is used to determine the eligible 3495 libraries. Requests for old data sets are directed to a 3495 only when the volume or volumes containing the data set are located within a 3495. Eligible device types are determined from the cartridge type and recording format.

JES3 initialization statements for the 3495

JES3 initialization deck must include SETNAME, DEVICE and HWSNAME statements for the 3495. These statements define the 3495 to JES3, ensuring that non-library requests are not allocated to 3495 tape drives.

Neither JES3 nor DFSMS verifies that a complete and accurate set of initialization statements are defined to the system. A 3495 definition that is either incomplete or inaccurate may result in jobs failing allocation or other unpredictable results.

Before defining the devices within a library to JES3, they must be properly defined to MVS using the hardware configuration definition (HCD) program. Unlike a JES2 environment, a JES3 operating environment requires the specification of esoteric unit names for the devices within a library. These unit names will be used in the required JES3 initialization statements.

Each device within a library must have exactly four (4) special esoteric names associated with it. These are:

1. The complex-wide name, this is always: "LDGW3495."
2. The library-specific name, this is an 8-character string composed of "LDG" prefixing the 5-digit library identification number.
3. The complex-wide device type.
4. A library-specific device type name, which consists of the 5-digit library device number prefixed by a 3-character device type identifier.

These esoteric names should be defined and the input for the JES3 initialization stream checker obtained before making changes to the JES3 initialization stream.

Device-specific library unitnames

A library-specific device type name, which consists of the 5-digit library device number prefixed by a 3-character device type identifier.The following example specifies the name to be used for each device type.

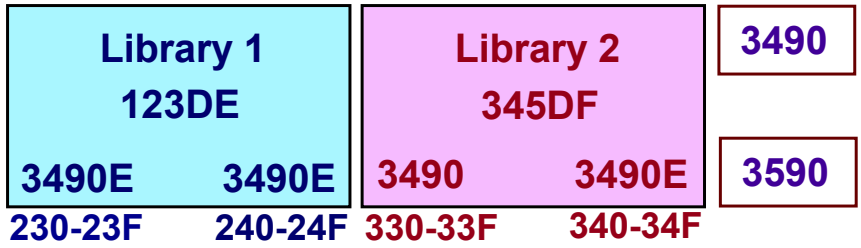
Device Type unitname	*Complex-wide Device Type unitname*	*Library-Specific Device Type unitname*	
3490	LDG3490	LDDnnnnn	
3490E	LDG3490E	LDEnnnnn	
3590-1	LDG3591	LDBnnnnn	
3590E1x	LDG359E	LDCnnnnn	(Note 1)
3590H1x	LDG359H	LDFnnnnn	(Note 1)
3592	LDG359J	LDJnnnnn	(Note 2)

Notes:

1. IBM TotalStorage Tape System 3590 Models E1x/H1x
2. z/OS DFSMS Software Support for IBM System Storage TS1120 Tape Drive (3592)

There is a different nomenclature for encrypted versus unencrypted for the 3592. LDLnnnnn indicates encrypted, LDKnnnnn indicated unencrypted (see APAR II14350).

8.39 MVS UNITNAMEs



Unit Name	Device Address Range
LDGW3495	230-23F, 240-24F, 330-33F, 340-34F
LDG123DE	230-23F, 240-24F
LDG345DF	330-33F, 340-34F
LDG3490	330-33F
LDD345DF	330-33F
LDE3490E	230-23F, 240-24F,340-34F
LDE123DE	230-23F, 240-24F
LDE345DF	340-34F

Complex-wide name always: LDGW3495
Library-specific name: LDGnnnn -- nnnnn 5-digit library ID

Device type	Complex Wide Device Type	Library Specific Device Type
3490	LDG3490	LDDnnnnn
3490E	LDG3490E	LDEnnnnn

MVS UNITNAMEs

MVS UNITNAMEs must be defined for all 3495 library device groups. The visual shows the UNITNAMEs defined for the library device groups in the configuration example. Note the following:

- The complex-wide name, LDGW3495, includes all devices in all libraries.
- The library-specific names, LDG123DE and LDG345DF, include devices from their respective libraries.
- The complex-wide device type names, LDG3490 or LDG3490E, include all devices of the same type, in all libraries.
- The library-specific device type names, LDD345DF, LDE123DF, and LDE123DE include all devices of the same type within their respective libraries.

In order for JES3 to manage SMS managed mountable requests for the ATLDS devices, the ATLDS specific unit names has to be added to the JES3 initialization stream (SETNAME and HWSNAME statements).

The DFSMS module IFG0JES3 uses the LDxccccc unit names when it provides library device unit type conversion for JES3. JES3 uses IFGXATL macro to invokes IFG0JES3 module services.

8.40 Coding ATLDS SETNAME statements

- The complex-wide library name must be included in all statements
- A library-specific name must be included for XTYPEs within the referenced library
  - **D SMS,LIB(libname),DETAIL** displays includes library ID
  - Library ID is just below the logo plate of the 3495
- The complex-wide device type name must be included for all XTYPEs of the corresponding device type in the complex
- A library-specific device type name must be included for the XTYPE associated with the devices within the library
- Do not specify generic (e.g. 3480, 3480X, 3490) or esoteric (e.g. TAPE, CART) unit names for library devices

Coding ATLDS SETNAME statements

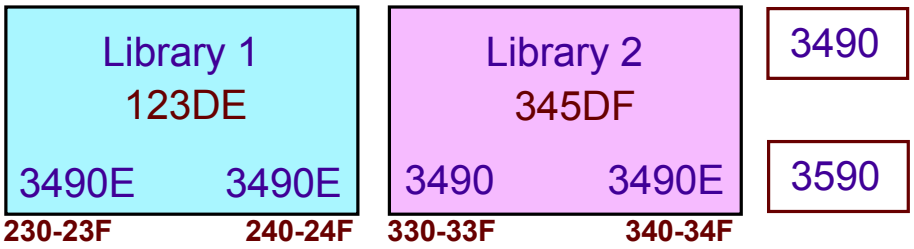
Include a SETNAME statement for each XTYPE group of ATL tape devices. The NAMES list for the SETNAME statement must include all required library device groups. It must not include any generic or esoteric names (such as 3490, 3480X, SYS3480R). The omission of these esoteric and generic names is required to prevent tapes from being mounted on unsuitable devices. The names specified must be defined to MVS as esoteric names.

The grouping of library devices into XTYPE groups is the same process for all other JES3 managed devices. The XTYPE groups are determined by the special esoteric names assigned during the HCD processing and associating the esoteric name to a device group.

The following guidelines will ensure correct generation of the SETNAME statements:

- There are always four esoteric names in the NAMES= list. These names are the special esoteric names that are used to identify library resident devices.
- There can never be esoteric names pertaining to different device types in the same list.
- There can never be two esoteric names for 3490, 3490E or 3590 device identifiers in the same list.
- There must be a LDEnnnnn value and vice versa if an LDG3490E exists. This is true for LDG3490 and LDDnnnnn, and LDG3591 and LDBnnnnn also.

8.41 ATLDS DEVICE and SETNAME statements




```
*-----LIBRARY TAPES-----
DEVICE,XTYPE=(D23490,CA),XUNIT=(0230,*ALL,S1,OFF),NUMDEV=32
DEVICE,XTYPE=(D13490,CA),XUNIT=(0330,*ALL,S1,OFF),NUMDEV=16
DEVICE,XTYPE=(D33490,CA),XUNIT=(0340,*ALL,S1,OFF),NUMDEV=16
* -----SETNAMES TAPE-----
SETNAME,XTYPE=D13490,NAMES=(LDGW3395,LDG345DF,LDG3490,LDD345DF)
SETNAME,XTYPE=D23490,NAMES=(LDGW3495,LDG123DE,LDG3490E,LDE123DE)
SETNAME,XTYPE=D33490,NAMES=(LDGW3495,LDG345DF,LDG3490E,LDE345DF)
```

Complex Wide Library Name | Library Specific Library Name | Complex Wide Device Name | Library Specific Device Name

SETNAMES with tape

The visual illustrates the two 3495 libraries defined for this configuration, as follows:

- Library 1 (sequence number 123DE) has only 3490E tape drives.
- Library 2 (sequence number 345DF) has both 3490 and 3490E drives.

The grouping of library devices into XTYPE groups is the same process for all other JES3 managed devices. The XTYPE groups are determined by the special esoteric names assigned during the HCD processing and associating the esoteric name to a device group.

SETNAME statements for the two libraries are shown in the visual. Note the following specifications:

- Three XTYPEs are required because of different unit names used for 3490E devices in the two libraries.
- Ordinary esoteric or generic unit names such as 3480, 3480X, 3490, SYS3480R and SYS348XR, are not specified.
- Installation specific esoteric names such as TAPE or CART are also not used to describe the library devices.

Note: Your library ID number is printed just below the logo plate of the 3495, on the side containing the convenience I/O station.

DEVICE statement example

Include a DEVICE statement for each tape library drive in the complex. The XTYPE name must be unique for each device type within a library and cannot span libraries. You should not specify the 3495 as a JUNIT, or use the DTYPE or JNAME parameters. The absence of the DTYPE, JNAME and JUNIT parameters will prevent inadvertent and unsuccessful use of a library device by a JES3 dynamic support program (DSP).

JES3 functions with limitations

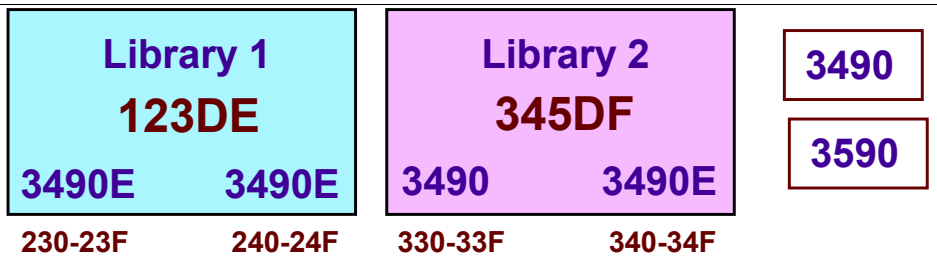
The following functions have limited capability for the 3495 in the JES3 environment:

- Support units
- Device connectivity
- Storage group states
- Scratch integrated cartridge loader requests
- Mixed JES3-managed and non-managed devices
- Library device groups specified in JCL

JES3 support unit limitations

3495 tape drives cannot be used as support units by JES3 dynamic support programs. Therefore, you should not specify DTYPE, JUNIT, and JNAME parameters in your initialization deck. JES3 does not prevent 3495 drives from being defined as support units. It also does not prevent them from being allocated to a dynamic support program. If this does occur, however, the dynamic support program must be canceled and restarted with a non-3495 tape drive.

8.42 ATLDS HWSNAME statements




```
HWSNAME,TYPE=(LDGW3495,LDG123DE,LDG123DF,LDE123DE,LDE345DF,
        LDD345DF,LDG3490,LDG3490E)
HWSNAME,TYPE=(LDG123DE,LDE123DE)
HWSNAME,TYPE=(LDG345DF,LDE123DF,LDD345DF,LDG3490)[1]
HWSNAME,TYPE=(LDE123DE,LDG123DE)[2]
HWSNAME,TYPE=(LDE345DF)
HWSNAME,TYPE=(LDD345DF,LDG3490)[1]
HWSNAME,TYPE=(LDG3490,LDD345DF)
HWSNAME,TYPE=(LDG3490E,LDE123DE,LDE345DF,LDG123DE)[2]
```

1. LDG3490 is a valid alternate for LDG123DF and LDD123DF because there are no 3490 drives in the other library.
2. LDG123DE is a valid alternate for LDG3490E and LDE123DE because that library contains only 3490E drives.

HWSNAME with tape

Build the HWSNAME statements for your library device groups using the following rules:

- The complex-wide library name includes all other LDGs as alternates.
- The library-specific name includes all LDGs for the corresponding library as alternates. The complex-wide device type name is also included as an alternate when all tape devices of a specific type are housed within a single 3495.
- The complex-wide device type name includes all library-specific device type names. When all devices of a specific type are contained within a single 3495, the complex-wide device type name is the same as the library-specific name. In this case, the library-specific name should also be specified as an alternate.
- The library-specific device type name includes the following alternate names:
  - When all drives within the 3495 are of the same device type, the library-specific device type name is the same as the library name. The library-specific name should be specified as an alternate.
  - When these are the only drives of this type in the complex, the complex-wide device type name is the same as the library-specific device type name. Therefore, the complex-wide device type name is specified as an alternate.

HWSNAME statement guidelines

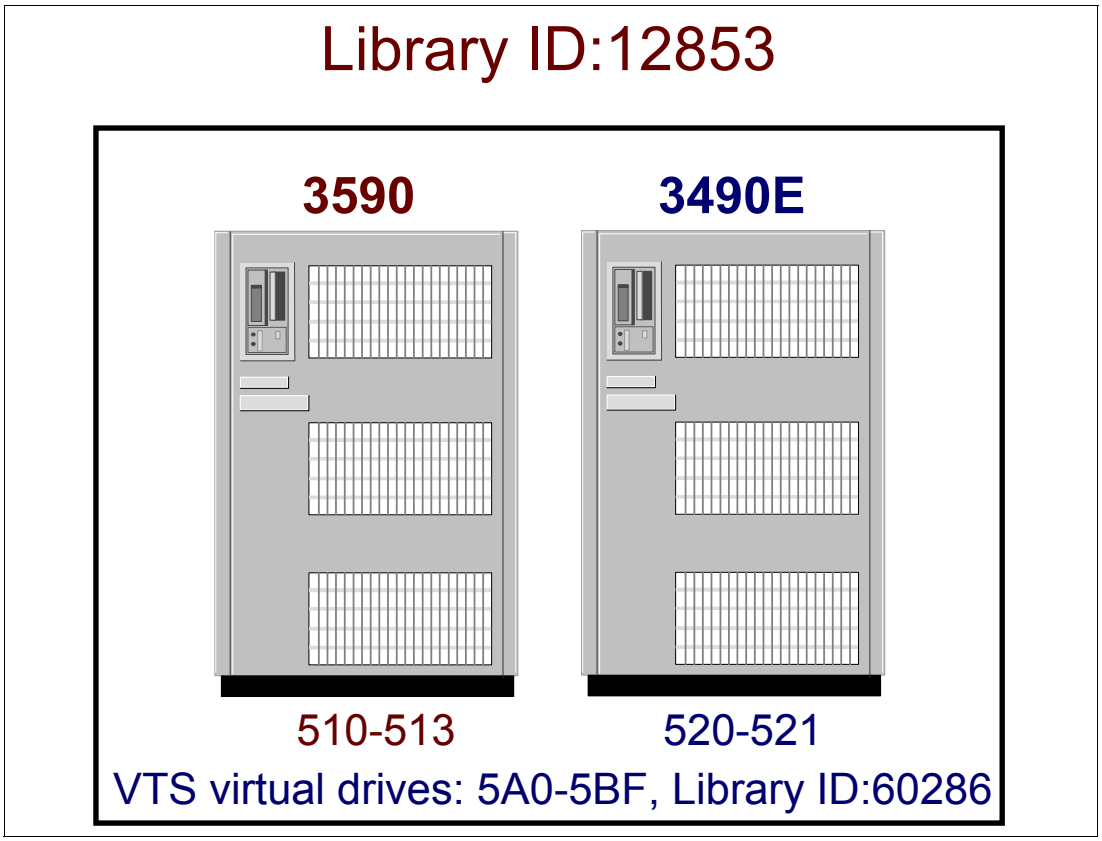
The following guidelines will ensure correct generation of the HWSNAME statements.

- There is a one to one correlation to HWSNAME statements as to special esoteric names defined in HCD.
- There can never be esoteric names in the TYPE= lists that are not one of the special esoteric names. Similarly, your existing HWSNAME statements for tape devices must not have any of these special esoteric names in their TYPE= lists.

Following is an example of a HWSNAME statement:

```
HWSNAME,TYPE=(LDGW3495,LDG123DE,LDG3490E,LDE123DE)
```

8.43 Virtual tape server (VTS) tape libraries



Virtual tape servers

A way for optimizing tape media is through the virtual tape server (VTS) tape library hardware. VTS can be used with or without tape mount management. It does not require ACS routine or other software changes.

You can reduce the number of tape cartridges, devices, and automated tape libraries by implementing an IBM TotalStorage® Virtual Tape Server (VTS) or an IBM TotalStorage Peer-to-Peer Virtual Tape Server (PtP VTS). This virtual tape subsystem consists of virtual tape devices, virtual tape volumes, tape volume cache (DASD), and hierarchical storage management software.

The PtP VTS addresses data availability, system availability, remote copy, and data vaulting needs by coupling together two stand-alone VTSs.

From a host perspective, the virtual system looks like multiple 3490E control units, each with 16 tape devices. Each emulated device is called a virtual tape device. The virtual system handles all 3490 tape commands. Each virtual device has the following attributes:

- Has a host device address
- Is included in the I/O generation for the system
- Is varied online or offline to a host
- Signals ready when a virtual volume is loaded
- Responds to and processes all 3490E tape commands
- Becomes not ready when a virtual volume is rewound and unloaded
- Indicates that it has a cartridge loader

- Can be associated with a pool of scratch volumes that allow very fast mount access for scratch mounts

Note: The active status of the cartridge loader depends on the availability of scratch volumes in the assigned pool.

VTS Tape Libraries

The IBM TotalStorage Peer-to-Peer Virtual Tape Server (PtP VTS) the IBM 3494 can be used to join two separate Virtual Tape Servers into a single interconnected system. Supported libraries are the IBM TotalStorage 3494 Tape Library and the IBM System Storage® TS3500 Tape Library. The two virtual tape systems can be located at the same site or at different sites that are geographically remote. This provides a remote copy capability for remote vaulting applications.

The Peer-to-Peer VTS appears to the z/OS host as a single automated tape library with 64, 128, or 256 virtual tape drives and up to 500,000 virtual volumes. The configuration of this system has up to 3.5 TB of Tape Volume Cache native (10.4 TB with 3:1 compression), up to 24 IBM TotalStorage 3592 tape drives, and up to 12 IBM TotalStorage 3590 tape drives models B1A, E1A, or H1A, and up to 16 host ESCON or FICON® channels.

A Peer-to-Peer VTS configuration is built from the following components:

- The 3494 Virtual Tape Controller (VTC)
- The 3494 Auxiliary Tape Frame Model CX1
- The 3494 Model B10 and B20 Virtual Tape Server attached to a IBM tape library
- Peer-to-Peer Copy features

VTS Tape Allocation in JES3 Environments

Setup recommendations:

- UNITNAME
  - Never specify a virtual tape drive as an alternate for another tape library specific unit name.
  - Separate the tape allocations to the VTS from other real tape drives in the same 3494 Tape Library.
- DFSMS
  - To define the VTS as a JES3-managed drive, use the following definitions in the JES3 initialization stream.
    - Define JES3 managed drives in the VTS through DEVICE statements.
    - Define JES3 drive names through SETNAME statements.
    - Define which device names are subsets of other device names through HWSNAME statements.
- Do <u>not</u> use the same generics/esoterics inside and outside tape library for JES3-managed tapes.

In a JES3 environment, where MVS allocation is not used, JES3 attempts to spread scratch mount requests across all library devices. For specific mounts, subsystem affinity cannot be used. In a JES3 environment, where JES3-managed devices get assigned at pre-execution time before MVS allocation is invoked, a different approach can be used. JES3 can be set up, through the definition of ADDRSORT=NO in its INISH deck, to use the devices in the defined order, evenly distributing workload across the VTSs.

8.44 DEVICE statements for VTS

```
* Example Library ID 12853 (Virtual 60286)
* Devices (native 3590) 510-513
DEVICE,XTYPE=(D13590,CA),XUNIT=(510,*ALL,,ON),NUMDEV=4
* Devices (native 3490E) 520-521
DEVICE,XTYPE=(D23490,CA),XUNIT=(520,*ALL,,ON),NUMDEV=2
* Devices (virtual 3490E) 5A0-5AF and 5B0-5BF
DEVICE,XTYPE=(ATLVTS,CA),XUNIT=(5A0,*ALL,,ON),NUMDEV=32
```

MVS UNITNAME grouping

<table>
<tr><th>510-513</th><th>520-521</th><th>5A0-5BF</th></tr>
<tr><td>LDB12853</td><td>LDE12853</td><td>LDE60286</td></tr>
<tr><td>LDG3591</td><td colspan="2">LDG3490E</td></tr>
<tr><td colspan="2">LDG12853</td><td>LDG60286</td></tr>
<tr><td colspan="3">LDGW3495</td></tr>
</table>

DEVICE statements for VTS

From a host perspective, the virtual system looks like multiple 3490E control units, each with 16 tape devices. Each emulated device is called a virtual tape device. The virtual system handles all 3490 tape commands.

All host interactions with data in a VTS are through virtual volumes and associated virtual tape devices; there is no direct access to the data on a physical cartridge or device. Host application data is written and read as if it is stored on a real Standard or Enhanced Cartridge System Tape; however, within the system it is really stored on DASD. All tape read and write commands are translated to read and write data records to or from DASD. Volumes residing on the DASD are called virtual volumes.

One of the key features of the VTS is its capability to automatically use the 3590 and 3592 tape technology cartridge storage capacity. With a VTS, volumes being created by the host applications are stored in a tape volume cache which is built from DASD devices. The size of the tape volume cache is greater than the capacity of a native cartridge. The tape volume cache can potentially contain hundreds of tape volume images called virtual volumes, depending on the size of the volumes and tape volume cache.

Through tape volume cache management policies, the VTS moves virtual volumes from the tape volume cache to a native cartridge managed by the VTS system. As virtual volumes are moved from the tape volume cache, they are stacked end to end on the cartridge and take up only the number of bytes written by the host, effectively using all of the storage capacity of the cartridge.

8.45 SETNAME statements for VTS

*** Example Library ID 12853 (Virtual 60286)**

- **This SETNAME is for native 3590 drives**
  - **SETNAME,XTYPE=D13590,NAMES=(LDGW3495,LDG12853,LDG3591,LDB12583)**
- **This SETNAME is for native 3490E drives**
  - **SETNAME,XTYPE=D23490,NAMES=(LDGW3495,LDG12853,LDG3490E,LDE12853)**
- **This SETNAME is for virtual 3490E drives**
  - **SETNAME,XTYPE=ATLVTS,NAMES=(LDGW3495,LDG60286,LDG3490E,LDE60286)**

MVS UNITNAME grouping

<table>
<tr><th>510-513</th><th>520-521</th><th>5A0-5BF</th></tr>
<tr><td>LDB12853</td><td>LDE12853</td><td>LDE60286</td></tr>
<tr><td>LDG3591</td><td colspan="2">LDG3490E</td></tr>
<tr><td colspan="2">LDG12853</td><td>LDG60286</td></tr>
<tr><td colspan="3">LDGW3495</td></tr>
</table>

SETNAMEs with VTS

The SETNAME statement is used for proper allocation in a JES3 environment. For tape devices, it tells JES3 which tape device belongs to which library. This is done by specifying the relationships between the XTYPE values (coded in the DEVICE Statement) and the LDG names shown in the visual. A SETNAME statement must be defined for each unique XTYPE in the device statements.

Device addresses 510-513 are 3590 drives.

Device addresses 520-521 are 3490E drives.

Device address 5A0-5BF are virtual 3490E drives.

The rules for this SETNAME statement are:

- Each SETNAME statement has one entry from each LDG category.
- The complex-wide library name must be included in all statements (LDGW3495).
- A library-specific name must be included for XTYPEs in the referenced library.
- The complex device type name must be included for all XTYPEs of the corresponding device type in the complex.
- A library-specific device type name must be included for the XTYPE associated with the devices in the library.

8.46 HWSNAME statements for VTS

*** Example Library ID 12853 (Virtual 60286)**

```
* Sysplex Wide Library Name
HWSNAME,TYPE=(LDGW3495,LDG12853,LDG60286,LDG3591,LDG3490E,
LDB12853,LDE12853,LDE60286)
* Library Specific Library Name
HWSNAME,TYPE=(LDG12853,LDB12853,LDE12853[,LDG3591])
HWSNAME,TYPE=(LDG60286, LDE60286)
* Sysplex Wide Device Name
HWSNAME,TYPE=(LDG3591,LDB12853)
HWSNAME,TYPE=(LDG3490E, LDE12853,LDG60286,LDE60286)
*Library Specifiv Device Name
HWSNAME,TYPE=(LDB12853[,LDG3591])
HWSNAME,TYPE=(LDE12853)
HWSNAME,TYPE=(LDE60286,LDG60286)
```

Note: LDG3591 could be used, but is not recommended in case a new library is added.

MVS UNITNAME grouping

510-513	520-521	5A0-5BF
LDB12853	LDE12853	LDE60286
LDG3591	LDG3490E	
LDG12853		LDG60286
LDGW3495		

HWSNAME statements for VTS

```
HWSNAME,TYPE=(devname[altname,...])
```

HWSNAME statement guidelines for libraries:

- Alternate names must be a subset of or equal to the primary device name.
- List all names as the primary device name once, even if no alternates are possible.
- Do not list names as alternate which span multiple libraries
- unless the primary does.
- The complex-wide library name LDGW3494 must include all other LDGxxxxx names as alternates.
- The library-specific name LDGxxxxx must include all LDG names for the corresponding library alternates.
- The complex-wide device type name LDG3490/LDG3490E must include all library-specific device type names.

8.47 Operator control of jobs in MDS

- What is queued in MDS?

```
*I S
IAT5619 ALLOCATION QUEUE    = 0000001  BREAKDOWN QUEUE = 0000000
IAT5619 SYSTEM SELECT QUEUE = 0000000  ERROR QUEUE     = 0000000
IAT5619 SYSTEM VERIFY QUEUE = 0000000  FETCH QUEUE     = 0000000
IAT5619 UNAVAILABLE QUEUE   = 0000000  RESTART QUEUE   = 0000000
IAT5619 WAIT VOLUME QUEUE   = 0000000  VERIFY QUEUE    = 0000000
IAT5619 ALLOCATION TYPE = AUTO
IAT5619 CURRENT SETUP DEPTH - ALL PROCESSORS = 0000000
IAT5619 MAIN NAME    STATUS     SDEPTH     DASD       TAPE
IAT5619 SC43      ONLINE    IPLD 020,000 05997,00000 00064,00000
IAT5619 DEFERCT=NO
IAT5619 SDEPZERO=WAIT
```

- Which jobs are on allocate queue ?

```
*I S A
IAT5634 ALLOCATE    QUEUE
IAT5636 JOB VAINIATL (JOB46281)
*I S A SUMM
IAT5030 00001 REQUESTS NEED DEVICE TYPE LDK31233
IAT5033     JOB VAINIATL (JOB46281),01 - NEEDS 00001 DEVICE
```

Operator control of jobs

Operators can determine which jobs are waiting for resources by issuing the `*I S` command. `*I S A` command lists jobs waiting in allocate queue. `*I S,A,J=jobno` for a specific job.

The `*I S` command to displays the status of jobs currently in setup or the status of volumes and data sets controlled by MDS. If none of the optional parameters is specified, the status of all mains in the complex and a summary of the MDS queues are displayed.

`*I S` command operands:

- ,ALWIO - allowed (ALWIO) and maximum (MAXIO) number of asynchronous I/O requests
- ,A[,SUMM][,J=jobno] - jobs in the MDS allocate queue
- ,E[,J=jobnos] - jobs in the MDS error queue
- ,R[,J=jobnos] - jobs in the MDS restart queue
- ,SS[,J=jobno] - jobs on the MDS system select queue
- ,SV[,J=jobno] - jobs on the MDS system verify queue
- ,U[,J=jobnos] - displays unavailable volumes and jobs waiting for unavailable volumes
- ,B - jobs currently having their resources deallocated
- ,D=dsn - displays data set status and the status of associated volumes
- ,DE=dsn - displays data set status, the status of associated volumes, and using jobs
- ,F - jobs currently in the MDS fetch queue
- ,THWSSEP - current HWS option for separation of scratch and specific requests
- ,V - jobs waiting to be verified by setup
- ,V=vol[,E] | ALL[,E] | RES - displays status of the volume and associated data sets
- ,W - jobs currently in the MDS WAITVOL queue (waiting for the `*S SETUP`)

8.48 Operator control of jobs in MDS

- **Why a job is on allocate queue?**

  *I S A J=46281
  IAT5642 MDS ALLOCATION NOT YET ATTEMPTED FOR JOB VAINIATL (JOB46281) ON SC43 RESOURCE NAVAIL
  IAT5648 MDS FAILED TO ALLOCATE THE FOLLOWING FOR JOB VAINIATL (JOB46281) ON SC43
  IAT5645 1 OUT OF 1 REQUESTS FOR DEVICE=0B22

- **What is not available?**

  *I D D=B22 N=1
  IAT8572 0B22 (OFF) SC43 THM125,JES=R,OS=R
  IAT8500 INQUIRY ON DEVICES COMPLETE

- **Make the missing resource available!**

  *V B22 ON SC43
  IEE302I 0B22 ONLINE
  IAT5510 0B22 VARIED ONLINE ON SC43
  IAT2000 JOB VAINIATL (JOB46281) SELECTED SC43 GRP=A
  ICH70001I VAINI LAST ACCESS AT 15:41:57 ON FRIDAY, NOVEMBER 7, 2008
  IEF233A M 0B22,PRIVAT,SL,VAINIATL,E1,VAINI.ATLTS.TAPE
  IAT5918 SC43 JES3V1R 0B22, , , , ,PRIVAT
  IEC705I TAPE ON 0B22,THM126,SL,COMP,VAINIATL,E1,VAINI.ATLTS.TAPE,MEDIA5
  IAT5918 SC43 JES3V1R 0B22, , , , ,THM126

Operator controlling jobs on allocate queue

When MDS allocation is not successful because needed devices, volumes, or data sets are unavailable, the job is queued for another attempt. If it remains on the allocate queue, MDS builds/refreshes an allocation requirements list (ARL) for the job. The ARL is a list of all resources unavailable to the job. The ARL table is scanned prior to giving the job another allocation attempt (unless the job is above the priority barrier) to determine if allocation should be attempted. You define the priority barrier (or SETUP barrier) with the SBAR parameter on the SELECT initialization statement. The ***I S A** command provides detailed information with a list the reasons why the job is in the indicated main device scheduler (MDS) queue, as follows:

```
*I S A J=46281
IAT5642 MDS ALLOCATION NOT YET ATTEMPTED FOR JOB VAINIATL (JOB46281) ON SC43
RESOURCE NAVAIL
IAT5648 MDS FAILED TO ALLOCATE THE FOLLOWING FOR JOB VAINIATL (JOB46281) ON SC43
IAT5645 1 OUT OF 1 REQUESTS FOR DEVICE=VOL001
```

Message IAT5645 indicates the resources that MDS could not allocate for the job and main indicated in message IAT5648.

Determine why the job is waiting

The *I D,D= command inquires on the device characteristics to determine why the volume on the specified device (LDK31233) is not available. The IAT8572 message indicates the that the resource (device) is offline. The operator response can be to vary the device online:

```
*V B22,ON,SC43
```

8.49 MVS dynamic I/O reconfiguration and JES3

- MVS devices, defined to JES3, can be added, deleted, or changed with HCD
  - JES3, execution, and shared devices
- JES3 tolerates XUNITs and JUNITs that are undefined or incorrectly defined in MVS
- If an undefined or incorrectly defined unit is part of a NUMDEV range, the device in error is ignored but the rest of the range is unaffected
- JES3 managed devices can be added in HCD if a DEVICE statement is added and a hot start with refresh is done

MVS dynamic I/O reconfiguration and JES3

You can use the Hardware Configuration Definition (HCD) program to add, delete, or change JES3 defined devices. These can be JES3 global devices (JUNIT), execution devices (XUNIT), or shared devices (both JUNIT and XUNIT). When making these changes, be careful not to introduce subgeneric splits, define devices to JES3 but not to MVS, or define devices as one device type to JES3 but a different device type to MVS.

If you make a mistake, JES3 will tolerate the error and initialize (if there are no other errors with higher impact); however, you should correct the error and perform a hot start with refresh at your earliest convenience.

Adding devices

Adding a device to HCD that is also to be defined to JES3 requires a corresponding device definition in the JES3 initialization stream. The device can be defined by an individual DEVICE statement or it can be included in a range of devices by using the NUMDEV parameter on a DEVICE statement, which defines the first device in the range. The device must be specified on the XUNIT, JUNIT parameter, or both. The processor can be defined by name or it can be defined implicitly by using the system name *ALL. If you use *ALL, the HCD configuration for every processor must define the device the same way.

If you are adding a new device type or changing the esoteric groupings of devices, you may also need to add or change SETNAME statements. To define the new device to JES3, you must perform a hot start with refresh using your modified initialization stream after activating the configuration in HCD. All local processors will be automatically restarted when the hot

start with refresh occurs, so you must activate the configuration on all processors before you perform the hot start with refresh.

Note: Even if you are adding the same device that was previously deleted without an intervening JES3 restart, a hot start with refresh must be performed after activating the configuration in HCD to allow the internal control blocks associated with the device to be built.

Changing devices

When you use HCD to change the characteristics of an existing JES3 defined device in HCD, you do not need to IPL the processor or restart JES3.

Deleting devices

Deleting a JES3 defined device from HCD does not require any immediate JES3 change; however, you should plan to delete the device from JES3 and perform a hot start with refresh with the modified initialization stream at your earliest convenience after activating the configuration.

HCD only needs to identify if a device is online or offline, and bases whether to allow the deletion from the MVS configuration solely on that criteria. If the device is online, it will not allow deletion of the device. If the device is offline, it will delete it.

FSS devices require special consideration if their configuration in MVS is being dynamically modified. An FSS might have more than one FSA device assigned to it. Some of the devices could be active and online, and some inactive and offline. HCD will allow dynamic deletion of the offline devices. However, a subsequent JES3 hot start with refresh will fail if any of the devices under the FSS are active, even if the DEVICE statements for the previously deleted through HCD devices have been removed from the JES3 Initialization stream. A hot start (without refresh) will fail whether the FSS is active or inactive.

Note: Do not use hot start to start JES3 after a dynamic deletion of FSS devices from the MVS configuration. Only use hot start with refresh to start JES3 after deactivating all FSA devices assigned to the FSS.

8.50 Move a DASD volume to a new address

- Use the RECOVER parameter to logically move a permanently resident volume to a new address
  - *V old_dev,OFF,ALL - Vary old device offline
  - *V new_dev,ON,ALL - Vary new device online
  - *V (new_dev) RECOVER main
    - IAT5918 main JES3V**xy** new_dev ,**z**, ,E,S,volser,S MOUNTED
    - IAT5510 new_dev VARIED ONLINE ON main
  - *V (CA07) RECOVER SC70
    - IAT5918 SC70 JES3V1V CA07,P, ,E,S,TWS001,S MOUNTED
    - IAT5510 CA07 VARIED ONLINE ON SC70
  - *I D,D=CA07

    IAT8572 CA07 (AV) SC70 TWS001,JES=P,OS=P

    IAT8572 CA07 (AV) SC65 TWS001,JES=P,OS=P

    IAT8500 INQUIRY ON DEVICES COMPLETE

Moving DASD volumes

With the **`*V`** command, use the **`RECOVER`** parameter to logically move a permanently resident volume to a new address. The **`RECOVER`** parameter forces vary online and volume verification processing for a single direct access set-up device. The device that you specify on this command is treated as if it has a device characteristic of removable. Other devices that have the same volume serial as the specified device are also treated as removable devices. Parameter restrictions are as follows:

- The issuing console must have level 15 authority
- You must specify a main name
- You cannot move non-DASD volumes
- You cannot move JES3 spool volumes
- You cannot move a DASD volume that has a mount pending
- You cannot move a DASD volume involved in an active DDR swap.

Attention: Use this parameter only with the approval of your system programmer. Improper use can damage your JES3 system.

Message IAT5918

JES3Vxy indicates a copy of the verify response message from an MVS processor described as follows:

x - verify descriptions

- 1 character, 0-F, or a blank (X'40'), corresponding to the following verify descriptions:

```
0 - xxxxxx VERIFIED - volume xxxxxx has completed MDS verification.
1 - xxxxxx,L MOUNTED - volume xxxxxx is mounted on the indicated device (L is label
type).
2 - UNIT NOT READY - the device is not ready.
3 - NO RESPONSE - no response has been received from MDS verify on the main.
4 - VERIFY TIMEOUT - a timeout occurred during execution of a channel program.
5 - NON-EXISTENT DEV- there is no unit control block (UCB) for the indicated device.
6 - PERM I/O ERROR - a permanent I/O error has occurred on the indicated device.
7 - VOLID READ ERROR - an error was encountered reading the volume label.
8 - xxxxxx ALLOCATED - device xxxxxx is allocated with this volume.
9 - DUPLICATE VOLID - the indicated volume serial number is in use on another
device.
A - xxxxxx OFFLINE - device xxxxxx is offline but contains a permanently resident
B - RESTART COMPLETE - end of initial verifies (during connect).
D - NO PATHS AVAIL - the device is offline to MVS.
E - EXPR DATE ERROR - the expiration date has not yet been reached.
F - TAPE LOAD CHECK - a load check error has occurred.
```

y - source of the verify request

- A - device arm response (errors only).
- B - initial verify response.
- C - SYSUNITS ACL update.
- M - volume mounted by operator.
- R - MDS message route.
- U - unload complete.
- V - vary online response.

z - UCB/STATUS byte

- P - MVS permanently resident
- M - MVS mounted
- blank - removable

JES3V1V indicates In the IAT5918 message in the visual:

```
1 - xxxxxx,L MOUNTED - volume xxxxxx is mounted on the indicated device (L is label
type).
V - vary online response.
```

IAT8572 message

- JES=
  - R the volume (if any) is removable by JES3.
  - M the volume is JES3 mounted.
  - P the volume is permanently resident to JES3.
- OS=
  - R the volume (if any) is removable to the operating system.
  - M the volume is reserved by the operating system.
  - P the volume is permanently resident to the operating system.
  - MS the volume is reserved by the operating system and is SMS-managed.
  - PS the volume is permanently resident to the operating system and is SMS-managed.

9

JES3 job scheduling - GMS

Each time an MVS initiator requests work, generalized main scheduling (GMS) selects and schedules a job for execution. The job that GMS selects depends primarily upon initialization parameters that you have specified.

Deadline scheduling and dependent job control (DJC), additional GMS functions, enable you to control when jobs execute. With deadline scheduling, you specify a deadline by which you want the job to run. JES3 periodically increases the job's selection priority in an attempt to run the job by the specified deadline. DJC allows you to create a network of related jobs.

An installation can have JES3 or Workload Manager (WLM) or both manage initiators for batch jobs. JES3 or WLM control of initiators is at the job class group level. To specify whether JES3 or WLM manages initiators for a job class group, the installation uses the MODE parameter on the GROUP initialization statement. If MODE=JES is specified or defaulted, the initiators are managed by JES3. If MODE=WLM is specified, the initiators are managed by WLM. The MODE parameter can also be changed dynamically using the `*F G` command. A group must be either WLM managed or JES3 managed; it cannot be WLM managed on one system and JES3 managed on another.

There are a number of reasons why you might choose WLM initiator management over JES3 initiator management:

- Fewer and simpler externals -- Less externals are needed in JES3 to control WLM-managed initiators and to perform workload balancing. Once the service administrator defines the performance goals and classification rules in the WLM policy, the MVS system takes over the job of starting and stopping initiators.
- Externals reflect customer expectations -- With JES3 initiator management, it is the installation's responsibility to determine the number of initiators to be started on each system, the correct mix of jobs, and so forth. The JES3 externals do not reflect an actual performance goal, such as one hour turn around time for jobs in class X.

  With WLM initiator management, initiators are managed according to the service classes and performance goals specified in the WLM policy. The performance goals are typically expressed in terms that are found in service level agreements (for example, one hour turn around time).
- Dynamic, goal oriented initiator management -- The system adapts to changing conditions and to how well the work is meeting its performance goals. The current JES3 initiator

management puts the responsibility on the system programmer/operator to manage the work. Workload balancing algorithms used by JES3 are difficult to define and static in nature; they require an operator command and sometimes even a JES3 hot start to change.

- WLM workload balancing across a SYSPLEX is automatic -- WLM decides when to start initiators and how many to start based on performance goals and the importance of batch work with respect to other work.

9.1 Generalized main scheduling

- Which job should execute next ?
  - Priority - Mix of job classes
  - Priority barrier - Aging of jobs
  - Main dependency - Workload balancing
- How many initiators should be active ?
  - Start INITs at initialization - as needed
  - Start INITs by operator
  - Let WLM start initiators

Generalized main scheduling

In response to an initiator “job select” request, GMS routines examine the jobs that are available for execution and determine which job should be scheduled to meet the installation-defined throughput objectives for the job.

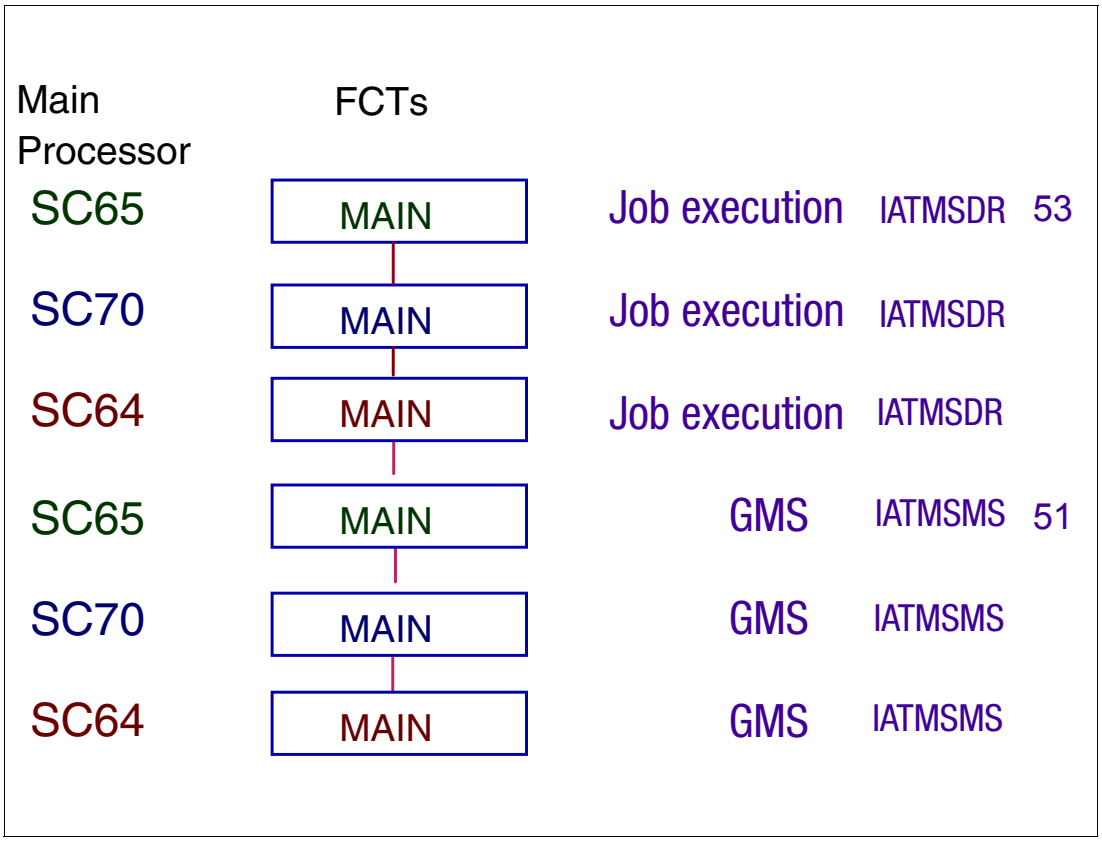
Main service selects jobs to be processed by MVS initiators. Main service selects a job for execution using the job selection algorithms established at JES3 initialization. MAINPROC, SELECT, CLASS, and GROUP initialization statements control the key variables in the job scheduling and job execution process.

GMS initiator options

The initiator allocation and deallocation options specified in the EXRESC parameter of the GROUP initialization statement control the conditions under which initiators are started and stopped when the MODE=JES is specified. Demand or dynamic initiator allocation occurs when jobs are available for execution and when the group is enabled. (Dynamic allocation is used to allocate all initiators when the first job of the group in the queue is eligible for main scheduling; demand allocation is used to allocate initiators as they are needed.) Demand or dynamic deallocation occurs when no jobs are available for execution.

If jobs are not continually available for execution, the overhead of starting and stopping initiators may be undesirable. In this case, the IPL allocation option should be used to allocate execution resources during processor connection, or the MANUAL allocation or deallocation option should be specified to indicate that all execution resources are not to be allocated or released until the operator disables the job class group with a `*F G` command.

9.2 MAIN FCTs for processing



MAIN FCTs for processing

Main service creates for each processor in the JES3 complex main processor control table (MPC). Each local contains also the MPCs of all processors. Each MPC has two function control tables (FCTs) associated with it. The higher priority FCT (IATMSDR) services the interprocessor communication; the lower priority FCT (IATMSMS) services job scheduling and termination.

IATMSDR is the driver module for the higher priority MAIN FCT. It initiates connect processing, and processes main service operator commands, staging area shortages, and subsystem restarts. The main service support is active for a job after GMS selection and through MVS termination of the job. Main service on the global interfaces with all other processors using the JESXCF services.

IATMSDR runs as a resident FCT for JES3 main I/O. Routines in IATMSDR:

- Connect Post Routine. Normally this is invoked when the JES3 system is brought up. It will also be invoked when a vary online is done.
- External Action Routine. This processes CANCEL requests for jobs and MAIN FLUSH requests.
- Main Service JESXCF Mailbox. This routine processes JESXCF System Event messages and connect messages in the main service JESXCF mailbox.

GMS processing phase

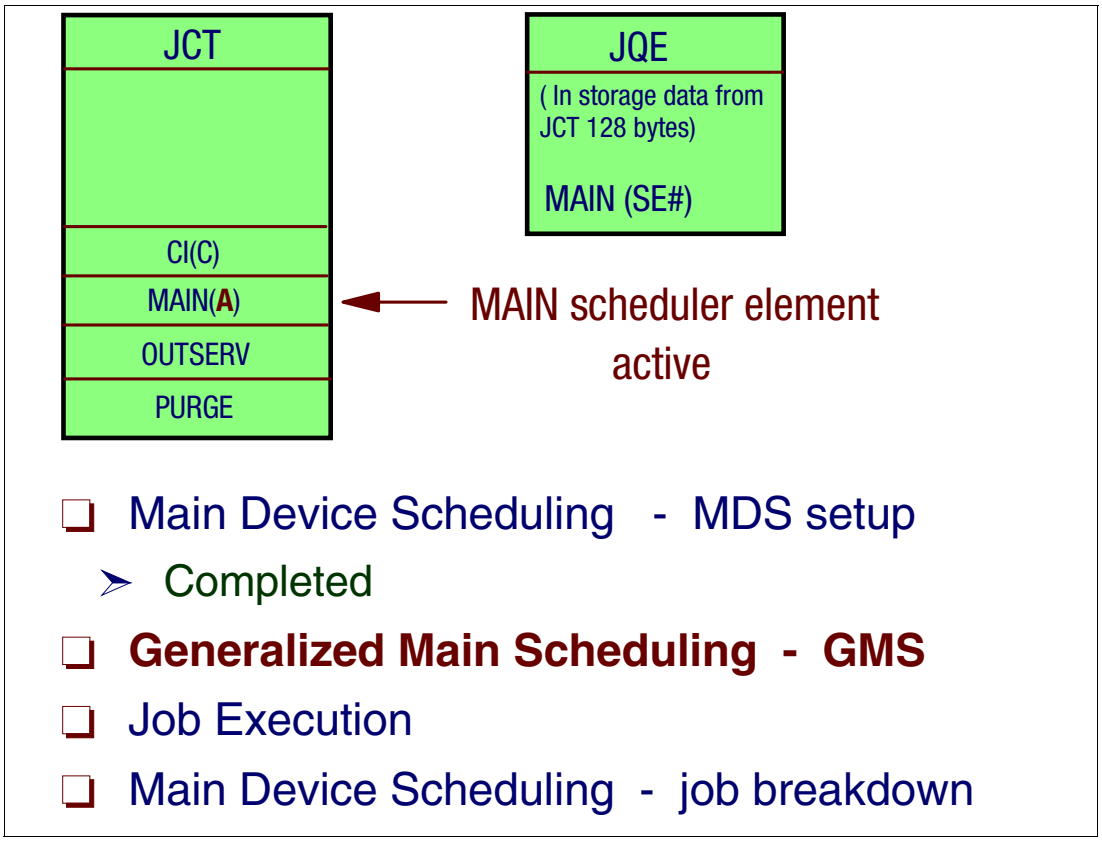
The FCTs (IATMSMS), that process the GMS phase of job processing, are built into the FCT chain during JES3 initialization. The GMS FCT gets posted when an initiator makes a job select request from that main and when a job is queued for system select.

IATMSMS GMS functions:

- Job selection and termination,
- Job flush processing
- JES3 initiator allocation and unallocation
- Resetting of job status at connect time for JES3 main processors.

In this visual, there are three mains, SC65, SC70, and SC64.

9.3 Job active in main scheduler element - GMS phase



Generalized main scheduling - GMS

Generalized main scheduling (GMS) is the second phase of job processing under the MAIN scheduler element. When a job completes MDS processing, it enters the GMS system select phase of processing (RQ index set to RQSELECT). Jobs in the GMS system select phase are waiting for an initiator's job select request.

The GMS routines schedule a job after it is placed on the selection queue, control the workload, and maximize system throughput. GMS provides a flexible framework for establishing priority between job classes within groups and between groups.

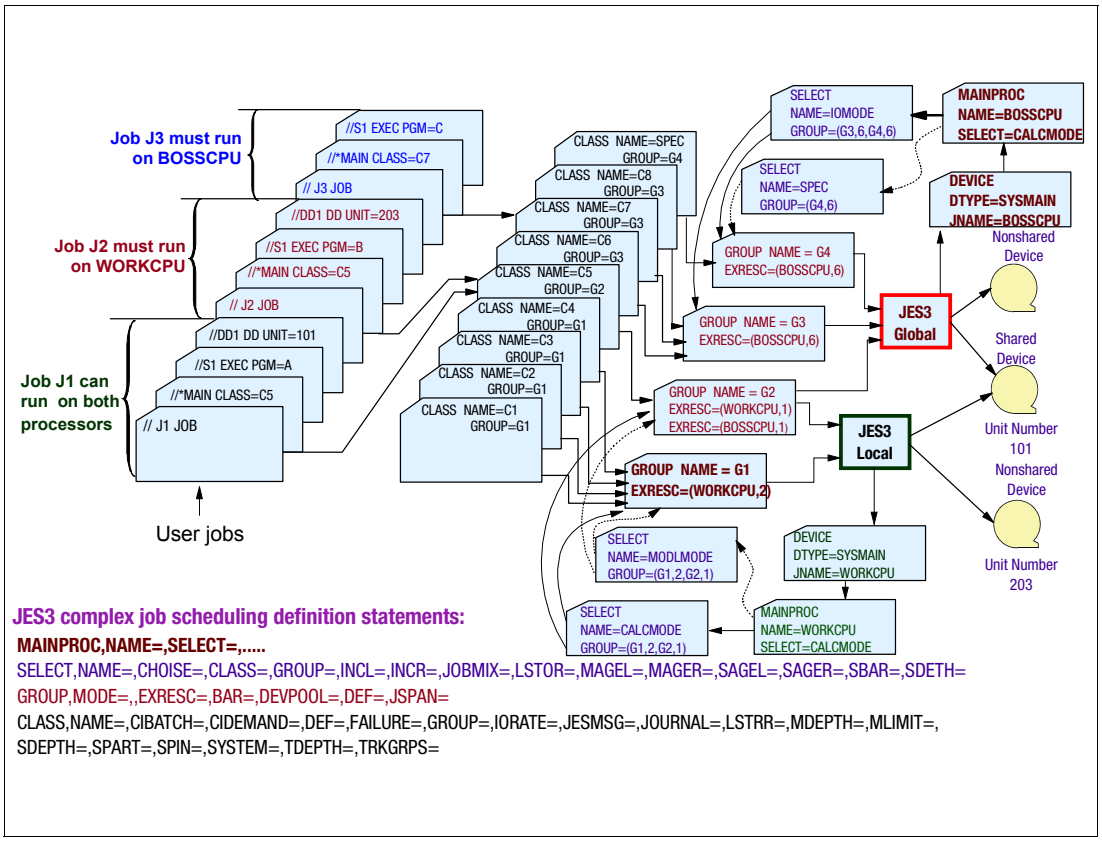
The GMS service allows many controls over job scheduling without requiring operator action or intervention. Some of these controls are:

- Job priority
- Job interaction within a group
- Job class mix
- Explicit and implicit processor dependency
- Logical storage size and I/O rate differentiation. Ignored when MODE=WLM specified
- Initiator availability
- Related job sequencing or dependent job control (DJC)

Initiator job select request

In response to an initiator "job select" request, GMS routines examine the jobs that are available for execution and determine which job should be scheduled to meet the installation-defined throughput objectives for the job.

9.4 GMS scheduling algorithms - JES3-managed initiators



GMS scheduling algorithms- JES3-managed initiators

When a job in a JES3-managed job class group has completed processing in MDS, the job is placed in the queue of jobs awaiting selection for execution. This queue is ordered by job priority, with the last jobs to arrive being placed last within the priority. Thus, the time at which a job completes setup processing partly determines its place in the queue.

Job selection

When a request for a job arrives from the JES3-managed initiator, the job is ready to be selected for execution. Jobs that are requested by JES3-managed initiators are still queued by priority. JES3-managed initiator job selection is driven by how you define job processing in the GROUP, CLASS, and SELECT initialization statements. All jobs classes are defined to JES3 by using the CLASS initialization statement. Each such class is associated with a job class group that is defined using the GROUP initialization statement. The GROUP initialization statement specifies the number of initiators to allocate on each system and how the initiators should be allocated and deallocated. A job class group can have one or more job classes associated with it.

JES3 starts initiators

The MVS start command issued by JES3 to start an initiator:

```
S INIT.groupname IATMSMS
```

After an initiator is started, it asks the JES3 global processor for a job. JES3 then examines the GMS select queue associated with the initiators job class group to see if there are jobs

that can be selected to run. After a job is selected, JES3 reads the job's control blocks and sends the initiator all the information it needs to setup the job for execution. When the job completes execution, the initiator returns the job to JES3. JES3 then updates the job's control blocks to show that the job has completed execution.

Job scheduling definition statements

For jobs requiring JES3-managed initiators, JES3 controls the initiator selection process by first determining whether the select mode on the processor on which the request originated includes the IORATE parameter (from the CLASS initialization statement) as a factor. This determination is made on the basis of what CHOICE parameter on the SELECT initialization statement was specified. If CHOICE was BMIX or FMIX, then IORATE is a factor in this job selection. If IORATE is a factor in this selection, the job selection algorithm computes the best and alternate rates for later use. If the CHOICE parameter was other than BMIX or FMIX, then IORATE is not a factor and best and alternates rates are not computed.

The best and alternate I/O rates are based on the total number of initiators with started jobs on the particular processor, and on the JOBMIX parameter specified on the SELECT initialization statement for this processor. For each number of active initiators, a set of jobs with low, high, and medium I/O rates is specified by the JOBMIX parameter. When the I/O rate computation begins, it determines the number of active initiators, and from this, the number of jobs for low, high, and medium I/O rates that the JOBMIX parameter specifies for this number of started initiators. The low, high, and medium numbers specified during initialization are the numbers of jobs that ideally should be executing for this total number of active initiators. The algorithm then computes which I/O rate (low, high, or medium) is farthest from the ideal, as specified during initialization. The I/O rate that most needs to be increased to meet the ideal becomes the best rate, and the one that needs to be increased second-most becomes the alternate rate. In case of ties, the choice favors low, then high, then medium for best or alternate. As far as I/O rate is a factor in job selection, the choice favors the I/O rate that needs to be increased the most.

GMS job selection

The job selection algorithm next moves to the queue of jobs ready for execution and looks up the first job in the group of the requesting initiator. The algorithm determines whether this first job is in hold status, either as the result of an operator command or because it is part of a DJC network. If the job is in hold status, the algorithm determines whether it has reached the end of the group's selection span. The JSPAN parameter on the GROUP initialization statement defines the span.

If the job being tried as a candidate for job selection could be executed on the processor from which the request originated, the algorithm determines the candidate's job class. The CLASS initialization statement parameters MDEPTH, MLIMIT, TDEPTH, and TLIMIT are all checked for this class. If any of these limits are met or exceeded for this job candidate, the scan checks for JSPAN and BAR parameters on the GROUP statement. If these are not exceeded, it moves to the next job candidate in this group and starts again.

The maximum number of job selection candidates examined in response to a job selection request is specified by the JSPAN parameter on the GROUP initialization statement. If the number of jobs scanned as candidates for selection reaches the value specified by JSPAN, the algorithm terminates the job selection pass. In this case, the initiator continues waiting, and the job-select request remains queued.

If the value specified by JSPAN has not been reached, the algorithm determines whether a priority barrier is effective at this point. If a priority barrier is reached, then the result is the same as if JSPAN has been reached: no more jobs in the queue can be scanned.

If the job has a scheduling environment (that is, the SCHENV= keyword was specified on the JOB statement), the job selection algorithm checks if the scheduling environment is available. If it is not available, the scan checks for the JSPAN and BAR parameters on the GROUP initialization statement. If these are not exceeded, it moves to the next job candidate in this group and starts again.

Using logical storage parameters

The selection algorithm next determines whether this job can fit into the available logical storage of this processor. If not, the algorithm checks whether the CHOICE parameter on the SELECT statement is set for FJOB (first job of the group in the queue). Specifying the FJOB parameter just tries the first job of the group; if it can be executed, it is selected, and if it cannot, no job is selected.

If the job selection candidate fits into storage on the processor and the CHOICE parameter is not FJOB, the algorithm looks back to the earlier determination of whether IORATE is to be considered in this selection. If it is not, a suitable job has been found, and this job is returned to the requesting initiator. If IORATE is a factor (CHOICE is specified as BMIX or FMIX), the algorithm compares the I/O rate of this job with its earlier determination for best I/O rate. If the I/O rate of this job is the same as the best I/O rate, this job is returned to the initiator as the selected job. If the I/O rate of this job is the same as the alternate I/O rate and the CHOICE parameter is set for FMIX, this job is returned to the initiator as the job selected. If the job does not match the best or alternate I/O rate, the algorithm checks for JSPAN and BAR and starts examination of the next job of the group on the queue.

Initialization statements for GMS

The visual shows the connection between MAINPROC, SELECT, GROUP, and CLASS initialization statements and how user jobs will be selected for execution. Three jobs (J1, J2, and J3) of class C5, C5, and C7 are in the input stream, and two of these jobs (J1 and J2) references a specific I/O device number with a DD statement UNIT parameter. One of these jobs (J2) references a non-shared device (203) and, therefore, must execute on the JES3 local processor (designated WORKCPU) in this example. Since job class C5 is available on the local main (WORKCPU), the job (J2) will for execution. The other job (J1) of class C5 references a specific I/O device number that is shared between the JES3 global processor and the JES3 local processor. Since class C5 is assigned to group G2 and has execution resources allocated to both processors, the job *could* run on either main if both the global and local mains *would* select class C5 jobs. However, the global main (BOSSCPU) does not select jobs from the job class group G2's job classes. Thus, job (J1) will be selected by the local main (WOTKCPU). The third job (J3) of class C7 will be selected by the global main (BOSSCPU). The global main's select mode is IOMODE. Under the IOMODE select mode jobs are selected from job class groups G3 and G4. Jobs in classes C6, C7, and C8 are assigned to group G3 and class SPEC to G4.

9.5 JES3 main definition - MAINPROC statement

- **MAINPROC,**NAME=main,SYSTEM=JES3,FIXPAGE=fixedpages,
  ID=msgprefix,MDEST=code,JESMSGLMT=(WTO_limits,interval),
  PRTPAGE=(csapages,auxpages),**SELECT=selmode**,
  SPART=partionname,TRKGRPS=(prigrps,secgrps),
  USRPAGE=nn

```
*I MAIN=ALL
IAT8643 MAIN INQUIRY RESPONSE
INFORMATION FOR MAINPROC SC70
  STATUS=(ONLINE,NOT-CONNECTED,DOWN,LOCAL)
INFORMATION FOR MAINPROC SC65
  FMID=HJS7750, STATUS=(ONLINE,CONNECTED,ATTACHED,GLOBAL)
MAINPROC INQUIRY RESPONSE COMPLETE
*I MAIN=SC65 X
IAT8643 MAIN INQUIRY RESPONSE 347
INFORMATION FOR MAINPROC SC65
  FMID=HJS7750, STATUS=(ONLINE,CONNECTED,ATTACHED,GLOBAL), PLEVEL=20,
  SLEVEL=02, ID='SC65=', MDEST=M2, SELECT=SELA, SPART=NONE,
  TRKGRPS=(1,1), JESMSGLMT=(00000000,00010), FIXPAGE=00005,
  PRTPAGE=(00025,00025), USRPAGE=00004
MAINPROC INQUIRY RESPONSE COMPLETE
*V SC70 OFF
IAT8180 SC70     VARIED OFFLINE TO JES3 ON GLOBAL
```

MAINPROC statement

The MAINPROC initialization statement to defines a processor as a JES3 main. The initialization stream must include one MAINPROC statement for each main that you want to define to JES3. JES3 supports up to 32 processors in a complex.

MAINPROC statement parameters

The MAINPROC initialization statement is used to define a processor as a JES3 main. The initialization stream must include one MAINPROC statement for each main that you want to define to JES3. The parameters are as follows:

NAME= Specifies the name of a JES3 main. You also use this name in operator commands and in the JES3 CLASS, DEVICE, GROUP, MSGROUTE, and SETACC initialization statements to refer to the main. It must be the first or second parameter on the statement.

Ensure that the name you specify matches the name you specify on the SYSNAME parameter in the IEASYSxx member of the MVS SYS1.PARMLIB data set that is used to IPL the system.

SYSTEM= Specifies that JES3 is running on this main. You do not need to code this parameter.

FIXPAGE= Specifies the number of pages of USAM protected data buffers (PBUFs) that JES3 is to fix in storage during JES3 initialization.

ID= Specifies a message prefix for every message received from this JES3 main and logged on the MDEST console (determined by the MDEST parameter).

JESMSGLMT= Specifies how you want JES3 to control the issuing of messages by establishing message limits.

MDEST= Specifies the routing information for messages that are about this main. This value may be specified as a JES3 destination class selected from the range M1 to M32, or as an MVS routing code.

PRTPAGE= Specifies the number of pages of storage that JES3 may use as USAM protected data buffers (PBUFs). For more information about defining buffers see *z/OS JES3 Initialization and Tuning Guide*, SA22-7549.

SELECT= Specifies the name of the job selection mode to be initially assigned to this main. The scheduling controls associated with this job selection mode should be appropriate for the main being defined by this MAINPROC statement. The selmode must match the NAME parameter on a SELECT initialization statement (except if the default JS3BATCH is used).

SPART= Specifies the spool partition that JES3 is to use for jobs that execute on the main defined by this statement. The partition name must match a partition name specified on an SPART statement. To specify the default spool partition, omit the SPART parameter.

TRKGRPS= Specifies the number of track groups (as defined by the GRPSZ parameter on the BUFFER or SPART statement) JES3 is to allocate to jobs that execute on this main.

GMS considerations

From a GMS point of view, there are two pieces of information that one needs to be remember.

1. The name of the processor will be used in the definition of initiator resources and constraints.
2. The initial select mode is defined on the MAINPROC statement. This is the initial or default “initiator configuration” for this main processor. The important thing to notice is that each processor has its own select mode.

*I MAIN=main command

The **`*I MAIN=main`** command displays information about main processors, the global processor, or all main processors.

There are two forms of the command, short and extended.

9.6 GMS scheduling defaults

- Main selection mode
  - SELECT statement — Default JS3BATCH
- Groups eligible under SELECT mode
  - GROUP statement — Default JS3BATCH
- Job classes in each GROUP
  - CLASS statement — Default JS3BATCH

GMS scheduling defaults

The default job selection environment is established when the SELECT, GROUP, and CLASS initialization statements are not specified. Jobs are then selected by priority only, in first-in-first-out order. In this case, two initiators are started on each processor, and default selection mode, group, and class are established with the name JS3BATCH.

SELECT statement

The SELECT initialization statement defines scheduling controls associated with a particular job selection mode. The initial job selection mode is assigned to a JES3 main using the SELECT parameter on the JES3 MAINPROC initialization statement. If a MAINPROC statement does not indicate a selection mode, the SELECT statement default values are assigned to that main.

GROUP statement

The GROUP initialization statement defines the characteristics of a JES3 job-class group and whether the initiators managed by this group are WLM managed or JES3 managed. A GROUP statement must define each job class group (except for the default group, JS3BATCH) named on a CLASS initialization statement.

CLASS statement

The CLASS initialization statement defines the characteristics of JES3 job classes. A CLASS statement must define each class that can appear on the //*MAIN JECL statement or JOB JCL statement.

9.7 Job selection mode - SELECT statement

- **SELECT**,NAME=modename,CHOICE=criteria
  ,**CLASS**=([jobclass,...] | [/jobclass,...]
  ,**GROUP**=([groupnam,initnums,...] | [/groupnam,...]
  ,INCL=nn,INCR=nn,JOBMIX=(n1,...),LSTOR=nnnnn
  MAGEL= nn,MAGER=nnn,SAGEL=nn,SAGER=nnn
  ,SBAR=[PRTY | nn],SDEPTH=nnn
- SELECTexample
  SELECT,NAME=SELA,GROUP=(A,9,S,8),CLASS=(A,B,S,T),
  SBAR=15,SAGEL=10,SDEPTH=20, INCR=4,MAGEL=10
  LSTOR=32000

```
*I G SC65 S MODE
IAT8930 SELECT - MODE     - SELA     - SC65
IAT8599 INQUIRY ON GMS COMPLETE
*F G SC65 S MODE SELA
IAT8456 GMS MODIFY - COMPLETE NO ERRORS     - SC65
IAT2002 LSTOR=32000 ALLOC=000000000 AVAIL=32000 SC65
```

SELECT statement

The SELECT initialization statement defines scheduling controls associated with a particular job selection mode. The initial job selection mode is assigned to a JES3 main using the SELECT parameter on the JES3 MAINPROC initialization statement. If a MAINPROC statement does not indicate a selection mode, the SELECT statement default values are assigned to that main. Each select mode defined can be dynamically changed using the:

```
*F G,main,S operator command
```

In addition, the commands **`*F G,main,G`** or **`*F G,main,C`** (C stands for class) can indirectly affect the select mode. A SELECT statement must be specified for each select mode indicated on a MAINPROC statement or in a **`*F G,main,S`** command.

SELECT statement parameters

The DELECT statements parameters are as follows:

NAME= Indicates the alphameric name of the job selection mode. This modename can be referenced with the SELECT parameter of the MAINPROC statement or the MODE subparameter on the **`*F,G,main,S`** command.

If you do not specify a SELECT initialization statement in your initialization stream, a default of JS3BATCH is generated. If you specify a SELECT initialization statement, then the NAME= parameter must be specified.

CHOICE= Specifies the job selection criteria to control the order of job selection on the main. The criteria are based on the size of the job and its I/O rate. JES3 uses

the specified scheduling choice to select the most suitable jobs for execution. The choices available are BMIX, FFIT, FJOB, and FMIX:

BMIX - indicates that the first job in the queue that fits on the main and has a best mix IORATE is scheduled. JES3 determines IORATE from the job's //*MAIN statement or from the CLASS statement (IORATE parameter) that defines the class to which the job belongs. If none of the jobs meet these criteria, the first job that fits and has the alternate mix IORATE is scheduled. If none of the jobs in the queue have the alternate IORATE, the first job to fit will be scheduled.

FFIT - specifies that the first job in the queue that fits on the main be scheduled. If none of the jobs fit, no job will be scheduled.

FJOB - specifies that the first job in the queue be scheduled if it fits on the main. Otherwise, no job will be scheduled.

FMIX - indicates that the first job in the queue that fits on the main and also meets the best or alternate IORATE requirement be scheduled. If none of the jobs in the queue meet these criteria, the first job that fits is scheduled.

CLASS= Identifies either job classes that can be scheduled under this mode or, if a "/" precedes all class names, indicates those job classes that cannot be scheduled under this mode. Use the CLASS parameter to identify classes to be treated in a specific way or to include or exclude classes from a group for purposes of scheduling under this mode.

GROUP= Indicates either the names of job class groups (groupnam) that can be scheduled under this SELECT mode and the number of initiators (initnum) to be assigned to each of these groups; or, if a "/" precedes all the group names, this parameter identifies those job class groups that cannot be scheduled under this mode. The subparameter groupnam must match the NAME parameter on a GROUP statement. Jobs in the groups defined here as eligible for scheduling will not be scheduled until the execution resources are allocated. (Allocation is controlled by the allocation option with the EXRESC parameter on a GROUP statement.)

Note: (1.) The *initnum* value on the GROUP parameter of the SELECT statement overrides the value of the initcnt parameter on the EXRESC parameter of the GROUP statement.

(2.) If the GROUP parameter specifies job class group that is WLM managed, the initiator count is ignored. If this job class group is switched to JES3 management using the `*F G` command, the initiator count that was specified will be in effect.

(3.) If the initiator count is omitted and the job class group is JES3 managed, a value of zero is assumed.

(4.) If you omit the SELECT statement GROUP parameter, all groups are eligible for scheduling. The SELECT statement may also specify priority aging rules for GMS and MDS processing.

INCL= Specifies a limit within the range of 0 to 15 past which a job's priority cannot be incremented by JES3 main device scheduling (MDS).

INCR= Specifies a decimal number from 0 to 15 that is automatically added to the priority of the job which is set up.

JOBMIX= Specifies 1 to 45 values (ranging from 1 to 15) which indicate the optimal I/O rate job mix for 1 to 15 active initiators. See *JES3 Initialization and Tuning Guide* for JOBMIX default values.

LSTOR= Defines the logical storage resource, in 1024-byte blocks, for a main. The purpose of this parameter is to allow the system programmer to control the number of jobs to be scheduled on a main in such a way as to fully use central dynamic storage and at the same time minimize the probability of excessive paging or thrashing. Initially it is recommended that LSTOR be defined to obtain maximum.

MAGEL= Specifies an aging priority limit (0-15) beyond which a job cannot be aged during GMS.

MAGER= Specifies the number (0-255) of times a job must be eligible for aging before its job priority is actually incremented. If 0 is specified, no aging is done.

SAGEL= Specifies an aging priority limit (0-15) beyond which a job cannot be aged during MDS setup for the job.

SAGER= Specifies the number of times (0-255) a job must be eligible for aging during JES3 MDS processing before its job priority is actually incremented. If 0 is specified, no aging is done.

SBAR= Specifies a job priority barrier. Jobs equal to or above this barrier which cannot obtain all their required resources (volumes, data sets, and available devices) will reserve resources as they become available to prevent lower priority jobs from obtaining them.

PRTY - indicates that the priority of the first job that cannot be setup is the priority barrier.

nn - specifies a job priority level from 0 to 15.

16 - indicates there is no job priority barrier.

SDEPTH= Specifies the maximum number (0-255) of jobs requiring operator mounts that may be set up at one time on any main for which this select mode is active. SDEPTH allows mains with different non-shared device configurations to be given a variable number of setup jobs, depending on the number of devices associated with the main.

Note: When deferred mounting is either specified in the JCL for any device (for example, UNIT=(TAPE,,DEFER)) or implicitly requested, by using tape library dataserver devices, JES3 bypasses pre-execution mount processing and does not include the job in its CLASS setup depth (SDEPTH) count unless DEFERCT=YES has been specified on the SETPARAM initialization statement, or the `*F S` operator command.

9.8 MDS job setup selection - GMS SELECT statement

```
SELECT,NAME = modename
        ,SBAR = PRTY | nn | 16          (DEF:  16)
        ,SAGEL = nn | 14                (DEF:  14)
        ,SAGER nnn | 0                  (DEF:   0)
        ,SDEPTH = nnn | 255             (DEF: 255)
        ,INCL= nn | 14                  (DEF:  14)
        ,INCR= nn | 1                   (DEF:   1)
```

SBAR	Job priority barrier - Jobs at barrier or above reserve resources
SAGER	Aging - Jobs can advance to next priority level
SAGEL	Aging limit priority level
SDEPTH	Maximum number of jobs requring operator mounts
INCL	Priority increment limit
INCR	Priority increment

MDS job setup selection

Module IATMDSL provides MDS job setup selection, final resource allocation, and issuance of operator messages to perform the initial volume mounting required for job execution.

IATMDSL first finds a job for which to attempt allocation. It does this by checking the dynalloc queue, the restart queue, and the allocate queue. For jobs already tried, it checks the ‘devices-required’ counts to see if the job has a chance to allocate successfully.

IATMDSL selects a setup processor based on the job's main mask, and the processor's setup depth. At this time the main mask is an indicator of processors which are eligible after fetch processing and where SMS resources are also available.

After selecting the setup main, if the job is unable to reserve resources and the job's JST is not in storage, the ARL (allocation requirements list of the resources that a job was not able to obtain on a previous allocation attempt) pre-allocation scan will be performed if the job has an ARL. The JST will not be read and allocation will not be attempted, if the pre-allocation scan fails. Also this setup main will be bypassed and the next eligible main will be checked.

IATMDSL invokes IATMDAL to try to allocate the job on the setup processor, if the JST was read in. If allocation is successful, mount messages are issued if appropriate, and IATMDSB is called to hard allocate the job's resources. Also, if the job had an ARL, the ARL is deleted.

If the job requires volumes to be mounted, then the job is put on the verify queue to wait for its volumes to be mounted.

If job allocation fails on the setup processor, another processor will be tried, if available; if no other processors are eligible, and the job stays on the allocate queue, the ARL will be built or updated and the JST will be released. If the ARL cannot be built or modified, the JST will be rewritten.

MDS Parameters on SELECT Statement

The SELECT initialization statement defines scheduling controls for a particular job selection mode. The initial job selection mode is assigned to a JES3 main using the SELECT parameter on the JES3 MAINPROC initialization statement. If a MAINPROC statement does not indicate a selection mode, the SELECT statement default values are assigned to that main. Each select mode defined can be dynamically changed using the **`*F G,main,S`** operator command. In addition, the commands **`*F G,main,G`** or **`*F G,main,C`** can indirectly affect the select mode. A SELECT statement must be specified for each select mode indicated on a MAINPROC statement or in a **`*F G,main,S`** command.

Several parameters on the SELECT statement affect the operation of MDS allocation on a processor basis (SDEPTH, SBAR, INCR, INCL, SAGER, SAGEL). Through these parameters, MDS allocation may be biased toward one processor (a larger SDEPTH), devices may be reserved but not entirely allocated to one processor (a higher SBAR), and jobs on a specific processor may be favored for selection (higher INCR, INCL, SAGER, and SAGEL parameters):

SBAR Specifies a job priority barrier. Jobs equal to or above this barrier which cannot obtain all their required resources (volumes, data sets, and available devices) will reserve resources as they become available to prevent lower priority jobs from obtaining them.

PRTY- Indicates that the priority of the first job that cannot be setup is the priority barrier.

nn - Specifies a job priority level from 0 to 15.

16 - Indicates there is no job priority barrier.

SAGEL Specifies an aging priority limit (0-15) beyond which a job cannot be aged during MDS setup for the job.

SAGER Specifies the number of times (0-255) a job must be eligible for aging during JES3 MDS processing before its job priority is actually incremented. If 0 is specified, no aging is done.

SDEPTH Specifies the maximum number (0-255) of jobs requiring operator mounts that may be set up at one time on any main for which this select mode is active. SDEPTH allows mains with different nonshared device configurations to be given a variable number of setup jobs, depending on the number of devices associated with the main.

INCL Specifies a limit within the range of 0 to 15 past which a job's priority cannot be incremented by JES3 main device scheduling (MDS). Note that only jobs requiring volume mounting or referencing a volume that was mounted for another job are incremented by setup.

INCR Specifies a decimal number from 0 to 15 that is automatically added to the priority of the job which is set up. If a job has a priority of 5 when it is set up, and INCR=4 is specified, the job's priority is elevated to 9 (or to the value specified in the INCL parameter, whichever is less) after the devices have been allocated and set up. This parameter expedites the processing of jobs once devices have been assigned to them.

9.9 SELECT mode commands

```
*I G,SC65,S
 IAT8930 SELECT - SBAR     - 00000015 - SC65
 IAT8930 SELECT - MAGER    - 00000000 - SC65
 IAT8930 SELECT - SAGER    - 00000000 - SC65
 IAT8930 SELECT - MAGEL    - 00000010 - SC65
 IAT8930 SELECT - SAGEL    - 00000010 - SC65
 IAT8930 SELECT - SDEPTH   - 00000020 - SC65
 IAT8930 SELECT - INCR     - 00000004 - SC65
 IAT8930 SELECT - INCL     - 00000014 - SC65
 IAT8930 SELECT - CHOICE   - FIRSTFIT - SC65
 IAT8930 SELECT - LSTOR=32000 ALLOC=000000000  AVAIL=32000 - SC65
 IAT8930 SELECT - MODE     - SELA     - SC65
 IAT8930 SELECT - MAXI     - ALL      - SC65
 IAT8599 INQUIRY ON GMS COMPLETE
*F G,SC65,S,SBAR,14
 IAT8456 GMS MODIFY - COMPLETE NO ERRORS     - SC65
*I G,SC65,S,SBAR
 IAT8930 SELECT - SBAR     - 00000014 - SC65
 IAT8599 INQUIRY ON GMS COMPLETE
```

SELECT mode commands

Each select mode defined can be dynamically changed using the **`*F G,main,S`** operator command. In addition, the commands **`*F G,main,G`** or **`*F G,main,C`** can indirectly affect the select mode. A SELECT statement must be specified for each select mode indicated on a MAINPROC statement or in a **`*F G,main,S`** command.

Use the **`*I G`** command to obtain the status of GMS components of JES3 and to display the name of the spool partition assigned for a specific main or all mains. A main's spool partition contains the spool data for each job that runs on that main unless other partitions were specifically assigned for the job's job class, SYSOUT data, or in the job's //*MAIN control statement.

On the visual, the **`*I G,SC65,S`** command displays the current SELECT mode options on system SC65.

To change any of the options, the **`*F G,SC65,S,SBAR,14`** command changes the SBAR value from 15 to 14.

To check an individual parameter, the **`*I G,SC65,S,SBAR`** displays the current value of the SBAR parameter.

9.10 Job-class group definition - GROUP statement

- **GROUP**,NAME=groupname,MODE=[JES|WLM]
  ,EXRESC=([procname|*ALL],initcnt,storsize,exopts)
  ,BAR=[PRTY|nn],DEF=YES
  ,DEVPOOL=(devopt,devname,devcount,devnum,options)
  ,JSPAN=nnnnn
- Example

  GROUP,NAME=GRPA,DEF=YES,
  EXRESC=(*ALL,9,,IPL,MANUAL),JSPAN=ALL,MODE=JES

```
*I G SC65 G
IAT8932 GROUP - GRPA - STATUS=ON  DI=0009 AI=0009 UI=0000 ALLOC=IPL UNAL=MAN BAR=NO JSPAN=ALL MODE=JES - SC65
IAT8932 GROUP - GRPJ - STATUS=ON  DI=0009 AI=0009 UI=0000 ALLOC=IPL UNAL=MAN BAR=NO JSPAN=ALL MODE=JES - SC65
IAT8932 GROUP - GRPS - STATUS=ON  DI=0008 AI=0008 UI=0000 ALLOC=IPL UNAL=MAN BAR=NO JSPAN=ALL MODE=JES - SC65
IAT8599 INQUIRY ON GMS COMPLETE
*F G SC65 G GRPA INIT 10
IAT8456 GMS MODIFY - COMPLETE NO ERRORS     - SC65
IAT2003 MPAINIT=0027 DI=0010 AI=0010 UI=0000 GRP=GRPA       - SC65
S INIT.GRPA        IATMSMS
IAT6100 ( DEMSEL ) JOB INITJES3 (JOB25631), PRTY=15, ID=STC
ICH70001I STC      LAST ACCESS AT 15:12:28 ON MONDAY, NOVEMBER 10, 2008
IEF695I START INIT     WITH JOBNAME INIT     IS ASSIGNED TO USER STC     , GROUP SYS1
IEF403I INIT - STARTED - TIME=15.12.28 - ASID=0086 - SC65
```

GROUP statement

The GROUP initialization statement defines the characteristics of a JES3 job-class group and whether the initiators managed by this group are WLM managed or JES3 managed. A GROUP statement must define each job class group (except for the default group, JS3BATCH) named on a CLASS initialization statement.

NAME= Specifies the name of a job class group. A job class is assigned to this job-class group by placing this group name on its CLASS statement.

MODE= Specifies whether the initiators managed by this group are WLM managed or JES3 managed.

EXRESC= Defines the execution resources, such as initiators, processors, and devices, which you want assigned to this job class group. Jobs in this group use devices assigned to the group to satisfy requests for mountable volumes. An EXRESC parameter must be specified for each main on which this group may be scheduled.

The initiator resources are defined on a main processor basis. Dedicated devices can be defined on either a processor basis or a complex wide basis:

- Initiator management is performed at the group level.
- Initiators can only process jobs from classes that comprise the group.

The EXRESC parameter must still be specified even if MODE=WLM, because JES3 still needs to know which system's jobs in this group are allowed to

execute. If a particular system name does not appear in the EXRESC parameter, jobs in that group will not be allowed to run on that system.

For the full description of the EXRESC parameter see *z/OS JES3 Initialization and Tuning Guide*, SA22-7549.

DEF= Indicates that this is the default group to be assigned to all job classes that have no GROUP parameter on the CLASS statement. Only one GROUP statement should specify the DEF=YES parameter.

DEVPOOL= Specifies that a pool of devices is to be dedicated to jobs in this group for volume mounting on all mains for which this group may be enabled. If the DEVPOOL parameter is specified, devices may not be dedicated in the EXRESC parameter.

The POOLNAMS parameter on the SETNAME statement, with appropriate XTYPE definitions on the DEVICE statement, may be used to define unique sets of devices with the shareability (physical attachment) desired for this group.

For the full description of the DEVPOOL parameter see *z/OS JES3 Initialization and Tuning Guide*, SA22-7549.

JSPAN= Specifies the number of jobs in this group to be examined in selecting a job to be scheduled.

nnnnn - indicates a decimal number from 1 to 65534.

ALL - indicates that all the jobs in the group are to be examined.

Execution resources allocation/release

JES3 supports several options for the starting and stopping of initiators:

- ...IPL,MANUAL,...

  This option allows JES3 to start the allowed number of initiators after the processor is connected following an IPL of the processor. Furthermore, JES3 is not to terminate the initiators if no work is available. Initiator management will be done by the operations staff rather than JES3. This can result in idle initiators.

- ...MANUAL,MANUAL,...

  This option places complete control of initiator management in the hands of the operations staff. JES3 will not make any adjustments to the number of initiators that are active. The group will be disabled following IPL and connect processing. It will be necessary for the operator to enable the group before the specified number of initiators will be started on the processor. This option will result in idle initiators. This is the most popular option(s) used by installations.

- ...DEMAND,MANUAL,...

  JES3 will start only as many initiators in the group on the specified processor as is necessary to place the available jobs into execution up to the maximum number allowed. However, JES3 is not allowed to terminate idle initiators when no work is available. This can result in idle initiators.

- ...DEMAND,DEMAND,...

  JES3 will start only as many initiators in the group as is necessary to place the available jobs into execution on the processor up to the maximum number allowed. Furthermore, JES3 will terminate idle initiators when no work is available. This will cause initiators to “come and go” depending on the availability of work.

- ...DYNAMIC,DYNAMIC,...

  JES3 will start all of the initiators allowed in the group whenever it detects a job eligible for execution. All of the initiators will stay active until there is no available work, at which time JES3 will terminate all of the initiators. This is similar to the previous set of options.

GROUP statement considerations

There are some options that should be avoided or you will see initiators start and stop or constantly have the operators manually restarting initiators. These include:

- ...DYNAMIC,DEMAND,...
- ...MANUAL,DEMAND,...
- ...MANUAL,DYNAMIC,...
- ...IPL,DEMAND,...
- ...IPL,DYNAMIC,...

MANUAL or DYNAMIC allocation and unallocation options make good sense in the environment where:

- Work is released for processing on a predefined schedule in a specific window (e.g. turned loose at 4 PM to be completed by 6 AM the following morning)
- Work in the group is available in sufficient quantities to prevent the thrashing of initiators.

For work that arrives with great inter-arrival times, DEMAND allocation and unallocation options make sense.

In the previous example, the initiators in GRPA are started one at a time as needed on each processor and remain until there is no work available for any of the initiators on that processor. At that time, all of the initiators on the processor will be terminated automatically by JES3 without operator intervention.

*F G main G group *opt* operator commands

The *opt* options include:

ON or OFF	Specifies that the group named is to be enabled (ON) or disabled (OFF) for scheduling on the named main.
INIT,nnn	Specifies the number of dedicated initiators to be assigned to the JES-managed group. The INIT option can only be modified when the group is in JES mode.
ALLOC,opt	Specifies that the group initiator allocation option is to be modified. The allocation option determines when the execution resources are to be allocated. Four options are available: DEMAND, DYNAMIC, IPL, or MANUAL.
UNALLOC,opt	Specifies that the deallocation option is to be modified. The deallocation option determines when the execution resources are to be released. Three options are available: DEMAND, DYNAMIC, or MANUAL.
BAR,nn	Specifies that the JES-managed group scheduling barrier be changed.
JSPAN,nnn	Specifies that the JES-managed group job span be changed.
MODE,JES | WLM	Specifies whether a group is to be managed by JES or by WLM.

9.11 JES3 job class definition - CLASS statement

- **CLASS**,NAME=classname,CIBATCH=[CLASS | JOB | ANY]
  ,CIDEMAND=[CLASS | JOB | ANY],DEF=YES
  ,FAILURE=[CANCEL | HOLD | PRINT | RESTART],GROUP=groupname
  ,IORATE=[MED | HIGH | LOW],JESMSG=[LOG|NOLOG]
  ,JOURNAL=[NO | YES],LSTRR=nn,MDEPTH=(procname,dept,...)
  ,MLIMIT=(classnames,limit,...),PRTY=nn,SDEPTH=nnn
  ,SPART=partitionname,TDEPTH=nnn,,TLIMIT=(classnames,limit,...)
  ,SYSTEM=[ANY | LOCAL | GLOBAL | procname \ /procname,...)
  ,SPIN=[NO | YES | +hh:mm | hh:mm | nnn | nnnK | nnnM]
  - CLASS example
    CLASS,NAME=A,SYSTEM=(SC65,SC70),GROUP=GRPA,
    MDEPTH=(SC65,5,SC70,8),MLIMIT=(SC65,B,3),MLIMIT=(SC70,B,5),
    TDEPTH=9,TLIMIT=(B,7)
    CLASS,NAME=B,SYSTEM=(SC65,SC70),GROUP=GRPB,
    MDEPTH=(SC65,3,SC70,5),MLIMIT=(SC65,A,8),MLIMIT=(SC70,A,5),
    TDEPTH=5,TLIMIT=(A,7)

CLASS statement parameters

The CLASS initialization statement defines the characteristics of JES3 job classes. A CLASS statement must define each class that can appear on the //*MAIN control statement. You can define up to 255 classes.

The CLASS statement has dependencies on the SPART initialization statement. During a hot start with refresh, JES3 does not process the SPART statement but uses the SPART statement from the last warm or cold start. If you add a SPART statement during a hot start with refresh, JES3 ignores it and issues error messages if the CLASS statement you use references the SPART statement that you attempted to add during a hot start with refresh. The CLASS statement parameters are as follows:

NAME= Specifies the name of the job class. This name corresponds to the CLASS parameter on the //*MAIN control statement. If you omit DEF=YES from all CLASS statements in the initialization stream, JES3 defines JS3BATCH as the default class.

CIBATCH= Indicates whether batch jobs of this class must have CI processing limited to certain processors. If specified for a class, this will override the value specified on the CIBATCH parameter on the STANDARDS statement.

JOB - Indicates CI processing must be performed on a system on which the job is eligible to run. Only the JES3 //*MAIN SYSTEM= JECL statement (NOT the job's scheduling environment) is considered for CI scheduling purposes when determining where the job is eligible to run.

CLASS -Indicates CI processing must be performed on a system on which the job's JOB CLASS is enabled.

ANY - Indicates CI processing may be performed on any processor regardless of job or class eligibility.

CIDEMAND= Indicates whether demand select jobs of this class must have CI processing limited to certain processors. which the job's JOB CLASS is enabled.

DEF= Identifies this class as the default job class. Specify DEF=YES on one CLASS statement only.

FAILURE= Specifies the job recovery option that JES3 is to use when a system failure affects jobs in this class. If a job has been registered as an element of automatic restart management, the FAILURE= keyword is ignored.

CANCEL - Print any job output that is in a SYSOUT class that is specified as TYPE=PRINT. After printing the output, cancel the job. Empty data sets destined for the INTRDR are automatically canceled.

HOLD - Place the job into the hold queue.

PRINT - Print any job output that is in a SYSOUT class that is specified as TYPE=PRINT. Then place the job into the hold queue.

RESTART - Restart the job from the first step. The job will be restarted on the processor on which it was active.

GROUP= Specifies the name of a job class group to which this job class is to be assigned. If the GROUP parameter is omitted, JES3 assigns this class to the default group (JS3BATCH).

IORATE= Specifies the default I/O rate for the jobs in this class. JES3 attempts to balance the mixture of jobs in execution based on the values that you have specified for the JOBMIX and CHOICE parameters on the appropriate SELECT statements. See the description of the SELECT statement and in *JES3 Initialization and Tuning Guide*, SA22-7549.

JESMSG= Specifies whether to allow (LOG) or suppress (NOLOG) writing WTOs and WTORs to the JESMSGLG data set for jobs in this job class. If this JESMSG= is not used, the JESMSG= keyword on the STANDARDS initialization statement is in effect.

JOURNAL= Specifies whether (YES) or not (NO) job-related information is to be saved in a job journal. This parameter may be overridden by individual jobs with the JOURNAL parameter on the //*MAIN JES3 control statement.

MDEPTH= Determines the maximum number of jobs in this class that can be run on the indicated main at any one time. Specifies a decimal number from 0 to 255. JES3 applies the limit to each service class separately where WLM is in use. More initiators than expected can be started where jobs and tasks in a given job class run under different service classes. To have limits applied correctly under WLM, all jobs, in the same job class, need to be in the same service class.

LSTRR= Specifies the logical storage reduction rate (0 to 99), if a job's logical region size is not specified in the LREGION parameter on the //*MAIN control statement. JES3 uses this parameter as a percentage of the region size to determine the job's logical region. If the LSTRR parameter is specified as 0, logical storage processing is effectively disabled for jobs not specifying LREGION. If specified, the LREGION parameter overrides logical storage processing. If LSTOR=0 is specified on the SELECT statement logical storage scheduling is disabled.

MLIMIT= Determines the maximum number of jobs of other job classes that can run on the indicated main and still let JES3 schedule jobs in this class. When any of the limits are exceeded, JES3 does not schedule any more jobs in the class defined by this CLASS statement on that main. JES3 only schedules jobs of this class when the number of jobs running from the other named classes is equal to or less than the assigned limit.

PRTY= Specifies the JES3 job priority (0 to 14) be assigned to each job in this class. The PRTY parameter may be overridden on a job basis by the PRTY parameter on the JOB statement. If you omit this parameter, the value is taken from the PRTY parameter on the STANDARDS statement.

SDEPTH= Specifies the maximum number of jobs in this class requiring MDS operator mounts that can be set up at one time. The value of nnn is a number from 0 to 255. When JES3 counts the number of jobs that are set up, JES3 considers a job to be set up from the time the job enters allocation until the time the job's devices are deallocated.

SPART= Specifies the spool partition that JES3 is to use for jobs in this job class. The partition name must match the partition name specified on one of the SPART statements. To accept the default spool partition, omit this parameter. A partition name specified on a //*MAIN JES3 control statement or on a SYSOUT initialization statement can override this parameter.

SPIN= Specifies whether the installation wants to allow or suppress spin off of JESLOG data sets during job execution.

SYSTEM= Defines the main name(s) or type of system to be used for jobs in this class.

TDEPTH= Specifies the maximum number of jobs of this class that can be scheduled into the total JES3 complex at one time. The value of nnn is a number from 0 to 255.

TLIMIT= Specifies the maximum number of jobs of other classes that can be scheduled into the total JES3 complex and still allow jobs in this job class to be scheduled. If any of the limits are exceeded, more jobs in this class will not be scheduled. Jobs of this class are scheduled only when the number of jobs running from other classes is equal to or less than the assigned limit.

TRKGRPS= Specifies the number of track groups (as defined by the GRPSZ parameter on the BUFFER or SPART statement) JES3 is to allocate to jobs within this class. For guidelines on how to determine the appropriate value for the TRKGRPS parameter for your installation, see *JES3 Initialization and Tuning Guide*, SA22-7549.

CLASS statement example

In the example in the figure, two job classes and their relationships are defined. JES3 will not allow jobs that specify an undefined job class to execute. Furthermore, JES3 supports 8 character job class names on //*MAIN CLASS= JECL statement. MVS JCL only allows a single character job class on the // JOB card.

JES3 allows the installation to group job classes via the GROUP keyword. There are no guidelines for implementing job class grouping; usually the installation does this based upon workload characteristics, service objectives or strictly political reasons. Some installations have only one job class to a group. In this example, there are two defined job classes belonging to different job class groups.

There are numerous constraints that can be specified affecting the scheduling of work. We will discuss only four at this time and use class A as the sample.

MDEPTH Defines the maximum number of jobs of this class that can be run simultaneously on each processor. In this case, only five jobs can be in execution on processor SC65 and eight on processor SC70.

MLIMIT Specifies the maximum number of jobs of other job classes that can execute on a given processor and still allow jobs in this class to be scheduled. If any class limit is exceeded, no more jobs in this class are scheduled on the given processor; that is, jobs in this class are scheduled only when the number of jobs running from other classes is equal to or less than the assigned limit. For example, on SC65 no class A jobs may be placed into execution if three jobs from class B are in execution on processor SC65. Similarly, a job from class B may not be placed into execution on SC70 if five jobs from class B are executing on SC65.

Note: These constraints are on the processor basis. They are limitations imposed on scheduling based on workload executing on a specified main processor. The constraints need not be symmetric across the processors and in this case are not. Furthermore, the MLIMIT constraints are not restricted to classes within the same group.

TDEPTH Limits the total number of jobs that may be in execution in the complex at any one time. In this case, the limit is set to nine. Thus if class A work is all that is available, only nine jobs could be in execution concurrently despite the fact that up to 13 appear to be allowed based on the MDEPTH specifications.

TLIMIT Limits the number of jobs from this class based upon jobs currently in execution from other classes (anywhere in the complex). In this case no class A jobs may be placed into execution if seven class B jobs execution anywhere in the complex.

Note: These constraints are complex-wide constraints. They impose scheduling constraints for a processor based upon the workload running in the complex.

9.12 Operator commands for CLASS

```
*I G SC65 C
IAT8934 CLASS - K       - STATUS=ON  - GRP=A       - SC65
IAT8934 CLASS - J       - STATUS=ON  - GRP=J       - SC65
IAT8599 INQUIRY ON GMS COMPLETE
*I C=K
IAT8609 CLASS INQUIRY INFORMATION 533
INFORMATION FOR CLASS I
  GROUP=A (JES), SPART=NONE, DEFAULT=NO
  DEFINED ON SC64, SC70, SC65
  ENABLED ON SC64, SC70, SC65
  LIMIT   SYSTEM/CLASS       MAXIMUM     CURRENT
  SDEPTH  ----                255          0
  SPIN=NO
*F G SC65 C K OFF
IAT8456 GMS MODIFY - COMPLETE NO ERRORS     - SC65
*F,C=I,SDEPTH=100
IAT8043 MODIFY COMPLETE FOR CLASS I
```

Operator commands for CLASS

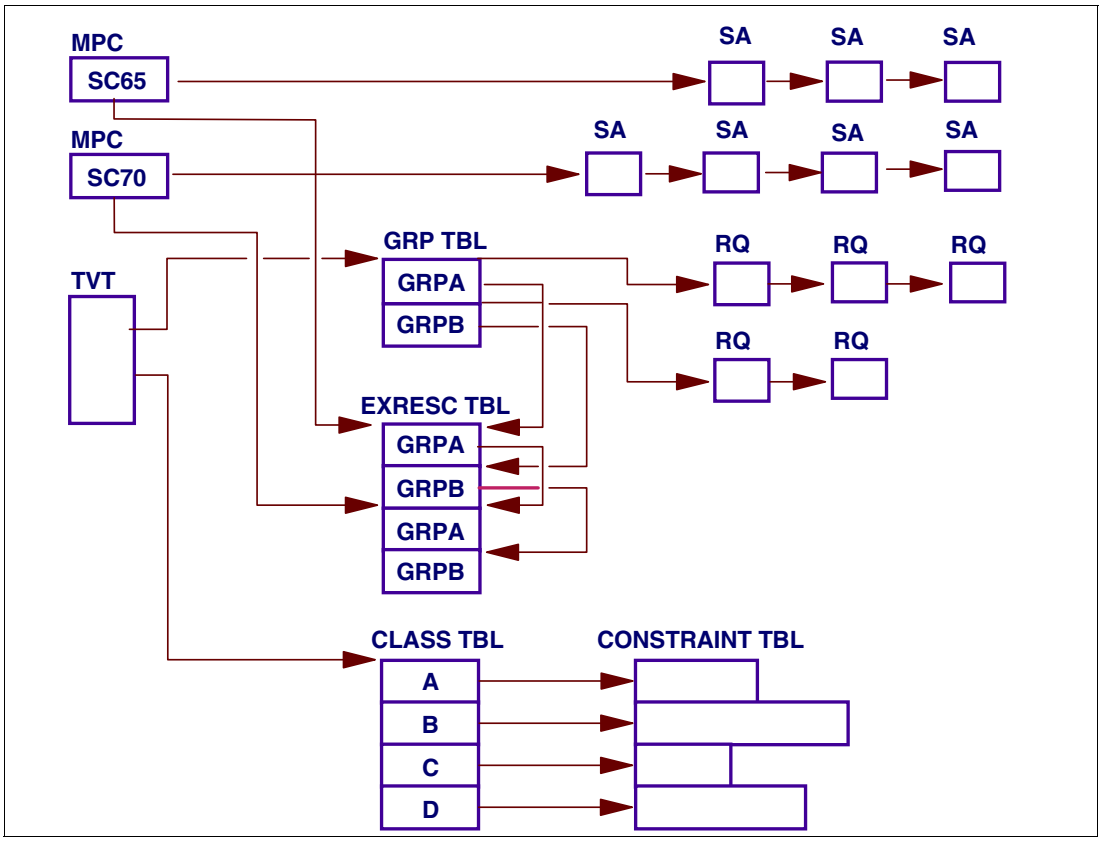
Inquiry commands:

- The **`*I C=class`** command to displays information about a job class, such as the group name, spool partition, and class limits.
- The **`*I G,main,C,class`** command displays status of a class. If the *class* parameter is omitted, the status of all of the classes is displayed.

Some modify commands:

- The **`*F G,main,CLASS`** command activates or deactivate a class.
- The **`*F C=class,SP=spart`** command modifies the spool partition.
- The **`*F C=class,SDEPTH=`** command modifies the setup depth (SDEPTH).
- The **`*F C=class,TDEPTH=`** command modifies the TDEPTH parameter.
- The **`*F C=class,MDEPTH=`** command modifies the MDEPTH parameter.
- The **`*F C=class,TLIMIT=`** command modifies the TLIMIT parameter.
- The **`*F C=class,GROUP=`** command modifies the group assigned to a class.
- The **`*F C=class,JESMSG=`** command modifies whether to create or suppress the JESMSGLG data set for jobs that run in this class.
- The **`*F C=class,SPIN=`** command modifies whether to spin off or not spin off the JESlog (JESMSGLG and JESYSMSG) data sets for jobs running in this class and whether the spin off is automatic or at a particular time.

9.13 GMS internal data areas



GMS internal data areas

The select mode information definitions are kept on spool. After the select mode is placed into affect, these tables reflect constraint conditions specified. There are several data structures that are important in GMS processing including:

- Staging areas (SA)
- Resqueue elements (RQ)
- Main processor control table (MPC)
- Main processor job class group table (MG) (Labeled GRP TBL in diagram.)
- Main processor job class group execution resource table (MGX) (Labeled EXRESC TBL in diagram.)
- Main processor job class table (MCL) (Labeled CLASS TBL in diagram.)
- Transfer vector table (TVT)

GMS control blocks for scheduling

The main processor control table (MPC) is used to select the processor on which the selected job will run. There is one and only one MPC associated with each MAIN FCT. It represents a processor defined by the MAINPROC statement and is the primary workarea for the GMS function. These tables are chained in the sequence in which the MAINPROC statements appear in the initialization stream. A number of important resources are chained off the MPC, as follows:

- SAs - the staging areas representing initiator requests from this processor are chained FIFO from the MPC.
- MGX - the group execution resources associated with this main processor (e.g. initiators) are chained from the MPC.

There are numerous counters kept in the MPC by GMS. These include:

- Number of jobs in execution
- Number of jobs by IORATE in execution
- Total and available LSTOR for the processor
- Various SELECT mode attributes: CHOICE, MAGER, MAGEL

GROUP tables

Each GROUP initialization statement results in an entry in this table. The entries in this table are numbered and sequenced in the order in which the GROUP statements appear in the initialization stream. The group table entry is the anchor for:

- All the jobs, from the job classes comprising the group, waiting to be sent to the initiator. These jobs are represented by a chain of RQs maintained in priority order.
- The execution resource table (initiator control table) for each processor relative to this group.
- DEVPOOL control blocks for those groups that have dedicated devices. Some group related parameters are also kept in the MG. These include:
  - BAR value
  - JSPAN value

EXRESC tables

There is one set of these tables per main processor. Each entry in a table represents the initiator control information (and dedicated device information) for a specific group on a specific processor. These tables are the representation of the EXRESC parameter specification on the GROUP initialization statement. Information includes:

- Number of initiators allowed
- Number of initiators waiting for select
- Number of initiators currently active
- Initiator allocation/unallocation options
- Various status flags

CLASS tables

The class table contains an entry for each of the defined job classes appearing in the initialization stream. The entries in this table are numbered and sequenced in the order in which the CLASS statements appear in the initialization stream. Each entry contains several items of interest including

- Pointer to the class constraints (if any)
- Group number to which this class belongs
- LSTRR value
- Enable/disable status of the class for each main processor
- Flags indicating scheduling constraints reached

The class constraint information represents the installations MDEPTH, TLIMIT, etc. specifications. The class constraint table is pointed to from the class table.

Staging areas (SA)

The staging areas represent requests from initiators for job selection or to return a job to JES3 for the next phase of processing following the completion of execution. The staging

areas are chained from the MPC of the processor where the initiator is active. SAs are chained in the order in which they are received. The SAs are processed (or re-processed) in the order that they are chained.

Resqueue (RQ) entries

Every job that is eligible for execution is represented by a resqueue element. These RQs are chained in priority order (oldest first) from the group table entry. Remember, initiators can only run jobs from classes comprising the group.

This queue is commonly referred to as the “GMS select queue”. Jobs from this queue are used to satisfy initiator requests. Normally, a job is taken from the select queue and placed into execution by GMS. The execution phase is commonly referred to as “on main”.

GMS select queue

To be eligible for being placed on the queue of jobs requires several things, as follows:

- The GROUP and CLASS must be enabled on some processor that satisfies the job's main mask requirements.
- No other job by that name may be in the queue of waiting jobs.
- The job must not be in operator hold.

Once the job is on the GMS select queue, the environment can obviously change with the result being that the job will remain enqueued but not selectable.

GMS internal data areas

So in the figure there are:

- Two MPCs, one each for SC65 and SC70
- Three initiator requests outstanding on SC65 and four outstanding on SC70. We will unilaterally declare them to be for job selection.
- There are two job class groups (GRPA and GRPB) defined. There are three jobs awaiting selection in GRPA and two in GRPA. (The RQ contains the job class for each of the these in case you are wondering).
- Each main processor has a set of execution resources. Each set contains entries for GRPA initiators and GRPB initiators.
- There are four job classes defined and associated with the two groups.
- There are job class constraints relative to the scheduling of jobs in each defined class.
- The TVT which has anchor pointers to all the JES3 address space control blocks.

9.14 JES3 GMS initiator management

- Number of active initiators dependent on:
  - Number allowed
  - Allocation options
  - Unallocation options
  - Available work

JES3 GMS initiator management

JES3 will vary the number of active initiators between zero and the maximum allowed. The number is based on available work when certain allocation and/or unallocation options are in effect.

To understand the conditions under which JES3 starts or stops initiators, one must first understand how JES3 categorizes the state of an initiator, as follows:

Waiting An initiator is in this state between the time GMS issues the S INIT.group_name command until the time the demand select request is recognized and satisfied by GMS. Remember, initiators are only started tasks (e.g. demand select jobs). At this time the initiator transitions into the next category.

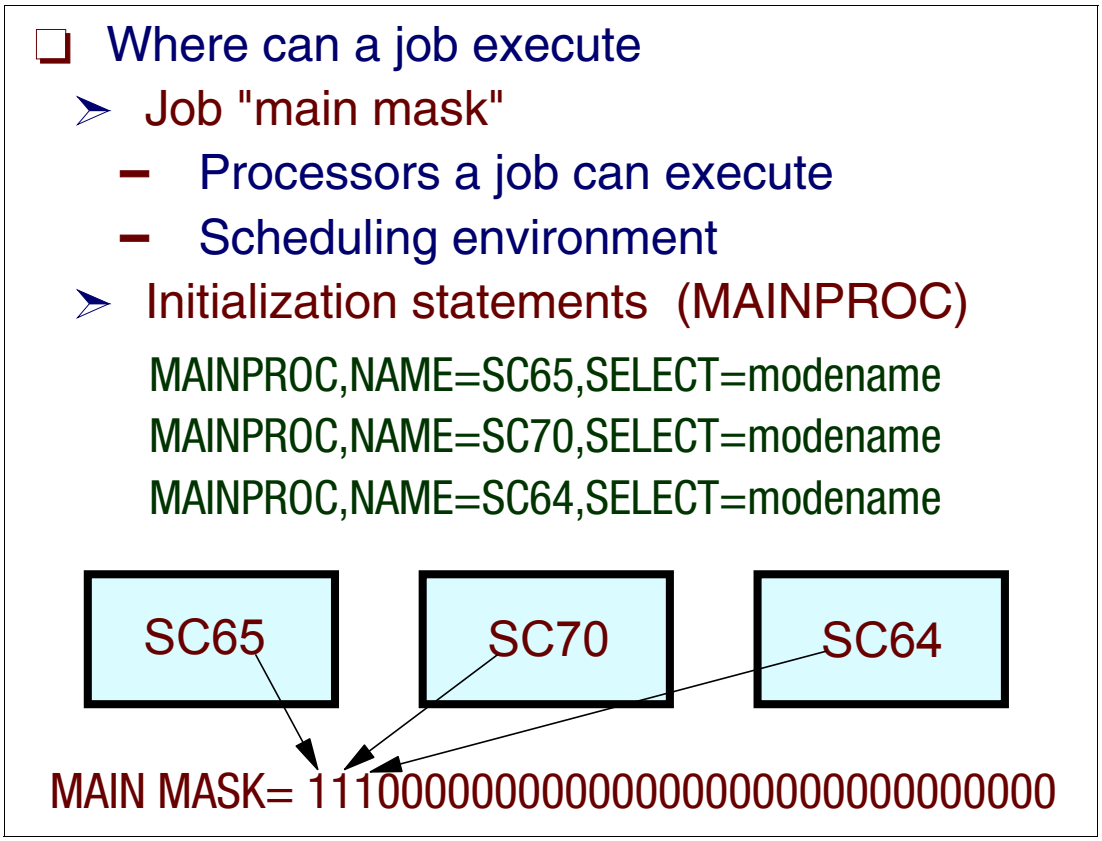
Allocated An initiator in this state is alive and well. It may be “idle”, in which case it will have an outstanding job select request and thus a staging area representing the open initiator will appear on the chain anchored from the appropriate MPC. The initiator may have a job in which case it will be in the next category.

In use The initiator currently is active and holding a job. This is the state you want your initiators to be.

JES3 keeps track of the number of initiators in each category and the number allowed. When GMS is allowed to manage the number of initiators the goals are relatively simple:

- The “waiting” plus the “allocated” should not exceed the “allowed”.
- The “waiting” plus the “allocated” minus the “in use” should not exceed the number of available jobs in the group.

9.15 Which system a job executes on



Which systems for execution - main mask

When a job enters the system, by default it is eligible to execute on any of the systems (LPARs). During its processing through converter/interpreter and main device scheduling, decisions are made about which system or systems they job can execute on.

Structure of the main mask

The installation main mask shows which processors a job can execute on. The lowest-order bit in the first byte of the mask represents the main defined by the first MAINPROC initialization statement in the initialization stream; the second lowest-order bit represents the main defined by the second MAINPROC initialization statement; and so on.

A job main mask is kept in storage in the RESQUEUE control block, (RQMAINS).

Processing that affects the main mask

If a job has SMS resource requirements, when SMS system select has completed processing, if SMS resources are not available on all processors that have connectivity to those resources, the job's main mask is updated with the results of SMS processing.

If a job has a scheduling environment specified for it, a separate main mask is created and stored in the RESQUEUE control block, (RQSCHEMM).

9.16 GMS job selection

- Occurs when:
  - New job added to queue
  - Initiator job select request added to queue
  - Main, group or class enabled
- Job selection process as asynchronous but serial process
- Driven by the Staging Area queue

GMS job selection

Consider the following to be in effect for processor SC65:

- Group GRPA enabled
- Class A enabled
- Maximum number of allowed initiators is at least one
- No initiators active
- DYNAMIC or DEMAND allocation option in effect.

If jobname JOBSY1A is placed on the GMS select queue and the main mask includes SC65, then GMS for SC65 is posted for initiator allocation in this group. GMS for SC65 will then awake, realize that initiators are needed for this group on this main and start the required number.

Started initiators

The number of started initiators differs for the two options. When dynamic allocation is used, JES3 starts the number of initiators necessary to reach the maximum allowed. This is calculated by subtracting the number in use or waiting from the maximum allowed. When demand allocation is in effect, JES3 will calculate the number of jobs in the group waiting for an initiator whose main mask includes this processor and are not in operator hold. This number will used to start initiators up to the maximum allowed. Again, JES3 takes into account the number of initiators in use and waiting so as not to exceed the maximum allowed.

In neither instance does JES3 take into consideration class constraint conditions in deciding how many initiators to start. It is possible that none of the jobs will be selectable by the started initiators and thus the initiators will be terminated (if JES3 is allowed to do so).

If the job's main mask had included SC70, and SC70 had the same initial state as SC65, then GMS on SC70 would do the same thing. The effect of these independent processes is a “race condition” and possibly over initiation of GRPA initiators in the complex.

Stopping initiators

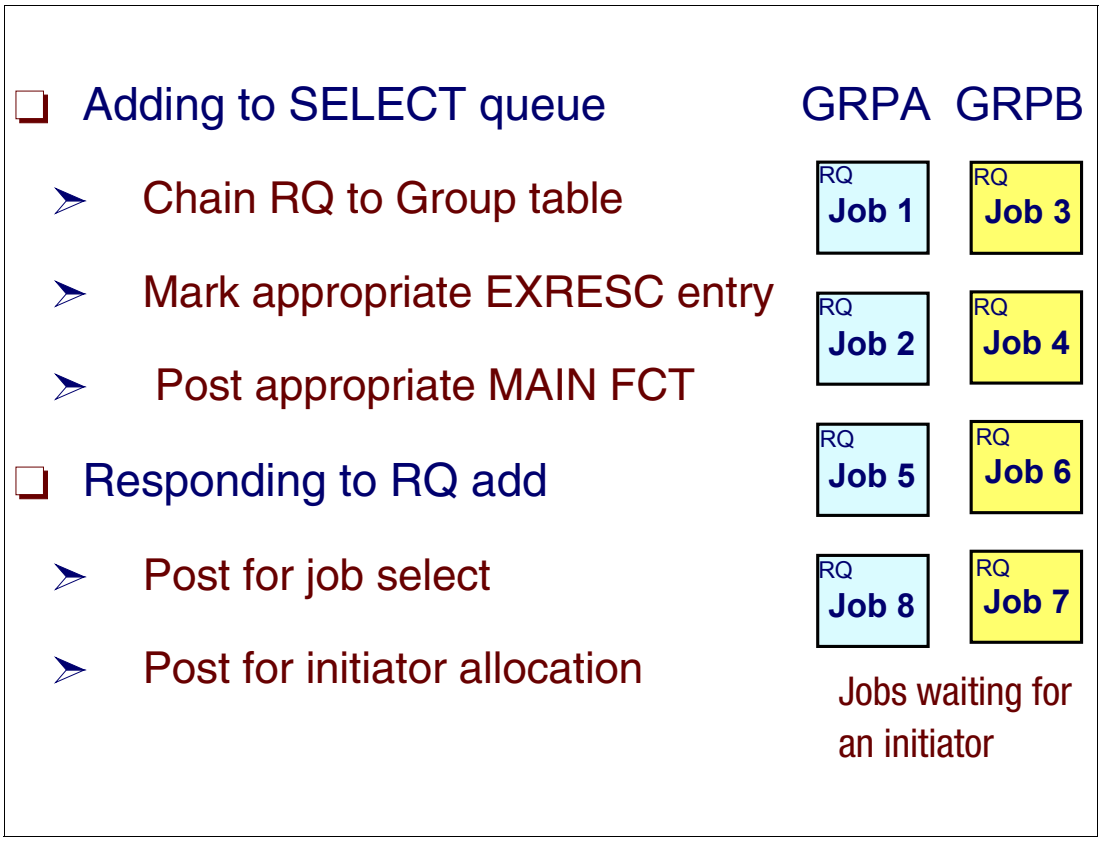
Initiator unallocation is a similar process. The basic idea is to reduce the number of “allocated” but not “in use” initiators to reflect the number of available jobs. In the case of dynamic unallocation, the number of jobs must be zero. For demand unallocation, the number of initiators terminated is the difference between the number of available initiators from the number of available jobs. Again, the estimate of the number of available jobs is a rough assessment of the jobs eligibility.

Stopping or terminating initiators is not done by MVS commands as when starting initiators. Basically, GMS searches the queue of initiators with select requests outstanding for a specific group (SAs anchored from the MPC) and returns the request with an indicator to terminate. The initiator then terminates like any other started task.

Maintaining initiator counts

Maintaining the “appropriate” number of initiators is very important to GMS. Job selection for the garden variety batch jobs in a group will be suspended until there are no pending allocations or unallocations for the initiators in the group. The process is somewhat iterative in that the unallocation process may invoke the allocation process and vice versa. Of course, the flow of jobs on the queue and operator actions will obviously call for initiator adjustment. If GMS is allowed to adjust the initiator count up and down and there is not a steady state of available jobs, you will notice considerable fluctuation in the initiator status for the group. This effect will be emphasized if the group is enabled on more than one main and the jobs in the queue are not main limited. To avoid this “initiator thrashing”, many installations maintain operator control over the heavily used groups. The thrashing can be limited by using an unallocation option of MANUAL.

9.17 New jobs to SELECT queue



New jobs to SELECT queue

Responding to jobs that are newly available for GMS scheduling is essentially a two stage process.

- Stage I - Adding to the SELECT queue

  The job is moved to the GMS select queue (most likely from the allocate queue). This process involves placing the job in the correct subchain in the proper sequence based on priority. The correct subchain is dependent on the job class and group structure and is anchored from the group table. The execution resource entries are marked to denote available work and the MAIN FCTs are posted in order that the GMS functions can be performed. GMS may be posted for possible initiator allocation required depending on the allocation options and state of the initiators.

- STAGE II - Responding to a RQ add

  Each MAIN FCT that is posted will run (serially) in response to the addition of a new RQ being added to some group. GMS will examine each execution resource table to determine if new work is available and respond by:

  – Posting itself to make a job selection pass.

  – Posting itself for initiator allocation if necessary.

9.18 IAT8674 message - response to *I J command

```
*I J=VAINICL*
IAT8674 JOB VAINICLJ (JOB25640) P=01 CL=J      MAIN(GMS SELECT)
IAT8699 INQUIRY ON JOB STATUS COMPLETE,      1 JOB  DISPLAYED
```

- **MAIN(status)**
  - FETCH
  - WAITVOL
  - SYSTEM SELECT
  - ALLOCATE
  - VOL UNAVAIL
  - VERIFY
  - SYSTEM VERIFY
  - MDS ERROR
  - GMS SELECT ← GMS
  - EXECUTING ← EXECUTION
  - BREAKDOWN
  - MDS RESTART
  - MAIN COMPLETE ←
  - OUTSERV WAIT
  - DEMAND SELECT ←
  - I/O WAIT
  - ENDFUNC ERROR
  - DSP ABEND

IAT8674 message - response to *I J command

The MAIN(status) is displayed using the `*I J` command. The MAIN scheduler element is ready to be processed or is being processed for the job. If no DSP is running yet, only MAIN appears in the message. Otherwise, *status* indicates where the job is in relationship to the functions of main service processing, as follows:

FETCH	The job is waiting for volume fetch processing.
WAITVOL	The job is waiting for *S setup processing.
SYSTEM SELECT	The job is on the system select queue.
ALLOCATE	The job is waiting for resources to be allocated.
VOL UNAVAIL	The job is waiting for an unavailable volume.
VERIFY	The job is waiting for volumes to be mounted.
SYSTEM VERIFY	The job is on the system verify queue.
MDS ERROR	The job is waiting for an operator decision. (MDS error queue)
GMS SELECT	The job is waiting to be selected for processing on the main.
EXECUTING	The job is in execution.
BREAKDOWN	The job is waiting for its resources to be deallocated (MDS breakdown).
MDS RESTART	The job is waiting for MDS restart processing.
MAIN COMPLETE	The job is complete on main.
OUTSERV WAIT	The job is in the process of being rescheduled.
DEMAND SELECT	A demand-select job that is waiting to be selected for main processing.
I/O WAIT	The ending function RESQUEUE is waiting for I/O to complete.
ENDFUNC ERROR	Error - the ending function RESQUEUE was not processed.
DSP ABEND	The main service scheduler element was ended by JES3 failsoft.

9.19 JES3 XCF group name

- **OPTIONS** XCFGRPNM=group_name
  - XCFGRPNM parameter is optional
  - JES3 determines XCF group name as follows:
    - 1. Use groupname from OPTIONS XCFGRPNM, if specified
    - 2. Use node name from HOME=YES NJERMT NAME= when no XCFGRPNM
    - 3. Use default node name of N1
    - 4. Optionally, use the JES XCF attach/detach exit IXZXIT03 if you are attaching your own JESXCF group
- The users of the JES3 XCF group are:
  - The JES3 Global and all locals
  - All converter/interpreter functional subsystem (C/I FSS) and writer functional subsystem (WTR FSS) address spaces
  - JES3DLOG

JES3 XCF group name

JES3 uses the JES common coupling services (JES XCF) for communicating data amongst the JES3 XCF group members. To allow both MVS and all the JES3 XCF members to know about each other, JES3 needs to be assigned to a XCF group. A group in XCF is defined as the set of related members of a multisystem application. It can span one or more of the systems in a sysplex and represents a complete logical entity to XCF. A member of an XCF group resides on one system in the sysplex and uses XCF services to communicate with other members of the same group.

XCFGRPNAM parameter specification

The XCFGRPNM parameter of the OPTIONS initialization statement specifies a unique XCF group name for JES3.

```
OPTIONS   XCFGRPNM=groupname
```

The XCFGRPNM parameter is optional. JES3 uses for the XCF group name:

1. The group name from the OPTIONS XCFGRPNM specification
2. The node name from the NJERMT NAME= specification of the HOME=YES node when no XCFGRPNM is defined
3. The default node name **N1**, when no networking and XCFGRPNM are defined.

The NAME parameter on the NJERMT statement that defines HOME=YES is the default for the XCFGRPNM parameter on the OPTIONS statement during a warm or cold start. If the home node is changed during a hot start with refresh and the XCFGRPNM parameter is not

specified, the value of XCFGRPNM that was determined from the home node definition during the last warm or cold start remains in effect.

> **Note:** The recommendation is to let the group name default to the home node name. An IBM recommendation - Use the same name for MVS sysplex name and JES3 XCF group. In support of these scenarios and any others in which your installation may want a JESXCF group name that differs from the node name, use the XCFGRPNM= keyword on the JES3 OPTIONS initialization statement. (Use this optional keyword to define a name that must be 1-8 alphanumeric characters including $, # and @.)

XCF group users

The users of the JES3 XCF group are:

- The JES3 global and all locals
- All converter/interpreter functional subsystem (C/I FSS) and writer functional subsystem (WTR FSS) address spaces
- JES3DLOG

9.20 XCF group and JES3 node display commands

- **SYS1.PARMLIB - COUPLExx - SYSPLEX=WTSCPLX4**
  - **D XCF**

    IXC334I 10.25.25 DISPLAY XCF 590

    SYSPLEX **WTSCPLX4**: SC64 SC65 SC70
- **NJERMT,NAME=WTSCPLX4,HOME=YES**
  - ***I NJE N=WTSCPLX4**

    IAT8659 - HOME **WTSCPLX4** PRTDEF A PRTTSO A PRTXWTR A PUNDEF B
- **MAINPROC,NAME=SC65,SYSTEM=JES3,...**

  DEVICE,DTYPE=PRTAFP1,JNAME=IAZFSS,...,FSSNAME=WTRIAZF

  FSSDEF,FSSNAME=CI8,TYPE=CI,PNAME=JES3CI,
  - **D XCF,G,WTSCPLX4**

    IXC332I 10.23.30 DISPLAY XCF 587

    GROUP WTSCPLX4:

    CI7 **CI8** IPDSWAY

    JES3DLOG PRINTWAY **SC65**

    SC70 WTRIAZF

XCF group and JES3 node display commands

XCF and JES3 display commands for sysplex, JES3 home node, and JES3 XCF group members:

```
D XCF
IXC334I  10.25.25  DISPLAY XCF 590
   SYSPLEX WTSCPLX4:   SC64              SC65                SC70
*I NJE N=WTSCPLX4
IAT8659 - HOME  WTSCPLX4  PRTDEF A PRTTSO A PRTXWTR A PUNDEF B
EJES510 USER-VAINI---/D XCF,G,WTSCPLX4
D XCF,G,WTSCPLX4
IXC332I  10.23.30  DISPLAY XCF 587
 GROUP WTSCPLX4:
                CI7                CI8                IPDSWAY
                JES3DLOG           PRINTWAY           SC65
                SC70               WTRIAZF
```

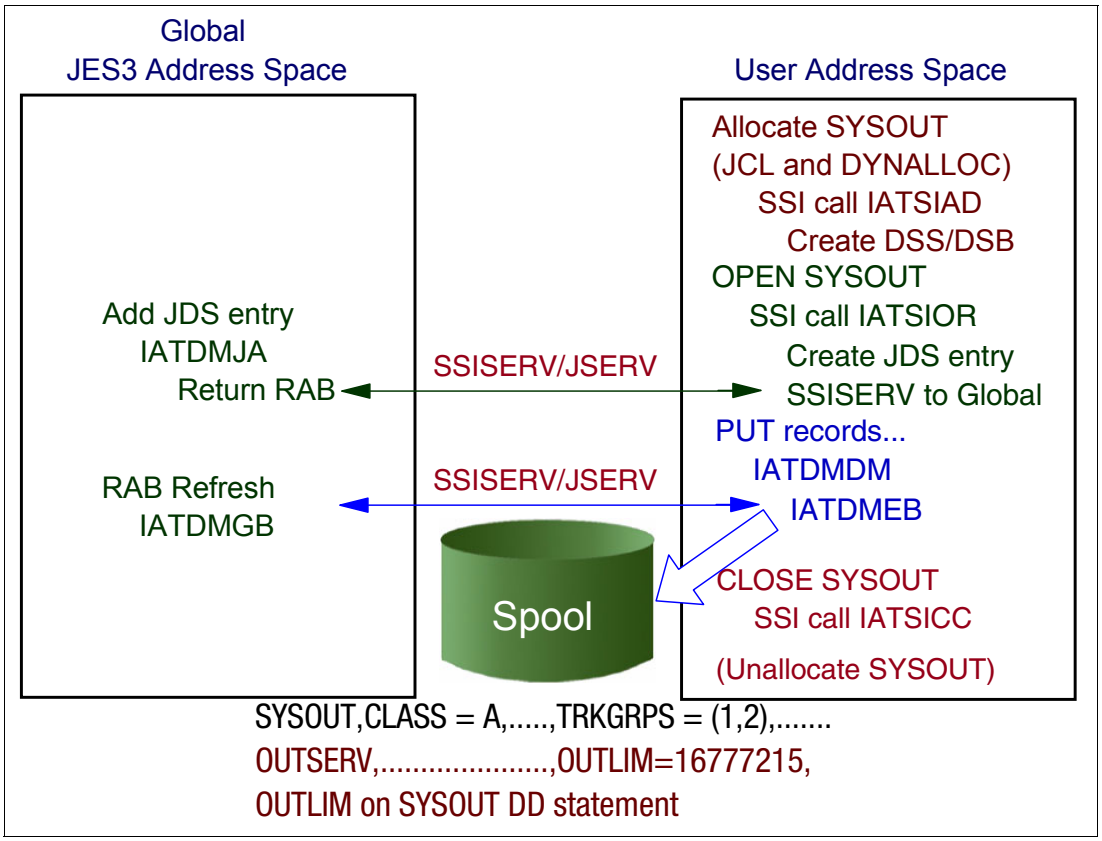
The DISPLAY XCF command to displays cross system coupling information in the sysplex.

```
D XCF[,S] - Displays (message IXC334I) a list of all systems currently
participating in the sysplex.
D XCF,GROUP,groupname - Displays (message IXC332I) the members of the specified group.
```

The JES3 *INQUIRY,NJE command to display the status of the networking nodes and communication lines.

```
*I NJE N=nodename
```

9.21 Job in execution - SYSOUT data set processing



Job in execution - SYSOUT data set processing

IATSIAD (JES3 SSI Allocation/Unallocation Routines) module services allocation and unallocation for all SYSIN and SYSOUT data sets. It also creates the DSB/DSS control blocks. The new SYSOUT data set name has the following format:

```
USERID.JOBNAME.JOBID.DSNUMBER.NAME
```

DSS control block

The data set status block (DSS) contains information for a SYSIN or SYSOUT data set. There is one DSS for each allocated spool data set. The DSS data includes:

- Type of data set
- Address of the associated DSB
- Address of the record allocation block (RAB) for an open SYSOUT data set

IATSIOR module services open, internal reader reopen and restart for all SYSIN and SYSOUT data sets. It builds the necessary control blocks to perform I/O for the data sets at open time. At restart the checkpointed information that was saved is used to reposition the data set.

JDS entry

A JDS entry is built at open time for the SYSOUT data set and passed to global in its entirety to be added to the job's JDS. This communication to the global is done via SSISERV. On the global module IATDMJA (USAM JDS Access Interface Routines) adds the JCD entry into the job's JDS chained SRF. For a new SYSOUT allocation, IATDMJA also performs the following:

- Builds a data set track allocation table or uses the existing job track allocation table to assign spool to the data set.
- Writes a dummy record to initialize the data set (MRF).
- Builds a record allocation block (RAB)
- Issues a JSERV to send the completed JDS to the caller.

RAB control block

The RAB is used to pass spool records to the executing job and contains:

- A spool record address that represents the first available record in the current track group
- A count of the number of records in the track group that are available for allocation
- The first spool record addresses of the track groups that were allocated to the job or data set

The IATSICC module services close and checkpoint for all SYSIN and SYSOUT data sets. At close time it allows for the completion of all I/O and frees up some of the resources used. For checkpoint, information about the data set is saved in a buffer and the buffer is written to disk.

TRKGRPS parameter

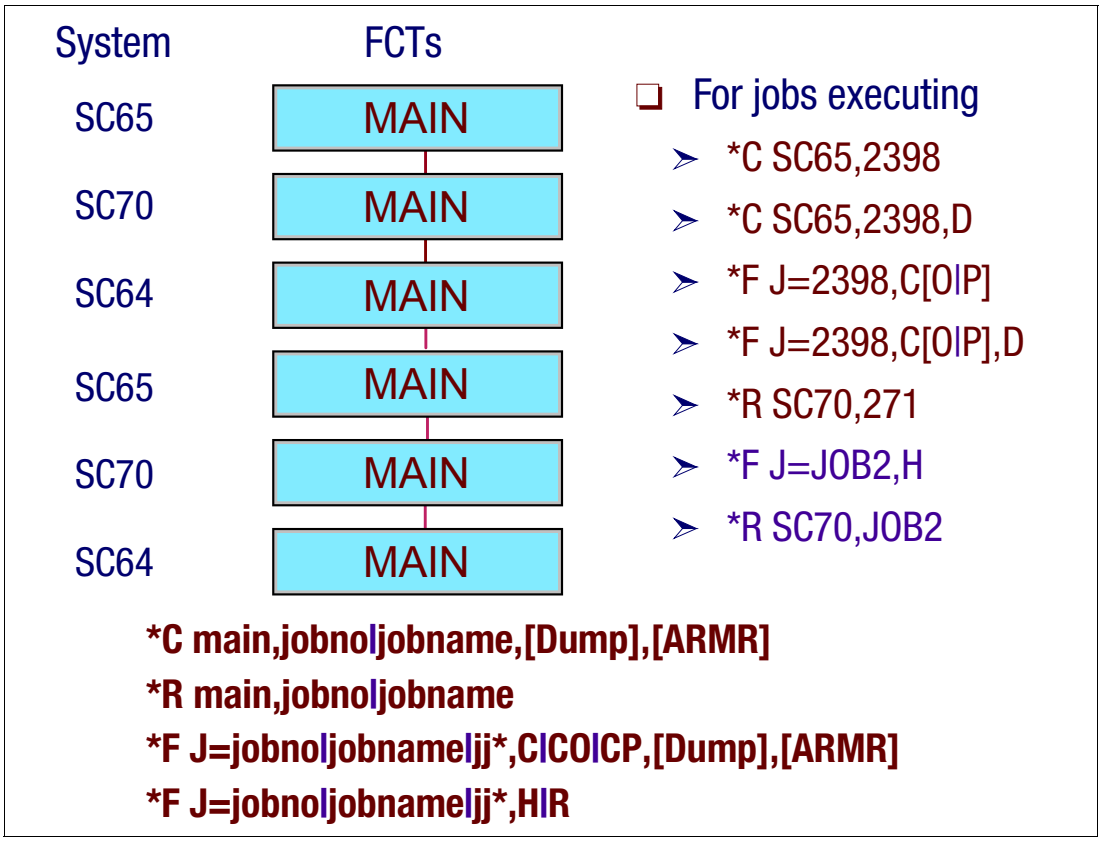
TRKGRPS parameter of the SYSOUT initialization statement specifies the number of track groups (as defined by the GRPSZ parameter on the BUFFER or SPART statement) JES3 is to allocate to jobs within this SYSOUT class. The primary allocation quantity specifies the number of track groups to be initially assigned to jobs in this SYSOUT class. The secondary allocation quantity specifies the number of track groups to be allocated to jobs in this SYSOUT class subsequent to their primary allocation. JES3 allocates the specified amount of spool space after the job uses up its initial allocation, and again (for an unlimited number of times) when the job uses up each secondary allocation and requests more spool space.

The TRKGRPS parameter on the SYSOUT statement overrides corresponding values on the CLASS and MAINPROC initialization statements and on the //*MAIN JES3 control statement.

JES3 Job execution monitoring OUTLIM parameter

The OUTLIM parameter of the OUTSERV initialization statement and the OUTLIM keyword of a SYSOUT DD statement specifies the default record limit for a SYSOUT data set. When this limit is exceeded, the installation IEFUSO SMF exit is entered. Upon return from the SMF exit, the job is either canceled or a new limit is used. The value of limit must be from 1 to 16777215. For more information concerning SMF exits, see *z/OS MVS Installation Exits*, SA22-7593.

9.22 Working with executing jobs



Canceling executing jobs

Use the `*C` command to stop execution of a job that is currently active on a main. An SVC dump can also be requested. The `*C` command cannot be used to cancel a job that is part of a DJC network; for such jobs, use the `*F N` command.

> **Note:** The `*F J=` command is the preferred way to cancel jobs that are in execution. The `DUMP` and `ARMR` options can also be specified on the `*F J=` command.

The `*F J` command can be used to:

- Hold a job.
- Release a job.
- Run a job.
- Cancel a job.
- Change a job's priority.
- Change a job's job class.
- Change a job's JESMSGLG logging status.
- Change a job's service class.
- Add a message to the job's JESMSGLG data set.
- Spin off the JESlog (JESMSGLG and JESYSMSG) data sets.
- Activate the C/I debug processing for a job.

Job cancel options

The canceling options are:

C Specifies that the job or jobs be canceled. If a job is in a hold priority, the job will not be purged from the system until the job or the priority is released.Otherwise the job is purged immediately.

CO Specifies that the job or jobs be canceled and that all data sets (including those on the output service hold queue) ready for output at the time the command is issued be processed.

CP Specifies that the job or jobs be canceled and that any print data sets ready for output at the time the command is issued be printed. Data sets on the output service hold queue (such as TSO/E and external writer data sets) are not considered print data sets.

Restarting executing jobs

Use the **`*R main`** command to restart a job that is already in execution. The **`*R main`** command causes the failure option for the job to be taken unless the job is registered with automatic restart management (ARM). If the job is registered with automatic restart management, the FAILURE option will be ignored and automatic restart management will decide if the job will be restarted. If automatic restart management decides not to start the job, then the job will be cancelled. If automatic restart management decides to start the job, all non-spin sysout data sets created during the previous execution will be deleted.

If you want to restart a job but do not want it to execute immediately, place the job in hold status with an **`*F`** command, and then issue the **`*R main`** command. The **`*R main`** command actually cancels the job on the main, making its resources available to other jobs. When the job can be released, using an **`*F`** command, it is rescheduled without having to be read into the system again.

Job FAILURE option

The job will be restarted provided the system failure option for the job is RESTART. The failure option can be specified in this order:

1. //*MAIN statement

   Indicates the job recovery option to be used if the system fails. If you do not code a FAILURE parameter on the //*MAIN statement, JES3 assigns the job the default failure option, which is defined during JES3 initialization for each job class.

2. CLASS initialization statement

   Specifies the job recovery option that JES3 is to use when a system failure affects jobs in this class.

3. STANDARDS initialization statement

   Specifies the default job recovery option to be applied to jobs active at the time of a main restart failure. This parameter is used if the FAILURE option is not indicated on the CLASS statement associated with the job and the job does not have a main MVS checkpoint/restart option specified in its JCL.

If RESTART was not specified, the job is processed according to its failure option.

9.23 Inquiry active command

```
*I A,C=class
*I A,D=[dspname | ALL | DEADLINE | DLINE | INTRDR[,R]]
*I A,G=group
*I A,J=jobno
*I A,[SRVCLASS=srvclass | SRVCL=srvclass]
*I A,main,[C=class | G=group | SRVCL=srvclass]
```

```
*I A D=INTRDR
IAT8522 JOB INTRDR   (JOB28541) ACTIVE ON INTRDR          7358.25 MIN
IAT8522 JOB INTRDR   (JOB19166) ACTIVE ON INTRDR          7358.25 MIN
IAT8523 INTRDR COUNTS - MAX=(020,020), ACT=0002, FCT=002, HWATER=0002
IAT8593 INQUIRY ON ACTIVE JOBS COMPLETE,      2 JOBS DISPLAYED
```

Inquiry active command

The **`*I A`** command used to display the following options:

- Status of jobs active on a particular DSP

  `*I A,D=dspname`

- Names of jobs waiting to be transmitted to other nodes through BSC/NJE, and the transmission paths for the jobs

  `*I A,D=NJESND`

  JES3 displays the following message for each job waiting to be transmitted:

  ```
  IAT8522 JOB jobno jobname ACTIVE ON NJESND TIME NAVAIL SCHED FOR nodename
  ```

  JES3 displays the following message for each job currently being transmitted:

  ```
  IAT8522 JOB jobno jobname ACTIVE ON NJESND lsender  time
  ```

- Status of JES3 jobs active on each main

  `*I A,main`

- Status of jobs in execution for the specified job class.

  `*I A,C=class`

- Status of jobs in execution for the specified job class group.

  `*I A,main,G=group`

- Status of jobs in execution for the specified service class.

  `*I A,main,SRVCL=srvclass`

- Status of active jobs in the deadline queue

  `*I A,D=DLINE`

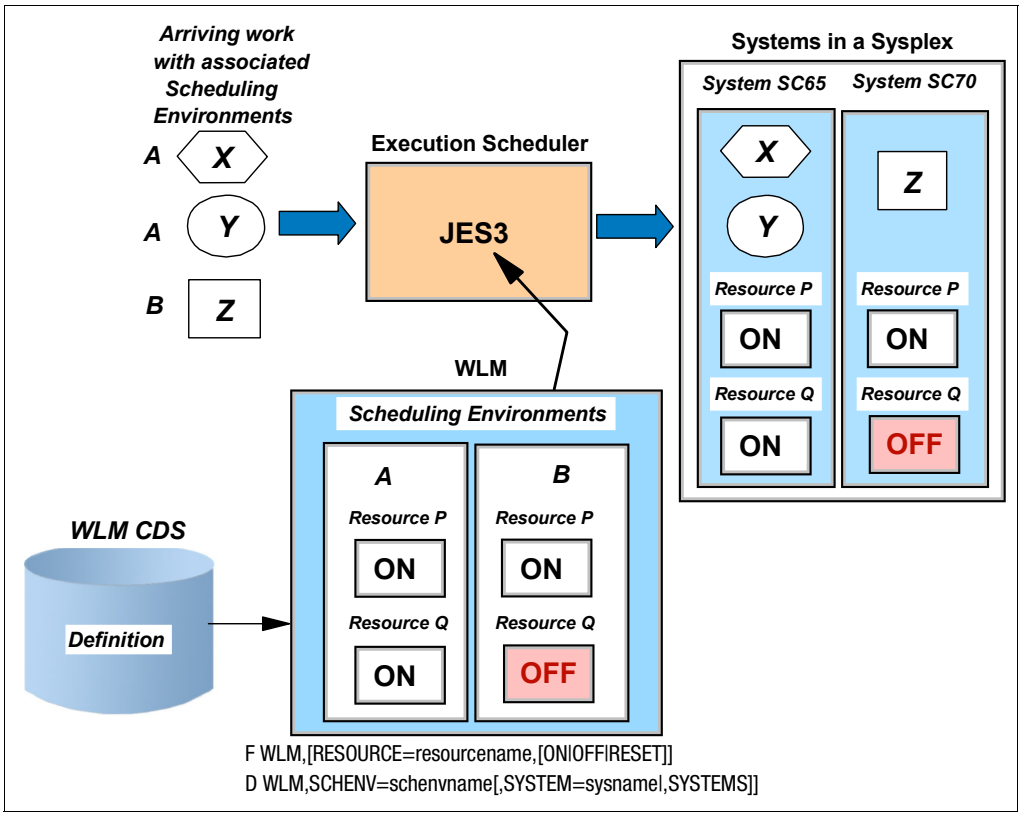
- Names of jobs that are currently being transmitted to other nodes

  `*I A,D=WTSCPLX1 - where WTSCPLX1 in the node name`

- Names of the logical senders handling the transmissions

> **Note:** A logical sender is a type of logical device that JES3 uses to communicate with a directly-connected node. JES3 creates and names one or two logical senders for each line that is connected to a directly-connected node provided the NJERMT statement for that node specifies MAXLINE=1, 2, or 3. To create a logical sender name, JES3 starts with a 6-character base name. The form of the base name is XYYYYY, where X is an alphanumeric character and YYYYY is a number between 00001 and 99999. JES3 verifies that the base name is unique by checking the base name against a list of existing base names. If the new base name is unique, JES3 adds it to the list. If the name is not unique, JES3 makes it unique by changing one or more characters of the name.

9.24 WLM scheduling environment example



WLM Scheduling Environment Example

The scheduling environment is a list of resource names along with their required states. If an MVS image satisfies all of the requirements in the scheduling environment associated with a given unit of work, then that unit of work can be assigned to that MVS image. If any of the requirements are not satisfied, then that unit of work cannot be assigned to that MVS image.

Scheduling environments help ensure that units of work are sent to systems that have the appropriate resources to handle them. A scheduling environment is a list of resource names along with their required states. Resources can represent actual physical entities, such as a data base or a peripheral device, or they can represent intangible qualities such as a certain period of time (like second shift or weekend).

Resources

These resources are listed in the scheduling environment according to whether they must be set to ON or set to OFF. A unit of work can be assigned to a specific system only when all of the required states are satisfied. This function is commonly referred to as resource affinity scheduling.

Scheduling environments assignments

JES3 checks the scheduling environment associated with each arriving batch job and then assigns the work to a system that matches that scheduling environment. In the example, both the X and Y jobs require that both Resource P and Resource Q be set to ON, so those jobs can be initiated only on System SC65 in the sysplex. The Z job requires that Resource P be

set to ON and that Resource Q be set to OFF. So that job can be initiated only on System SC70.

In a sysplex containing only one system, scheduling environments have some degree of usefulness, as JES3 will hold batch jobs until the required states become satisfied. In a multi-system sysplex, the full power of scheduling environments becomes apparent, as work is assigned only to those systems that have the correct resource states (the resource affinity) to handle that work.

Presently, JES2 and JES3 are the only participants that use scheduling environments, although the concepts could certainly apply to other types of work, too.

Changing workload manager resource states

Resource states are used by workload management in conjunction with scheduling environments to ensure that work is scheduled only on a system with the appropriate resources to handle that work.

The complete syntax for the MVS MODIFY WLM command is:

```
F WLM,[RESOURCE=resourcename,{ON|OFF|RESET}]
```

WLM - The name of the address space where WLM executes

RESOURCE=resourcename - Changes the state of resourcename.

ON - Specifies that if the required resource state in a scheduling environment is ON, that requirement will be satisfied on the target system.

OFF - Specifies that if the required resource state in a scheduling environment is OFF, that requirement will be satisfied on the target system.

RESET - Specifies that this resource setting will satisfy neither an ON nor an OFF resource requirement. Therefore if a scheduling environment includes resourcename in its list of resources (whether ON or OFF), then that scheduling environment will not be available on the target system.

Display workload manager scheduling environment

To display sysplex-level information about a scheduling environment, you can issue the following command from an MVS console:

```
D WLM,SCHENV=schenvname[,SYSTEM=sysname|,SYSTEMS]
```

SCHENV=schenvname - Displays status information for the specified scheduling environment (schenvname). You can display multiple scheduling environments by using wildcard characters. The multiple-character symbol (*) and the single-character symbol (?) can be used in any position.

SYSTEM=sysname - Displays the state of the scheduling environment and the availability of each resource referenced by the scheduling environment on the designated system.

SYSTEMS - Displays the state of the scheduling environment on all active systems in the sysplex.

If neither SYSTEM=sysname or SYSTEMS is specified, SYSTEMS is the default.

9.25 Why use scheduling environments

- Distribute and balance batch workloads
  - Define resource names for batch job scheduling
    - Vector facility - Encryption - Second shift
  - Associate resource name with
    - Real system resources
- Define in user installation terminology
- Initiate batch jobs
  - Where resources exist
  - Where jobs should or should not execute
  - When jobs are desired to execute

Resource names

Resource names are really indicators on an MVS system, that show the presence or absence of a real resource. The resource can be an actual physical entity, such as a data base or a hardware feature (vector processor or cryptography). It can also be an intangible quality, such as a certain time of day or a certain day of the week. However, understand that the resource names are abstract and have no actual meaning or direct relationship to physical resources.

Resource names have no real meaning and may have no direct relationship to physical resources in existence. You can establish a resource name such as *DB2A* and then schedule jobs where a give instance of a DB2 subsystem either exists or does not exist. However, the name DB2A could equally be XYZ and the concept would be the same and work equally well.

Scheduling batch jobs

Scheduling environments are intended to provide a method for an end user to declare which resources are required for a job and have the job scheduled only on systems having the required resources. It is a way in which you can cause batch work to be initiated:

- Where a jobs resources exist
- Where jobs are not desired to execute
- When jobs are desired to execute

9.26 JES3 and scheduling environments

- Scheduling environments defined to WLM
- Specified for jobs with SCHENV parameter on job card
  - //JOBSCHEN JOB .....,SCHENV=schenv-name
- WLM keeps track of where SCHENV is available
- JES3 uses IWMSEDES service for availability on a given MVS
- JES3 keeps an main mask for systems where requested scheduling environment is available
  - For jobs with scheduling environment specified

JES3 and scheduling environments

WLM administrative definitions are necessary to define the resource names that can be requested by jobs.

SCHENV parameter on job card

You can specify the name of a WLM scheduling environment, using the SCHENV parameter on the JOB statement. A scheduling environment is a list of resources and their required settings. By associating a scheduling environment name with a job, you ensure that the job will be scheduled only on a system that satisfies those resource state requirements.

Note: An input service user exit (IATUX29) or a converter interpreter exit (IATUX08 or IATUX09) can assign a scheduling environment to a job by storing the scheduler environment name into the JCT field JCTSCHEN.

WLM IWMSEDES service

The WLM IWMSEDES service determines if a scheduling environment is available on a specified system. A scheduling environment is a list of resource names and their required states. If all of the resources are in the required state, the scheduling environment is available. If any of the resources is not in the required state, the scheduling environment is not available. JES3 is a user that issues the IWMSEDES service.

When a job specifies or requires a scheduling environment, JES3 maintains a main mask for the job that contains the systems where the scheduling environment is available.

9.27 Operator commands

```
*I J=VAINSCHE
IAT8674 JOB VAINSCHE (JOB25659) P=01 CL=A       MAIN(GMS SELECT)
IAT8699 INQUIRY ON JOB STATUS COMPLETE,       1 JOB  DISPLAYED
*I J=VAINSCHE,W,X
IAT8674 JOB VAINSCHE (JOB25659) P=01 CL=A       MAIN(GMS SELECT)
IAT8564 SCHENV=SC65, SRVCLASS=BATCHLOW, GROUP=A (JES)
IAT8685     SC70     - MAIN OFFLINE/NOT CONNECTED
IAT8685     SC65     - SCHEDULING ENVIRONMENT NOT AVAILABLE
IAT8687     JOB WAITING/ACTIVE 00000 HOURS  10 MINUTES  48 SECONDS
IAT8699 INQUIRY ON JOB STATUS COMPLETE,       1 JOB  DISPLAYED
D WLM,SCHENV=SC65,SYSTEM=SC65
IWM037I  17.17.34  WLM DISPLAY 125
  SCHEDULING ENVIRONMENT: SC65
   DESCRIPTION:                        Ditto
   SYSTEM:                             SC65
   STATUS:                             NOT AVAILABLE
                                  REQUIRED  CURRENT
    RESOURCE NAME          STATE        STATE
   *SC65                   ON            RESET
F WLM,RESOURCE=SC65,ON
IWM039I RESOURCE SC65 IS NOW IN THE ON STATE
IWM056I SCHEDULING ENVIRONMENT SC65 IS NOW AVAILABLE
IAT2000 JOB VAINSCHE (JOB25659) SELECTED SC65     GRP=A
ICH70001I VAINI    LAST ACCESS AT 17:04:44 ON WEDNESDAY, NOVEMBER 12, 2008
IEF403I VAINSCHE - STARTED - TIME=17.19.00 - ASID=0040 - SC65
```

Scheduling environment operator commands

The following commands display information related to scheduling environments:

```
*I J=jobid,W,X,[T=term | ID=userid | SUBID=userid]
*I J=jobid,X,W
```

W - Displays the reason a job is waiting to be scheduled, and the amount of time the job is waiting.

T=term - Displays the status of jobs from the designated terminal group.

ID=userid - Specifies the userid that belongs to the owner.

SUBID=userid - Specifies the user that belongs to the job submitter.

If specified, the userid is obtained from the //*MAIN JCL statement. If it is not specified on the //*MAIN JCL statement, the userid of the job owner (ID=userid) is used by default.

X - Displays extended information for a job such as the job's scheduling environment and service class.

Displaying scheduling environment status

The MVS operator command **D WLM,SCHENV=schenvname**displays status information for the specified scheduling environment (schenvname).

The command displays status information for the specified scheduling environment (schenvname). You can display multiple scheduling environments by using wildcard

characters. The multiple-character symbol (*) and the single-character symbol (?) can be used in any position.

Where:

SYSTEM=sysname Displays the state of the scheduling environment and the availability of each resource referenced by the scheduling environment on the designated system.

Changing resource states

Resource states are used by workload management in conjunction with scheduling environments to ensure that work is scheduled only on a system with the appropriate resources to handle that work. See *z/OS MVS Planning: Workload Management*, for more information about resources and scheduling environments.

The complete syntax for the command is:

```
F WLM,[RESOURCE=resourcename,{ON|OFF|RESET}]
```

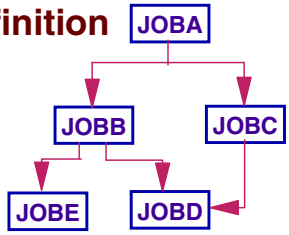
The syntax for the MODIFY WLM command is:

```
F WLM,[RESOURCE=resourcename,{ON|OFF|RESET}]
```

It is used to change WLM resource states.

9.28 Defining a DJC network

❑ DJC NETWORK Definition



//*NET NETID = | ID =name[,parameter]... Syntax

The parameters:

ABCMP | AC = [NOKP | KEEP]	NHOLD | HC = n
ABNORMAL | AB = [D | F | R]	NRCMP | PC = [HOLD | NOHO | FLSH]
NORMAL | NC = [D | F | R]	OPHOLD | OH = [NO | YES]
DEVPOOL = (ANY | NET [,device-name,n]...	RELEASE | RL = (jobname[,jobname]...)
DEVRELSE = [YES | NO]	RELSCHCT | RS = n
NETREL | NR = (netid,jobname)	

//*NET ID=JOBNET,HC=0,RL=(JOBB,JOBC),OH=YES	**Job A**
//*NET ID=JOBNET,HC=1,RL=(JOBD,JOBE)	**Job B**
//*NET ID=JOBNET,HC=1,RL=(JOBD)	**Job C**
//*NET ID=JOBNET,HC=2,AB=D	**Job D**
//*NET ID=JOBNET,HC=1,NC=F,AB=D	**Job E**

Dependent job control (DJC)

Dependent job control (DJC) is 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. Success or failure of one job can result in execution, holding, or cancellation of other jobs. Job dependencies may occur because of data dependencies or may be defined to achieve better device utilization or to manage job streams.

DJC network definitions

The visual has a sample DJC network of 5 jobs. The //*NET statement show the hold counts and the jobs that are to be released. Job E only executes if job D abends (AB=D). Jobs in a DJC network are of two types:

- Predecessor jobs, which must be completed before another job.
- Successor jobs, which must not be executed until one or more jobs are completed.

//*NET control statements

To define a DJC network, the user must include a //*NET control statement in the JCL stream for each job in the network. The //*NET control statement specifies the dependency that must be satisfied before the job can be scheduled for processing. Jobs normally must wait for scheduling until a predecessor job completes. A predecessor job is a job that must complete execution before this job can be scheduled. Jobs that have one or more predecessor jobs are called successor jobs. Nonstandard DJC jobs may be defined with the inclusion of //*PROCESS DJCPROC statements. Jobs belonging to a DJC network cannot be registered with automatic restart management (ARM).

Using parameters on the //*NET statement, you can make execution of a job depend on how a predecessor terminated, either normally or abnormally. When a predecessor job completes, a successor job can be processed based on any of the following conditions:

- Can have the count of predecessor jobs it is waiting for decreased by one. When the count reaches zero, the successor job is queued for execution.
- Can be flushed from the system. The successor job and all of its successors are canceled, printed, and flushed from the system.
- Can be retained until the operator releases it. The successor job and all of its successors are kept from being scheduled. The job is released only when its immediate predecessor is resubmitted or the operator decreases the predecessor job number.

//*NET JECL statement parameters

Following are the parameters on the //*NET control statements to define a DJS network:

NETID= **name** - Specifies the name of the DJC network for this job. All jobs put into the system with the same NETID name form a DJC network. To add a job to an existing DJC network, specify the NETID name for that job.

ABCMP= [**NOKP | KEEP**] - Indicates what action JES3 is to take if the job abnormally terminates.

NOKP - Indicates that JES3 is to purge the DJC network if the job abnormally terminates and has not been resubmitted by the time the other jobs in the network have completed. JES3 purges the network unless successor jobs or subnetworks are missing. If the ABCMP parameter is omitted, NOKP is the default.

KEEP - Indicates that the DJC network is to be kept in the system until (1) the job is resubmitted and completes normally or (2) the operator forces the network from the system. Use KEEP to make sure that the network is not purged until the operator takes proper action.

Note: If the job abnormally terminates, you can resubmit it to the DJC network, and the network will be retained until the job completes.

ABNORMAL= **[D | F | R]** and **NORMAL=[D | F | R]** - Indicates the action JES3 is to take for this job when any predecessor job completes execution normally or abnormally. If the ABNORMAL parameter is omitted, the default is R, and, if the NORMAL parameter is omitted, the default is D.

D - Requests that JES3 decrease this job's NHOLD count, which indicates the number of predecessors for this job. When the NHOLD count becomes zero, JES3 can schedule this job.

F - Requests that JES3 flush this job and its successor jobs from the system. JES3 cancels the job, prints any output, and cancels all successor jobs presently in the system, regardless of their normal or abnormal specifications. However, JES3 admits into the system all successor jobs that enter after the DJC network has been flushed. To flush those jobs, the operator must cancel the jobs or the network.

R - Requests that JES3 retain this job in the system and not decrease the NHOLD count. R suspends the job and its successor jobs from scheduling until either the predecessor job is resubmitted or the operator decreases the NHOLD count.

DEVPOOL= **(ANY | NET[,device-name,n]...)** - Identifies devices to be dedicated to this DJC network. The system allocates these devices only to jobs in the network. The DEVPOOL parameter should be coded on the //*NET

statement that establishes the network; it is ignored on other //*NET statements.

ANY - Indicates that jobs in the network can use any dedicated or undedicated device. JES3 tries to allocate from the dedicated pool before allocating any undedicated devices.

NET - Indicates that jobs can use only devices dedicated to the network.

device-name,n - Identifies a dedicated device. Code as many device-names with numbers as will fit on one statement. device-name specifies (1) a device name defined to JES3 by the installation during initialization or (2) a device-type defined to the system in HCD. n is the number of named devices. n is a number from 1 through 32,767.

DEVRELSE= **[YES | NO]** - Indicates when devices dedicated to the DJC network are to be released. The DEVRELSE parameter can be coded in several jobs in the network, but must not be coded in the first job. If no network job containing DEVRELSE=YES completes, the system releases the devices when it purges the network.

YES - Requests that JES3 release all devices at the end of this job. Completion of any job that specified DEVRELSE=YES causes the devices dedicated to the network to be released.

NO - Requests that JES3 release all devices only when the last job in the network ends.

NETREL= **(netid,jobname)** - Indicates that this job must be executed before the named job in another DJC network can be executed. The NETREL parameter can be specified only once for each job of a DJC network.

netid - Identifies the NETID for the successor job.

jobname - Names the JOB statement for the successor job.

NHOLD= **n** - Indicates the number of predecessor job completions required before this job can be released for scheduling. The predecessor number can include jobs from another DJC network. n is a number from 0 through 32,767.

When the predecessor number reaches 0, the job is scheduled for execution. The system reduces this number:

- When each predecessor job completes execution.
- By operator command.
- When a program in a predecessor job issues an assembler DJC WTO macro.

If you specify NHOLD=0 or omit the NHOLD parameter, this job has no predecessor jobs. JES3 can schedule it for immediate execution.

If the NHOLD count is incorrect, the following can occur:

If n is greater than the actual number of predecessor jobs, JES3 does not release this job for execution when all of its predecessor jobs complete execution.

If n is less than the actual number of predecessor jobs, JES3 prematurely releases the job for execution.

NRCMP= **[HOLD | NOHO | FLSH]** - Indicates that a network job that completed normally is being resubmitted and that JES3 must erase all references to the job before the job reenters the network.

HOLD - Indicates that JES3 is to hold the job until it is released by the operator.

NOHO - Indicates that JES3 is to allow the job to be scheduled as system resources become available.

FLSH - Indicates that JES3 is to flush the job from the system.

OPHOLD= [NO | YES]

NO - Indicates that the job is to be processed normally without operator intervention. If OPHOLD is omitted, NO is the default.

YES - Indicates that JES3 is to hold the job until it is released by the operator.

RELEASE= **(jobname[,jobname]...)** - Indicates that this job must be executed before the named job(s) in this DJC network can be executed.

jobname - Names the JOB statement for a successor job. You can specify from 1 through 50 successor jobnames.

RELEASE is the only parameter on the //*NET statement that can be split and continued on the next statement. To continue the RELEASE parameter, end the statement with the comma following a jobname and continue the next statement with the next jobname. The left parenthesis appears at the beginning of the jobname list and the right parenthesis appears at the end of the list. For example:

```
//*NET NETID=EXAMPLE,RELEASE=(JOB35,JOB27Z,MYJOB,
//*WRITJB,JOBABC)
```

RELSCHCT= **n** - Controls early set up of a dependent job's resources. Set up begins when the NHOLD count becomes less than or equal to n. n is a number from 1 through 32,767.

If you specify RELSCHCT=0 or omit the RELSCHCT parameter, JES3 does not set up dependent jobs early.

Note: Use this parameter carefully; RELSCHCT can tie up devices and data sets for long times. Do not specify the RELSCHCT parameter:

- For a job that may have catalog dependencies.
- For a job that contains one or more //*PROCESS statements.

9.29 DJC network processing

- Early DJC network JCL Scan
- DJC network definition to system
  - First job of DJC network causes network to be defined
- DJC network job scheduling
  - Controlled with //*NET control statement specifications
    - If no NHOLD parameter - job is eligible for scheduling
  - EARLY completion indication by PREDECESSOR
    - Job issues a WTO - "JESDJC**n** jobname netid"
    - **n** = 1 - normal -- **n** = 2 - abnormal completion
- Modifying the DJC network
  - *F N operator command
- DJC network termination
  - All jobs completed
    - No missing successor jobs *and* no missing subnetworks
  - Job pending count is equal to zero

Early dependent job control (DJC) JCL scan

The CI DSPs process the JCL for all jobs in a DJC network through the prescan phase, regardless of the progress of predecessor jobs through the system. As a result, most JCL and control statement errors in those dependent jobs can be detected and corrected, and the job can be resubmitted before its release. When the required predecessor jobs have indicated that dependent jobs can be released for execution, a POSTSCAN DSP is subsequently scheduled to complete postscan processing.

DJC network definition to system

The first job of a given DJC network entering the system causes the specified DJC network to be defined to JES3. All subsequent jobs with the same DJC network identification become members of that DJC network.

The first DJC job of a particular DJC network can use the DEVPOOL parameter of the //*NET control statement to reserve devices for the entire network. When reserving devices, the user can code the DEVPOOL parameter to refer to the requested devices by name. This parameter should refer to the names defined by the POOLNAMS parameter on the SETNAME initialization statement.

It is important to reserve devices for a DJC network if the DJC jobs pass data sets from one to another; this means that they have similar setup requirements. If devices are not reserved for a DJC network, the DJC jobs contend with other jobs in the system for the available devices when they enter setup. Since DJC jobs are normally held before setup and they are only

released for setup when their predecessor jobs have completed, other jobs can take over the devices that the DJC network will soon need again.

Both volume mounting operations and the time required by successor jobs to get through the system can be reduced by reserving the commonly required devices for the network. User exit IATUX24 allows you to examine information coded on a //*NET statement. You can examine the network id and the list of requested devices. A return code allows you to accept or reject the device request.

DJC network scheduling

The NHOLD parameter on the //*NET control statement specifies the number of predecessor jobs that must complete before the job is eligible for scheduling. If no NHOLD parameter is specified, then the job is eligible for immediate scheduling. If the NHOLD parameter is specified or if the job is in an operator-hold state, only the converter/interpreter and prescan phases of C/I service are scheduled. Postscan processing is suspended until the job is released when all predecessors complete execution. It is possible to make a job eligible for device setup before its predecessor jobs complete execution. To do this, code the NHOLD and RELSCHCT parameters on the job's //*NET statement. The values of these parameters determine when the job becomes eligible for device setup.

The job becomes eligible for device setup when its NHOLD value is equal to or less than its RELSCHCT value. JES3 reduces the NHOLD value by 1 each time:

- A predecessor job completes execution
- A job (the job need not be part of the DJC network) issues the following form of the DJC write-to-operator message:

  ```
  JESDJC1 jobname net-id
  ```

> **Note:** The variable 'jobname' refers to the name of the job to be terminated; NHOLD values for successor jobs will be decremented.

If a job becomes eligible for device setup before its predecessor jobs complete execution, JES3 schedules the job up to but not including generalized main service. JES3 then places the job in DJC-hold status.

Modifying the DJC network

Use the `*F N` operator command to:

- Hold an entire DJC job network or a specific job within the network
- Release an entire DJC job network or a specific job within the DJC network
- Cancel an entire DJC network or a specific job within the DJC network

Terminating the DJC network

A DJC job network is purged when all of the following conditions are satisfied:

- All jobs in the DJC network are completed
- There are no missing successor jobs in the DJC network
- There are no missing subnetworks
- The job pending count is equal to zero

The job pending count is the number of abended jobs that have been resubmitted, or have abended with the ABCMP=KEEP parameter specified on the //*NET statement. Specifying this parameter ensures that the DJC network will be retained in the system until the job is resubmitted and completed normally or until the DJC network is flushed by operator commands.

9.30 DJC network execution flow

```
IAT6160 JOB NET JOBNET   NOW ENTERING SYSTEM
IAT6100 (JOB25676) JOB JOBE     (JOB25675), PRTY=01, ID=VAINI        NET-ID=JOBNET   SUB=JOB25494
*I N ID=JOBNET
IAT8578 NET-ID=JOBNET   TOT=0000005 COMP=0000000 PEND=0000000 MISSC=N DATE=11/13/2008 TIME=12:34:29
*I  N ID=JOBNET J=JOBA
IAT8580 NET-ID     JOB NAME JOB NUM   NHOLD CT   SUCCESR   REL/SCH   STATUS
IAT8581 JOBNET   JOBA         JOB25669 00000000    0000002    0000000  OH
*F N ID=JOBNET R
IAT8034 NET-ID=JOBNET   IS BEING RELEASED
IAT2000 JOB JOBA     (JOB25669) SELECTED SC65     GRP=A       NET-ID=JOBNET
IEF404I JOBA - ENDED - TIME=12.56.32 - ASID=003B - SC65
IAT7100 (MAIN    ) *S DJCUPDAT,JOBA    ,JOBNET  ,1
*S DJCUPDAT,JOBA    ,JOBNET  ,1
IAT7306 DJCUPDAT ACTIVE
IAT2000 JOB JOBB     (JOB25670) SELECTED SC65     GRP=A       NET-ID=JOBNET
IAT2000 JOB JOBC     (JOB25671) SELECTED SC65     GRP=A       NET-ID=JOBNET
IEF404I JOBB - ENDED - TIME=12.56.32 - ASID=003C - SC65
IAT7100 (MAIN    ) *S DJCUPDAT,JOBB    ,JOBNET  ,1
*S DJCUPDAT,JOBB    ,JOBNET  ,1
IEF404I JOBC - ENDED - TIME=12.56.32 - ASID=003F - SC65
IAT7100 (MAIN    ) *S DJCUPDAT,JOBC    ,JOBNET  ,1
*S DJCUPDAT,JOBC    ,JOBNET  ,1
IAT7305 SUCCESSOR JOB JOBE     FOR NET JOBNET   BEING FLUSHED
IAT7100 (DJCUPDAT) *F N,ID=JOBNET  ,J=0025675,C
*F N,ID=JOBNET  ,J=0025675,C
IAT8036 JOB JOBE     (JOB25675) OF NET-ID=JOBNET   IS BEING CANCELED
IAT2000 JOB JOBD     (JOB25673) SELECTED SC65     GRP=A       NET-ID=JOBNET
IEF404I JOBD - ENDED - TIME=12.56.32 - ASID=0043 - SC65
IAT7100 (MAIN    ) *S DJCUPDAT,JOBD    ,JOBNET  ,1
*S DJCUPDAT,JOBD    ,JOBNET  ,1
IAT7310 NET JOBNET   HAS COMPLETED
IAT7315 DJC PROCESSING HAS COMPLETED
```

DJC network execution flow

Operator messages are provided to inform of normal DJC processing completion and to identify specific DJC error conditions. The DJC function includes the following three subfunctions of DJC network management:

- Initialization of a job network
- Scheduling and supervision of a job network
- Termination of a job network

Operator messages

The following message is issued for the DJC network entering the system. In this example the DJC network is the one shown on the “Defining a DJC network” on page 481.

```
IAT6160 JOB NET JOBNET   NOW ENTERING SYSTEM
```

The following message is issued for every job entering the network.

```
IAT6100 (JOB25676) JOB JOBE (JOB25675),PRTY=01,ID=VAINI NET-ID=JOBNET   SUB=JOB25494”
```

Operator commands

The ***I N,ID=JOBNET,parameters** command lists or displays the status of all active DJC networks. If none of the optional parameters is specified, this command provides statistics for each defined DJC network in the JES3 system. The statistics include network ID, total number of jobs in the DJC network, the number of completed jobs in the DJC network

(including jobs which have abended) and the number of jobs which have abended and are eligible for re-submission. The display also indicates whether there are missing successor jobs or sub-networks.

The **`*F N`** command alters the DJC network status. In the example **`R (`**release) is requested.

DJCUPDAT DSP

The DJCUPDAT DSP is activated by the **`*S DJCUPDAT`** command, which is sent using the INTERCOM macro by JES3 code when the first network enters the system.

Once the first job **JOBA** of the DJC network ends, the DJCUPDAT DSP is started. (The DJCUPDAT DSP is non-cancellable. The following message is issued:

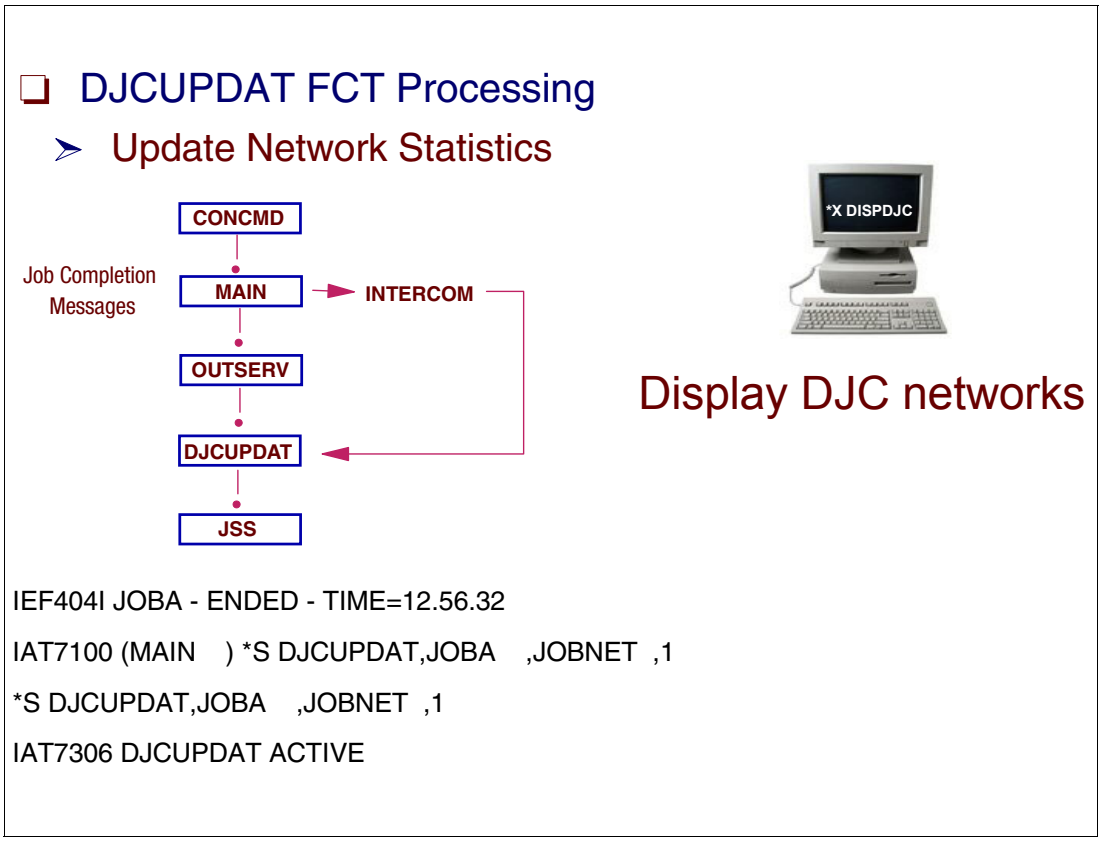
```
IAT7130 '*C DJCUPDAT  ' REJECTED, REFUSED BY 'DJCUPDAT'
```

The DJCUPDAT DSP (IATDCUP) updates the job net control block (JNCB) and net control blocks (NCBS) associated with a job net when a job within a net has terminated either normally or abnormally, or the net is to be modified or cancelled by the operator. The first DJCUPDAT invocation for the DJC network JOBNET will release jobs **JOBB** and **JOBC**.

At the successful completion of **JOBC** the DCJUPDAT DSP will flush **JOBE** as requested by the DJC network definition. The following message is issued:

```
IAT7305 SUCCESSOR JOB JOBE     FOR NET JOBNET   BEING FLUSHED
```

9.31 DJCUPDAT FCT processing



DJCUPDAT FCT processing

The DJCUPDAT FCT gets control through an INTERCOM macro. When DJC jobs complete execution, the MAIN FCT issues an INTERCOM macro with the command

```
*S DJCUPDAT,jobname,netid,n (n=1 or 2)
```

The DJCUPDAT FCT gets posted from the INTERCOM service and updates the DJC network status using the information in the INTERCOM message. The number ***n*** indicates whether the job ***jobname*** ended normally (1) or abnormally (2).

DISPDJC DSP

The DISPDJC facility displays the status of a dependent job control network. Displaying DJC network information can be obtained by issuing the following command:

```
*X DISPDJC
```

9.32 Status of DJC jobs

```
*I N ID=JOBNET LIST
IAT8580 NET-ID    JOB NAME JOB NUM  NHOLD CT  SUCCESR REL/SCH STATUS
IAT8581 JOBNET    JOBA     JOB25678 00000000  0000002  0000000 C
IAT8581 JOBNET    JOBB     JOB25679 00000000  0000002  0000000 C
IAT8581 JOBNET    JOBD     JOB25680 00000001  0000000  0000000 H
IAT8581 JOBNET    JOBE     JOB25682 00000001  0000000  0000000 C,H,N
```

- Status of the job indicated as follows:
  - Completed (C)
  - Abnormally completed (AC)
  - In network hold (H)
  - Failed at converter/interpreter (F)
  - In DJC operator hold (OH)
  - Null job no longer an active or participating member of the DJC network (N)
  - Job eligible for scheduling and might be active (E)

Status of DJC jobs

The visual shows the possible status of the DJC jobs in the network, as shown in the previous visual.

DJC status

The current DJC status of the specified job or jobs when you **`*X DISPDJC`**. The STATUS indicators are defined as follows:

AC The job completed abnormally

F The job failed at the reader/interpreter or the converter/interpreter

C The job is complete

OH The job is in DJC operator hold

H The job is in DJC hold

N The job is null

E The job is eligible for scheduling and may be active

Displaying DJC network (DISPDJC)

The following operator command displays the status of a dependent job control network on a and makes the output available either on the console or into a file that can be printed.

```
*X DISPDJC,NET=JOBNET
```

DISPDJC DSP example

An example display of the JOBNET DCJ network. Note that JOBC is missing (was not submitted as part of the network):

```
*X DISPDJC OUT=(PRT) NET=JOBNET

                        ***  NET-ID=JOBNET     ***     08.318   14:49:05
** JNCB PARAMETERS **  FLAG1=04    FLAG2=00    TOTAL COUNT=0000004  COMPLETED=0000003  PENDING=0000000
                                        H/R       ACTION       ATTRIBUTES    ATTRIBUTES
    JOB NAME    STATUS    SUCCESSORS   COUNT    PARAMETERS    FLAG1  FLAG2  FLAG3  FLAG4  FLAG5  FLAG6
    JOBA       COMPLETED   JOBB                                 43     04     00     00     20     88
                           JOBC
    JOBB       COMPLETED   JOBD                NRML=D ABNML=R   47     00     0C     05     20     88
                           JOBE
    JOBD      IN NET HOLD              00001   NRML=D ABNML=D   04     00     24     00     00     88
    JOBE      JOB IS NULL              00001   NRML=F ABNML=D   44     60     21     80     00     88
```

10

WLM batch initiator management

Workload management can dynamically manage the number of batch initiator address spaces in a JES3 environment. The control of the batch initiator management can be selectively turned over to WLM for one or more job class groups. WLM starts new initiators, as needed, to meet the performance goals of this work.

WLM-managed initiators are specified by setting MODE=WLM for a JES3 job class group. As many job class groups can be switched to WLM-managed mode as required. Job class groups can also be switched back to JES-managed mode.

When a job in a WLM-managed job class group has completed processing in MDS, the job is placed in the queue of jobs waiting to be selected for execution by service class. This queue is ordered by main service arrival time (i.e. when the job completed C/I processing). Unlike jobs in JES-managed groups, priority is not used to order the jobs on the queue, because priority is one of the many criteria that can be used to assign a service class for a job. Because jobs are ordered by main service arrival time and not priority, the time the job completes setup processing does not determine its place in the queue.

When a request for work arrives from a WLM-managed initiator, the source of the request narrows the choice in two ways:

1. The service class identification of the initiator limits the choice to jobs with that service class.

   WLM classification is responsible for assigning a service class and optionally a report class to a job based on the classification rules in the WLM policy. A service class is a group of work which has the same performance goals, resource requirements, or business importance. For workload management, you assign a service goal and optionally a resource group to a service class. WLM starts initiators on a service class basis.

   Note: Even though WLM management operates at a job class group level, WLM-managed initiators select work by service class and not by job class group like JES-managed initiators. Therefore, the queue of work for a particular service class can have jobs from different job class groups.

2. The processor on which the initiator is started limits the choice to jobs that can run on that processor.

Unlike job selection for JES-managed groups, following job selection parameters are ignored for selecting jobs in WLM-managed groups.

- IORATE on the CLASS statement
- CHOICE on the SELECT statement
- JOBMIX on the SELECT statement
- LSTOR on the SELECT statement
- JSPAN on the GROUP statement
- BAR on the GROUP statement

10.1 WLM batch initiator management

- Batch initiator management
  - JES3
    - Externals do not exactly reflect customer expectations
  - WLM
    - Fewer, simpler externals
    - Externals reflect better customer expectations
    - Dynamic, goal oriented initiator management
    - Workload balancing across a sysplex
    - Expectations-to-feedback correlation
    - Dynamic work and resource management
  - Both JES3 and WLM initiators

JES3 batch initiator management

Managing batch work has become increasingly complex within a sysplex due to the varying types of non-batch work that compete for resources and due to the number of jobs and systems in the sysplex. With JES3 initiator management, the system programmer is responsible for defining the number of initiators and where they will reside, and the operator is responsible for monitoring the sysplex, and making changes as necessary via ***MODIFY** commands. There are some problems with this:

- The externals do not reflect customer expectations. Instead of specifying that you want 1 hour turn around time for certain classes of batch work, you must somehow figure out the number of initiators that are needed to get 1 hour turn around time. If you add more work to the system, how do you determine what the correct number of initiators is?
- JES3 does provide some workload balancing controls, but they are difficult to define and do not react to changing conditions. If a system goes down, which system picks up the work? If batch initiators are running on that system, does the operator even know that he needs to start more initiators on other systems? How fast can someone react when there is a performance problem or when someone dumps a large number of jobs into the system? Why should a human have to be involved with making these kinds of decisions anyway (besides the initial definition of the performance goals).

WLM managed initiators

The solution is to allow WLM to manage initiators for a job class group. The advantages of doing this is that WLM will manage initiators based on performance goals provided by the installation, and unlike JES3, will be able to adapt to changing conditions.

WLM batch initiator management

Before the introduction of MVS workload management (WLM), MVS required you to translate your data processing goals from high-level objectives about what work needs to be done into the extremely technical terms that the system can understand. This translation requires high skill-level staff, and can be protracted, error-prone, and eventually in conflict with the original business goals. Multi-system, sysplex, parallel processing, and data sharing environments add to the complexity.

MVS workload management provides a solution for managing workload distribution, workload balancing, and distributing resources to competing workloads. MVS workload management is the combined cooperation of various subsystems (CICS, IMS/ESA®, JES, APPC, TSO/E, z/OS UNIX System Services, DDF, DB2, SOM, LSFM, and Internet Connection Server) with the MVS workload management (WLM) component.

WLM managements of batch initators

WLM management of batch initiators has the following advantages. Fewer, simpler externals or less externals are needed in JES3 to control initiators and to perform workload balancing. Once the WLM service administrator defines the goals and classification rules for batch work, the system takes over the job of starting and stopping initiators. Externals reflect customer expectations. With JES managed initiators it is the customer's responsibility to determine the number of initiators to be started on each system, the correct mix of jobs etc. The externals do not reflect an actual performance goal such as 1 hour turnaround time for jobs in class X. How do you translate 1 hour turn around time into initialization statements? The WLM batch initiator management starts initiators for work in job class groups as stated by performance goals specified in the WLM policy. The performance goals are typically expressed in terms that are found in service level agreements (e.g. 1 hour turn around time).

WLM resource management

Workload management provides automatic work and resource management support that dynamically adapts as needed. It manages the trade-offs between meeting your service goals for work and making efficient use of system resources. With one common terminology, workload management provides feedback to support performance reporting, analysis, and modelling tools. The feedback describes performance achievements in the same terms as those used to express goals for work.

JES3 and WLM managed initiators

Workload management can dynamically manage the number of batch initiator address spaces in a JES2 or JES3 environment. You can selectively turn over control of the batch initiator management to WLM for one or more job classes. WLM will start new initiators, as needed, to meet the performance goals of this work.

By specifying or defaulting MODE=JES on the JES2 JOBCLASS statement or the JES3 GROUP statement, you indicate that the initiators for the job class should be JES-managed, as in the past. By specifying MODE=JES, you keep the job class in JES-managed mode. (JES will manage the batch initiators for that job class, in the same way it has in prior releases.) By specifying MODE=WLM, you put that class into WLM-managed mode.

10.2 WLM and JES3 interactions

- WLM scheduling of work is based on job classifications, rather than job class groups
  - WLM provides a service to classify jobs
  - JES3 provide sampling data so WLM has a local and sysplex view of work queue
  - The JES3 still controls
    - Mains processors where jobs can run
    - Holding/releasing jobs and duplicate jobnames
  - WLM controls:
    - The order of jobs waiting to be scheduled
    - The number of initiators on each main

WLM and JES3 work classification

WLM classification rules are the installation's rules defined to categorize work into service classes, and optionally report classes, based on work qualifiers. Service class represent a group of work which has the same performance goals, resource requirements, or business importance. For workload management, a service goal and optionally a resource group is assigned to a service class.

Classification using work qualifiers

A work qualifier is what identifies a work request to the system. The first qualifier is the subsystem type that receives the work request. For JES3 environment the WLM subsystem type is JES. Other work qualifiers for subsystem type JES include; accounting information, job priority, scheduling environment name, XCF group name, JES3 subsystem name (from the IEFSSNxx parmlib member), job class, job name, user ID (used for RACF security checks).

When WLM receives a work request, it searches the classification rules for matching work qualifiers. Because a piece of work can have more than one work qualifier associated with it, it may match more than one classification rule. Therefore, the order in which you specify the classification rules determines which service classes are assigned.

There is only one set of classification rules in the WLM service definition for a sysplex. They are the same regardless of what service policy is in effect; a policy cannot override classification rules.

Sampling data

The data returned to WLM from each MVS initator represents the data collected during one sampling interval, an instantaneous, non-cumulative snapshot of the address space. The data is not tied to a particular job. WLM cannot just start batch initiators by itself; it needs information from JES3. Although WLM and SRM know about performance goals and system capacity, it is JES3 that knows about the backlog of jobs in the job queue, how many are eligible to run, and on what systems the jobs are eligible to run. From time to time, JES3 examines the job queue and provides information to WLM about the job queue in the form of sampling data. This includes the number of jobs that are eligible or ineligible to run from a system and sysplex perspective. Since the job queue is managed by the JES3 global processor, the global is responsible for collecting the sampling data and passing it to the JES3 local address spaces. The JES3 local address spaces then pass the information to WLM on their respective systems.

JES3 control of jobs to WLM

JES3 collects sampling information for WLM by examining jobs that are in one of the following phases of JES3 processing:

- Jobs waiting to be scheduled for main service, for example, jobs waiting for a class or group to be enabled, jobs waiting because of a duplicate job name condition.

  Jobs that are waiting to be scheduled for main service or that are active in MDS processing are considered ineligible to run. That is, they cannot be selected by initiator at this time because they are waiting for something. For example, if the job is in MDS allocation, it must first have its resources allocated before it can be selected by an initiator.
- Jobs in MDS (setup) processing, for example, jobs waiting for their resources to be allocated or for volumes to be mounted.
- Jobs in GMS select queue.

  If a job is in GMS select, it may or may not be eligible to run for any number of reasons. To determine whether a job on the GMS select queue is eligible to run, the following checks are performed:

WLM initiator job selection

Main service arrival time is used to control the order for jobs waiting for a WLM managed initiator. A job in a WLM managed group which requires setup may actually get selected for execution faster than it normally would if it was in a JES managed group.

10.3 Comparison of JES3/WLM job initiation

JES3	WLM
❑ Initiators, number and where, are controlled by JES3	❑ Initiators, number and where, are controlled by WLM and JES3
❑ Externals expressed in execution resources	❑ Externals expressed in goals, importance, capacity
❑ Start and stop of initiators defined in initialization and through commands	❑ Initiators are associated with a service class and sterted by WLM
❑ Jobs are selected within a job class group	❑ Jobs are selected within a service class
❑ Some workload balancing controls exist, but difficult to define	❑ Dynamic goal-oriented initiator management

WLM/JES3 job initiation comparison

JES3 managed initiators are entirely controlled by JES3. That is, JES3 determines how many initiators to start, where to start them, and when to start them based on initialization stream parameters (for example, the EXRESC parameter on the GROUP statement) and operator commands (`*F,G,system,G` command).

WLM-managed initiators are controlled entirely by WLM. That is, WLM determines how many initiators to start, where to start them, and when to start them based on information in the WLM policy, available processing capacity and work waiting for execution.

Initiator externals

Externals for JES3 managed initiators are expressed in terms of resources, not goals. For example, the installation must specify in the initialization stream how many initiators should be started, on what systems they should be started, what is the correct job mix based on the number of initiators etc.

Externals for WLM managed initiators are expressed in terms of goals and importance in the WLM policy. For example, batch work in job class A has 1 hour turn around time but is less important than the IMS workload. Batch work in class B has a discretionary goal and should be run when there is excess capacity.

Starting initiators

JES3 initiators are started and stopped based on installation definitions and backlog of jobs. It does not take into consideration how the systems are performing, whether the work is

meeting its goals, or whether the work is as important as other work. If JES3 is told to start 100 initiators on a main processor, it will do it without regard to whether its the correct or not. In addition, once jobs are selected for execution there is no guarantee that they will actually make much progress, if there is CPU contention etc.

WLM-managed initiators are started and stopped based on performance goals, available capacity, and importance, and based on input from JES3 (for example, the number of jobs eligible to run).

Initiator job selection

JES3 initiators are associated with a job class group. A job class group consists of one or more job classes defined with the CLASS initialization statement. An initiator that is started for a job class group only selects work from that group. A job class group may consist of jobs from different service classes.

WLM initiators are associated with a service class. An initiator that is started for a service class only selects work from that service class. A service class may consist of jobs from different job classes.

Note: A service class is a group of work which has the same performance goals, resource requirements, or business importance.

Initiator workload balancing

JES3 selects jobs for execution within a job class group in priority order. If there is more than one job with the same priority, the jobs are ordered by the time they arrive on the GMS select queue.

WLM initiators select jobs for execution within a service class by their main service arrival time. For the most part, this is the time that the job completed C/I processing. Priority does not influence whether a job is selected for execution first, so changing the priority will have no effect, unless it causes the job's service class to be changed.

For JES3 managed initiators, some workload balancing controls exist but they are difficult to define and static in nature. That is, they do not adapt to changing workloads and system conditions. The following initialization parameters can be used to perform some crude workload balancing:

- JOBMIX on the SELECT statement
- CHOICE on the SELECT statement
- LSTOR on the SELECT statement
- IORATE and LSTRR on the CLASS statement

JES3's workload balancing externals are ignored for WLM managed initiators. WLM automatically makes adjustments based on performance goals, importance, and available capacity.

10.4 Classifying jobs

- WLM classify service is used to assign a service class and report class at end of C/I for batch jobs
- WLM classification work qualifiers include:
  - Subsystem name
  - Job name (transaction name)
  - Job class (transaction class)
  - PERFORM parameter from the job card
  - Owner user ID
  - Priority
  - Accounting information

WLM classifying work service - IWMCLSFY

The purpose of this IWMCLSFY service is to factor in available information about an arriving work request in order to associate a service class and possibly a report class with it. When JES3 receives a batch job, after conversion JES3 invokes the WLM IWMCLSFY service using IATXWCLF macro. JES3 reclassifies jobs when the installation activates a new WLM policy from a different service definition than the old policy. In this case all jobs are reclassified.

WLM classification is typically done for a job at the end of C/I processing after IATUX09 has been called. However, a job can also be reclassified at the following times:

- When a new WLM service definition is installed and activated. When a new service definition is installed and activated, the classification rules may have changed and service classes may have been added and deleted. As a result, the service class and report class that is assigned to the job may change (we won't know until we reclassify it).
- When a job's priority is changed via an operator command or due to deadline scheduling. Since priority is one of the inputs to classification, the service class may change if the priority is changed.
- When a job's class is changed via an operator command. Since job class is one of the inputs to classification, the service class may change if the priority is changed.
- Jobs that were in the queue prior to JES3 release 8 do not have a service class assigned. These jobs will be classified when release 8 is brought up for the first time.

A job is not reclassified when its priority is changed due to aging. This is because aging only updates the priority in the in-storage control blocks and therefore is temporary.

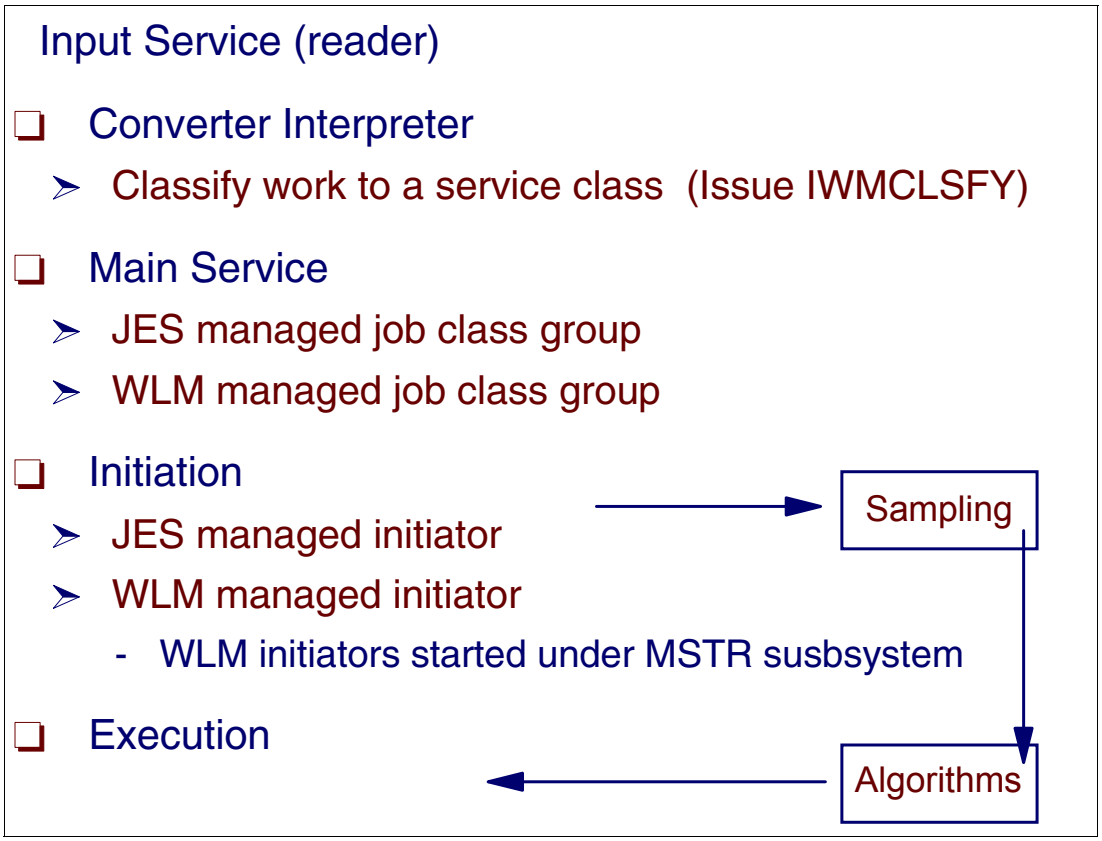
WLM classification work qualifiers

WLM classification is used to assign a service class and optionally a report class to a job using the classification rules that are defined in the WLM policy. Subsystem JES work qualifiers include:

- Subsystem JES
- Transaction name (job name)
- Transaction class (job class)
- Owning user id
- Priority
- Accounting information from the job card
- PERFORM parameter from the job card

Note: The installation can use this information in the classification rules to assign a service class and report class to the job. See *MVS Planning: Workload Management*, SA22-7602 for more information.

10.5 JES3 batch job flow



JES3 batch jobs classification service

The WLM IWMCLSFY service allows the caller to associate incoming work requests with a service class. The caller provides the work request qualifiers, such as priority, userid, and accounting information. Workload manager then checks the classification rules defined in the workload management ISPF application and provides a service class token representing the service class and report class (if any).

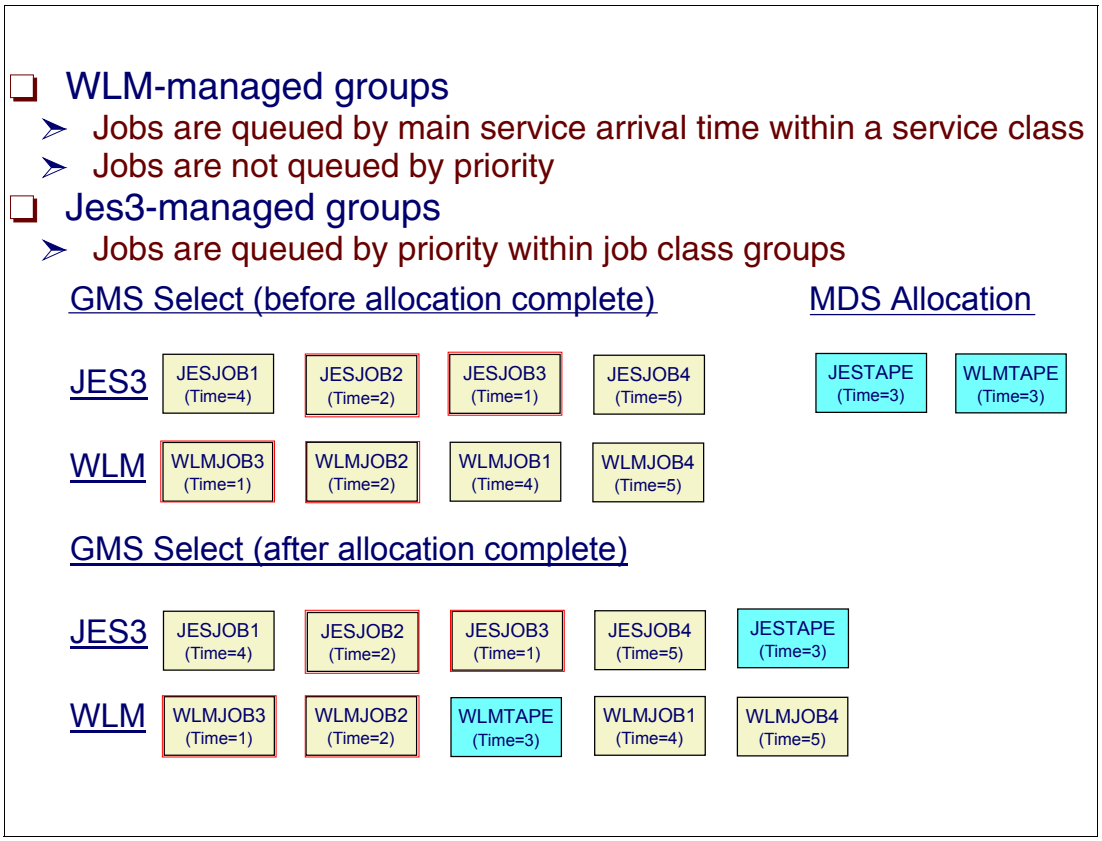
Converter/interpreter classification

The JES3 batch job flow is essentially the same as in prior releases. However, there a few differences with WLM batch initiator management. The first is that jobs are classified at the end of Conversion/Interpretation. Classifying a job means that the job is assigned a service class and optionally a report class based on the classification rules defined in the WLM policy. This is done for jobs in both JES3 and WLM managed job class groups.

Main service initiator management

From time to time, JES3 provides WLM with sampling information about the backlog of jobs in the queue for job class groups that are WLM managed. The sampling information contains information about the number of jobs eligible to run and ineligible to run from a system and sysplex basis. The sampling information is input to WLM's algorithms and is used to determine whether a service class is meeting its performance goals. As a result, WLM may decide to start or stop initiators for a service class. Sampling data is not provided for JES managed job class groups since WLM is not responsible for controlling JES managed initiators. WLM's involvement for jobs in JES managed job class groups start when the job begins execution. The WLM managed initiators are started under MSTR (master) subsystem.

10.6 JES3 job selection queues



JES3 job selection queues

Because main service arrival time is used to order jobs waiting for a WLM managed initiator, a job in a WLM managed group which requires setup may actually get selected for execution faster than if it was in a JES managed group. For example, assume we have the following situation shown in the visual.

Example of jobs in GMS select queue

Two jobs are waiting in MDS processing for their tapes to be mounted - one job is in a JES managed group (JESTAPE) and the other is in a WLM managed group (WLMTAPE). Assume that there are 4 jobs waiting to be selected for execution in both the JES and WLM managed groups. Assume that 2 of the jobs in each group completed C/I processing after the jobs that require tape mounts. That is, their main service arrival time is after the tape mount jobs. Assume that all jobs in both groups have the same priority and that they have about the same execution time.

Jobs with tape mounts

If the tape mounts are satisfied for both jobs at the same time, the job in the WLM managed group will be selected for execution first. This is because JES managed jobs are ordered by priority and the JES managed job will be added to the end of the queue. Jobs in WLM managed groups are ordered by main service arrival time; therefore, the WLM managed job will be added in the middle of the queue. As a result, the amount of time the job actually holds onto the resource will be shorter as compared to JES managed jobs, since it will take longer for the JES managed job to be selected for execution.

10.7 Reclassifying jobs

- A job may be reclassified when:
  - Priority changed with *F,J=job,P=xx command or by deadline scheduling
  - Class changed with *F,J=job,CLASS=class command
  - A new WLM service definition is installed and new policy activated
- A job is not reclassified due to priority aging
  - MAGER/MAGEL or SAGER/SAGEL on SELECT statement
- A new service defintion can be installed
  - A new policy is activated

Reclassify jobs

Using JES3 commands, the change of a priority level or jon class can cause a job to be reclassified:

- Priority changed via **`*F,J=xx,P=xx`** command or via deadline scheduling Class changed via **`*F,J=xx,CLASS`**=class command.
- A new WLM service definition is installed and a policy is activated.

Change priority of a job

A job is reclassified when its class or priority is changed because class and priority are two of the inputs to classification. As a result, the job's service class may change if either of these attributes are used in the classification rules.

Change WLM service definition

A job is also reclassified when a new WLM service definition is installed and a new policy is activated because the WLM classification rules may have changed, and service classes may have been added or deleted. In this case, all jobs that have completed C/I processing but have not completed execution are reclassified. Jobs that are in execution will be reclassified by SRM. Note that activating a new WLM policy without a service definition change does not cause jobs to be reclassified since classification rules and service classes do not change.

Priority aging on a job

A job is not reclassified when priority aging is used to update the job's priority. That is, SAGER/SAGEL or MAGER/MAGEL is specified on the SELECT statement. Unlike deadline scheduling, priority aging only updates the job's priority temporarily for the purposes of moving the job up in the queue ahead of other jobs. Once the job has completed setup or has been selected for execution, the priority is restored to its original value. Priority aging for job selection purposes (MAGER/MAGEL) is ignored for jobs in WLM managed groups.

Installing a new service definition

When a new WLM service definition is installed and a new policy is activated, jobs are reclassified. This is necessary because the classification rules may have changed, and service classes may have been deleted. If a job's service class was set via the MVS **RESET** or ***F J=** command, JES3 will attempt to use the same service class that was assigned to the job. If the service class is still defined in the WLM policy, the job will be assigned the same service class. Otherwise, the service class determined via the classification rules will be used.

A new service definition can be activated while JES3 is running or while JES3 is down. When JES3 recognizes that the service definition has changed, it suspends job selection and issues the following highlighted message:

```
IAT2011 WLM RECLASSIFICATION IS IN PROGRESS
```

No jobs will be selected for execution at this time, even jobs in JES managed job class groups. After all of the jobs have been reclassified, job selection is resumed and the following message is issued:

```
IAT2016 WLM RECLASSIFICATION HAS COMPLETED
```

10.8 Change Service Class of Job

- Change a batch job's service class
  - ***F J=xxx,SRVCLASS=srvclass**
    - Similar to MVS RESET jobname,SRVCLASS= command but job doesn't have to be executing
- If job is executing, MVS is notified of service class change
- If the service class is set with a command, the service class is used if possible when a WLM service definition change occurs
- SRVCL= can be used instead of SRVCLASS=

Command to change job service class

There are two ways for the operator to change a job's service class:

- With the MVS **`RESET jobname,SRVCLASS=srvclass`** command

  The **`RESET`** command can only be issued while the job is in execution. When this command is issued, JES3 is notified and updates the job's control blocks so that MVS and JES3 have the same view of the job.

- With the JES3 **`*F J=job,SRVCLASS=srvclass`** command

  The **`*F J=`** command can be issued anytime after the job has completed C/I but before it has completed main service. If the job is executing when the command is issued, MVS is informed of the change. The **`RESET`** command can be issued for all jobs included demand select jobs; the **`*F J=`** command can only be issued for batch jobs.

Note: If possible, you should not have jobs in JES and WLM managed job class groups going to the same service class. Doing so weakens the relationship between the number of WLM managed initiators and queue delay (the amount of time waiting for an initiator) that is observed. For example, if a velocity goal is used, queue delay is included in the velocity calculations for WLM managed initiators. If a service class contains jobs that run under JES and WLM managed initiators, queue delay will be included in the velocity calculations for the service class for some but not all jobs.

10.9 WLM Services used by JES3

- ❑ **IWMCONN - Informs WLM that JES3 exists**
- ❑ **IWMDISC - Disconnects JES3 from WLM**
- ❑ **IWMBQRY - Used in Extended Status**
- ❑ **IWMPQRY - Used to get active policy information**
- ❑ **IWMBREG - Registers a batch job queue**
- ❑ **IWMBDREG - Deregister a batch job queue**
- ❑ **IWMRESET - Change service class of work in execution**
  - **- Quesce work currently in execution**
  - **- Reclassify work currently in execution**
- ❑ **IWMCLSFY - Classify work request**
- ❑ **IWMBSET - Obtain a new service class token**
- ❑ **IWMBRIP - Tells WLM it needs to start an Initiator**
- ❑ **IWMBREQ - Asks WLM start an Initiator for *F J=xxx,RUN**
- ❑ **IWMBSMP - Passes batch job queue delay samples**
- ❑ **IWMBLOC - Asks WLM for best system to execute job**

WLM service to connect to WLM

During JES3 initialization JES3 connects to WLM by invoking the IWMCONN service. The purpose of this service is to connect a calling address space to WLM.

IWMDISC service

IWMDISC allows the caller to disconnect from the workload management services. This means that the input connect token can no longer be passed to workload management macros such as IWMCLSFY and IWMRPT. When a program disconnects, any enclaves associated with the input connect token are deleted from the system. Any SRBs running in the enclave are run as preemptible SRBs at the priority of the home address space. Any enclave TCBs are converted to ordinary TCBs.

IWMPQRY service

The purpose of this service is to return a representation of the active policy which could be used to explain how the sysplex is being managed and could be used in conjunction with current measurements to evaluate the condition of the system/sysplex.

IWMRESET service

The IWMRESET macro allows the caller to perform the same functions as the **RESET** system command. The caller can change the service class of work currently in execution, reset to a new service class, quiesce work currently in execution, and reclassify work currently in execution according to the service policy in effect.

IWMCLSFY service

Once JES3 has connected to WLM, it is allowed to classify jobs using the IWMCLSFY service. The output from the IWMCLSFY service is a service class name and a WLM classification token. Classification of jobs occurs only on the global system. During classification, the classification rules in the WLM policy are used to assign a service class and optionally a report class to a job. The service class is important since it represents the performance goals and importance for the job, and WLM starts initiators on a service class level. The report class is not that important from a JES3 point of view and is only used for reporting purposes.

After a job has been classified, JES3 becomes aware of the job's service class. In order to make WLM aware of the service class and to eventually allow WLM to start initiators for it, the service class must be registered with WLM (IWMBREG). Unlike classification, which is done only on the global, the service class must be registered on every system in the sysplex. WLM will not start initiators on a particular system for a service class unless it is registered on that system.

Now that the service class is registered, JES3 needs to provide sampling data which describes the number of jobs that are waiting for initiators. JES3 also needs to track pre-execution delays for jobs. Both of these pieces of information are used by WLM to determine whether a service class is meeting its performance goals, and whether initiators should be assigned to (or unassigned from) the service class. Note that WLM does not start initiator address spaces every time a service class needs one. WLM keeps a pool of idle initiators, not bound to any service class, and dynamically binds them to a service class as needed.

Once an initiator has been assigned to a service class, it goes through the normal job select interface just like JES3-managed initiators.

10.10 Registering service class queues

- A JES3 service class queue must be registered on each system where WLM should start initiators
- Service class queue is registered when
  - The first job in the service class is available for execution - (IWMBREG service)
  - When a job's service class is changed as a result of an operator command -(IWMRESET service)
- When a policy change occurs and jobs need to be reclassified
- Registration accomplished via the IWMBREG service
  - Input is 16 byte queue token = JESXCF group name and service class name

Registering service class queues

Before WLM will start initiators for a service class on a particular system, the JES3 service class queue (queue of jobs having the same service class) must be registered on that system. On the global, a service class is typically registered with WLM the first time a job in that service class is scheduled for main service.

IWMBREG service

When a new WLM service definition is installed and activated. When this occurs, classification rules may have changed and service classes may have been added or deleted. As a result, jobs need to be reclassified. If a job is assigned a service class that is not known to JES3, the service class must be registered.

IWMRESET service

When a job's service class is changed as a result of a command. For example, the operator may issue a ***F J=job,SRVCLASS=** or an MVS **RESET jobname,SRVCLASS=** command to change the service class directly. Or the service class may be changed indirectly as a result of the job needing to be reclassified when the operator changes a job's priority via the ***F J=job,P=prty** command or changes a job's class via the ***F J=job,CL=class** command. If the service class assigned to the job is one that is not known to JES3, the service class must be registered.

A service class is registered on a local processor the first time the service class appears in the sampling information sent by the local. This will be covered in more detail later.

Registration is accomplished by issuing the IWMBREG macro. The input to IWMBREG is a 16 byte queue token that consists of the JESXCF group name and the service class name. By adding the JESXCF group name to the queue token, multiple JES3 complexes can exist in the same SYSPLEX and still allow WLM to start initiators separately for each JES3 complex.

Policy change

A new service definition can be activated while JES3 is running or while JES3 is down. When JES3 recognizes that the service definition has changed, it suspends job selection and issues the following highlighted message:

```
IAT2011 WLM RECLASSIFICATION IS IN PROGRESS
```

No jobs will be selected for execution at this time, even jobs in JES managed job class groups. After all of the jobs have been reclassified, job selection is resumed and the following message is issued:

```
IAT2016 WLM RECLASSIFICATION HAS COMPLETED
```

Registered service classes

Once service classes have been registered, how do we get rid of them? Every 20 minutes, the WLM FCT is posted to perform a deregistration scan. The service class tables are searched for service classes that do not have any jobs referencing them and a time stamp is set when the service class became empty. This time stamp is reset if a job that uses the service class is scheduled for main service. If a service class has not been referenced by a job for more than 60 minutes, the service class s deregistered. The reason 60 minutes is interesting is that once a service class has been deregistered, all history information is lost. The history information is used by WLM for dealing with sparse arrival problems (i.e. when jobs are submitted into the service class infrequently).

When a service class is deregistered, the following message is written to SYSLOG:

```
IAT2012 WLM DEREGISTRATION SUCCESSFUL FOR SRVCLASS srvclass - system
```

If the deregistration request fails, the following message is written to the console:

```
IAT2013 WLM DEREGISTRATION FAILED FOR SRVCLASS srvclass, RETURN CODE =
                 REASON CODE = rsncode
```

If a service class is deregistered, sampling data is sent to all of the local processors. Since the service class that was deregistered will no longer appear in the sampling data, this will also cause the locals to deregister the service class.

10.11 Batch job delays

- WLM needs to know about delays prior to execution so it can compute queue delay
  - Queue delay = amount of time waiting for a WLM initiator
  - Only delay WLM can affect is the queue delay
  - No affect how long a job is in hold or wait for resources
- JES3 keeps track of delays - (pre-execution delays)
  - Operational delay
  - JES3 scheduling delay
  - Resource delay
  - Queue delay
  - C/I delay (tracked but not used to compute queue delay)
- JES3 delays are passed to initiator at job select

Batch job delays

When a job enters the system, there are a number of things that can affect how fast a job gets selected for execution. For example, the job could be put into operator hold, the job's resources may not be available, the job's class and group may not be enabled, or all initiators may be in use. For jobs in WLM managed job class groups, the only one of these delays that WLM has any control over is the amount of time a job waits for an initiator, which is known as queue delay. If a service class is not meeting its goals and waiting for an initiator is a significant part of the reason why the goals are not being met, WLM can help the service class by adding more initiators. WLM cannot control how long a job is in operator hold, how long it takes to allocate its resources etc. so these times must be factored out of WLM calculations. JES3 is responsible for keeping track of these delays that are not managed by WLM and passing them to the initiator during job select. WLM then uses this information to calculate queue delay (time a job waits for an WLM managed initiator).

JES3 operational delays

The following delays are tracked by JES3:

- Operator hold - The operator issued a **`*F,J=xx,H`** command to hold the job.
- Priority hold - The operator issued a **`*F,Q,P=xx,H`** or **`*F,Q,H`** command to hold the job's priority or all priorities.
- Spool hold - The operator issued a *F,Q,DD=ddname,HOLD command to put a spool volume in hold.

- Class/group unavailable - The class or group is not enabled on one or more systems, a **`*F,G`** command was issued to disable the class or group.
- System unavailable - The system has not connected to the JES3 global or is varied offline via a. **`*V main,OFF`** command
- The job is on the MDS WAITVOL queue waiting for a **`*S S`** command to be issued before allocating the job's resources.
- The job is on the MDS ERROR queue because of an error in MDS processing. The job is waiting for a **`*C ,S`** command to cancel the job or a **`*R S`** command to restart the job through setup

JES3 scheduling delays

Delays caused by JES3 or installation defined limits or algorithms such as duplicate job name wait, class limits. Although JES3 maintains JES scheduling delay separate from operational delay, the two value are added together before they are passed to the initiator. This includes duplicate job name delay and the delay caused because the TDEPTH, TLIMIT, MDEPTH, or MLIMIT has been exceeded

JES3 resource delays

Delays waiting for resources to be allocated or to become available as follows:

- The job's SMS resources are unavailable (the job is on the MDS system select queue).
- The job's JES3 resources (devices, volumes, data sets) are unavailable.
- The job's scheduling environment is unavailable on one or more systems.
- The job's scheduling environment is undefined.
- The job is on the MDS verify queue waiting for the operator to mount its volumes.
- The job is on the MDS volume unavailable queue because the operator issued a **`*F,S`** command to make a volume unavailable.
- The job is in MDS fetch or system verify processing.

Queue delays

The amount of time spent waiting for an initiator. That is everything else is available except an initiator.

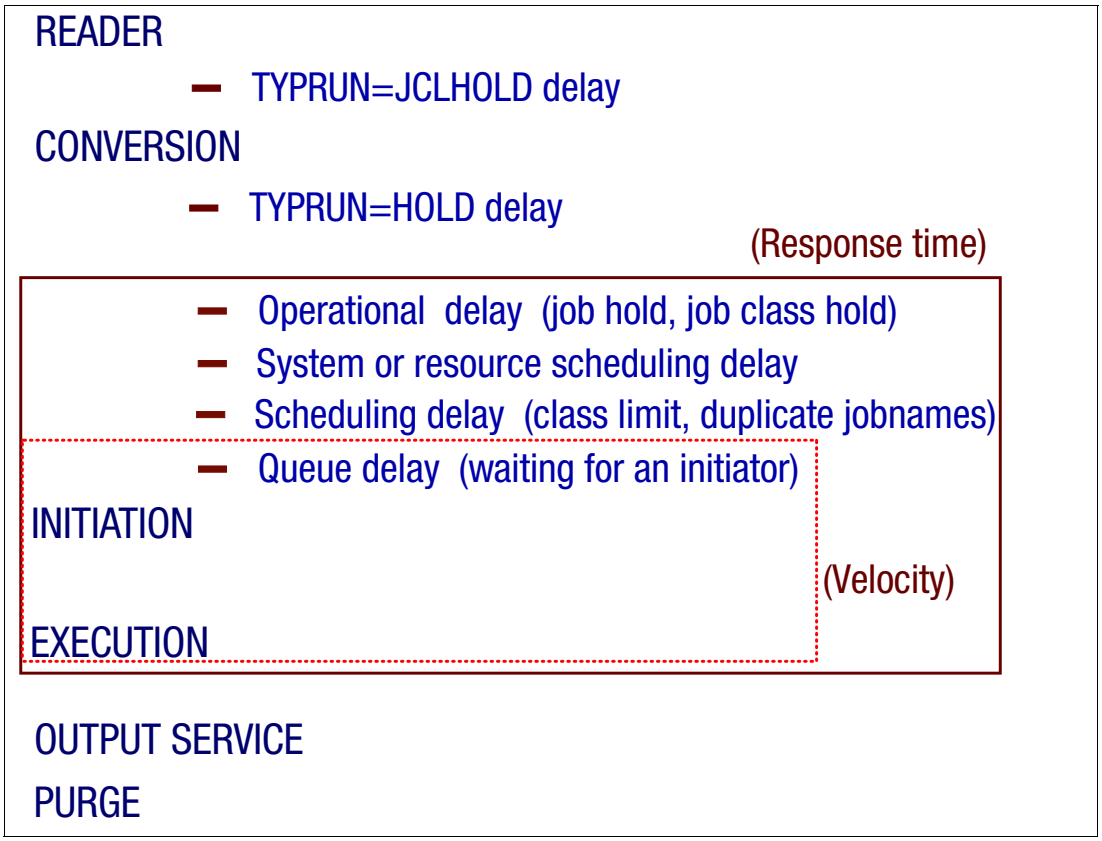
C/I delays

The amount of time it took to complete converter/Interpreter processing in JES3. This includes the amount of time the job spent waiting to be scheduled for C/I; for DJC jobs, the amount of time spent waiting for a predecessor job to complete; as well as the actual time spent in C/I processing. The clock starts running for conversion delay as soon as the job is added to the JES3 job queue during input service. The exception is if TYPRUN=HOLD is specified. c In this case, since the delay is user inflicted, the clock starts running when the job is released from hold.

Job select and delays

When a job is selected for execution, WLM uses the time the job was selected for execution, the main service arrival time, and the delay times to compute the amount of time spent waiting for an initiator.

10.12 Pre-execution job delays



Response time goal delays

This visual shows the delays that are kept for response time goals. The delay time starts following conversion and includes the time waiting for an initiator. The goals types are:

- **Discretionary**

  Work is run when the system resources are available.

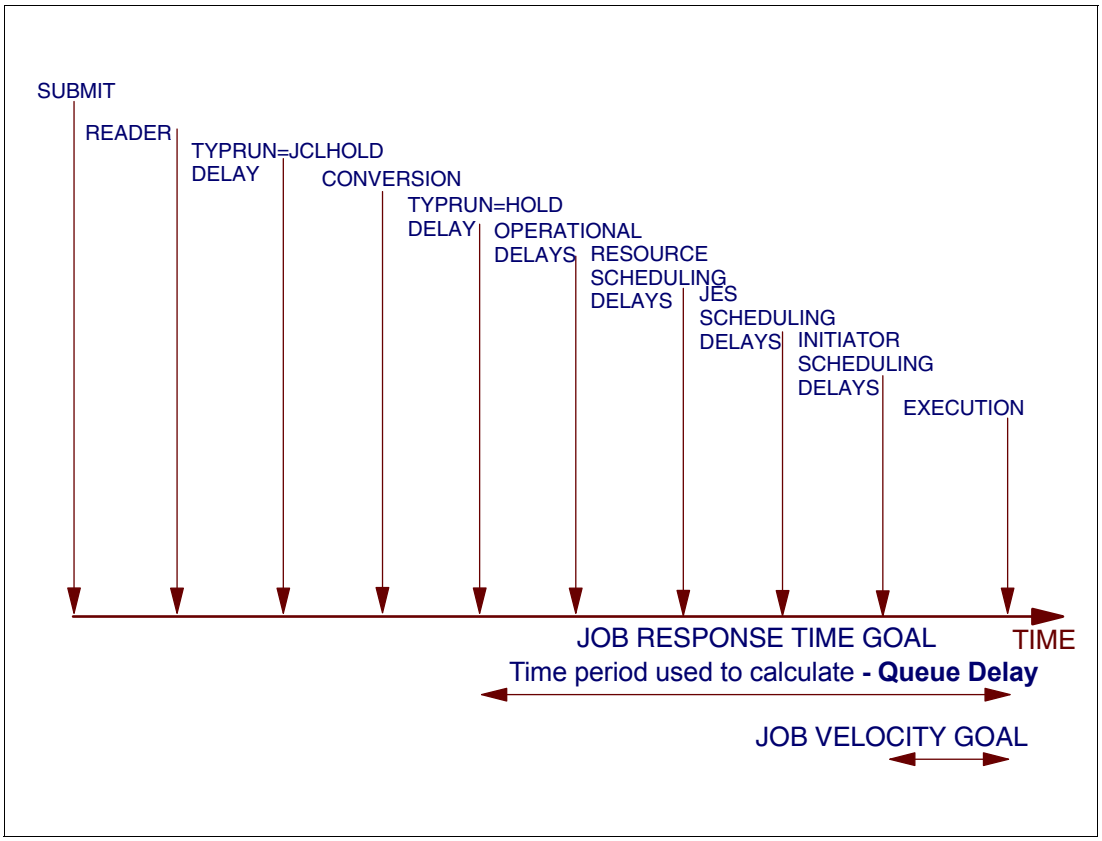
- **Response Time Goal**

  Response time goals are measured from the time the job completes C/I processing (main service arrival time) until it completes execution. Conversion delay is still tracked by JES3, but it is not included as part of the goal. TYPRUN=HOLD time is not included in either the conversion delay or in the response time calculation. If you are currently using velocity goals for batch work, because TYPRUN=HOLD and conversion delay were included in the definition of response time, you may wan to reevaluate this and see if a response time goal is more appropriate.

- **Velocity Goal**

  Velocity is a measure of how fast work should run when ready (without being delayed for WLM-managed resources). For jobs in JES-managed job class groups, the definition of velocity is unchanged. For jobs in WLM-managed job class groups, velocity includes queue delay (the amount of time waiting for an initiator). By including queue delay in the velocity calculation, WLM can determine whether adding another initiator will help a service class meet its performance goals. If you have velocity goals defined for batch service classes, the achieved velocity will decrease as a result of queue delay being included in velocity calculations. You may need to adjust your velocity goals.

10.13 Job response time goal



Job response time delays

This visual shows the delays in a different format, but is the same as the last visual. What is different is that it shows the delays kept for job velocity goals, which is just the time spent waiting for an initiator. JES3 keeps track of the different delays prior to execution, with the exception of user delay, for all jobs (even ones in JES managed groups). These delays are reported in RMF™ reports and SMF records.

Response type goals

Response time goals include from the time when the job completed C/I processing (main service arrival time) to the time the job completed execution. Conversion delay is still tracked but is not included as part of the response time goal. TYPRUN=HOLD is not included in either the conversion delay or in the response time calculation.

Velocity type goals

If you are currently using velocity goals for batch service classes because of the TYPRUN=HOLD and conversion delay, you may want to consider switching to use response time goals, provided that other delays such as resource delay are not significant. For JES managed job class groups, velocity is unchanged. But for WLM managed job class groups, velocity includes the amount of time spent waiting for an initiator (queue delay). By including queue delay in the velocity calculations, WLM can determine whether adding another initiator would help a service class meet its performance goals. If you have velocity goals for batch service classes, the actual velocity will decrease as a result of queue delay being included in the calculations. Therefore, you may have to adjust your velocity goals.

10.14 Controlling batch job selection

- WLM dynamically controls initiators
  - Available capacity - Goals - Pending work
- User controls:
  - Job class limits for simultaneous execution
    - TDEPTH
  - Start or stop selection of batch jobs
    - Job class group ON/OFF
  - Use system affinity
    - //*MAIN SYSTEM=
  - Schedule a specific batch job
    - *F J=nnnnn,RUN
  - Use scheduling environments
    - //jobname JOB ....,SCHENV=schenv_name

WLM batch job selection

WLM controls the starting and stopping of WLM managed initiators. However, WLM does honor all the user controls for jobs that have always been part of JES3 scheduling of jobs to an initiator. Jobs that have specific systems that they must execute on are honored by WLM. Also, class limits are honored by WLM, and scheduling environment resource limits are honored.

WLM improves the balancing of WLM managed batch initiators between systems of a sysplex. On highly utilized systems, the number of initiators is reduced while new ones are started on low utilized systems. This can improve sysplex performance with better use of the processing capability of each system. WLM attempts to distribute the initiators across all members in the sysplex to reduce batch work on highly used systems while taking care that jobs with affinities to specific systems are not hurt by WLM decisions. Initiators are stopped on systems that are utilized over 95% when another system in the sysplex offers the required capacity for such an initiator. WLM also increases the number of initiators more aggressively when a system is low utilized and jobs are waiting for execution. Batch initiator balancing improves the performance and throughput of batch workload over the sysplex.

User and operator control for job selection

This visual contains the various ways a job might be selected by an initiator based on user specifications in the JCL, initialization statements control, or operator commands. These methods are described in the following figures.

10.15 Batch initiator job selection

- WLM performs multisystem workload balancing
- For batch, WLM determines when and where to start initiators based on information from JES3
  - WLM/SRM know about performance goals and system capacity
  - JES3 knows about the backlog of jobs, how many are eligible to run, and on what systems they are eligible to run

WLM initiator job selection and workload balancing

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

WLM starting initiators

WLM initiator workload balancing across a sysplex is automatic. WLM decides when to start initiators and how many to start based on performance goals and the importance of batch work with respect to other work. WLM management of initiators does not necessarily imply that there will be an equal number of initiators on each system.

WLM-managed initiators are controlled entirely by WLM. WLM determines how many initiators to start, when to start them, and where to start them based on performance goals in the WLM policy, backlog of jobs, and system capacity.

JES3 knowledge of backlog

When the WLM FCT is posted for sampling, it looks for jobs that are waiting to be scheduled for main service, or have been scheduled for main service but not executing. The following queues are checked for jobs:

- Queue of jobs (JQE's) waiting to be scheduled for main service because of a duplicate job name or because their main, class, or group is unavailable. Jobs that are waiting to be

scheduled for main service for other reasons such as operator hold are not included since they are not on a readily scannable queue. That does not matter for sampling purposes since this would not cause WLM to start any more initiators if these jobs were included.

- MDS related queues prior to execution. This includes the MDS fetch, system select, allocate, verify, system verify, volume wait, error, and unavailable volume queues.
- The GMS select queue for WLM managed initiators is used when a job is ready to be selected for execution when a job is added to the GMS select queue. For JES3 managed job class groups, jobs are queued from the job class group table. For WLM managed job class groups, jobs are queued from the service class table for the service class assigned to the job. Only the service class related GMS select queues are searched for jobs since WLM does not start initiators for JES3 managed groups.

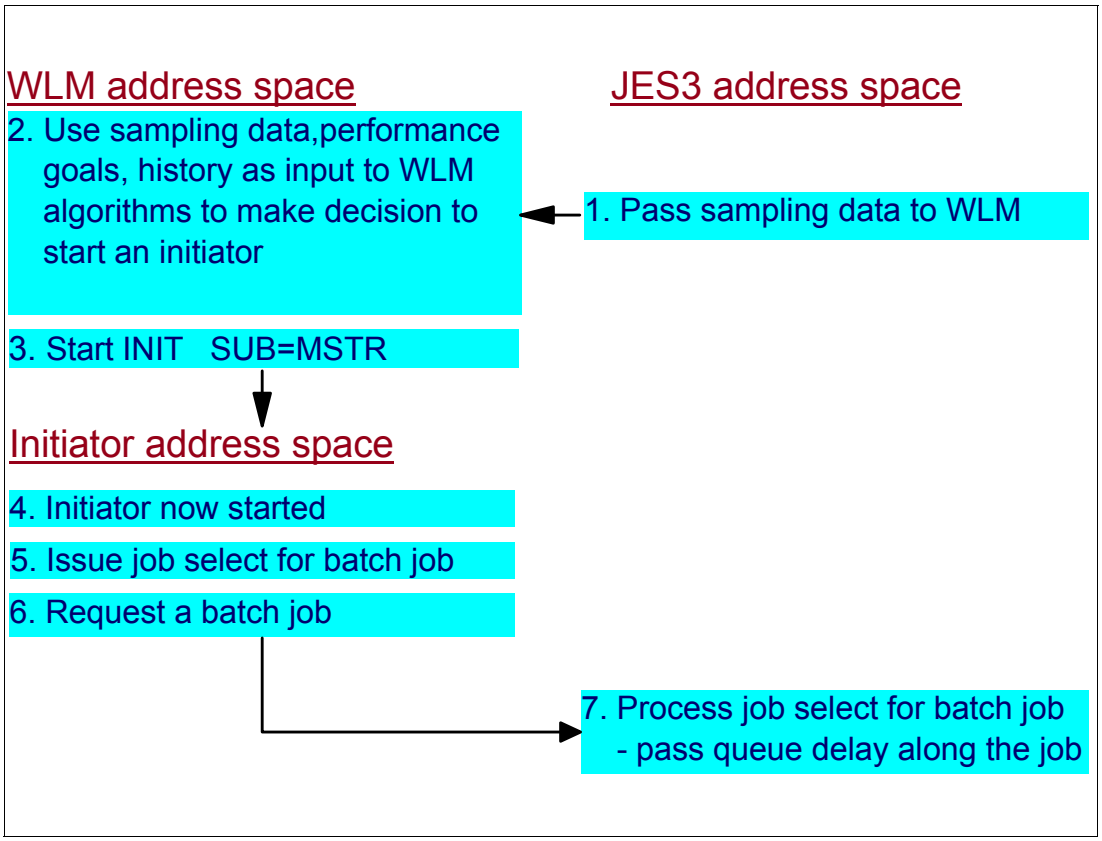
JES3 sampling data

For each job, a WLM job sampling element (WJS) is created in the WLM data space. The WJS contains information such as the job number, main eligibility mask, service class name, WLM classification token, and group sequence number. This information is used by the WLM subtask to determine whether the job is eligible or ineligible to run. There are three WJS queues pointed to by the WLM data area:

- Jobs waiting to be scheduled for main service
- Jobs active in MDS processing
- Jobs waiting to be selected for execution (GMS select queue anchored from the SRVC)

The WLM FCT actually does very little during sampling in order to keep the overhead under the IATNUC task to a minimum. The WLM subtask is one responsible for analyzing the data and passing it to WLM/SRM on the glob processors, and to the WLM subtasks on the local processors.

10.16 Initiator startup and job select



WLM initiator start

JES3 controls the starting and stopping of JES3 managed initiators. Therefore, there should never be a case where JES3 selects a job for an initiator that is being stopped. However, JES3 does not control the starting and stopping of WLM managed initiators. When a WLM initiator is finished processing a job, it issues a job select for another job just like JES3 initiators do. This causes the JES3 job select SSI routine, IATSIJS, to get control and send a job select request to be sent to the global. However, at any time, WLM can tell the initiator that it should no longer select work from that service class (this is called unbinding from the service class). This may even occur while the JES3 global is in the process of sending back a job to the initiator. If a job is being selected when the initiator decides to no longer select work, the job will appear to be stuck in execution.

Initiator startup for WLM managed initiators is a lot simpler from JES3's point of view. Here are the differences between the two:

- WLM initiator starts under master subsystem the initiator address space, not JES3
- JES3 does not get involved with allocating, opening, and writing to the STCINRDR DD
- JES3 does not create a job in the JES3 job queue for WLM managed initiators. As a result, the first SSISERV is not done and the MEMDATA is not created until later in the flow. In addition, JES3 will never see a demand select request for a WLM initiator because there is no JES3 job number assigned.

In order for WLM to start initiators, the service class is registered with WLM. This makes WLM aware that JES3 has work that references the service class. However, without additional

information, WLM does not know how many initiators to start. In order for WLM to determine how many initiators need to be started, JES3 must collect sampling information for each service class and pass it to WLM. JES3 also collects sampling information for each report class assigned to a job.

WLM initiators

WLM initiators are assigned a sequence number when they first select a job. This sequence number identifies the initiator and is put into the job's JDAB and RQ when the job is selected for execution. This allows us to determine which initiator is processing the job.

WLM initiator address space

The flow of control for WLM managed initiators is as follows:

1. From time to time, JES3 provides sampling information to WLM about the backlog of jobs.
2. WLM uses the sampling information, the performance goals in the WLM policy, and history information about each service class as input to its algorithms. As a result, WLM may decide to start an initiator for a service class.
3. An address space is started under the master subsystem. JES3 is not involved at this point so no JES3 job number is assigned.
4. The initiator is now started and ready to issue a job select request.
5. The initiator issues a job select request. Since this is the first time JES3 is involved with the initiator, a MEMDATA header and MEMENTRY 1 need to be created at this time before the job select request is processed.
6. An SSISERV request is issued to send a request to the global to ask for a batch job. Since this is a WLM managed initiator, the initiator will request a job from the service class assigned to the initiator.
7. When the global receives the job select request, it will choose a job on the GMS select queue for the service class and send job information back to the initiator. The initiator will use this information to initialize the environment so that the job can run (this includes creating the MEMENTRY 2 for the batch job).

10.17 WLM initiators and sampling data

- No more jobs in the service class
  - When to stop initiators
- JES3 sampling data
  - Determines how many initiators WLM should start
  - Notifies WLM about jobs in the select queue
    - Which system jobs can execute on
    - Number of jobs in the service class
    - Status of jobs
- Sampling interval
- Number of jobs eligible for each LPAR

Jobs in service classes

When WLM decides that an initiator should no longer select work from a service class when there are no longer any jobs in the select queue for that service class. This causes JES3 to send an initiator deselect request to the global to tell JES3 that the initiator(s) is no longer selecting work from the service class. When the deselect request arrives on the global, GMS searches the execution queue to see if JES3 selected a job for the initiator (remember the initiator sequence number is in the job's RQ). If a job was selected for the initiator, the job is unselected and put back on the GMS select queue. If a job was not selected, the GMS destination queue is searched for the original job select and the request is purged.

Note: When an initiator unbinds from a service class, the initiator address space does not necessarily go away. Typically, the address space is returned to a free pool where it can be used by other service classes or the same service class at a later time. When an initiator rebinds to service class, a new initiator sequence number is assigned to distinguish it from earlier incarnations.

JES3 sampling data

In order for WLM to determine how many initiators need to be started, JES3 must collect sampling information for each service class and pass it to WLM. JES3 also collects sampling information for each report class assigned to a job. Unlike the service class, JES3 does not externalize the report class assigned to the job in any operator commands or messages. From time to time, JES3 examines the job queue looking for jobs that are waiting to be scheduled for main service or have been scheduled for main service but not executing. It then

examines each job to determine whether or not it is eligible to run on at least one system. The following information is then collected about each service class:

On a sysplex level:

- The number of jobs that are eligible to run on at least one system.
- The number of jobs that are ineligible to run on any system because of class limits.
- The number of jobs that are ineligible to run on any system because of other reasons such as operator hold, class/group disabled, resources not available etc.

For each system:

- The number of jobs that are eligible to run on that system
- The number of jobs that are ineligible to run on that system

Sampling information

JES3 collects sampling information for WLM by examining jobs that are in one of the following phases of JES3 processing:

- Jobs waiting to be scheduled for main service - for example, jobs waiting for a class or group to be enabled, jobs waiting because of a duplicate job name condition.
- Jobs in MDS (setup) processing - for example, jobs waiting for their resources to be allocated or for volumes to be mounted.
- Jobs in GMS select

Jobs that are waiting to be scheduled for main service or that are active in MDS processing are considered ineligible to run. That is, they cannot be selected by initiator at this time because they are waiting for something. For example, if the job is in MDS allocation, it must first have its resources allocated before it can be selected by an initiator. If a job is in GMS select, it may or may not be eligible to run for any number of reasons. To determine whether a job on the GMS select queue is eligible to run, the following checks are performed:

- Are there any connected and online systems?
- Is the job in hold?
- Is the job's class and group enabled?
- Are the class limits exceeded?
- Is there available spool space?
- Is the job's scheduling environment available?

Sampling interval

When sampling is complete, the WLM subtask sets up a new sampling interval timer. When the time interval expires, the timer exit will post the WLM FCT for sampling. The sampling interval varies from 2 to 60 seconds depending on whether the sampling data changes from interval to interval. If the data changed from the last interval, the sampling interval is set to 2 seconds, if it is not already set to 2 seconds. If the sampling data did not change for 2 consecutive intervals, the sampling interval is increased by 1 second until it reaches 60 seconds. When the sampling interval is set to a value larger than 2 seconds, this is called slow down mode. Since the sampling information did not change from interval to interval, amount of time between intervals is increased so that less overhead is incurred.

Eligible systems for execution

It doesn't make any sense for WLM to start initiators on systems where no jobs are eligible to run, so the sampling information provided to WLM includes the number of eligible jobs from a system perspective. There are a number of things that affect where a job is eligible to run including system affinity, resource availability, class/group status, and system specific class limits (MDEPTH/MLIMIT).

10.18 Make a job run now

- Make a job run now
  - *F J=xxx,RUN
- Only allowed for jobs in WLM-managed groups
- WLM managed initiator started specifically for job
- Not remembered across a hot start
- Undo the RUN option

Make a job run now

For JES3 managed initiators, the operator, via a series of commands, could cause a job to be selected for execution ahead of other jobs. This is not possible for WLM managed initiators since initiators cannot be started via operator commands and jobs are not selected by priority within a service class.

The new **`*F J=job,RUN`** command allows you to cause a job to be selected for execution ahead of other jobs. This command is only allowed for jobs that are in WLM managed groups. When this command is issued, the best system that is eligible to run the job is selected and a WLM managed initiator is started specifically for the job. This command should be used sparingly since it defeats the purpose of WLM's initiator management algorithms.

If a hot start is performed, the fact that a RUN command was issued for the job is lost. If a WLM managed initiator had been started as a result of the RUN command, it will terminate and another RUN command will have to be issued.

Note: JES3 will allow you to issue multiple RUN commands for a job. Each RUN command will cause an initiator to be started. However, it really won't help the job get selected for execution any faster and will just waste system resources by starting up multiple initiators.

Undo RUN option

A **`*F J=job,H`** command will undo the **`RUN`** command and put the job in hold. The job can then be released from hold and it will be scheduled for execution just like any other job.

10.19 Make job run now considerations

- The following selection criteria are ignored:
  - Operator hold - job will be released from hold
  - Priority hold - priority hold will be ignored for purposes of job select
  - TDEPTH, MDEPTH, TLIMIT, and MLIMIT exceeded
- The following selection criteria are not ignored:
  - Class disabled
  - Group disabled
  - Spool space not available
  - Scheduling environment not available
  - Job is in DJC or spool hold
  - All eligible systems offline/not connected

Run command considerations

When the **RUN** command is issued, JES3 determines which systems are eligible to run the job and passes them to WLM. WLM then selects the best system and starts an initiator on that system. When JES3 selects a job for execution, there are a number of things that JES3 checks before allowing a job to run. When a **RUN** command is issued, some of these selection criteria are ignored. The following are criteria, preventing selection for execution, are ignored:

- Operator hold - The job will be released from hold.
- Priority hold - Priority hold will be ignored for the purposes of selecting the job for execution. Once the job completes execution, priority hold will still be in effect. As a result, the job will not be scheduled for output service until the priority is released from hold
- TDEPTH, TLIMIT, MDEPTH, MLIMIT exceeded

The following selection criteria are not ignored:

- Class disabled - The job will run if the class is enabled on at least one system. This check is not ignored since many customers use classes to control where a job will run.
- Group disabled - The job will run if the group is enabled on at least one system.
- Spool space not available
- Scheduling environment not available - The job will run if the scheduling environment is available on at least one system.
- Job is in DJC or spool hold
- All eligible systems offline/not connected

10.20 Make job run example

- If a job is not eligible to be selected for execution:

```
*F J=25702,RUN
IAT8037 JOB VAINRUN  (JOB25702) RUN REQUEST REJECTED, JOB NOT ELIGIBLE TO RUN
IAT8095 SC64     - MAIN OFFLINE/NOT CONNECTED
IAT8095 SC70     - JOB CLASS DISABLED
IAT8095 SC65     - SCHEDULING ENVIRONMENT NOT AVAILABLE
```

- If a job is eligible to be selected for execution:

```
IAT8033 JOB VAINIRUN   (JOB25703) RUN REQUEST ACCEPTED - SC65
```

Make job run example

If the RUN command is accepted, a special WLM-managed initiator is started to run the job.

If the RUN request is rejected because the job is not eligible to be selected for execution, message IAT8037 is issued and message IAT8095 is issued for each system that the job is allowed to run on. IAT8095 tells why the job is not eligible to execute on that particular system.

```
*F J=25702,RUN
IAT8037 JOB VAINRUN  (JOB25702) RUN REQUEST REJECTED, JOB NOT ELIGIBLE TO RUN
IAT8095 SC64     - MAIN OFFLINE/NOT CONNECTED
IAT8095 SC70     - JOB CLASS DISABLED
IAT8095 SC65     - SCHEDULING ENVIRONMENT NOT AVAILABLE
```

If the RUN request is accepted, message IAT8033 is issued. IAT8033 contains the system name that was chosen for the RUN request and where the WLM managed initiator will be started.

```
IAT8033 JOB VAINIRUN   (JOB25703) RUN REQUEST ACCEPTED - SC65
```

10.21 WLM-managed initiator starting

```
1. IAT6100 (JOB19166) JOB VAINRUN  (JOB25700), PRTY=01, ID=VAINI        NET-ID=*NONE    SUB=JOB25494

2. IAT2012 WLM REGISTRATION SUCCESSFUL FOR SRVCLASS BATCHHI  - SC65
   IAT2012 WLM REGISTRATION SUCCESSFUL FOR SRVCLASS BATCHHI  - SC70

3. *I J=VAINRUN,W
   IAT8674 JOB VAINRUN  (JOB25700) P=01 CL=C       MAIN(GMS  SELECT)
   IAT8685        SC65        - NO INITIATORS STARTED IN SRVCLASS
   IAT8685        SC70        - NO INITIATORS STARTED IN SRVCLASS
   IAT8687     JOB WAITING/ACTIVE 00000 HOURS  00 MINUTES  15 SECONDS
   IAT8699 INQUIRY ON JOB STATUS COMPLETE,       1 JOB  DISPLAYED

4.   IWM034I PROCEDURE INIT STARTED FOR SUBSYSTEM JES3
     APPLICATION ENVIRONMENT SYSBATCH
     PARAMETERS SUB=MSTR
5.   IEF196I        1 //INIT     JOB MSGLEVEL=1
     IEF196I        2 //STARTING EXEC INIT
      : :
     IEF196I        2 IEFC001I PROCEDURE INIT WAS EXPANDED USING SYSTEM
     IEF196I LIBRARY SYS1.IBM.PROCLIB
     IEF196I        3 XXIEFPROC   EXEC   PGM=IEFIIC

6.   IEF403I INIT - STARTED - TIME=17.51.38 - ASID=0051 - SC65
     IEF196I IEF236I ALLOC. FOR INIT INIT
     IEF196I IEF237I DMY  ALLOCATED TO IEFINDMY

7.   IAT2000 JOB VAINRUN  (JOB25700) SELECTED SC65     SRVCLASS=BATCHHI
8.   IEF403I VAINRUN - STARTED - TIME=17.51.38 - ASID=0051 - SC65
```

Starting a WLM-managed initiator

The following shows an example of a WLM managed initiator starting and a job being selected by a WLM managed initiator.

1. The job enters the system.
2. When the job is scheduled for main service, JES3 registers the service class with WLM. The service class must be registered and sampling information must be provided to WLM before any initiators can be started. It is registered on all systems in the JESPLEX the first time a job that references the service class is scheduled for main service. This allows WLM to start initiators on any system where the job can run. These messages appear in SYSLOG only and are for debugging purposes only.

Inquiry command for job

3. Since the job is not running, an **`*I,J=,W`** command is issued for the job to see why it is not running. The output shows that the job is waiting to be selected for execution and there are no initiators started in the service class. Note the difference between JES3 and WLM managed initiators. JES3 initiators select by job class group and WLM initiators select by service class.

Initiator starts

4. WLM recognizes that an initiator must be started and issues IWM034I to indicate that an initiator is being started. WLM does not issue a START command like JES3 to start the initiator, so you won't see a START command. This message appears in SYSLOG only.

5. Since WLM initiators are started under master and not JES3, their JESJCL output appears in SYSLOG. The INIT procedure that is shipped with MVS contains a lot of comment statements which are not shown. In order to avoid cluttering up SYSLOG with this information, we suggest removing most (if not all) of these comment statements.
6. The WLM initiator has successfully started. Unlike JES3 initiators, it doesn't have to go through C/I or main service in JES3 before starting.
7. JES3 returns a batch job to the initiator and the initiator issues message IAT2000 to the operator. For WLM managed initiators, the IAT2000 message contains the service class name, not the job class group.

Job selected

8. The job that was selected actually starts running.

10.22 WLM policy changes

- To define new classification rules or add or delete service classes:
  - Changes made using WLM ISPF dialog
  - Install new WLM service definition using WLM ISPF dialog
  - Activate new service definition/policy using WLM ISPF dialog or VARY WLM,POLICY=command
- When new service definition is activated, WLM deregisters all service classes and SRM reclassifies all jobs in execution
- JES3 must reclassify all jobs prior to execution and extract new service class info for jobs in execution so it can update its control blocks
- Job selection is suspended while reclassification is done

WLM policy changes

The WLM service definition contains the service classes, performance goals, classification rules, scheduling environments and other information that WLM uses to manage the system. In order to add new classification rules or add or delete service classes, you must do the following:

- Make the required changes using the WLM ISPF dialog
- Install a new WLM service definition via the WLM ISPF dialog
- Activate the new WLM service definition/policy via the WLM ISPF dialog or by issuing a **`VARY WLM,POLICY=policy`** command

New service definition

When a new WLM service definition is installed and activated, WLM deregisters all service classes that have been registered by JES3, and SRM reclassifies all jobs that are in execution. This necessary because the classification rules may have changed, and as a result, new service classes may be assigned to jobs. When a policy change occurs, JES3 is notified via an ENF signal that this has occurred. If a new WLM service definition was installed, JES3 reclassifies all jobs that have not been selected for execution, and re-registers all service classes known to JES3. While JES3 is reclassifying jobs, job selection is suspended to prevent a job from being selected for execution with an incorrect service class. Note that if a job was selected for execution at this time, SRM would detect that the WLM classification token is from an earlier WLM service definition and reclassify the job at execution. In addition to reclassifying jobs, JES3 will also extract the new service class information for all jobs that are in execution so it can update the jobs' control blocks to reflect

the current service class that is assigned. Unlike reclassification processing, job selection will not be suspended while this occurs.

Service class recommendations

The following are general recommendations you should follow when defining service classes.

- Service classes should not include jobs in JES-managed and WLM-managed job class group. Doing so weakens the relationship between the number of WLM-managed initiators and the queue delay that is observed. For example, if a velocity goal is used, queue delay is included in the velocity calculations for WLM-managed initiators. If a service class contains jobs in JES-managed and WLM-managed job class groups, queue delay will be included in the velocity calculations for some jobs but not all of the jobs in the service class.
- Service classes should be unique across a JES3 complex. If you run multiple JES3 complexes within a sysplex, keep in mind that WLM policies are sysplex-wide and WLM manages the performance goals for a service class at a sysplex level (for example, performance goals based on the sysplex performance index). If a service class is shared between two JES3 complexes, it is possible for a service class to be meeting its goals from a sysplex, but not meeting its goals in a particular JES3 complex within a sysplex.
- If class limits are specified on the CLASS initialization statement, jobs in that class should map to a single service class. If jobs in the class map to multiple service classes, more initiators may be started than necessary because each service class will be treated independently with respect to the class limits. When JES3 reports to WLM the number of eligible jobs in the service class, it will apply the class limits to each service class individually.

10.23 Defining WLM service classes

- Service Class - How it's made?
  - Name
  - Description
  - Workload
  - Resource Group
  - Performance Period
  - Duration
- Goal types
  - Average Response Time
  - Response Time and Percentile
  - Velocity
  - Discretionary (usually typical goal for batch work)
- Goal importance (1 to 5)
  - Importance
  - CPU Protection

Defining WLM service classes

A service class is a named group of work within a workload with the following similar performance characteristics:

- Performance goals
- Resource requirements
- Business importance to the installation

Workload management manages a service class period as a single entity when allocating resources to meet performance goals. A service class can be associated with only one workload. You can define up to 100 service classes.

You can assign the following kinds of performance goals to service classes: average response time, response time with percentile, velocity, and discretionary. You assign an importance level to the performance goal. Importance indicates how vital it is to the installation that the performance goal be met relative to other goals.

Because some work has variable resource requirements, workload management provides performance periods where you specify a series of varying goals and importances. You can define up to eight performance periods for each service class. You can also assign a service class to a resource group if its CPU service must be either protected or limited.

Parts of a service class

Following are the descriptions that make up a service class definitiion. Some specifications are optional.

Name
: Service class name. Service class names must be unique within a service definition.

Description
: Service class description. An area of 32 characters describing the service class. The descriptive text is available to performance monitors for reporting.

Workload
: The name of the workload associated with this service class. You can associate only one workload per service class in a service definition. A workload is a group of work to be tracked, managed and reported as a unit. Also, a group of service classes.

Resource group
: The name of the resource group associated with the work in this service class. You can assign only one resource group per service class in a service policy. You can override the resource group assigned to a service class in each service policy.

  A resource group is an amount of processor capacity. It is optional. Unless you have some special need to limit or protect processor capacity for a group of work, you should skip defining resource groups and let workload management manage all of the processor resource to meet performance goals. You use a resource group to:

  - Limit the amount of processing capacity available to one or more service classes
  - Set a minimum processing capacity for one or more service classes in the event that the work is not achieving its goals
  - Define a minimum and maximum amount of capacity sysplex-wide, or on system-level

Perf. period
: There is one goal per performance period and a performance goal, importance, and duration for a service class. You set up multiple performance periods for work that has changing performance requirements as work consumes more and more resources. You can specify up to eight performance periods.

Duration
: Number of service units for this performance period. Duration specifies the length of the period in service units. If the work included in this service class period does not complete when the number of service units have been used, the work moves into the next performance period. You do not specify a duration on the last defined period.

Ave. resp. time
: The average response time is for transactions completing within the period in terms of hours, minutes, and seconds. Decimal points are accepted. Response time varies from 15 milliseconds to 24 hours.

  Response time is the expected amount of time required to complete the work submitted under the service class, in milliseconds, seconds, minutes and hours. Specify either an average response time, or response time with a percentile. Percentile is the percentage of work in that period that should complete within the response time. Percentile must be a whole number. You must specify a system response time goal, not "end-to-end". That is, workload management does not control all aspects of system performance, so response time scope is confined to the time workload management has control of the work. This time includes the time the work is using or waiting for CPU, storage, or I/O service.

Note: Workload management does not delay work, or limit it, to achieve the response time goal when extra processing capacity exists.

Percentile resp. A percentile response time is a percentile of work to be completed in the specified amount of time. Percentile boundaries vary from 1 to 99. Amount of time is in hours, minutes, or seconds. Decimal points are accepted. Response time ranges from 15 milliseconds to 24 hours.

Velocity Measure of how fast work should run when ready, without being delayed for WLM-managed resources. Velocity ranges from 1 to 99.

```
Velocity = 100 * using_samples / (using_samples + delay_samples)
```

using_samples

- The number of samples of work using the processor
- The number of samples of work using non-paging DASD I/O resources (in a state of device connect).

delay_samples

- The number of samples of work delayed for the processor
- The number of samples of work delayed for storage
- The number of samples of work delayed for non-paging DASD I/O.

Discretionary Workload management defined goal. Work is run as system resources are available.

Importance The relative importance of the service class goal. Importance is a reflection of how important it is that the service class goal be achieved, Workload management uses importance only when work is not meeting its goal. Importance indicates the order in which work should receive resources when work is not achieving its goal. Importance is required for all goal types except discretionary. Importance applies on a performance period level and you can change importance from period to period. Importance is in five levels: 1 to 5, 1 being the highest importance.

CPU protection Whether long-term CPU protection should be assigned to this service class. By specifying YES in the "CPU Critical" field when defining a service class, you ensure that work of lower importance will always have a lower dispatch priority.

Several options are available to help performance administrators protect critical work. Although applicable to several other subsystem types, CICS and IMS work will particularly benefit from the CPU protection feature:

Long-term storage protection - When you assign long-term storage protection to critical work, WLM restricts storage donations to other work.

Long-term CPU protection - When you assign long-term CPU protection to critical work, you ensure that less important work will generally have a lower dispatch priority. (There are some rare exceptions, such as when other work is promoted because it is holding an enqueue for which there is contention.)

Exemption from transaction response time management - Declares that a specific CICS/IMS region will not be managed to the response times of the CICS/IMS transactions that it processes.

11

JES3 output processing

JES3 output service performs three distinct functions:

- Queueing output

  Normally, output data produced by a job is placed in one of four output service queues when the job terminates. Spin-off data sets are placed in an output queue while the job is still in execution. The three output queues are:

  – Output service writer queue (**Q=WTR**): This queue contains data sets waiting for output processing by JES3 managed devices. Output service automatically selects data sets for processing based on their selection characteristics such as output class, output priority, and output device-related requirements. You can use JES3 commands to place these data sets in operator-hold status. You can also use JES3 commands to modify a data set's selection characteristics or move the data set to the HOLD queue

  – You can also use JES3 commands to Output service hold queue (**Q=HOLD**): This queue contains data sets that are awaiting output processing by other than JES3-managed devices. These data sets must be processed by the function for which they are held (system application printer interface (SAPI) application, external writer or TSO). The function that processes the data set can then change data set characteristics, release it for JES3 processing, or cause JES3 to purge it. If necessary, the operator can force a JES3 writer to process the data set or issue a modify (*F) command to move the data set to the WTR for JES3 device processing. modify a data set's selection characteristics or move the data set to the HOLD queue.

  – MVS/BDT work queue (**Q=BDT**): This queue contains SNA/NJE networking job or networking system output streams. MVS/BDT sends these job or system output streams to the proper node within a SNA/NJE network. You can use JES3 operator commands to hold, release, or cancel networking requests from the queue.

  – TCP/IP work queue (**Q=TCP**): This queue contains TCP/IP networking job or networking system output streams. TCP/IP/NJE sends these job or system output streams to the proper node within a TCP/IP network. You can use JES3 operator commands to hold, release, or cancel networking requests from the queue.

- Scheduling output

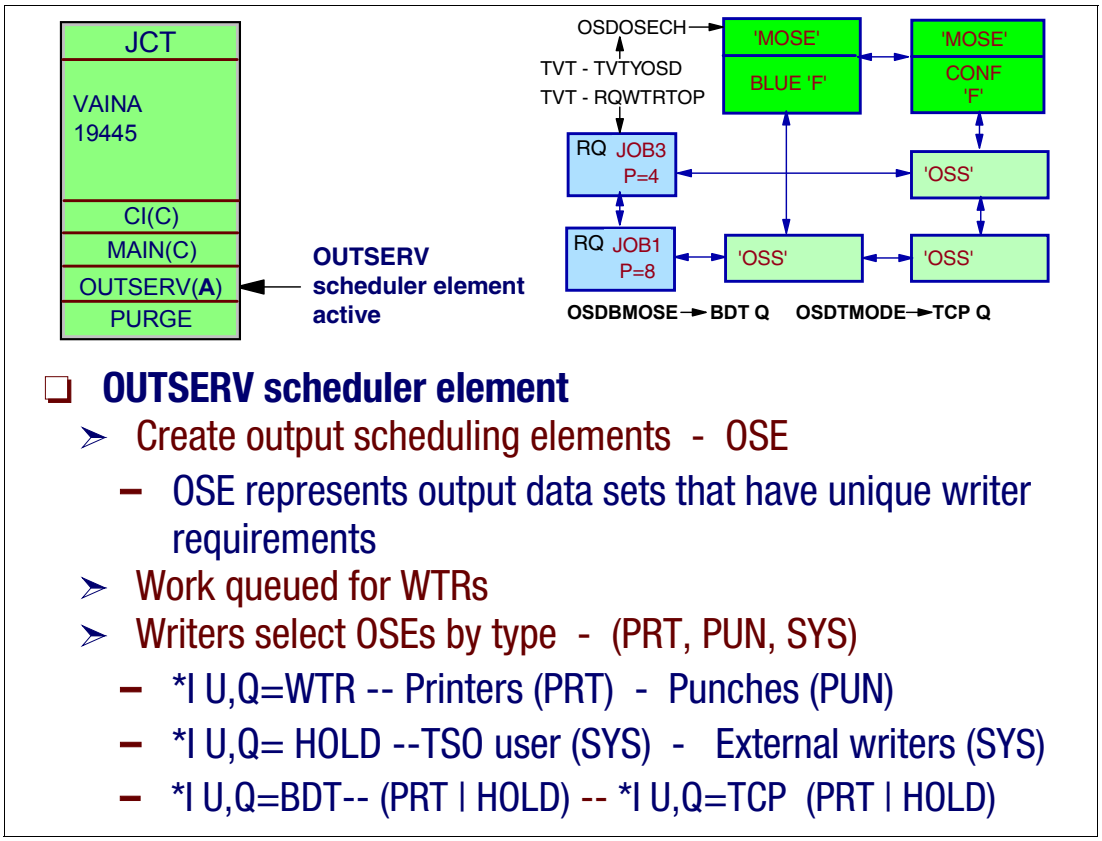
  JES3 output service schedules OSEs to writers in one of two ways:

- An OSE is used to scan the 'writers waiting for work' queue and the available-devices queue to find a device that can process the OSE.
- A set of writer scheduling parameters is used to search the OSEs for the first perfect-fit OSE or for the OSE which best fits the requirements of the writer requesting a job. You can specify these parameters on the DEVICE or OUTSERV initialization statement by coding the WC and WS parameters. The operator can change these parameters when calling, starting, or restarting a writer. The operator does this by specifying the WC and WS parameters on the *X, *S, or *R commands. If two or more OSEs fit the requirements of this writer equally well, JES3 schedules the OSE with the highest JES3 job queue priority. The JES3 job queue priority is based on the job priority specified on the JCL JOB statement.

► Writing output

JES3 writer support consists of a writer driver, writer scheduling (selection) routines, device-dependent routines, command-processing routines (also called message-processing routines) and spool-access routines (for print and punch writers). In most cases, the writer support is provided within the JES3 global address space. Certain devices, however, use device-dependent routines that operate in a separate address space called an output writer functional subsystem (FSS) address space. In this case, the writer driver and the command-processing routines operate in the JES3 global address space and communicate with the output writer FSS using the functional subsystem interface (FSI). The device-dependent routines, also called a functional subsystem application (FSA), and the spool-access routines operate in the output writer FSS.

11.1 Output service processing



Output service scheduled

Output service routines manage SYSOUT data sets generated by jobs during execution by preparing and maintaining a queue of jobs destined for the following destinations:

- Print devices - (PRT)
- Punch devices - (PUN)
- TSO users - (SYS)
- External writers (non-JES3 controlled writers) - (SYS)

OUTSERV scheduler element

OUTSERV, the third scheduler element for a job, is represented by two DSPs responsible for the management of SYSOUT data sets generated by jobs during execution.

Output scheduling element (OSE)

Output service uses a basic work unit, the OSE, to represent all the output service characteristics of the output data sets of a job. Output service characteristics include format, print size, paper requirements, number of copies, and other specifications that tailor the output to the user's needs. Every job is associated with at least one OSE; an OSE represents one or more data sets with similar output requirements. After OSE construction, the OSEs are queued for waiting writers (WTRs).

Types of output

There are four types of output that the JES3 processes. When output data sets are created and are available for output service processing, output service builds an OSE that allows to

the output data set to be processed to its destination. The four types of output data sets that are queued for processing by creating an OSE are as follows:

Q=BDT MVS/BDT work queue: This queue contains SNA/NJE networking job or networking system output streams. MVS/BDT sends these job or system output streams to the proper node within a SNA/NJE network. You can use JES3 operator commands to hold, release, or cancel networking requests from the queue.

Q=WTR Output service writer queue: This queue contains data sets waiting for output processing by JES3 managed devices. Output service automatically selects data sets for processing based on their selection characteristics such as output class, output priority, and output device-related requirements. You can use JES3 commands to place these data sets in operator-hold status. You can also use JES3 commands to modify a data set's selection characteristics or move the data set to the HOLD queue.

Q=HOLD Output service hold queue: This queue contains data sets that are awaiting output processing by other than JES3-managed devices. These data sets must be processed by the function for which they are held (system application printer interface (SAPI) application, external writer or TSO). The function that processes the data set can then change data set characteristics, release it for JES3 processing, or cause JES3 to purge it. If necessary, the operator can force a JES3 writer to process the data set or issue a modify (`*F`) command to move the data set to the WTR for JES3 device processing.

Q=TCP TCP/IP/NJE network jobs: This queue can be either network job streams or network SYSOUT streams. This command might have a length of 126 characters if the command is issued from an input device that permits that command length.

11.2 Processing output data sets

- //SYSPRINT DD SYSOUT=F
  - JCL DD statement requesting print
- OUTPUT JCL statement
  - JCL user control statement requesting processing of a DD
- JES3 //*FORMAT user control statement
  - JES3 user control statement requesting processing of a DD
- Initialization statements for processing output
  - OUTSERV statement - installation defaults for output
  - SYSOUT statement - processing options each output class

SYSOUT data sets

The SYSOUT data set is processed according to the following processing options, in override order:

1. The options specified on this SYSOUT DD statement.
2. The options specified on a referenced OUTPUT JCL statement.
3. The options specified on a referenced on a JES3 //*FORMAT statement.
4. The installation default options for the requested output class via the initialization statements

A SYSOUT data set's processing options can be defined on a DD statement and an OUTPUT JCL statement and a JES3 //*FORMAT statement.

SYSOUT DD statement

The SYSOUT parameter on a DD statement identifies a data set as a system output data set, usually called a SYSOUT data set. The SYSOUT parameter also:

- Assigns this SYSOUT data set to an output class. The processing options of each output class are defined during JES initialization.
- Optionally requests an external writer to process the SYSOUT data set rather than JES. An external writer is an IBM- or installation-written program.
- Optionally identifies the forms on which the data set is to be printed or punched.
- Optionally refers to a JES2 /*OUTPUT statement for processing parameters.

OUTPUT JCL statement

The OUTPUT JCL statement specifies processing options for a SYSOUT data set. These processing options are used only when the OUTPUT JCL statement is explicitly or implicitly referenced by a SYSOUT DD statement. JES3 combines the options from an OUTPUT JCL statement with the options from the referencing DD statement.

OUTPUT JCL statements are useful in processing the output of one SYSOUT data set in several ways. For example, a SYSOUT data set can be sent to a distant site for printing, as shown in statement OUT1, while it is also printed locally, as shown in statement OUT2:

```
//OUT1  OUTPUT  DEST=STLNODE.WMSMITH
//OUT2  OUTPUT  CONTROL=DOUBLE
//DS    DD      SYSOUT=C,OUTPUT=(*.OUT1,*.OUT2)
```

The OUTPUT JCL statement supports over 70 keyword parameters for various processing options specifications. All parameters are optional

Dynamic output - Before a program creates dynamically a SYSOUT data set, the program must; describe the processing options for the SYSOUT data set, and allocate the data set.

dynamic allocation with dynamic output. When you use dynamic output together with dynamic allocation, the options for SYSOUT data set's processing options are similar to the options available through the OUTPUT and DD JCL statements.

The JES3 //*FORMAT statement does not support dynamic definitions.

JES3 //*FORMAT statement

The //*FORMAT PR statement to specifies JES3 processing instructions for SYSOUT data sets that are printed. These instructions permit special processing of SYSOUT data sets, such as:

- Multiple destinations.
- Multiple copies of output with different attributes.
- Forced single or double space control.
- Printer overflow checking.

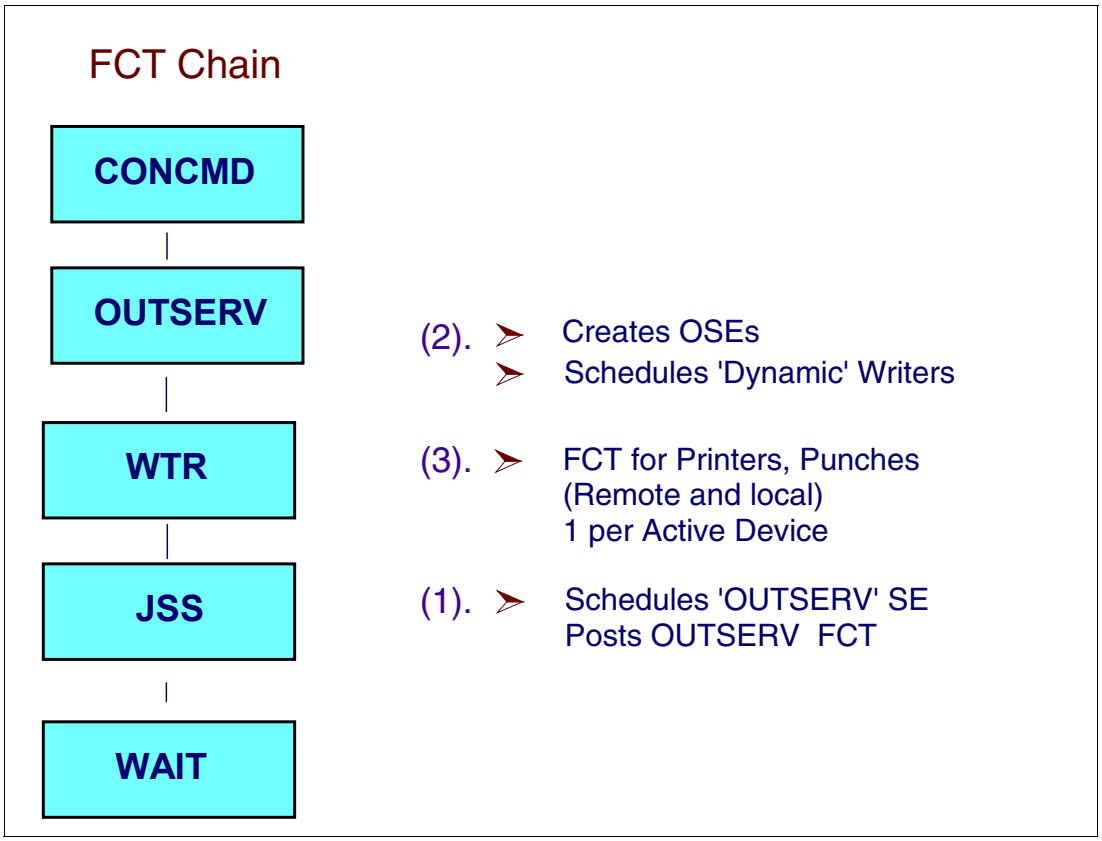
//*FORMAT PR statements can be either specific or nonspecific. A specific //*FORMAT PR statement contains a DDNAME parameter that specifies something other than a null value, such as DDNAME=ddname or DDNAME=JESYSMSG. A nonspecific //*FORMAT PR statement contains DDNAME=, with no value (null) specified for the DDNAME parameter.

You can code multiple specific //*FORMAT PR statements for a particular SYSOUT data set to specify special requirements for different copies of the data set. In addition, you can code a //*FORMAT PU statement for the same SYSOUT data set, thereby both printing and punching it.

You can also code multiple nonspecific //*FORMAT PR statements. In this case, the system produces only one copy of each data set, combining any parameter values specified on the statements.

The //*FORMAT PR statement supports less than 20 keyword parameters. All parameters are optional

11.3 Output service processing



Output service processing

The OUTSERV DSP and WTR DSPs are the DSPs responsible for the management of SYSOUT data sets generated by jobs during execution.

JSS When JSS schedules the OUTSERV scheduler element for a job, the OUTSERV FCT is posted and the resqueue entry is placed on a RQ chain for OSE construction.

OUTSERV The OUSERV DSP (IATOSDR - Output Service Driver) first performs preliminary work necessary to schedule the writing of data sets to the output devices. This work consists of generating OSEs for work to be done by output service writer routines. So, the OUTSERV DSP differs from many DSPs in that it schedules work to other DSPs. The OUTSERV FCT may also schedule a dynamic writer to process the newly created OSE.

WTR A WTR DSP (IATOSWD - Writer Driver Module or IATOSFD - FSS Writer Driver Module) handles printing and punching of output. The WTR DSP reads from spool and writes to the specific output device, thereby performing device dependent functions. Output can be sent to a variety of devices. For locally attached device that are driven by JES3 the global address space, the WTR DSP writes directly to the device. For non-locally attached devices that are not driven by the JES3 global address space, the WTR DSP passes output to the appropriate interface. This can be RJP, SNARJP, NJE services, the internal writer, or a functional subsystem (FSS).

11.4 Output service initialization statements

- **OUTSERV** and **SYSOUT** statements for OSE construction
  - Output service installation standards and default values
  - SYSOUT class definitions

OUTSERV,CARRIAGE =6,CB=N
,CDSTOCK=5081,CHARS = GS10
,FLASH = (NONE,255),FORMS = 1PRT
,MODIFY = (NONE,0),OUTLIM=16777215
,OUTSVFCT=1,STACKER = C
,TRAIN = PN,THRESHLD=-1
,WS = (D,T,F,C,U,FL,CM,SS),NPRO=90
,WC = c,.....,
,SNAGROUP=NO,EXTOSENUM=YES

Initialization statements

The OUTSERV and SYSOUT statements are used to define output service characteristics. The OUTSERV initialization statement specifies default values and standards for the output scheduling element (OSE) to be used on output devices; for example: printers, punches, or RJP (remote job processing). These defaults apply to every built OSE, regardless of the device that handles the output, provided other overrides do not take effect.

Characteristics assigned to writers are specified in the JES3 initialization statements. These characteristics may be changed by the operator. For example, if a device is active, a RESTART command could change the characteristics. Normally, characteristics of dynamic writer are changed when JES3 is restarting. An operator may allow a hot writer to assume some or all of the installation-default set of characteristics or the operator may specify an override set, thereby having total control over hot writers.

The OUTSERV initialization statement specifies default values and standards for the output scheduling element (OSE) to be used on output devices; for example: printers, punches, or RJP (remote job processing) These defaults apply to every built OSE, regardless of the device that handles the output, provided other overrides do not take effect. The following are the defaults for the OUTSERV statement:

```
OUTSERV,CARRIAGE=6,CB=N,CDSTOCK=5081,CHARS=GS10,FLASH=(NONE,255),FORMS=1PRT,
MODIFY=(NONE,0),OUTLIM=16777215,OUTSVFCT=1,STACKER=C,TRAIN=PN,THRESHLD=-1,
WS=(D,T,F,C,U,FL,CM,SS,PM),NPRO=90,SNAGROUP=NO,EXTOSENUM=YES
```

See *z/OS JES3 Initialization and Tuning Reference*, SA22-7550 for detailed parameter descriptions.

11.5 SYSOUT initialization statement

```
SYSOUT,CLASS= class | STANDARD,CARR= name  | STANDARD
        ,CHARS= id1,...,CONTROL= PROGRAM | SINGLE | DOUBLE
        ,COPIES= (num,n,...),DEST= destination
        ,FLASH= ( id,cnt ) | STANDARD,FORMS= name | STANDARD
         ,HOLD= (TSO | EXTWTR | 3540),INT=(YES | NO)
         ,MODIFY= (name,trc) | STANDARD | (NONE)
         ,PRTY= nn,OVFL= ON | OFF,PRTY=nn,SPART=partitionname
         ,STACKER= C | S   | STANDARD,TRAIN= name
         ,TYPE= PRINT | PUNCH | RSVD | DSISO | USER1 | USER2
         ,THRESHLD= 1 | STANDARD | limit,TRUNC= YES | NO
         ,TRKGRPS= (1,2 | prigrps | prigrps,secgrps)
         ,COMPACT= comptab,CHNSIZE= nnn | DS | (nnn,mmm)
```

SYSOUT initialization statement

The SYSOUT initialization statement defines SYSOUT class characteristics. The SYSOUT statement is required for each JES3 output class that requires other than TYPE=PRINT processing (JES3 initially set all SYSOUT classes to TYPE=PRINT).

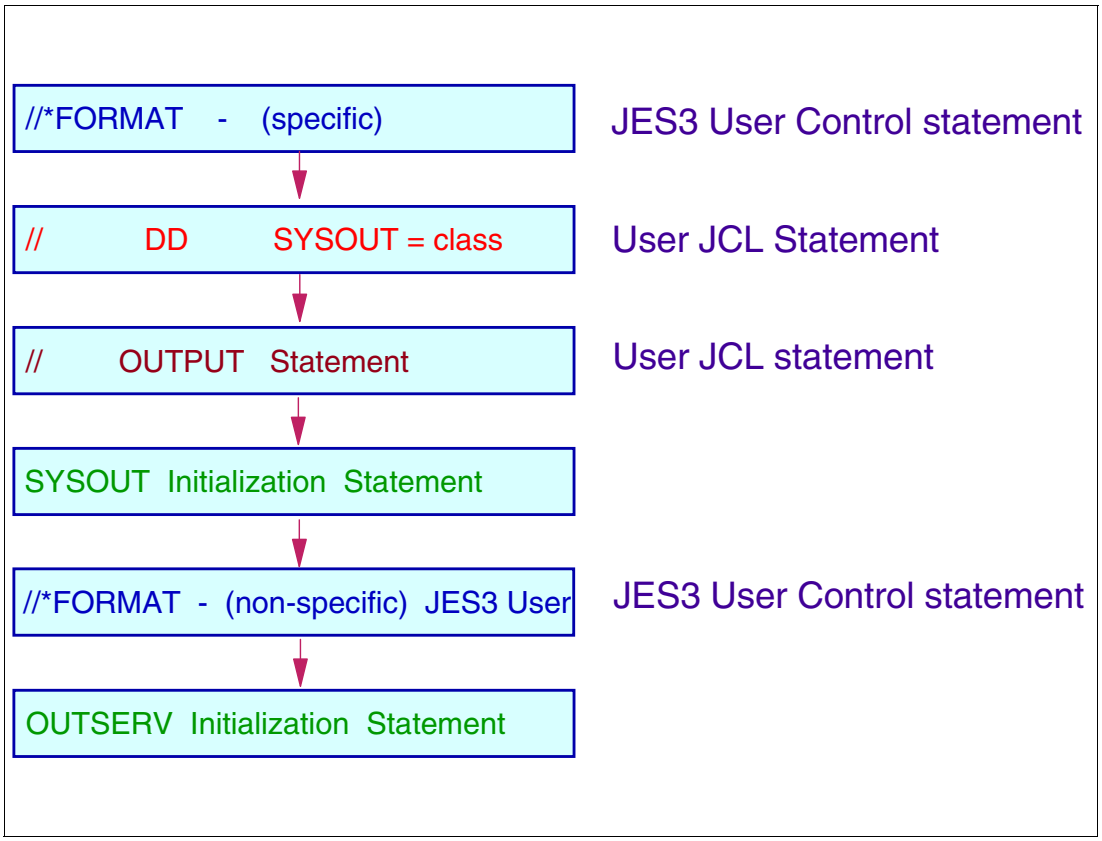
The SYSOUT statement parameters are applicable only to JES3 processing.

Also, be aware that if the SYSOUT is associated with an output descriptor that is defined by the OUTPUT JCL statement or TSO OUTDES command, then the output characteristics are merged for SYSOUT on the HOLD queue.

Note: The CARRIAGE, CHARS, FLASH, FORMS, MODIFY, STACKER, TRAIN and THRESHLD parameters on this SYSOUT statement override corresponding values on the OUTSERV and DEVICE statements.

See *z/OS JES3 Initialization and Tuning Reference*, SA22-7550 for detailed parameter descriptions.

11.6 SYSOUT order of overrides



Order of overrides for OSE construction

To build the OSE that contains the composite of all the specifications of all the statements, IATOSDO uses as many as five control blocks in a specific order. The visual shows the merge order used to create OSEs:

1. The output service resident data area (OSD), which contains information from the OUTSERV initialization statement.
2. A non-specific FORMAT statement overrides the OUTSERV initialization statement.
3. The SYSOUT class table (SCT), which contains information from the SYSOUT initialization statement.
4. The in-storage job data set output control block (JDO), which contains information from the spooled OUTPUT JCL statement.
5. The job data set control block (JDS), which contains information from the SYSOUT DD JCL statement.
6. A specific FORMAT statement overrides all the other statements. initialization statement.

Note: If a DD has both a direct FORMAT statement and an OUTPUT statement, both are used to create separate OSEs.

Override processing

The following examples illustrates the visual override processing. The initialization statements would exist in the installation's initialization stream. The JCL statements are those that a programmer might use for his job.

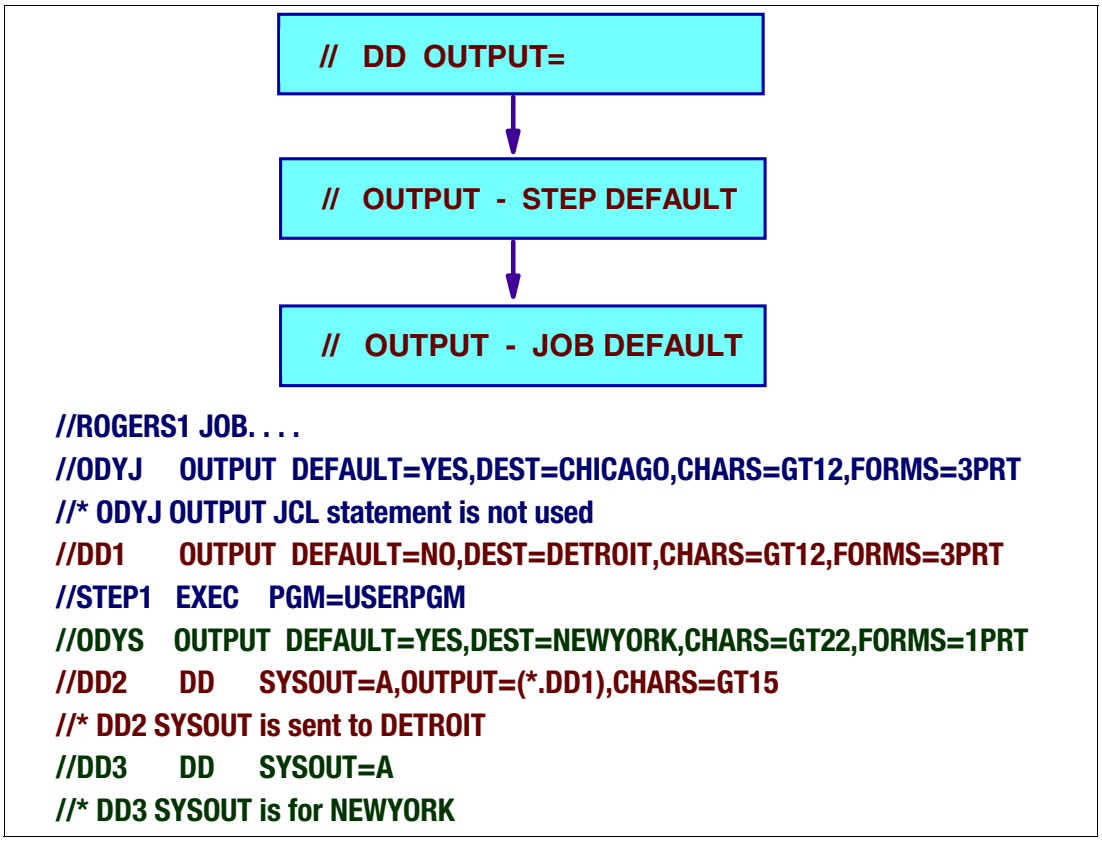
Initialization statements:

```
OUTSERV   FORMS=1PRT,CHARS=GT10          - (OUTSERV statement)
SYSOUT CLASS=F,FORMS=2PRT            - (SYSOUT statement)
```

JCL Job Statements

```
//D75DMZ1  JOB    . . .
//*FORMAT         PR,DDNAME=,DEST=POK     - (non-specific statement)
//*FORMAT         PR,DDNAME=DD2,DEST=CHICAGO   - (specific statement)
//STEP1    EXEC   PGM=USERPGM
//DD2      DD     SYSOUT=(F,,3PRT),CHARS=GT15,DEST=NEWYORK       - (DD statement)
```

11.7 OUTPUT JCL order of override



OUTPUT statement overrides

The OUTPUT statement can be specified in two ways:

- Either directly or
- By default (through the use of "DEFAULT=YES" on the OUTPUT statement)

  The default OUTPUT statement parameters apply to all SYSOUT DD statements and there can be a step default and a job default.

The direct OUTPUT statement parameters apply only to the SYSOUT DD statement that references it. Module IATIIOS processes the OUTPUT statement and saves the parameters specified on the OUTPUT statement in the job data set control block (JDS). Module IATSIOD processes the dynamic OUTPUT statements. When building OSEs, module IATOSDO merges any values specified on the OUTPUT statement, found in the JDS, into the OUTSERV work OSE.

OUTPUT statement example

The visual shows the order of overrides if multiple OUTPUT statements are used in the same job, where **//*** statements are JES3 comment statements when specified in JCL.

JCL Job Statements with OUTPUT statements

```
/ROGERS1 JOB. . . .
//ODYJ     OUTPUT  DEFAULT=YES,DEST=CHICAGO,CHARS=GT12,FORMS=3PRT
//* ODYJ OUTPUT JCL statement is not used
//DD1       OUTPUT  DEFAULT=NO,DEST=DETROIT,CHARS=GT12,FORMS=3PRT
```

```
//STEP1   EXEC    PGM=USERPGM
//ODYS    OUTPUT  DEFAULT=YES,DEST=NEWYORK,CHARS=GT22,FORMS=1PRT
//DD2       DD      SYSOUT=A,OUTPUT=(*.DD1),CHARS=GT15
//* DD2 SYSOUT is sent to DETROIT
//DD3       DD      SYSOUT=A
//* DD3 SYSOUT is for NEWYORK
```

OUTPUT statement merge order

The merge order used with a default OUTPUT statement in the OSE merge process, JES3 incorporates the control block information in the order listed below:

- First - the OSD, which contains information from the OUTSERV initialization statement.
- Second - the SCT, which contains information from the SYSOUT statement.
- Third - the JDO, which contains information from the default OUTPUT JCL statement.

Note: More than one OUTPUT default can be used for a DD statement. The output default is not cumulative like the default for

A default OUTPUT specification can also be used as a direct reference.

Default OUTPUT specifications have no effect on system data sets (JESMSG, JESJCL, and SYSMSG). The JESDS= keyword must be used to have OUTPUT specifications applicable to system data sets.

11.8 Sample job for OSE construction

❑ Sample Job

```
//VAINIFO  JOB (999,POK),EXPERT,MSGLEVEL=1,MSGCLASS=A
//*FORMAT PR,DDNAME=,FORMS=SPECIAL
//*FORMAT PR,DDNAME=STEP1.SYSUT2,FORMS=CONF
//*FORMAT PR,DDNAME=STEP2.SYSUT2,FORMS=SPEC
//STEP1    EXEC PGM=IEBGENER
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//SYSUT2   DD SYSOUT=F
//SYSUT1   DD *
  STEP1
//STEP2    EXEC PGM=IEBGENER
//OUTNJE OUTPUT DEST=WTSCPLX9
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//SYSUT2   DD SYSOUT=(F,,BLUE),OUTPUT=(*.OUTNJE)
//SYSUT1   DD *
  STEP2
OUTSERV,........,FORMS=1PRT,WC=(F,A,D,E,S),..
SYSOUT,CLASS=A,.........,FORMS=1PRT
SYSOUT,CLASS=F,.........,FORMS=GRAY
```

OSE construction

This visual describes a sample job that will be used to show OSE construction for the output data sets for the job:

- JESMSGLG
- JESJCL
- SYSMSGLG
- SYSUT2 - step 1
- SYSUT2 - step 2

//*FORMAT statement

The FORMAT statement can be specified in two ways--either directly or by default (through the use of "DDNAME=," on the FORMAT statement). The default FORMAT statement parameters apply to all SYSOUT DD statements for the job. The direct FORMAT statement parameters additionally apply to the specified SYSOUT DD statement. Input service module IATISFR processes the FORMAT statement and saves its parameters in the format parameter buffer (FRP). When building OSEs, module IATOSDO (OUTPUT Service Driver OSE Management) merges any value specified on a FORMAT statement, found in the FRP, into the work OSE. Both types of FORMAT statements are shown in the example.

SYSOUT DD statements

For SYSOUT DD statements parameters are saved in the job data set control block (JDS). When building OSEs, module IATOSDO merges any values specified on the SYSOUT DD statement, found in the JDS into the OUTSERV work OSE. There are two DD statements

specifying the SYSOUT keyword in the example, both for SYSUT2 but in different steps. The OUTSERV and SYSOUT statements that apply to this job are shown in the visual.

The STEP2 SYSUT2 DD statement specifies also the OUTPUT parameter to associate the SYSOUT data set explicitly with an OUTPUT JCL statement. JES3 processes this SYSOUT data set using the options from this DD statement combined with the options from the referenced OUTPUT JCL statement and the //*FORMAT PR JECL statement.

OUTPUT JCL statements

If a job's JCL has OUTPUT JCL statement references, module IATIICM (CI Compatibility Module) will use scheduler JCL facility (SJF) services to retrieve DD related information from the job's scheduler work blocks (SWB). When an OUTPUT SWB is found, module IATIIOS (OUTPUT SWB Spooling Routine) is invoked to process the SWB. The IATIIOS processing creates output JDS entries using data from OUTPUT SWBs, updates each OUTPUT SWB with the relative OUTPUT statement number and spools the SWB.

When the sample job is executed, the following processing options will be assigned tor the SYSOUT data sets:

DD Stmt		Destination	Forms	Class
JESMSGLG		ANYLOCAL	SPECIAL	A
JESJCL		ANYLOCAL	SPECIAL	A
JESYSMSG		ANYLOCAL	SPECIAL	A
SYSUT2	STEP1	ANYLOCAL	CONF	F
SYSUT2	STEP2	ANYLOCAL	SPEC	F
SYSUT2	STEP2	WTSCPLX9	BLUE	F

JES3 processing of SYSOUT data sets

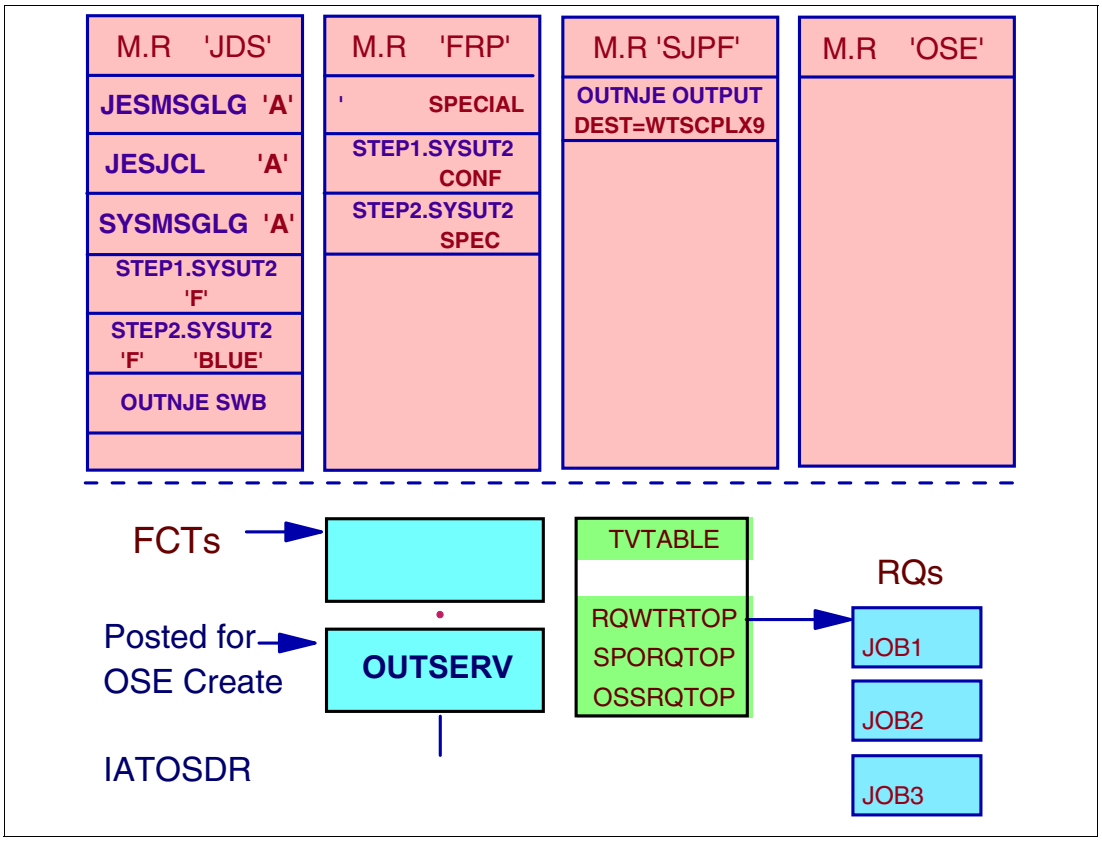
SYSOUT data set output information (DOI) is collected and spooled during job execution. The DOI (IATYDOI) maps the user-specific output descriptors for SYSOUT data sets for use during output service.

Subsystem interface module IATSIAD (JES3 SSI Allocation/Unallocation Routines) calls, during allocation of SYSOUT data sets, module IATSIODA (OUTPUT SWB Processing Routines) to build an output reference table that contains all references by this data set to an output descriptor (i.e. JCL OUTPUT statement).

At open time SSI module IATSIOR(JES3 Open and Restart SSI Routines), when the JDS entry is built, calls module IATSIOD to build an OUTPUT Statement Number Table if the SYSOUT data set directly references ONLY static output descriptors. This table will be added to the end of the JDS ENTRY being built.

IATSIOR, at open time, calls module IATSIOD to build a JDS OUTPUT entry into a DOI for OUTPUT references that have been dynamically created before allocation time. IATSIOD also spools the OUTPUT descriptors using a USAM access method.

11.9 Control blocks at OSE create



OSE construction

The output scheduling element (OSE) is the heart of the control data base. It summarizes the scheduling characteristics for SYSOUT data sets for a job, each OSE representing data sets within a job with similar output characteristics, requirements, and scheduling characteristics.

The job's control blocks are shown as follows:

JDS The JDS contains the output data sets that are to be printed. These data sets will have their OSEs created under the OUTSERV FCT.

FRP The FRP control block contains the information from the //*FORMAT statements. The first entry is for the statement that applies to all data sets and the second entry is a specific statement that applies to the SYSPRINT data set.

OSE The OSE control block is empty at this point of processing. When the OSEs are created, they will be moved from a work area into the OSE control block and written to the spool.

SJPF The SJPF data set contains the SWB for OUTNJE OUTPUT JCL statement.

Output service elements (OSEs)

The output service elements (OSE) is used to hold processing options for SYSOUT data sets. Each OSE representing data sets within a job with similar output characteristics, requirements, scheduling characteristics, and so forth.) An OSE is divided into three sections.

1. A **fixed section** resides at the beginning of every disk and writer wo OSE and forms that OSE's header. This fixed header section contains information about this particular buffer and a pointer to the next disk OSE.
2. A **variable section** contains the common output characteristics describing how data sets should be printed and/or punched and describes OSE status. There may be more than one variable section in a disk OSE. A variable section is associated with one or more corresponding data set sections.
3. A **data set section** exists for every print and/or punch instance of a SYSOUT data set. This section contains information unique to the print/punch requirements of the output data set and its present status, the pointer to the actual spool data set, and JDS entry.

OUTSERV scheduling

Module IATOSDR is dispatched by the multifunction monitor when the TVT field, OSEFLAGS, is posted for the following types of processing:

- Normal output and the RQ placed on the OSSRQTOP chain
- Spin-off output and the RQ placed on the SPORQTOP chain

After OSE construction for a job, the job's RQ is placed on the RQWTRTOP chain waiting for a writer (WTR) DSP to select it.

The TVT points to the chain of jobs that required processing as follows:

RQWTRTOP Points to the first RQ of a chain of RQs that requires write processing.

SPORQTOP Points to the first RQ of a chain of RQs that requires OSE build for spin-off processing.

OSSRQTOP Points to the first RQ of a chain of RQs that requires OSE build for normal processing.

Within each RQ on the above chains, the following chain fields are used to chain the RQ entries on each chain:

```
RQWTRCHN    points to the next RQ that requires writer processing.
RQSPNCH     points to the next RQ that requires spin-off processing.
RQGRPCHN    points to the next RQ that requires output service processing
```

11.10 OSE construction steps

- ❑ Select output data set from JDS
- ❑ Create OSE for data set in 'Override' order
- ❑ Enter user exit IATUX72 - if RC 8 or a dummy IATUX72 exit
  - ➢ IATUX19 - Examine/modify temporary OSE
- ❑ Compare OSE with 'others' in Job
- ❑ Place data set entry with OSE
- ❑ Next JDS entry ?

OSE construction

JES3 builds OSEs, which represent output scheduling characteristics, processing options, of a job's SYSOUT data sets by using initialization statement defaults, //*FORMAT JECL statements, or MVS JCL DD and OUTPUT JCL statements. The same characteristics can be specified on several statements. Characteristics specified on one statement can be overridden by statements processed later. Conversely, if no specification is made for a particular required characteristic, one of the statements used provides a default value.

OSE construction follows a specific pattern that merges the parameters specified on the three types of statements mentioned above. Module IATOSDR calls module IATOSDO to build OSEs in two different hierarchical structures depending on whether an OUTPUT or FORMAT statement is specified for the SYSOUT data set. If neither statement is specified, IATOSDR uses the same hierarchy as that built for the FORMAT statement.

OSE construction steps

The OSE construction routines work as follows:

1. A JDS entry is selected.
2. In order to construct the data set characteristics, an OSE is built in a work areas to contain a composite of all the various processing options specifications on initialization, //*FORMAT JECL, and MVS DD and OUTPUT JCL statements.
3. User exit IATUX72 is given a chance to modify the OSE under construction. If IATUX72 is a dummy exit, or if it indicates via its return code that IATUX19 should be called, then user IATUX19 is called. Otherwise the call to IATUX19 is skipped.

When exit IATUX19 is entered you can override the data set characteristics in the OSE.

4. This work OSE is then compared with all previous OSEs created for this job
5. If there is a match, a data set entry is created under the OSE that matches. If there is no match, a data set entry is created under this new OSE for the job.
6. Loop to step 1 if more JDS entry exists.

User exit IATUX19

When an OSE is initially constructed, or when it is rebuilt, either IATUX19 or IATUX72 can be used to examine or modify it. When an OSE is moved to the writer queue without being rebuilt, only IATUX72 can be used to examine or modify it.

Note: Installations that do not need to process an existing OSE that is moved to the writer queue can use the IBM-supplied IATUX72, which is a dummy exit. IATUX19 will then be used to process OSEs when they are built or rebuilt.

Installations that want to use an existing IATUX19 to process OSEs, but also need to process output that is moved to the writer queue, can code IATUX72 so that it uses return code 8 when called under the OUTSERV FCT. This directs the output service driver to call IATUX19.

Installations that want to use a single exit to process OSEs both when they are built/rebuilt and when they are moved to the writer queue can use IATUX72 for that purpose. IATUX72 should use return code 0 or 4 so that IATUX19 is not called by the output service driver.

User exit IATUX72

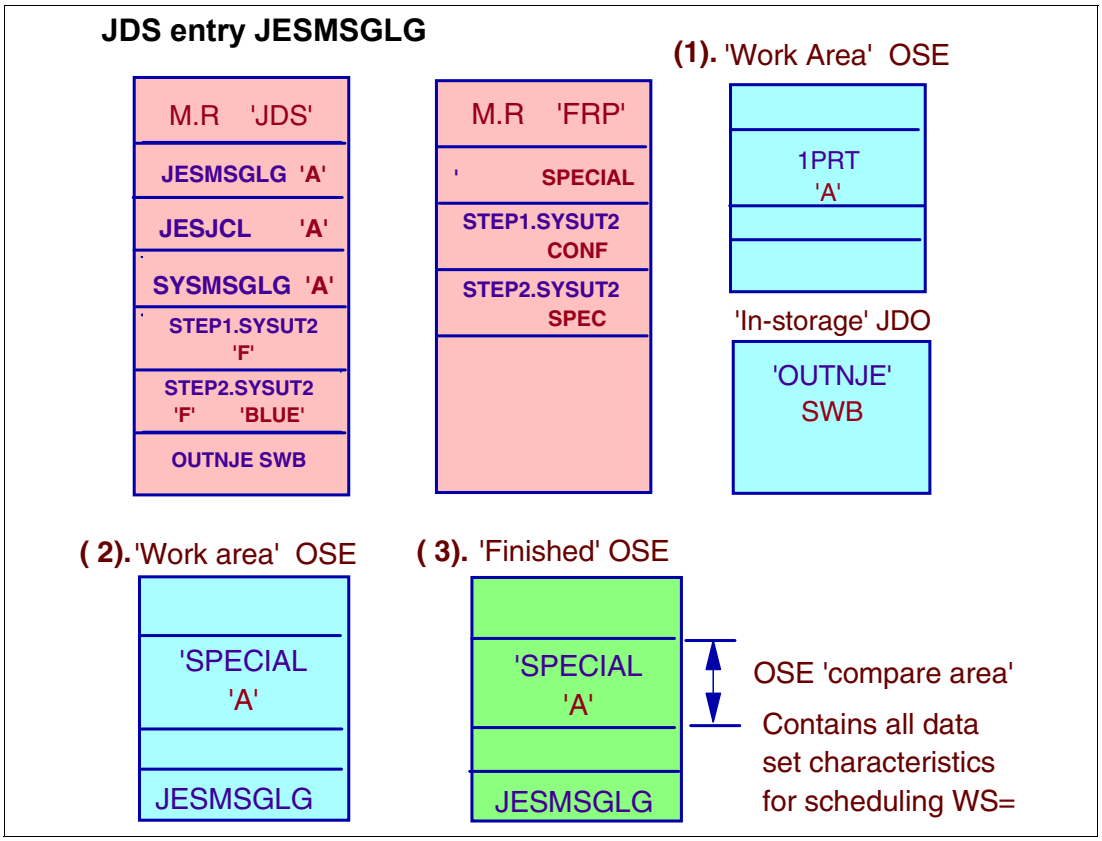
Exit IATUX72 (Examine/Modify a Temporary OSE or an OSE Moved to Writer Queue) allows to change the contents of the output scheduling element (OSE) being constructed for a SYSOUT data set.

When called from the output service OSE management module IATOSDO, the OSE passed as input is a temporary OSE. IATUX72 may process the OSE or indicate that exit IATUX19 is to be called to process the OSE before it is written to spool.

When called from the output service modify implementation (module IATMOOI), SYSOUT API (SAPI) processor (module IATOSSO), or processor SYSOUT (PSO) scheduler (module IATOSPC), the OSE is an existing OSE that is being moved from the hold queue to the writer queue.Exit IATUX72 has the opportunity of examining and, if desired, changing the information before the OSE is rewritten to spool.

Note: When an OSE is initially constructed, or when it is rebuilt, either IATUX19 or IATUX72 can be used to examine or modify it. When an OSE is moved to the writer queue without being rebuilt, only IATUX72 can be used to examine or modify it.

11.11 OSE construction - 1/3



OSE construction

Using the JCL specified in “Sample job for OSE construction” on page 546, this visual shows the construction of the OSE for the first JDS entry JESMSGLG as follows:

1. An OSE is constructed in the work area and the data set characteristics are moved into the "compare area".

   IATOSDO processes default OUTPUT and FORMAT statements. Separate “Work” OSE default templates are created for each of these statements. All OUTPUT related statements are taken out of the JDS and put on a separate chain in order that output values can be merged into the “Work” OSEs at a later point.

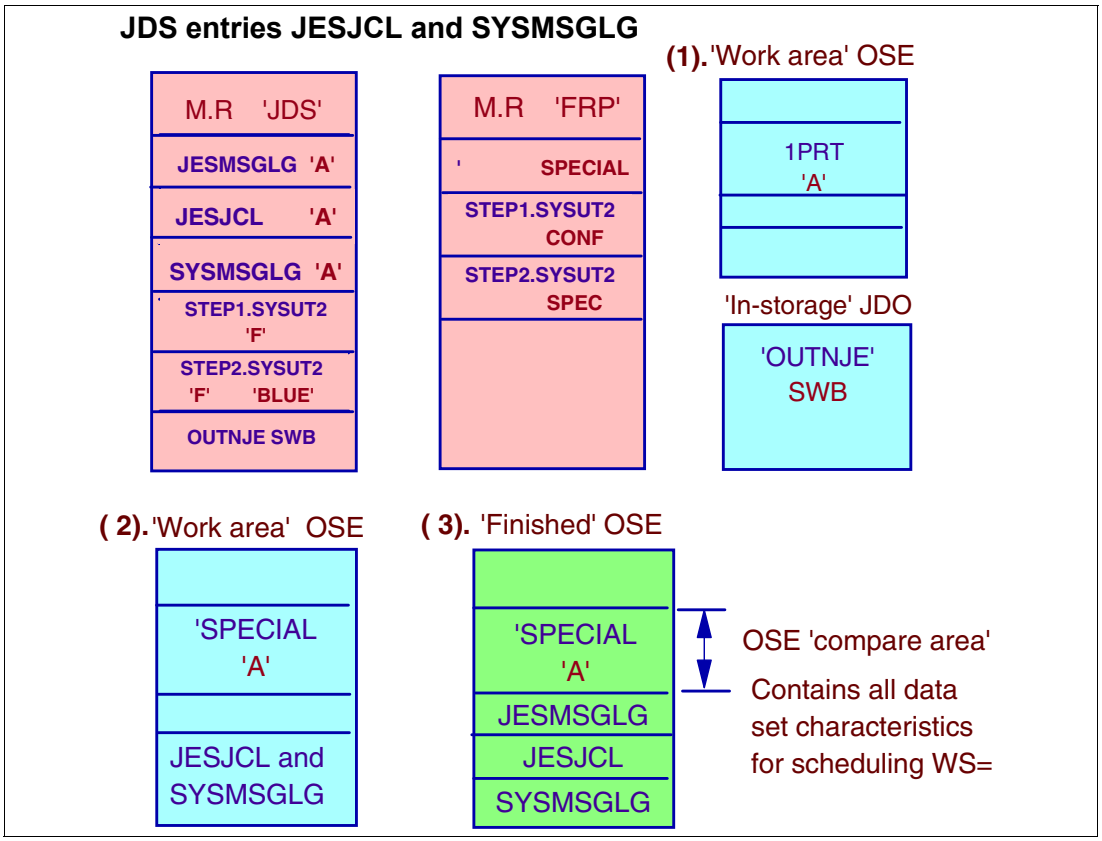
   For FORMAT processing, Each data set for the current RQ is checked to find out if any FRPs exist for it. If any FRPs for the data set are found, the JDS is set to indicate the type of FRP found. As default OSEs are created for each type of data processed by the OUTSERV writers and as default override FRPs are found, the OSEs are changed according to the FRPs.

2. The non-specific FORMAT override is merged into the OSE under construction.

   There are no previous matching OSEs, so the data set entry is added to this new OSE.

3. The work area OSE is moved to a "finished" OSE area.

11.12 OSE Construction - 2/3



OSE construction

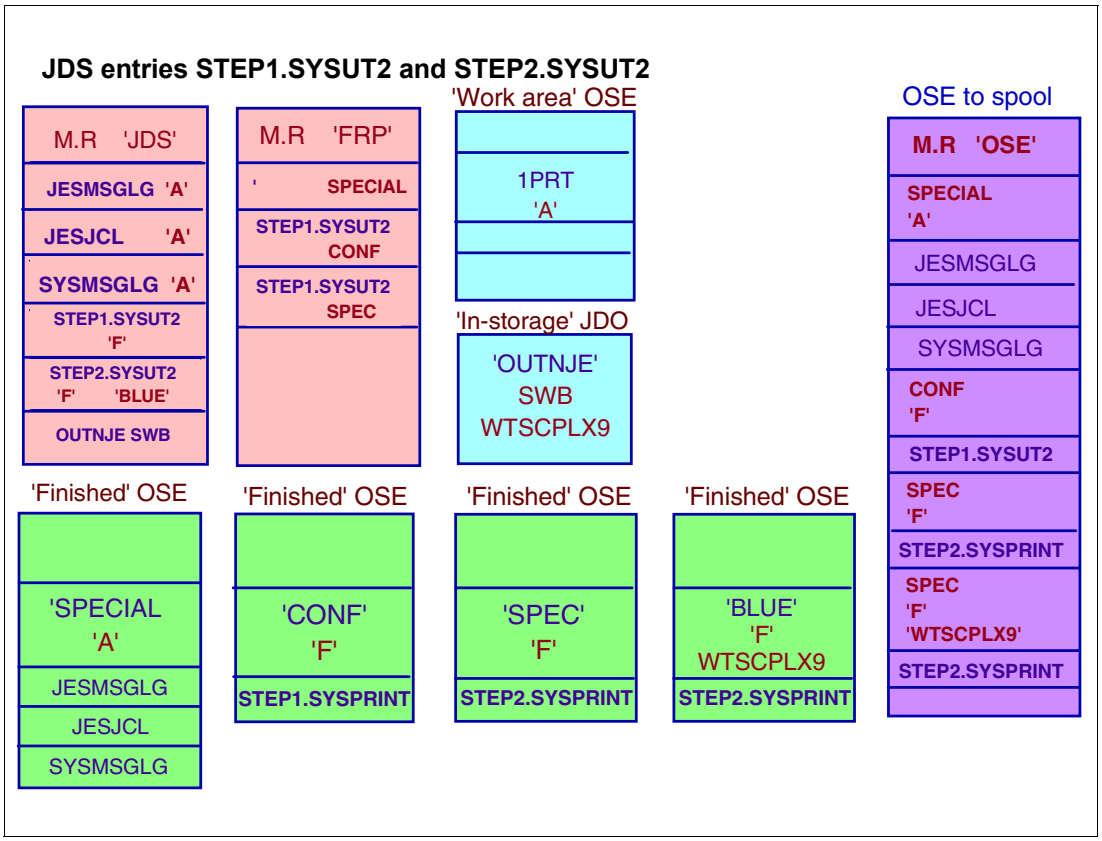
Since output service uses a basic work unit, the output service element (OSE), to represent all the output service characteristics of the output data sets of a job. Output service characteristics include format, print size, paper requirements, number of copies, and other specifications that tailor the output to the user's needs. Every job is associated with at least one OSE; an OSE represents one or more data sets with similar output requirements. OSEs change during output processing and exist during the processing of the OUTSERV scheduler element in the following forms:

- An OUTSERV work OSE (the OSE as it is initially created during OUTSERV DSP processing)
- A DOSE (the OSE as it resides on disk)
- A writer work OSE (the writer's working copy of a disk OSE)

Sample job OSEs

This visual shows the construction of OSEs for the second and third JDS data set entries; JESJCL and SYSMSGLG. Since the output processing options are the same, by comparing the “compare area” as shown in the visual, of any previous OSEs, for these two data set and they match the already existing OSE “compare area” for the JESMSGLG data set, the data set entries for these two are added to the existing matching OSE.

11.13 OSE construction - 3/3



Final OSEs

The data set entries for STEP1.SYSUT2 and STEP2.SYSUT2, are processed through OSE construction and they do not match any previous OSEs, by using the “compare area” of previous OSEs for this job and the "finished" OSEs are separate from each of the other OSEs for the job.

The second STEP2.SYSUT2 “Finished” OSE is created using the OUTPUT JCL statement information in JDO. Therefore, the job has the final following OSEs for the following output data sets:

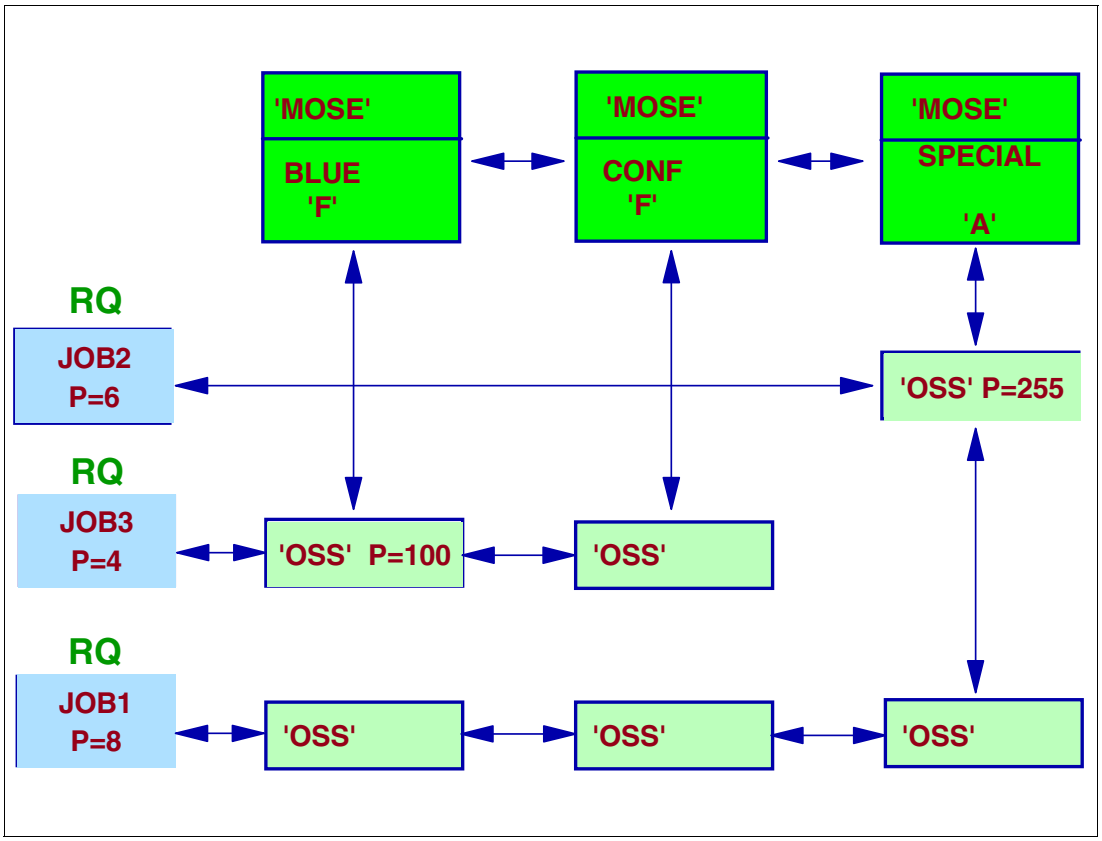
STEP1.SYSPRINT This output data set is from DD SYSUT2 in step 1.

STEP2.SYSPRINT This output data set is from DD SYSUT2 in step 2.

STEP2.SYSPRINT This output data set is from DD SYSUT2 in step 2 but this copy is sent to an NJE node as specified in the OUTPUT statement reference on the DD statement, (OUTPUT= (*.OUTNJE)) in the JCL which references a step OUTPUT statement in STEP2 which specifies DEST=WTSCPLX9.

When all the OSEs are constructed, they are moved into a spool buffer and written to spool, as shown in the visual.

11.14 In-storage MOSE/OSS chains



RQ/MOSE/OSS structure

The MOSE is a in-storage control block that summarizes all scheduling characteristics of OSEs with identical characteristics. The RQ/MOSE/OSS structure shown in the visual describes output service processing by illustrating the connection of the TVT/RQ structure, the MOSE/OSD structure, and the OSS structure.

Output service uses the MOSE structure to improve scheduling performance. The OSD (mapped by IATYOSD, the output service driver data area) contains a pointer to the MOSE control structure. The output service scheduling element (OSS) is a control element that specifies availability of OSEs and which jobs are associated with a MOSE. The MOSE and the job's resqueue both contain pointers to the OSS. The OSS contains a pointer to its MOSE and may contain pointers to additional OSS control blocks. A count of available OSEs in the OSS permits the scheduling of the job output.

JES3 maintains also OSE information, for performance reasons, in a data space (JES3OST) resident OST (OSE Summary Table) control blocks.

In-storage control blocks for scheduling

The MOSE is a in-storage control block that summarizes all scheduling characteristics of OSEs with identical characteristics. The RQ/MOSE/OSS structure shown in the visual describes output service processing by illustrating the connection of the TVT/RQ structure, the MOSE/OSD structure, and the OSS structure.

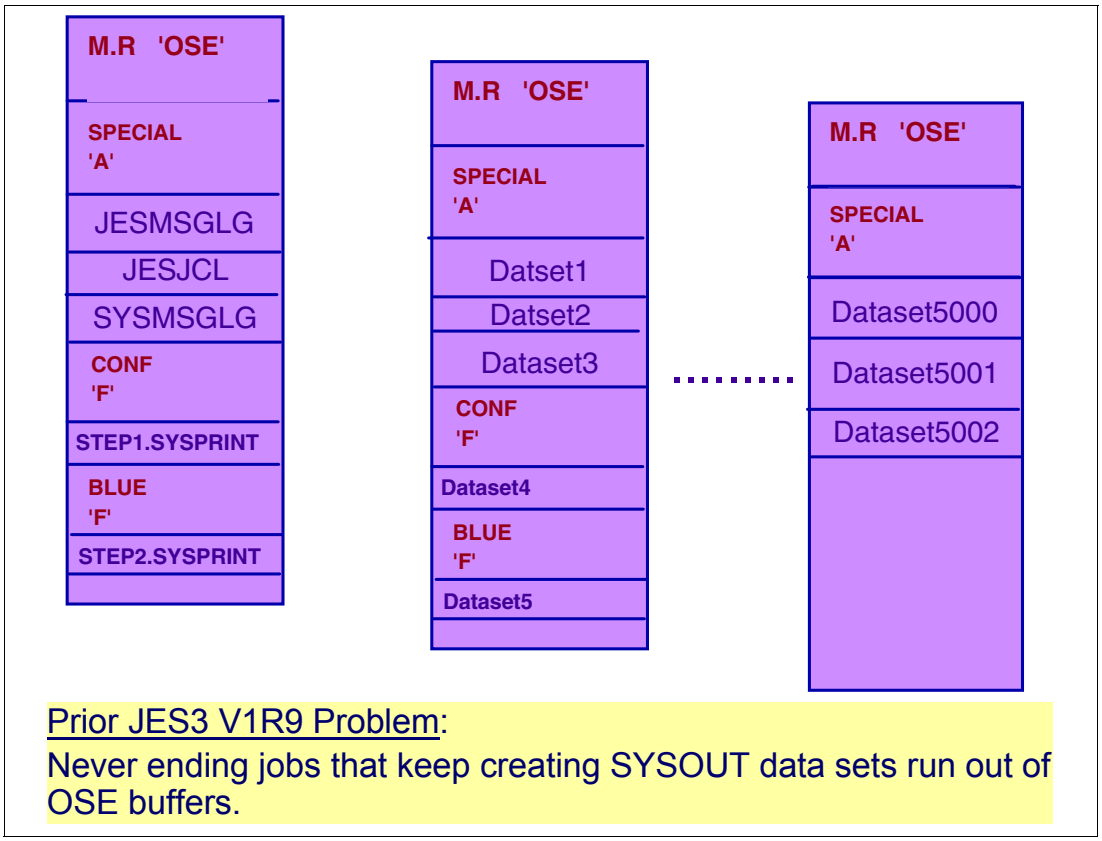
Output service uses the MOSE structure to improve scheduling performance. The OSD (mapped by IATYOSD, the output service driver data area) contains a pointer to the MOSE control structure.

The output service scheduling element (OSS) is a control element that specifies availability of OSEs and which jobs are associated with a MOSE. The MOSE and the job's resqueue both contain pointers to the OSS. The OSS contains a pointer to its MOSE and may contain pointers to additional OSS control blocks. A count of available OSEs in the OSS permits the scheduling of the job output.

The OSS specifies the availability of OSEs. The OSS represents data sets (under the OSEs) for the MOSEs. Each OSS element is chained as follows:

```
RQOSSTOP    points to the first OSS entry for a job (RQ).
OSSCHAIN    points to the next OSS entry for a job (RQ).
OSSNEXT     points to the next OSS entry under a master OSE (MOSE).
OSSRQADD    points to the job's RQ entry.
OSSBUFF     points to the number of first OSE buffers relating to this OSS.
```

11.15 OSE chained SRFs - long running job



OSE chained SRFs - long running jobs

JES3 output service, by its original design, was limited to 2 bytes for the Output Scheduling Element (OSE) buffers. Even worse, the code that worked with the buffer numbers did so with signed instructions. As a result, the limit was 32K buffers. APAR OW55574, which closed January 2003, bought a little time by changing the instructions to unsigned so that you could now get up to 64K buffers.

Increasing the limit to 64K didn't fix the problem for everyone. The real fix to the problem had to be to increase the capacity of the buffer number to four bytes instead of two. This makes it possible for a job to have 64K times as many buffers as it had before. For the application of the customer who submitted the requirement, their print application can now run for about of 65,536 weeks, or 1,260 years, give or take.

z/OS V1R9 JES3 lets a batch job or started task that needs to spin off output do so virtually indefinitely (barring other issues such as running out of spool space). Applications can therefore stay up longer with less need to recycle it.

Long running job support

The ability to create OSE buffers greater than 64K is dependent on the EXTOSENUM keyword on the OUTSERV initialization statement. This keyword takes a YES or NO value. However, the default is YES, and it does not require reading the initialization deck to take this default, so you have this function enabled.

11.16 Defining output devices to JES3

- Writer device characteristics
  - Defined with Initialization statements
  - Changeable By Operator
- "SETUP" Characteristics
  - "HELD" versus "CHANGEABLE" -- "NO" VERSUS "YES"
    - CARRIAGE / FCB NAME
    - FORMS
    - TRAIN
  - CHARS, FLASH, BURST, STACKER, MODIFY - 3800
- HEADER/ TRAILER Page Options
- LINE / PAGE Limit Specifications
  - Maximum or Minimum Limits

Defining output devices

The DEVICE statement is used to define I/O devices to JES3. Output service does not use the DEVICE statement exclusively, but the DEVICE statement is important to output service because it specifies:

- The "setup" characteristics of the device
- The header and trailer options for the printed output
- The type of data the device is allowed to process

How device characteristics are set and changed

Parameters on the DEVICE statement act as the defaults when the device is initialized. Once the device is initialized, some of these defaults can be modified by the ***CALL**, ***START**, and ***RESTART** operator commands or by the type of output sent to the device. If you perform a warm or cold start on your system, the initial defaults are used and the modifications are lost.

Forms, FCB/carriage tape, and train

These three characteristics are grouped together, since they have similar effect on output. The OUTSERV DSP routes output to a printer having the proper "setup" characteristics or to a printer having characteristics that may be changed to match the requirements of a data set. You can define devices with characteristics that are not to be changed. Many installations have some printers with specified, unchangeable characteristics and other printers whose characteristics are changeable as required.

Line and page limit

A characteristic assignable to output devices is that of line limit. You can use this limit to cause data sets of less than 10,000 lines, for example, to print on one printer and data sets with more than 10,000 lines to print on another. In this manner, you can route low volume output to a slow printer and high volume output to a fast printer. A characteristic similar to line limit assigned to output devices is that of page limit. You can use this limit to cause data sets of less than 2,000 lines, for example, to print on one printer and data sets with more than 2,000 lines to print on another. You can route low volume output to a slow printer and high volume output to a fast printer through this method, also.

Character set image and forms flash cartridge

These two characteristics are associated with the IBM page-oriented printers (such as the IBM 3800 printer), allowing you to change the print character set and the print forms to be formatted or "flashed".

11.17 Printer DEVICE statement

```
DEVICE,JNAME = PRT1
        ,JUNIT=JUNIT parms
        ,BURST = YES|NO
        ,CARRIAGE = (YES|NO , name)
        ,CKPNT = nnnnn
        ,DGROUP=LOCAL|groupname
        ,DGRPONLY=NO|YES
        ,DTYPE =PRTxxxxx
        ,DYNAMIC =YES|YES,timeout|NO
        ,FORMS = (YES|NO , name)
        ,HEADER = YES|NO
        ,LINELIM = nnnnnnn{+}
        ,PAGELIM = nnnnnnn{+}
        ,PM = (LINE|PAGE|(LINE,PAGE)
        ,RECORDS=nnn
        ,XLATE=YES|NO
        ,WS = STANDARDS|(D,T,F,C,U,FL,CM,SS,CL,L,P,PM)
        ,WC = STANDARDS|class
* AFP printer parameters
        ,CHARS = (YES|NO , id1,....)
        ,FLASH = (YES|NO , name)
        ,MODIFY = (YES|NO , name, trc)
        ,STACKER = (YES|NO , C|S)
        ,BURST = (YES|NO , M)  - 3800 only
        ,TRAIN = (YES|NO , name)
```

DEVICE statement

The DEVICE statement is used to define the output devices for printers, punches, (both local and remote through RJP). The set of valid parameters of a DEVICE initialization statement depends on the device type (DTYPE). The visual shows the set of parameters that can be used for all printers and some of the AFP printer parameters. For full description of the DEVICE statement, *see JES3 Initialization and Tuning Reference.*

No initialization stream overrides are supported for DEVICE statements.

Changes to the DEVICE statements through warm start, cold start, or hot start with refresh for all parameters. Omitting one statement might cause a subgeneric split or loss of a device. Omitting all statements for a device causes loss of the device as a JES3-managed device. Hot start with refresh applies for all parameters.

Determining which output parameters apply

The parameter values used for output vary according to the interaction between the device defaults and parameter values in the OSE based on writer-selection criteria. The following is a guideline for understanding which output parameters apply:

How device characteristics are set and changed: Parameters on the DEVICE statement act as the defaults when the device is initialized. When the device is initialized, some of these defaults can be modified by the *CALL, *START, and *RESTART operator commands or by the type of output sent to the device. If you perform a warm or cold start on your system, the initial defaults are used and the modifications are lost.

How output characteristics are associated with a data set: Defaults specified on the OUTSERV and SYSOUT initialization statements and JCL or JES3 control statements are used to build the OSE for each instance of output.

How data set characteristics are associated with a device: For each parameter defined in the OSE (for example: CHARS or FORMS), one of the following is true:

- If the parameter is not one of the writer-selection criteria, the value in the OSE is ignored and the current value defined for the device is used for the output.
- If the parameter is one of the writer-selection criteria and is not changeable on the device, only an OSE whose parameter value matches the value on the device will be selected for output.
- If the parameter is one of the writer-selection criteria and is changeable on the device, the value on the device is changed to match the parameter value of the OSE to be selected for output.

Writing output

JES3 writer support consists of a writer driver, writer scheduling (selection) routines, device-dependent routines, command-processing routines (also called message-processing routines) and spool-access routines (for print and punch writers).

In most cases, the writer support is provided within the JES3 global address space. Certain devices, however, use device-dependent routines that operate in a separate address space called an output writer functional subsystem (FSS) address space. In this case, the writer driver and the command-processing routines operate in the JES3 global address space and communicate with the output writer FSS using the functional subsystem interface (FSI). The device-dependent routines, also called a functional subsystem application (FSA), and the spool-access routines operate in the output writer FSS.

The spool-access routines in an output writer FSS read and write spool data from USAM protected data buffers (PBUFs) in CSA. JES3 does not release a PBUF until all the records in the PBUF have been copied to a private area buffer in preparation for passing them to the FSA. To allocate enough pages of storage for PBUFs used by all the output writer FSSs in the JES3 complex, use the PRTPAGE parameter on the MAINPROC statement.

Controlling the starting and stopping of writing output

There are several methods of controlling the starting and stopping of writing output:

- The operator controls the writer and its associated devices through the *CALL, *START, *RESTART, or *CANCEL commands. The writer notifies the operator when it is waiting for work and remains available for processing.
- The starting and stopping of the writer and its associated devices is controlled by JES3 output service based on the availability of output devices and output data set requirements that exist at any given time. When no more data sets with defined characteristics for the writer are available for processing, the writer is automatically terminated.

  Running an output writer FSS as a dynamic writer could slow output processing, because an address space must be brought up or down each time the writer is started or stopped.
- A Timeout Value can be defined for a writer. This value controls how long a writer remains idle before terminating.

  The value is specified on the DYNAMIC= keyword of the DEVICE initialization statement (for example, DYNAMIC=(YES,timeout value)). If you specify DYNAMIC=(YES) (with no timeout value), the writer stops automatically when no more output is available for processing. If you specify DYNAMIC=(NO) (with no timeout value), the writer remains active indefinitely even when no output is available for processing.

11.18 Writer selection parameters

- **WS= parameter defines the writer selection criteria**
  - The ORDER is important when data sets are selected

C	- CARRIAGE TAPE or FCB (printers only)
CL	- SYSOUT Class
CM	- Copy Modification (3800 only)
D	- DESTINATION
F	- FORMS
FL	- FLASH (3800 only)
L	- LIMIT Scheduling (lines, pages)
SS	- STACKER (3800 only)
P	- PRIORITY
PM	- PROCESS MODE (line or page)
T	- DEVICE TYPE (PRT,PUN,SYS)
U	- TRAIN UCS

- **WC= parameter specifies writer classes.**
  - The ORDER is the processing order by the writer

Writer selection parameters

Specifies the writer selection criteria. The value of the selection criteria indicates the items JES3 output service checks, in order of importance, when selecting a data set for output processing on this device. (These WS parameter values remain in effect over a hot start.)

Specify the selections in order of importance.

WS=STANDARD specifies that the values on the WS parameter of the OUTSERV initialization statement is to be used.

Writer selection (WS)

The writer selection parameter (WS) list is specified on the OUTSERV and on a DEVICE initialization statement.This list may be modified by a keyword on the **CALL**, **START**, or **RESTART** commands. The parameters that can be specified are shown in the table in the visual.

Writer class (WC)

The writer class parameter (WC) specifies writer classes. This parameter indicates SYSOUT classes in the order they are processed by the output service writers for this device. Data sets for a class not specified will not be selected. The WC values specify valid SYSOUT classes that appear on the SYSOUT initialization statements.

WC=STANDARD specifies that the values on the WC parameter of the OUTSERV initialization statement is to be used.

11.19 Data set processing to writers

- Data sets selected for processing
  - In order of WS parameters
- Select job with 'Best Match' OSE
  - Queued for Output
- Output OSEs for job in 'Best Match' order
  - Mark ALL OSEs in job
    - "Pending for same printer"

Processing data sets to writers

The writer selection (WS) list is specified on the OUTSERV initialization statement or on the DEVICE statement for the writer. This list may be modified by a keyword on the CALL, START, or RESTART commands. Data sets are selected in the order that the WS parameters are specified for the device.

Consider a WS= specification as follows:

```
WS=(D,T,F,P,CL)
```

Attention: Omission of any of the other WS= parameters indicates that no checking is to be done for the missing parameter.

Using the WS as specified above, a writer request for a job means finding a job with the highest priority data set whose destination is consistent with the writer's device and whose forms and UCS train (or chars) are set up on the writer's device (or are allowed to change). When there are several jobs with output that meets these particular requirements, the best match job is selected. (For example, a job with no set-up changes is a better match than one with set-up changes.)

When a job is selected to the writer, the OSEs are processed in the best match order and all OSEs that can process on the device are marked pending for the device. This prevents a different writer from selecting the same OSEs.

SYSOUT class (CL)

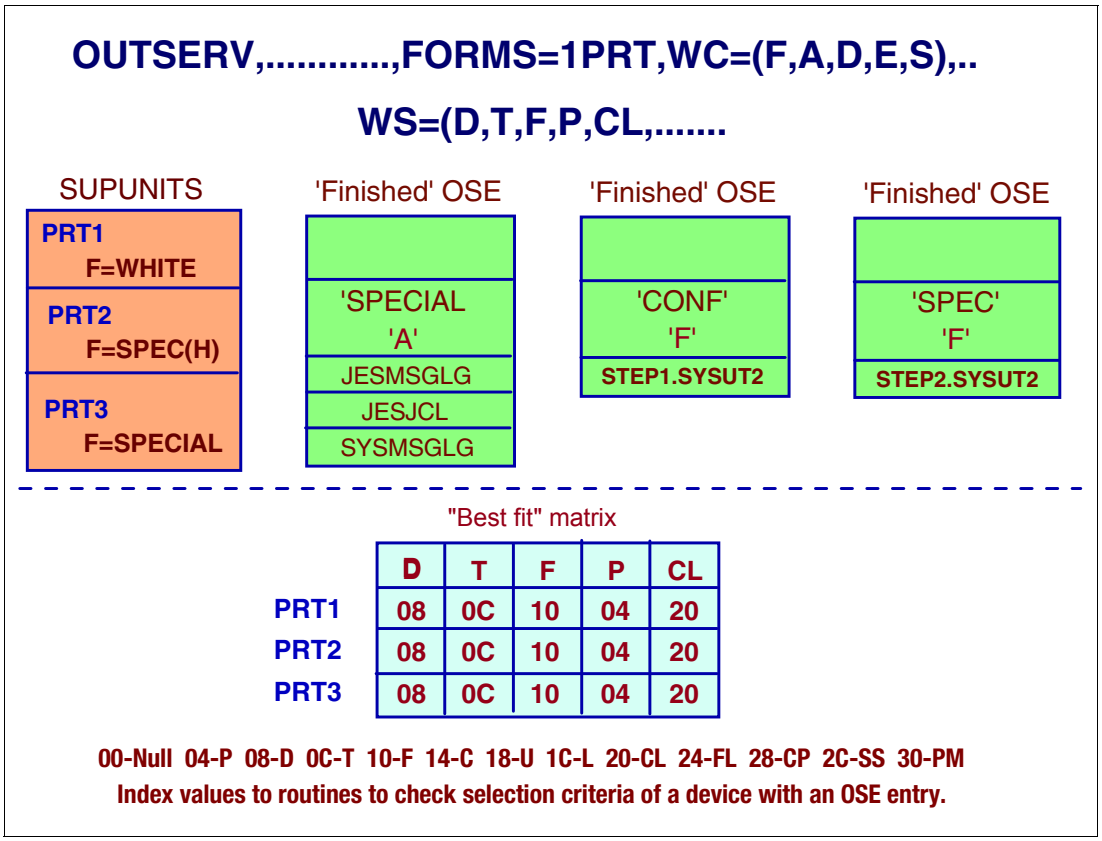
If SYSOUT class (CL) is used in the writer selection list, the CL parameter lists the SYSOUT classes that will be processed by the writer. The SYSOUT classes are controlled for each device either by an installation default for all writers or an individual specification a a single writer. The WC=class specification, is one or more output classes (A to Z, 0 to 9) that can be processed by this writer. The WC= parameter allows a maximum of 36 SYSOUT classes.

Use the WC= keyword on the **`*S devname`** or **`*R devname`** command to reassign the output classes that the writer can process. This command can be issued while the writer is active. Output classes of subsequent activity are affected. Regardless of the classes specified, they can be ignored if output class (CL) is not specified in the WS= keyword.

In the figure for this section, the default specification specified on the OUTSERV statement in the initialization statements specifies the only SYSOUT classes (F,A,D,E, and S) are to be used. The order is which they are listed indicates the class F output is processed by the others if priority (P) is specified in the WS parameters.

Note: Anytime WC parameters are specified for an individual writer or in the initialization stream, the WC classes that are not specified are added behind the ones that are specified with the exception of CL and P. Those two must be specifically specified to be used.

11.20 Scheduling OSEs to writers



Scheduling OSEs to writers

The SUPUNITS is the control block that contains the characteristics of the JES3 device. Module IATOSWS and IATOSFG matches the characteristics of a device to that of a job's SYSOUT (OSE) to start writers and/or select work. There are 5 different SUPUNITS formats: print, punch, networking, SNA RJP, and BSC RJP.

In the example in the visual, the scheduling of the just created OSEs is attempted by the OUTSERV FCT using the default WS= parameters from the OUTSERV statement or WS= parameters specified on each device that are now in the SUPUNITS entry. The writer classes are used from the OUTSERV statement or the ones specified in the SUPUNITS entry for each device.

Output service scheduling builds a mask to determine the best match OSEs to schedule to the available devices.

Select best fit OSE

JES3 uses the "best fit" approach to match SYSOUT data sets with an available device (writer). The following paragraphs describe the "best fit" approach.

First, JES3 compares the characteristics for the data sets to the characteristics of a writer based on the writer's selection criteria. If JES3 finds that more than one data set is eligible to be processed by a writer, then JES3 uses the following primary factors to determine the best fitting data set to a writer:

- Characteristics order of importance in the writer-selection list

- Job and data set priority

Once the best fitting data set is determined, the remainder of that job's data sets are processed applying the "best fit" approach within the boundary of the same job.

Writer characteristics

An output service writer has characteristics that can be changed (changeable characteristics) or cannot be changed (non-changeable characteristics) during work selection processing. Some of the writer's characteristics can be toggled between changeable and non-changeable by using the hold option for that particular characteristic. Non-changeable characteristics listed in the writer-selection criteria require the data set to match exactly the device characteristics; otherwise the data set is considered ineligible.

Changeable characteristics allow a writer to make adjustments to the writer-selection criteria list to accommodate data sets with different characteristics than are currently in use on the device. Note that priority always plays a role in work selection.

On the visual example writer device PRT2 can only process output that specifies forms “SPEC”. The **H** in the forms definition specifies that only designated forms are to be used until you change this status. **R** would specify that JES3 can request that different forms be placed on this device.

For example, to change the forms for printer PRT2, enter command:

```
*S PRT2,F=(STANDARD,R)
```

11.21 Output Operators Commands

- Operator command structure
  - *I U, parameters
  - *F U,parameters
  - Calling hot writers
    - *X WTR,parameters
  - Controlling printers
    - *S PRTx,parameters
    - *R PRTx,parameters
    - *C PRTx,parameters

Output service operator commands

You can use the JES3 **`*I`** and **`*F`** commands to inquire about, modify, or delete data sets on the writer, hold, or BDT queue.

Use the **`*X WTR`** command to control output devices such as hot writers.

Once output devices are active, you can control what is processing by using the **`*S`**, **`*R`**, and **`*C`** device commands.

***I U**
Enter the **`*I U`** command to display information for work currently on the output service writer queue (Q=WTR), the output service hold queue (Q=HOLD), or the output service MVS/BDT queue (Q=BDT). The following subsections describe the parameters you can include in the command. If you do not specify a particular queue, the display contains information for the writer queue.

***F U**
Use the **`*F U`** command to modify requirements for work currently on the output service writer queue (WTR), the output service hold queue (HOLD) or the output service BDT queue (BDT).

***X WTR**
Use the **`*X WTR`** command to invoke a hot writer that will drive a selected output device or a device chosen by JES3. The first call for a hot writer does not start output processing. The **`*X`** command invokes a unique writer program for the device and allows you to establish required writer characteristics.

***S devname** You must enter a **`*S devname`** command to start the writer, after a hot writer has been called, a dynamic writer has been invoked the first time, or any writer in manual mode has been called. This is an opportunity for you to ensure that various writer characteristics have been properly assigned before actual printing or punching.

You can change writer characteristics using the various parameters of the **`*S devname`** command without having to reenter the **`*X WTR`** command.

***R devname** You can use the **`*R devname`** command to respecify writer characteristics when you must stop a writer that has been started and then restart it. When you use the *RESTART,devname command, JES3 interrupts the current writer activity and allows you to respecify output writer characteristics before JES3 continues to process work.

***C devname** Unless a writer is ended (canceled), JES3 either continues to schedule the data sets that the writer is configured to process or waits for you to start it with a new configuration, using a **`*S devname`** or **`*R devname`** command.

A dynamic writer automatically ends when there is no more output in the queue to process. If necessary, a dynamic writer can be removed from JES3 scheduling by varying it offline. Varying the device offline, however, will not interrupt current activity. To end the current activity on the dynamic writer and to prevent further scheduling of the writer, use the **`*V`** command and then use the **`*C`** command. Ending a writer also ends an output writer functional subsystem, if one is controlling the device. (The writer must be in an idle state; if it is not, only the current data set on the writer is canceled.)

11.22 Output queues

WTR Queue: Q = WTR

- All data sets for processing to printers, punches

HOLD Queue: Q = HOLD

- All data sets for external writers, TSO users
  - SYSOUT,CLASS = ,.....,HOLD = EXTWTR
  - SYSOUT,CLASS = ,.....,HOLD = TSO

BDT Queue: Q = BDT

- Output for SNA NJE users

TCP Queue: Q = TCP

- Output for TCP/IP/NJE users

Output queues

The output queues for SYSOUT data sets are as follows:

Q=WTR Output service writer queue (Q=WTR): This queue contains data sets waiting for output processing. Output service automatically selects data sets for processing based on their selection characteristics such as output class, output priority, and output device-related requirements. You can use JES3 commands to place these data sets in operator-hold status. You can also use JES3 commands to modify a data set's selection characteristics.

Process SYSOUT and SYSOUT application program interface applications can also process work on the output service writer (WTR) queue.

Q=HOLD Output service hold queue (Q=HOLD): This queue, sometimes called the hold-for queue, contains data sets that are to be processed by other than JES3-managed devices. These data sets must be processed by the function for which they are held (external writer or TSO). The function that processes the data set can then change data set characteristics, release it for JES3 processing, or cause JES3 to purge it. If necessary, the operator can force a JES3 writer to process the data set or issue a modify (`*F`) command to move the data set to a WTR for JES3 device processing.

Q=BDT This queue contains SNA/NJE networking job or networking system output streams. MVS/BDT sends these job or system output streams to the proper node within a SNA/NJE network. You can use JES3 operator commands to hold, release, or cancel networking requests from the queue.

Q=BDT This queue contains TCP/IP/NJE networking job or networking system output streams. TCP NJE sends these job or system output streams to the proper node within a TCP/IP/NJE network. You can use JES3 operator commands to hold, release, or cancel networking requests from the queue.

Output in the HOLD queue

It is possible for a TSO user to access the output job control language (JCL), system messages, and system output (SYSOUT) data sets of a batch job. The user that submits the batch job must first make the job's output available to TSO. There are two ways the user can do this:

- Assign the job's output that is to be accessed to a SYSOUT class for which you have specified HOLD=TSO.
- Specify on the // JOB statement MSGCLASS parameter a SYSOUT class for which have been specified as TYPE=RSVD and assign the job's SYSOUT data sets to a SYSOUT class for which have also been specified as TYPE=RSVD.

You must have previously specified HOLD=TSO and TYPE=RSVD on the appropriate SYSOUT initialization statements. To access the output of a batch job, which is on the JES3 spool, the TSO user must issue the TSO command, OUTPUT.

To make output available for external writers, the SYSOUT statement can be used to EXTWTR hold a particular class. Doing this specifies that the data set is to be returned to an external writer.

11.23 Writer hold classes

- EXTWTR SYSOUT class
  - SYSOUT,CLASS =W,HOLD=EXTWTR
  - XWTR -- IBM-supplied cataloged procedure

```
//XWTRPROC PROC
//IEFPROC  EXEC  PGM=IASXWR00,PARM='PA',REGION=20K
//IEFRDER  DD UNIT=TAPE,VOL=(,,,35),DSNAME=SYSOUT,DISP=(NEW,KEEP),
//         DCB=(BLKSIZE=133,LRECL=133,BUFL=133,BUFNO=2,RECFM=FM)
```

- User job creating SYSOUT
  - //MYDD DD SYSOUT=W
  - //MYDD1 DD SYSOUT=(A,STDWTR)
  - //MYDD2 DD SYSOUT=(A,PLOTTER)

Defining writer HOLD=EXTWTR class

Output data sets placed on the output service hold queue are processed by the following system routines:

- An external writer is an MVS routine that handles output destined for devices not supported by JES3 output service, such as DASD or magnetic tape devices.
- This visual shows examples of external writer specifications in a user's JCL. The SYSOUT class of the MYDD statement matches the initialization statement specification of HOLD=EXTWTR.

The other two DD statements reference an external writer name that causes the output to be held for an external writer of the name to process the output. The IBM supplied external writer XWTR processes output with an external writer name of STDWTR.

External writer application routines execute in an address space other than the JES3 address space. This type of writer is functionally independent of JES3 and operates as a completely separate MVS job. However, the external writer/SAPI application interacts with JES3, via the subsystem interface, to request data sets for processing. A subset of the output service scheduling function called PROCESS SYSOUT and system application printer interface are invoked as a result of this kind of request.

No attempt is made by output service to schedule external writer applications as a result of constructing OSEs requiring their services; it is the responsibility of the operator to start external writers as required. JES3 posts started writers that are waiting for work when incoming jobs require processing.

11.24 Output service writers

- Dynamic writer
  - Automatically started by Ouput Service
  - DEVICE,...,DYNAMIC=YES[,timeout]
  - When output exists - (OSE) and device available
  - Output processed on job basis
  - All data sets printed ?
    - Get next job --- If no job - writer terminates
- Hot writer
  - Called by operator - *X WTR,.....
  - DEVICE,...,DYNAMIC=NO[,+]
  - Output processed on job basis
  - All data sets printed ?
    - Get next job --- If no job - writer "Waits" for work
- Operator command: *F W,device,DYN=YES|NO

Output service writers

The Writer DSP drives a device and selects data sets to ensure that data sets reach their destination. Writer DSPs may be categorized in two distinct groups: dynamic writers and hot writers. Subcategories within those two groups are external writers, writers that operate in the JES3 address space, and writers that operate in the FSS address space.

Dynamic writers

JES3 output service starts the writer and its associated devices, based on the availability of output devices and the current output data set requirements. After JES3 initialization, you must use the ***S** command the first time you use a device associated with a dynamic writer. After that, printing or punching begins automatically for properly prepared devices that are in the ready state. You can use the ***S**, ***R**, and ***C** commands to control dynamic writers while they are active. The dynamic writer will stop immediately after no suitable output is available for processing by the writer.

A dynamic writer is scheduled when:

- There is work (one or more data sets) in the output queue
- There is an output device available

Dynamic writers reduce the amount of control operations personnel have over when and how writing is performed. They do allow operator interaction while the writer is active (that is, changing of setup characteristics). The dynamic writer terminates when no more work is available or a higher priority DSP is waiting for the device.

Hot writers

A hot writer is invoked using a ***X** command. The command can either be issued by you or by JES3, depending on how the DEVICE associated with the writer is defined to JES3. If the DEVICE specifies NO for the DYNAMIC keyword, the writer is controlled exclusively by you. If the DYNAMIC keyword specifies YES and a nonzero value for the timeout, the writer is eligible for an automatic ***X** by JES3. The writer notifies you when it is waiting for work and remains available for processing. The writer will wait for as long as the timeout period defined for the device. If no new work becomes available it will then terminate. You control the writer using the ***X**, ***S**, ***R**, and ***C** commands. You can use the *MODIFY,W command to restrict a device to be started as a hot writer or to change the associated timeout value. If the device is defined with the DYNAMIC keyword set to YES and a timeout value of zero, the device is eligible for a dynamic writer.

Defining dynamic writers

DYNAMIC=YES|NO on the DEVICE statement specifies whether or not you want this device started dynamically when there is work available to print, punch, or to be sent to a remote printer or punch. If you specify NO, you can only start this device by calling a hot writer using the JES3 ***X WTR** command. You can change the value of this parameter during system operation using the ***F W** command.

11.25 Output writer operator commands

- Device Characteristics Obtained From
  - DEVICE Statement
  - OUTSERV Statement
- Placed into SUPUNITS Table for devices

CARRIAGE	CHARS	FORMS
FLASH	MODIFY	STACKER
WS =	WC =	TRAIN - (UCS)

- Characteristics may be Changed by Operator
  - *S PRT1,F = (WHITE,H)
  - *R PRT1,U = (PN,H)

Output writer commands

Every printer JES3 global device obtains its device characteristics from the OUTSERV and DEVICE statements. The DEVICE statement parameters override the OUTSERV statement parameters when they are the same keyword. The final value is saved in the SUPUNITS table for the device.

The values stored in the SUPUNITS entry for a device may be updated by the operator when the JES3 **`*S`** and **`*R`** commands for that device are issued.

Dynamic writers are often used for volume printing on stock paper. These writers allow JES3 to control changing the setup characteristics for devices thereby reducing the amount of control operators have over when and how writing is performed. Hot writers give operations personnel total control of output handling. Operators enter commands to call and control hot writers. Hot writers remain available, even when there is nothing to print.

Using commands to control printing

You can change writer characteristics using the various parameters of the **`*S devname`** command without having to reenter the **`*X WTR`** command.

You can use the **`*R devname`** command to respecify writer characteristics when you must stop a writer that has been started and then restart it. When you use the **`*R devname`** command, JES3 interrupts the current writer activity and allows you to respecify output writer characteristics before JES3 continues to process work.

11.26 Controlling hot writers

- OUTSERV defaults specified
  - OUTSERV,WS = (D,T,F,C,U,FL,CM,SS)
- SUPUNITS updated when writer schedules
  - Hot writer WS parms are merged with defaults
    - OUTSERV,WS = (P,CL,D,T,F,C,U,FL,CM,SS)
- Which jobs require "white" forms
  - *I U,F = WHITE,N = ALL
- Call a hot writer

***X WTR,OUT=PRT1,F=(WHITE,H),WS=(P,CL),WC=(J,V)**

Controlling hot writers

When you call the hot writer, you can specify parameters that allow you control the writer during its activity and to change parameters that are currently in effect. By specifying one or more appropriate parameters at the time you enter the **`*X WTR`** command, you can:

- Suspend writer output until a specific device becomes available
- End the writer if the device is unavailable
- Control the forms, band or train, or carriage tape to be used
- Control printer device type specifications
- Create header and burst records
- Specify the writer-selection characteristics to be used and their order
- Specify output classes for this writer
- Specify the maximum or minimum number of lines or pages allowed per data set
- Specify the number of seconds that should elapse or pages that should be written before a checkpoint is taken
- Specify the diagnostic mode
- Stop the writer between data sets

When you specify WS= on the call command, the WS= parameters specified are merged with the default WS parameters from the OUTSERV statement.

The SUPUNITS entry for the device is updated with the call parameters.

Since forms of WHITE was specified on a WTR call command, if you want to determine which jobs require WHITE forms, issue the **`*I U,F=WHITE,N=ALL`** command.

11.27 Query writer status

```
*I D D=PRTWAY
IAT8562 PRTWAY      (AC ) WTR   F=STD(H) CH=(GT15) DYN=Y
IAT8562 PRTWAY                  TIMEOUT=NONE
IAT8562 PRTWAY                  H=Y B=N M=A
IAT8562 PRTWAY                  L=0+ PG=0 CK=200P DGRPY=N MK=C
IAT8562 PRTWAY                  NPRO=90 MODE=FSS FSS=PRINTWAY
IAT8562 PRTWAY                  C=STD3 PM=LINE,PAGE
IAT8562 PRTWAY                  SS=C CM=(NONE,0)
IAT8562 PRTWAY                  SETUPMSG=Y
IAT8562 PRTWAY                  PDEFAULT=(NONE)
IAT8562 PRTWAY                  WS=(CL,U)
IAT8562 PRTWAY                  WC=J DGRP=TCPIP
IAT8562 PRTWAY                  XLATE=Y
IAT8562 PRTWAY                  JOB WTR (JOB25655)
IAT8567 SC64=       SC70=         SC65=
IAT8500 INQUIRY ON DEVICES COMPLETE
```

WTR commands

The command **`*I D,D=`**devnum or /devnum or devname specifies the number (by 3-digit or 4-digit hexadecimal number) or name of one or more output devices. If omitted, the number specified in the N parameter or the first ten JES3-managed devices attached to the global are displayed. A slash (/) preceding the device number is not required. Device numbers can be specified by any of the following formats:

```
ddd
dddd
/ddd
/dddd
```

Note: The devname or device number can be specified. FSS printers do not have device numbers.

Writer device command examples

Display the numbers of all devices in the complex defined to JES3 with each of five device numbers starting with the number 3D0:

```
*I,D,D=3D0,N=5
```

Display the status of each occurrence in the complex of the device defined on mains SC49 and SC50 by the device number 3D0:

```
*I,D,D=482,(SC49,SC50)
```

11.28 Printer checkpoints and notes

- Printer specified checkpoints
- Checkpoint taken at specified interval while printing
  - Specified on DEVICE Statement

Printer	Defaults
3800	10000
3211	2000
other printers	1000

- Output service takes 'Notes' every 100 lines while printing

Restarting printer commands

The frequency with which checkpoints are taken is specified by the CKPNT parameter on the DEVICE initialization statement. The actual frequency with which checkpoints are taken is approximately the value specified by CKPNT (CKPNTPG or CKPNTSEC for FSS-supported devices) but is never more than twice the specification. For example, if the default is used (1000 records), a restart would cause printing to resume between 1000 and 2000 records prior to the current position. Because they are buffered devices, the printed output on remote writers might be misleading; the record count includes data that has been transferred to the buffer but not yet printed. If a restart with repositioning has been performed, the checkpoint intervals might not be on even boundaries.

If no valid checkpoint exists for the restarted data set, printing or punching resumes at the start of the current copy of the current data set.

Output service also maintains a note every 100 lines. Printers can be restarted at the last note taken.

Note: Also make sure that the DD statements for all data sets in the JES3 procedure are correct and are for the correct global. For the checkpoint data sets in particular, note the group name shown in IAT3040 when you start JES3. If the group name is incorrect, it means you have specified checkpoint data sets for the wrong global or possible an old set of checkpoint data sets that you are no longer using but have not deleted.

11.29 Restarting printers

- Reschedule job currently on printer
  - *R PRT1,RSCD or *R PRT1,RSCD,C or
  - *R PRT1,option

Options	Meaning
C	Last Checkpoint
N	Last Note
J	All Data sets for Job
G	Only Data sets 'this' printer

- Place current data set printing into 'HOLD'
  - *R PRT1,HOLD

Restarting printer commands

Use the **`*R devname`** command with the appropriate parameter to stop print or punch activity for the current data set and to immediately restart activity at a prior point in the job. The options are as follows:

RSCD Specifies that the writer is to perform a scheduling pass.

To reschedule a job currently on a printer or after respecifying writer-selection characteristics, you should issue the **`*R,devname,RSCD`** command to force the writer through a scheduling pass with the new criteria. At this point the writer will find work to process or terminate. In either case, the new characteristics will remain in effect until you respecify new ones.

C Specifies restarting print or punch activity from the last checkpoint taken for the current data set.

G Specifies that all data sets for the current job that were processed by the restarted device are to be printed.

J Specifies that JES3 resqueue all completed data sets of the appropriate type (that is, PRT or PUN) for the current job. JES3 does not resqueue spin-off data sets.

N Specifies to restart printing of the current data set at the last internally-noted checkpoint (100 to 200 lines or records). If there is no internally-noted checkpoint, printing resumes at the last checkpoint or, if there is no checkpoint, at the beginning of the data set. This parameter is not valid for TSO/E writers. For a 3800 printer, N is treated as if C were specified (internally-noted checkpoints are not used).

Note: The C and N parameters are only valid on the **`*R devname`** command. Use the *R devname command with the appropriate parameter to stop print or punch activity for the current data set and to immediately restart activity at a prior point in the job. The current data set on a printer is the data set that is visible at the operator orientation point (OOP). On a 3800 printer, the OOP is located slightly above the transfer station.

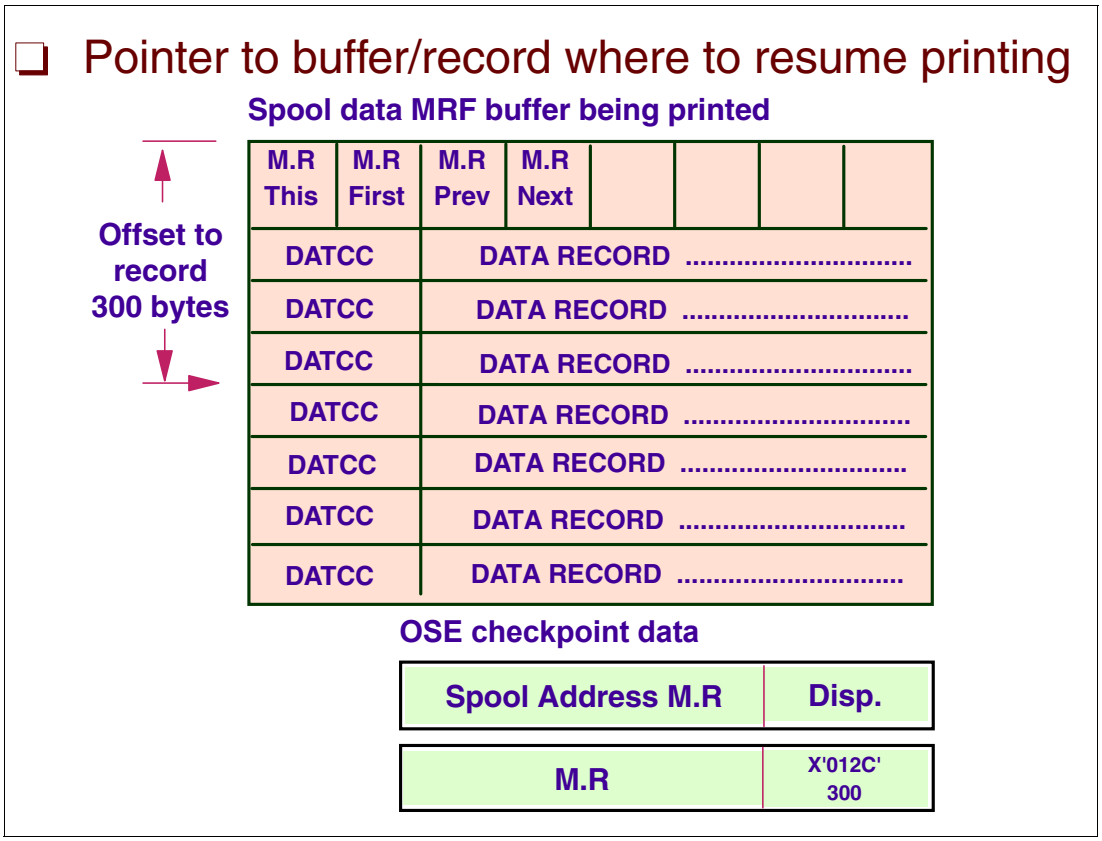
RSCD option

Data sets that are currently printing may be placed into an operator hold status by issuing an operator command.

```
*R PRT1,H
```

After respecifying writer-selection characteristics, you should issue the **`*R,devname,RSCD`** command to force the writer through a scheduling pass with the new criteria. At this point the writer will find work to process or terminate. In either case, the new characteristics will remain in effect until you respecify new ones.

11.30 Checkpoints and notes



Checkpoints and notes

This visual illustrates how writer checkpoints and notes are taken and maintained by output service. The checkpoint and note are 8-byte in length. The first 6 bytes are the M.R of the data buffer in which the checkpoint or note was taken. The other 2 bytes is the displacement into the data buffer to the record where the checkpoint or note is taken.

In the example, M.R This is the M.R and the displacement is 300 bytes (012C in hex) to the record in the buffer. If the printer is currently positioned to print that record, that record becomes either the checkpoint or the note.

From the note, the M.R specifies the actual spool buffer to be read into storage that contains where printing is be resumed. The displacement (Disp.) indicates to offset down into the buffer to the record that restarts the printing.

Note: Notes are taken every 100 lines. Each printer definition allows a checkpoint to be defined for restarts and repositioning of data sets that are printing.

.

11.31 Repositioning output on printers

- Forward space repositioning
  - From current line printed
    - *S PRT1,R = 100 or *S PRT1,R = 10P
- Backspacing a data set
  - From current line printed
    - *S PRT1,R = -100 or *S PRT1,R = -10P
- Operator restart command
  - When issued: Restarts from beginning of data set
    - *R PRT1,R=100 or *R PRT1,R=10P

Restarting output

The frequency with which checkpoints are taken is specified by the CKPNT parameter on the DEVICE initialization statement. The actual frequency with which checkpoints are taken is approximately the value specified by CKPNT (CKPNTPG or CKPNTSEC for FSS-supported devices) but is never more than twice the specification. For example, if the default for the 1403 is used (1000 records), a restart would cause printing to resume between 1000 and 2000 records before the current position. Because they are buffered devices, the printed output on remote writers might be misleading; the record count includes data that has been transferred to the buffer but not yet printed. If a restart with repositioning has been performed, the checkpoint intervals might not be on even boundaries.

Repositioning print data sets

Repositioning a printing data set (`R=`) specifies that the data set be spaced forward nnnnnnnn number of lines or nnnnnnnn number of pages (P) from the start of the current copy of the current data set, or from the point in the current job specified by the C or N parameter. For a page-oriented device, you must specify the number in terms of pages (P).

If the data set is spaced forward past the end of the current copy, message IAT7006 is issued and the output writer is stopped. Spacing by page on a non-3800 printer, when pages are not defined in the data set, also causes message IAT7006 to be issued and the output writer to be stopped.

Use of *S and *R for repositioning

For repositioning of output on printers, operators should be made aware of the following:

***S** Issuing the `START` command repositions from the current line in the data set.

To backspace from a position in the data set, use the `START` command.

***R** Issuing the `RESTART` command repositions from the beginning of the data set.

Note: You can backspace a 3800 or 3900 printer to any page of any copy of a data set that is not yet completely stacked. If you backspace it further, the printer is repositioned to the beginning of the data set currently being stacked.

11.32 Operator commands

```
*I  U,J=?,N=7
 IAT8131 JOB JES3 (JOB00000), T=PRT, L=139702, PG=0, SR=139702,
 IAT8131 JOB JES3 (JOB00000), BY=19068196.
 IAT8131 JOB VTAM44 (JOB23479), T=PRT, L=242, PG=0, SR=242, BY=12252.
 IAT8131 JOB OPTSO (JOB23480), T=PRT, L=1547, PG=0, SR=1547, BY=12252.
 IAT8131 JOB SYSLOG (JOB23474), T=PRT, L=4, PG=0, SR=4, BY=8168.
 IAT8131 JOB SDSF (JOB23494), T=PRT, L=51, PG=0, SR=51, BY=12252.
 IAT8131 JOB ZFS (JOB23495), T=PRT, L=207, PG=0, SR=207, BY=114352.
 IAT8131 JOB RMF (JOB23485), T=PRT, L=197, PG=0, SR=197, BY=118436.
 IAT8119 NUMBER OF JOBS FOUND : 1824
*I U,Q=BDT
 IAT8131 JOB VAINI (JOB25494), D=WTSCPLX9, L=4, PG=0, SR=4, BY=4084,
 IAT8131 JOB VAINI (JOB25494), TG=TCP00000.
 IAT8119 NUMBER OF JOBS FOUND : 1
 IAT8141 NUMBER OF TCP JOBS FOUND : 1
*I U,Q=BDT
*I U,Q=HOLD
*I U,Q=WTR
*I U
```

OUTSERV Inquiry Commands

Output service inquiry command for jobs is the **`*I U`** command. Using **`J=?`** just displays the first 10 jobs as shown in the visual.The IAT8119 message shows the total number of jobs are queued for output processing.

Your selection of the proper "Q=" keyword value on the **`*I U`** command dictates what output you want.

The choices are the following:

`*I U,Q=TCP` Use the **`*I U,Q=TCP`** command with the appropriate parameters to display information about TCP/IP/NJE network jobs.

`*I U,Q=BDT` Use the **`*I U,Q=BDT`** command with the appropriate parameters to display information about SNA/NJE network jobs. SNA/NJE network jobs may be either network job streams or network SYSOUT streams.

`*I U,Q=HOLD` Use the *INQUIRY,U,Q=HOLD command to display a summary of output in the HOLD queue.

`*I U,Q=WTR` Use the *INQUIRY,U,Q=WTR command to display a summary of output in the WTR queue.

`*I U` To display job output on the WTR service queue as the **`Q=WTR`** is not required as it is the default.

11.33 Operator commands for spool data sets

- What data sets are in 'HOLD' in WTR queue
  - ***I U,H=Y,J=?,N=ALL**
    - IAT8131 JOB JES3 (JOB00000), T=PRT, H=Y, L=24, PG=0, SR=24, BY=4084.
    - IAT8131 JOB WTRIAZF (**JOB25652**), T=PRT, H=Y, L=6700, PG=0, SR=6701,
    - IAT8131 JOB WTRIAZF (JOB25652), BY=69428.
- What is held for a given job?
  - ***I U,H=Y,N=5,J=25652,DSN=?**
    - IAT8131 JOB WTRIAZF (JOB25652), T=PRT, H=(OPER), L=2, PG=0, SR=2,
    - IAT8131 JOB WTRIAZF (JOB25652), BY=4084, COPIES=1,
    - IAT8131 JOB WTRIAZF (JOB25652), DD=.WTRIAZF.SYS00001(1),
    - IAT8131 JOB WTRIAZF (JOB25652), **DSN=STC.WTRIAZF.JOB25652.D0000009.?.**
    - IAT8131 JOB WTRIAZF (JOB25652), T=PRT, H=(OPER), L=21, PG=0, SR=21,
    - IAT8131 JOB WTRIAZF (JOB25652), BY=4084, COPIES=1,
    - IAT8131 JOB WTRIAZF (JOB25652), DD=.WTRIAZF.SYS00002(1),
    - IAT8131 JOB WTRIAZF (JOB25652), DSN=STC.WTRIAZF.JOB25652.D000000A.?.
- Release a 'HOLD' data set
  - ***F U,J=25652,DSN=...D0000009,NH = N**
    - IAT8122 CHANGED JOB WTRIAZF (JOB25652) DATA SET=STC.WTRIAZF.JOB25652.D0000009.?
    - IAT8118 NUMBER OF DATA SETS MODIFIED : 1

Output commands for data set control

Knowing the amount of work yet to be processed by output service allows you maximum flexibility when scheduling output for printing or punching or for processing by routines other than JES3 output writers. Use the **`*I U`** command to examine, at various levels of detail, the status or amount of work in the output service writer queue (Q=WTR), the output service hold queue (Q=HOLD), or the output service BDT queue (Q=BDT).

JES3 output data sets have a variety of unique characteristics. To display output information, you specify one or more of the output characteristics as criteria for what is to be displayed. Unless specifically stated, you can combine characteristics when using the *INQUIRY,U command. For example, to display information for all output in the WTR queue created by job TEST with an output class of T, enter:

```
*I,U,J=TEST,CL=T,N=ALL
```

Data sets in hold status

Data sets waiting for processing can be placed into HOLD status. To inquire which data sets are in HOLD status, use:

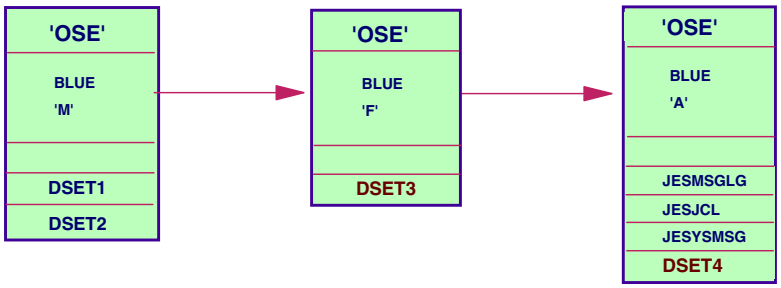
`H=Y`	As a filter asking for output data sets that are in hold status
`J=?`	For all jobs in the output queue specified
`N=ALL`	To get more than 10 jobs listed
`DD=?`	For the data set names

DSN= If you want information for a specific data set in the indicated job, include either the DD= keyword or the DSN= keyword on the **`*I U`** command. If you do not include the DD= or the DSN= keyword on the command, the response contains all data sets for the specified job. The data sets are identified by general type: print (PRT), punch (PUN), and any other qualifiers that you specify in the command.

To release a data set that is in hold status, use the **`*F U`** command. Specify the data set name, the job number, and if there are multiple data sets with the same name in the specified job, specify the sequence number S= as shown in the **`*I U`** command output. To change the HOLD status, specify **`NH=N`** (new hold equal NO).

11.34 Printing large data sets

- Data sets DSET1 and DSET2 - 1 million lines each
- Multiple copies of same data set requested
  - Data set DSET3 has 200,000 lines and 10 copies
  - Data set DSET4 has 5500 lines



All data sets schedule to SAME printer

Printing large data sets

JES3 provides a method to allow control of printing large data sets in a job to make sure they do not get queued for the same printer.

In the example shown in the visual, the job has multiple large data sets, DSET1 and DSET2, and a data set DSET3 which has 10 copies. DSET4 is the other data set and is much smaller. Normal output service processing would be to schedule all the data sets to the same printer as shown in the visual. A job that has work created in this manner is eligible to be concurrently processed by several output service writers that have the same processing characteristics. Normally, similar pieces of work for a job are assigned to the same writer.

THRESHLD parameter

The THRESHLD parameter should be used when the user wants to permit concurrent printing or punching of multiple copies of data sets or a large volume of data sets for a job on several output service writers. The THRESHLD parameter assumes that the data set size is the number of records in the data set multiplied by the number of copies. Data sets that equal or exceed the value specified are queued as separate piece of work for output service writers.

The THRESHLD parameter specifies the default maximum size for a SYSOUT data set. The maximum value that may be specified is 99999999. THRESHLD=-1, which is the default, indicates that no threshold processing is in effect. This parameter is used by output service to build and queue OSEs for writers. The THRESHLD parameter assumes that data set size is the number of records in the data set multiplied by the number of copies. Data sets that equal or exceed the value specified are queued as separate piece of work for output service writers.

11.35 THRESHLD specification

- Allow large data sets to print simultaneously
- Can be specified in 4 different ways

1 OUTSERV,.....,THRESHLD = 1500000

2 SYSOUT,CLASS=M,.....,THRESHLD = 100000

3 //*FORMAT PR,DDNAME = DSET2,THRESHLD = 5000

4 //DD1 OUTPUT THRESHLD=2000000
//DD2 OUTPUT DEFAULT=YES,THRESHLD=3000000

THRESHLD specification

The THRESHLD parameter can be used to avoid large data sets in a single job being scheduled to the same device, when it is desired to have large data sets print simultaneously on different printers. The THRESHLD parameter specifies the default maximum size for a SYSOUT data set. The maximum value that may be specified is 99999999. THRESHLD=-1, which is the default, indicates that no threshold processing is in effect. This parameter is used by output service to build and queue OSEs for writers. The THRESHLD parameter assumes that the data set size is the number of records in the data set multiplied by the number of copies. Data sets that equal or exceed the value specified are queued as separate piece of work (under separate OSEs) for output service writers.

Using THRESHLD parameter

There are four ways to specify this parameter for output processing as follows:

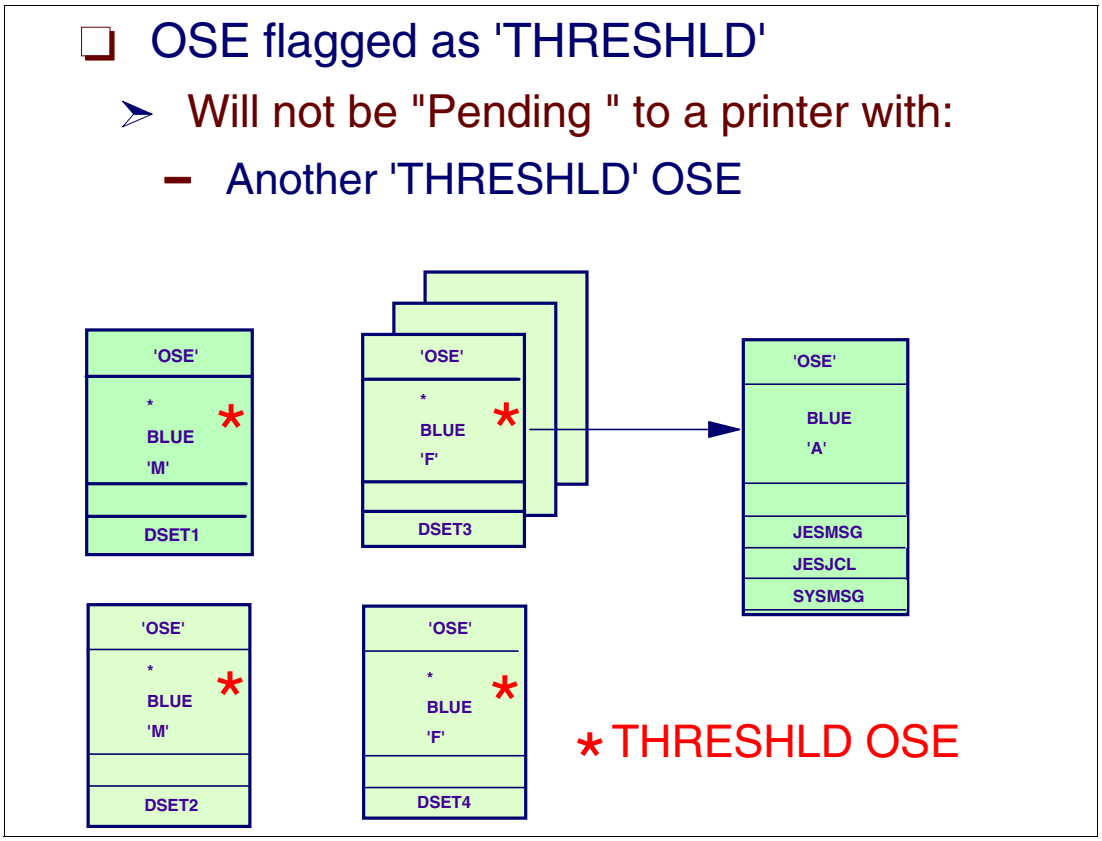
OUTSERV Specifying the THRESHLD parameter on the OUTSERV initialization statement sets an installation default for all data sets greater than the specified line count.

SYSOUT A THRESHLD count can be set be SYSOUT class using the SYSOUT initialization statement. All data sets with this SYSOUT class meet the THRESHLD state when the specified line count is exceeded.

//*FORMAT The submitter of the job can specify a THRESHLD count by data set name to cause THRESHLD processing when printing the job.

//DD1 OUTPUT To control whether a SYSOUT data sets from a job are printed together or as separate units of work, use a statement in the JCL.

11.36 Scheduling THRESHOLD OSEs



Scheduling THRESHLD OSEs

The THRESHLD parameter is used to cause printing of data sets meeting a THRESHLD specified count on printers separate from other non-THRESHLD data sets in the same job. A job that has work created in this manner is eligible to be concurrently processed by several output service writers that have the same processing characteristics. Normally, similar pieces of work for a job are assigned to the same writer. This parameter should be used when the user wants to permit concurrent printing or punching of multiple copies of data sets or a large volume of data sets for a job on several output service writers. This parameter is ignored when the copy distribution for a data set is 3000 or more.

JES3 calculates the size of the sysout data set(s) as the number of records multiplied by the number of copies requested. When the size exceeds the THRESHLD value, JES3 creates a new unit of work on a data set boundary, and queues it for printing.

Flag OSEs

When output service is scheduling the OSEs to a printer, data sets meeting THRESHLD status have their OSEs flagged as a "THRESHLD" OSE. When scheduling the OSEs to the same printer, OSEs flagged as "THRESHLD" are not marking as pending to the active printer for the job. This allows another printer device to select the same job and schedule the "THRESHLD" OSE. This allows large THRESHLD OSEs to print simultaneously on different printers.

Note: JES3 schedules one non-THRESHLD OSE together with a THRESHLD OSE.

11.37 DSISO specification

- Purge spool space when data set has printed
  - SYSOUT,CLASS=M,....,TYPE=(PRINT,DSISO)
- Track group allocation to large sysout data sets
  - SYSOUT,CLASS =M,.....,TRKGRPS=(9,9),..... ,TYPE=(PRINT,DSISO)
  - SYSOUT,CLASS =M,.....,TRKGRPS=(9,9),..... ,TYPE=(PRINT,DSISO),... ,THRESHLD=10000000
- Spin-off SYSOUT data sets
  - SPIN=(UNALLOC | NO) on DD statement
  - FREE=CLOSE on DD statement

DSISO specifications

When there are multiple large output data sets in a job, JES3 provides a method to delete the data sets as soon as they are printed. The default is to purge all the spool space used by a job when the job purges. The DSISO parameter allows data sets to be purged from the spool, freeing the spool space used before the job purges.

SYSOUT initialization statement

DSISO is specified by SYSOUT class in the SYSOUT initialization statement. TYPE=(PRINT,DSISO) must be specified. So if all large output data sets are in the same SYSOUT class, the spool space is purged when the data set finishes printing.

DSISO specifies that each data set in this class is to have its own track allocation table (TAT). This increases spool utilization because each data set can be purged when the data set processing is complete instead of when the job completes.

Note: This parameter does not apply to the JESMSGLG, JESJCL, or JESYSMSG data sets. If you specify the SPART or TRKGRPS parameter on this statement, you must also specify the DSISO parameter.

TRKGRPS parameter

When a job needs spool space for the first time, JES3 allocates one or more track groups to the job. You can specify how many track groups JES3 should allocate by using the TRKGRPS parameter on the MAINPROC, CLASS, a SYSOUT initialization statements. The person

submitting a job can specify the track group allocation size on the //*MAIN JES3 control statement. When a job uses up its first, or "primary", allocation of spool space, JES3 allocates more track groups. You can also specify, using the TRKGRPS parameter, how many units JES3 may allocate for each additional, or "secondary", allocation. JES3 continues allocating secondary quantities of spool space until the job needs no more space.

When large data sets are being created when a job is in execution, by SYSOUT class you can specify on the SYSOUT initialization statement (TRKGRPS=(9,9)) to send multiple track groups to the user memory from the JES3 address space.

> **Note:** If you specify the TRKGRPS parameter on the SYSOUT statement, you must also specify the DSISO parameter.
>
> The TRKGRPS parameter on this SYSOUT statement overrides corresponding values on the CLASS and MAINPROC initialization statements and on the //*MAIN JES3 control statement.

Spinoff SYSOUT data sets

The **SPIN** JCL parameter specifies that the output for a SYSOUT data set is to be made available for processing:

- Immediately upon unallocation - JCL DD SPIN=UNALLOC parameter or SPIN(UNALLOC) parameter for TSO/E ALLOCATE command.

  UNALLOC - Indicates that the system makes the data set available for processing immediately when the data set is unallocated. If you dynamically unallocate the SYSOUT data set, either explicitly or by specifying FREE=CLOSE, the system makes the data set available for processing immediately. If you do not dynamically unallocate it, the SYSOUT data set is unallocated at the end of the step, and the system will make it available for processing then.

- At the end of the job - SPIN=NO.

  NO - Indicates that the system makes the SYSOUT data set available for processing as a part of the output at the end of the job, regardless of when the data set is unallocated.

FREE option

The FREE JCL parameter specifies when the system is to unallocate the resources used for this DD statement's data set. The resources can be devices, volumes, or exclusive use of a data set.

Specifying FREE will not release the enqueue on the data set until the last step that requires the data set completes processing.

```
FREE=[END | CLOSE]
```

- **END** - Requests that the system unallocate the data set at the end of the last step that references the data set.
- **CLOSE** - Requests that the system unallocate the data set when it is closed.

If the DD statement specifies FREE=END and a DISP subparameter of PASS, the data set is not unallocated until the end of the job or until used for a later DD statement with a disposition of other than PASS. Do not specify FREE=CLOSE on a DD statement with a ddname of JOBLIB or STEPLIB; CLOSE is ignored.

If you specify SPIN=NO with FREE=CLOSE, the sysout data set will be unallocated, but not printed until the end of the job.

11.38 External writers, PSO interface, and SAPI

- Capability to process SYSOUT data sets to "External devices" -- PSO SSI Function Code 1
  - Devices not controlled by JES3
  - IBM supplied procedure - XWTR
    - XWTR processes output with **STDWTR** external writer name and destination **LOCAL** (as default)
  - User can write own external writers
  - Output to tape, DASD, printer, plotter, and punch
- SAPI Interface
  - SYSOUT Application Program Interface
  - SSI Function Code 79

External writers (PSO - Process SYSOUT) and SAPI interface

In general, external writer routines execute in an address space other than the JES3 address space. This type of writer is functionally independent of JES3 and operates as a completely separate MVS job. However, the external writer interacts with JES3, via the subsystem interface, to request data sets for processing. A subset of the output service scheduling function called Process SYSOUT and SYSOUT application program interface are invoked as a result of this kind of request.

No attempt is made by output service to schedule external writers as a result of constructing OSEs requiring their services; it is the responsibility of the operator to start external writers as required. JES3 posts started writers that are waiting for work when incoming jobs require processing.

IBM external writer

The IBM-supplied external writer is defined and invoked by the cataloged procedure named XWTR, which can serve as the base or a model for a procedure you would write for your own output writer. Start and modify commands for the IBM-supplied external writer:

```
S {XWTR|membername}[.identifier][,devicetype|,[/]devnum]
     [,volumeserial][,classes][,keyword=option[,keyword=option]...]

F [XWTR.|jobname.]identifier {,{CLASS|C}=[classes] }
        {,{DEST|D}=[LOCAL] |remote-workstation-name}
        {,{FORMS | F}=[forms-name] }
```

```
{,{JOBID | J}=[JOBnnnnn | Jnnnnn  ] |[STCnnnnn | Snnnnn ] |[TSUnnnnn | Tnnnnn ]}
{,{WRITER|W}=[STDWTR ] | user-writer-name }
```

User written external writers

If you want to add your own output writing routine to the external writer consider whether your routine will be needed more often than the IBM-supplied routine. If your routine will be invoked to write most of the external writer's output, you might want to replace the IBM-supplied with your own routine, so that your routine will be called by default. You can retain the IBM-supplied routine by renaming it to an alias.

Replacing the IBM-Supplied Routine: You can replace the IBM-supplied routine with your routine by:

- Renaming the IBM-supplied routine (IEFSD087) to an alias.
- Naming your routine IEFSD087
- Installing your routine in SYS1.LINKLIB

The external writer will call your routine by default. You can request the IBM-supplied routine by coding its alias on the SYSOUT= parameter. For example, where IBMWRITR is the alias you've given to the IBM-supplied routine.

```
//MYDATA  DD  SYSOUT=(H,IBMWRITR)
```

Note: Do not code STDWTR as a writer name. STDWTR and INTRDR are and are reserved for JES3 used as a parameters in the MVS operator's **MODIFY** command.

The JES3 ***F U** command assigns a external writer name STDWTR to a SYSOUT data set with the **NW=*** parameter specification. For example:

```
*F U Q=WTR CL=F NCL=G NQ=HOLD
*F U Q=HOLD CL=G NW=* ND=LOCAL
```

SAPI interface

Although both the SYSOUT Application Program Interface (SSI Function Code 79) and Process SYSOUT (SSI Function Code 1) allow applications to retrieve SYSOUT from JES spool using a variety of criteria, there are several important differences between the two function calls. IBM recommends that applications use the SAPI, as it is richer in function, as well as having better performance characteristics than the process SYSOUT call.

External writer/SAPI application routines execute in an address space other than the JES3 address space. This type of writer is functionally independent of JES3 and operates as a completely separate MVS job. However, the external writer/SAPI application interacts with JES3, through the subsystem interface, to request data sets for processing. A subset of the output service scheduling function called PROCESS SYSOUT and system application printer interface are invoked as a result of this kind of request.

No attempt is made by output service to schedule external writers/SAPI applications as a result of constructing OSEs requiring their services; it is the responsibility of the operator to start external writers as required. JES3 posts started writers that are waiting for work when incoming jobs require processing.

11.39 Process SYSOUT (PSO) interface

- Current users of PSO interface -- SSI Function Code 1
  - TSO/E OUTPUT command
  - External writer
  - Vendor products
  - IBM products
  - Customer written programs ....
- Application Program Interface (SAPI) -- SSI Function Code 79
  - IBM Infoprint Server (Extended mode)
  - Vendor products
  - Does not replace PSO

PSO interface

The process SYSOUT (PSO) is an interface to JES3 to allow access and control of SYSOUT data sets from other address spaces. It is used primarily by TSO OUTPUT and RECEIVE commands and external writers. Process SYSOUT (PSO) is an interface used to process the output on the writer and hold queues. The PSO interface allows applications to view output on the JES spool data sets before a device prints the output allowing the end user to eliminate any unwanted output. Any application using PSO will NOT be able to process IP addresses; the exception is a TSO user. JES3 is changed to ensure that a TSO user can process any SYSOUT, which is displayed to the user from the TSO **STATUS** command, by using the TSO **OUTPUT** command. The SYSOUT returned to the TSO user from the TSO **OUTPUT** command can contain an IP address. The TSO user is not informed that the SYSOUT contains an IP address and cannot select by using an IP address.

z/OS Infoprint Server

Infoprint Server is an optional feature of z/OS that uses z/OS UNIX System Services. This feature is the basis for a total print serving solution for the z/OS environment. It lets you consolidate your print workload from many servers onto a central z/OS print server. Infoprint Server delivers improved efficiency and lower overall printing cost with the flexibility for high-volume, high-speed printing from anywhere in the network. With Infoprint Server, you can reduce the overall cost of printing while improving manageability, data retrievability, and usability. z/OS Infoprint Server lets you consolidate your print workload from many servers to a central z/OS print server. z/OS handles high volume printing, logs accounting histories, and provides data security and recovery. With z/OS, you can control host and LAN printing. You do not need to continually upgrade your server so it can handle more print requests.

11.40 SAPI overview

- JES3 as the print SERVER of choice
- Multiple SAPI sessions
  - Provide applications ability to obtain SYSOUT
    - Many clients at same time
    - From a single address space to JES3
- SSI code to access JES3 SYSOUT - 79
  - More SSOB fields than PSO allowed

SAPI overview

The SYSOUT Application Program Interface (SSI function code 79) allows JES3 to function as a server for applications needing to process SYSOUT data sets residing on JES3 spool. Use of the SAPI SSI call allows a user-supplied program to access JES3 SYSOUT data sets independently from the normal JES3-provided functions (such as print or network). Users of this function are application programs operating in address spaces external to JES3. SAPI supports multiple, concurrent requests from the applications' address spaces. Each issuer of the IEFSSREQ macro is referred to as an "application thread."

SAPI and PSO comparison

Although both the SYSOUT Application Program Interface (SSI Function Code 79) and Process SYSOUT (SSI Function Code 1) allow applications to retrieve SYSOUT from the JES3 spool using a variety of criteria, there are several important differences between the two function calls. IBM recommends that applications use the SAPI, as it is richer in function, as well as having better performance characteristics than the Process SYSOUT call. Some of the differences that SAPI provides are:

- The ability to multi-task data set selection and processing calls from within an application.
- A richer selection criteria, including the use of wildcard characters for attributes.
- A greater number of SYSOUT data set characteristics returned to the application than does Process SYSOUT.
- The application has the ability to retrieve information contained in the scheduler work blocks (SWBs)
- A greater degree of modification ability of selected SYSOUT data sets.
- A count facility that Process SYSOUT does not provide.

11.41 SSI function code 79

- Access JES3 SYSOUT on spool
  - IEFSSREQ - SSOB extension (IAZSSS2)
  - Multiple requests per address space
- An application thread can make three types of requests with SAPI
  - SAPI PUT/GET request
    - Information contained whitin SWBs can also be returned
    - SJCREQ or SWBTUREQ REQUEST=RETRIEVE
  - SAPI COUNT request
  - SAPI BULK MODIFY request

SSI function code 79

There is no need to migrate existing applications that use PSO (SSI function code 1). If you choose to exploit the new facilities of System Application Printer Interface (SAPI), you must follow these guidelines in converting your applications from using PSO to System Application Printer Interface:

- Specify the IAZSSS2 (SSS2) mapping macro wherever the IEFSSSO (SSSO) mapping macro was previously specified.
- Ensure GETMAIN/FREEMAIN processing obtains enough storage for the IAZSSS2 macro included in the SSOB extension.
- Initialize the entire IAZSSS2 mapping macro with zeros. Fill in eye catcher, length, version, and type fields.
- Change all references of SSSO to SSS2.

  In many cases, the names of the fields from the IEFSSSO to the IAZSSS2 have the SSSO prefix changed to the SSS2 prefix. However, fields in the SSSO that are both input and output fields have been separated into two, distinct, fields in the new SSS2. Examine all references to the SSSO making sure that the appropriate fields in the SSS2 are being used to provide the same function that your application expects.

SAPI request types

An application thread can make three types of requests with SAPI. Each is independent of, and mutually exclusive with the others.The IAZSSS2 is the SSOB extension for SAPI. Field

SSS2TYPE indicates which of these three possible types of requests the application thread is issuing:

- SSS2PUGE - indicates a SAPI PUT/GET request
- SSS2COUN - indicates a SAPI COUNT request
- SSS2BULK - indicates a SAPI BULK MODIFY request

PUT/GET request

Initiates data set selection, and optionally can provide disposition processing for the data set returned in the previous SAPI PUT/GET call. PUT/GET request processing occurs when an application thread issues the IEFSSREQ macro to initiate data set selection. The input SSOB and SSS2 control blocks, provided by the application thread, specifies the selection criteria used to select a data set. The application thread can use a wide variety of selection criteria to select a SYSOUT data set to be processed.

Information contained within the SYSOUT data set's scheduler work blocks (SWBs) can also be returned to the application thread. Much of the information contained within the SWB is normally not processed by JES3, and therefore much more information about the data set can be retrieved from the SWB than is returned in fields of the SSS2. Examples of such information contained within the SWB are NAME, BUILDING, ADDRESS, and so on.

The application thread needing to retrieve this SWB information, sets either SSS2FSWB or SSS2FSWT in flag byte SSS2MSC1 when issuing a PUT/GET request. The setting of SSS2FSWB implies SSS2FSWT processing as well. JES3 then provides the application thread the information that can be used when the application thread invokes the SJF services to retrieve this SWB information. These services are either SJFREQ REQUEST=RETRIEVE or SWBTUREQ REQUEST=RETRIEVE.

COUNT request

Returns the count of entries that can be scheduled without returning a particular data set. JES3 counts the number of schedulable elements (OSEs) matching the input selection criteria and returns the count to the application thread in field SSS2CDS. An application thread does not receive a data set in the SAPI COUNT call. Included in the information returned are the total byte count, record count, line count, and page count.

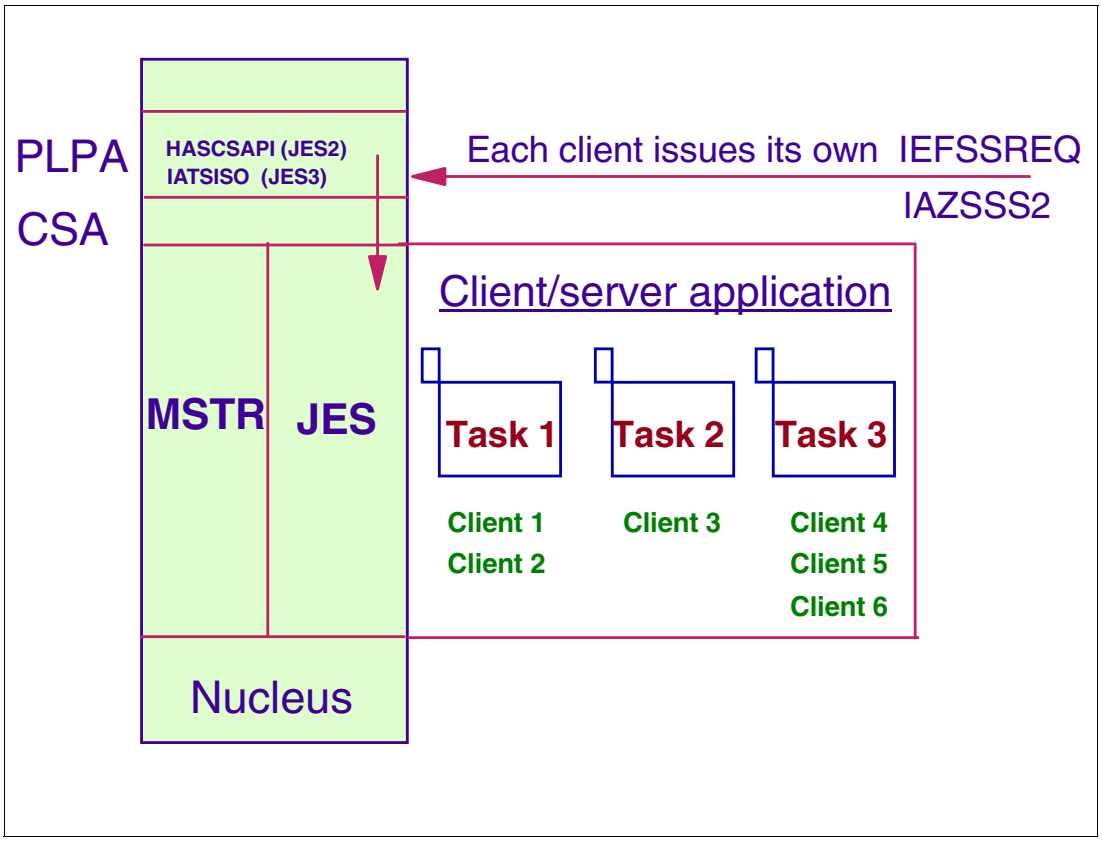
BULK MODIFY request

This request is used to modify selected attributes of one or more data sets. With a BULK MODIFY request, the application thread can select SYSOUT data set(s) for modifications. Modification of data sets matching the input selection criteria occurs with the setting of information in flag byte SSS2UFLG.

- SSS2SETC class update - The class of each data set is changed to the specified class in the SSS2CLAS field.
- SSS2DELC delete processing - Each data set is deleted.
- SSS2ROUT destination update - The destination of each data set is changed to the specified destination in the SSS2DES2 field.
- SSS2RLSE release processing - Each data set is moved to the WRITER queue in JES3. Release processing is applicable only to data sets on the JES3 Output Service HOLD queue.

  Processing for a BULK MODIFY request occurs for each data set matching the application thread's selection criteria. It is important to understand job boundaries can be crossed.

11.42 Multiple requests per address space



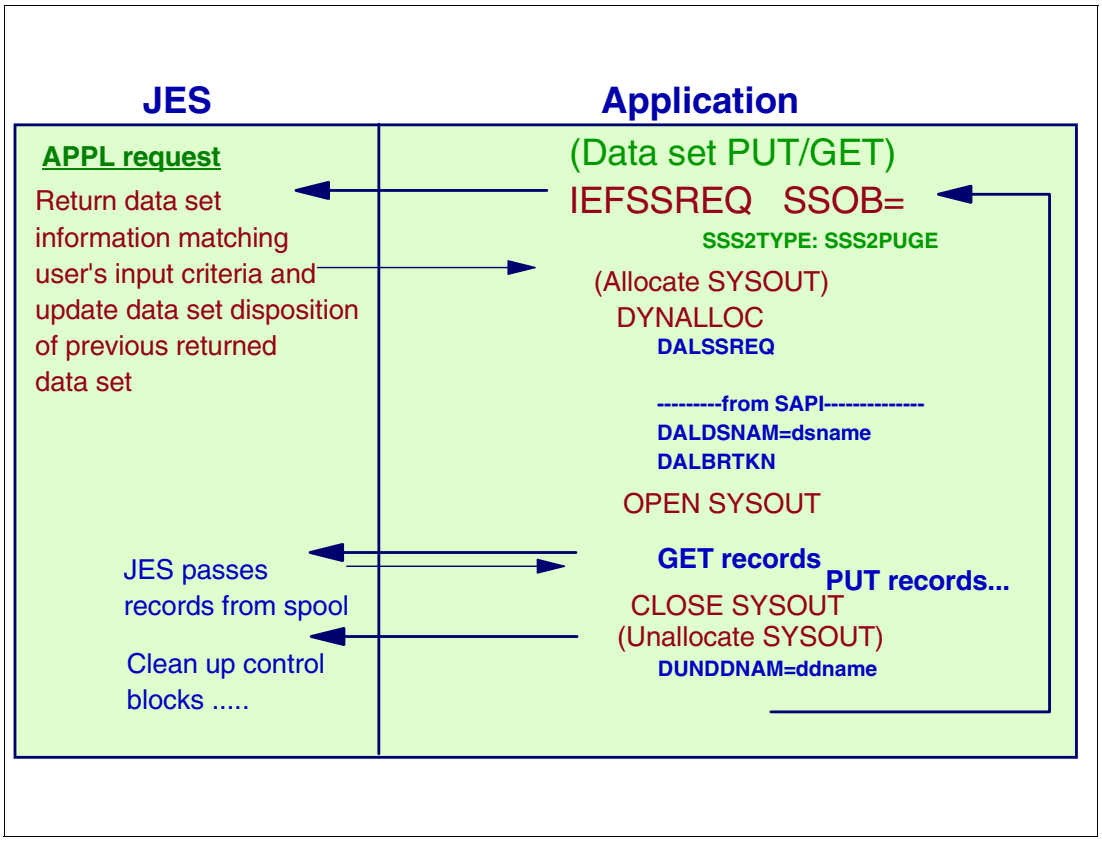
SAPI requests

SAPI supports multiple, concurrent requests from the applications' address spaces. Each issuer of the IEFSSREQ macro is referred to as an "application thread."

Multiple clients

Each unique SSOB/SSS2 pair supplied as input on the IEFSSREQ request is viewed as a separate thread by JES3. You can multi-task these requests within your application's address space, or even issue multiple IEFSSREQ requests (supplying different SSOB/SSS2 pairs) from within a single task in your application's address space. A task that issues the original IEFSSREQ can transfer the SSOB/SSS2 control block pair to another task within your address space for subsequent IEFSSREQ requests. However, if this is done and the originating task (which JES3 considers to be the owner of that specific thread) fails, then JES3 cleanup occurs for resources associated with that SSOB/SSS2 pair. If the transferred task attempts to issue another IEFSSREQ with that same SSOB/SSS2 pair after such a termination occurs, incorrect processing occurs because JES3 has disconnected from that SSOB/SSS2 pair.

11.43 Using the SAPI interface



Using the SAPI interface

An application thread can make three types of requests with SAPI. Each is independent of, and mutually exclusive with the others. Field SSS2TYPE indicates which of these three possible types of requests the application thread is issuing:

- SSS2PUGE - indicates a SAPI PUT/GET request
- SSS2COUN - indicates a SAPI COUNT request
- SSS2BULK - indicates a SAPI BULK MODIFY request

PUT/GET request

This request initiates data set selection, and optionally can provide disposition processing for the data set returned in the previous SAPI PUT/GET call.

PUT/GET request processing occurs when an application thread issues the IEFSSREQ macro to initiate data set selection. The input SSOB and SSS2 control blocks, provided by the application thread, specifies the selection criteria used to select a data set. The application thread can use a wide variety of selection criteria to select a SYSOUT data set to be processed.

Once the application thread receives a data set from the JES3, you must allocate (through a dynamic allocation with the data set name that is returned from SSS2DSN) the data set to process it. During this allocation, dynamic allocation requires DALBRTKN text unit. JES3 performs the initialization of this text unit. The application thread must move the address from field SSS2BTOK into a text unit pointer field for the JES3-provided DALBRTKN text unit. The

actual processing of the SYSOUT data set depends upon your specific application. After your application thread has completed processing of the data set, it then unallocates the data set with the text unit of DUNDDNAM specifying the DDNAME of the returned data set from the original allocation. The allocation/unallocation of the data set must occur once per returned data set.

The PUT processing occurs when the application thread subsequently issues a following IEFSSREQ macro to select another data set. You can use fields in the optional disposition section of the SSS2 to change certain attributes of the previously obtained data set from the prior IEFSSREQ call.

DALBRTKN text unit

Before allocating the returned data set from an IEFSSREQ request, create a DALBRTKN text unit for the allocation request and move the contents of field SSS2BTOK into the text unit pointer for DALBRTKN.

Dynamic allocation

Once the application thread receives a data set from the JES3, you must allocate (through a dynamic allocation with the data set name that is returned from SSS2DSN) the data set to process it. During this allocation, dynamic allocation requires DALBRTKN text unit. JES3 performs the initialization of this text unit. The application thread must move the address from field SSS2BTOK into a text unit pointer field for the JES3-provided DALBRTKN text unit. The actual processing of the SYSOUT data set depends upon your specific application.

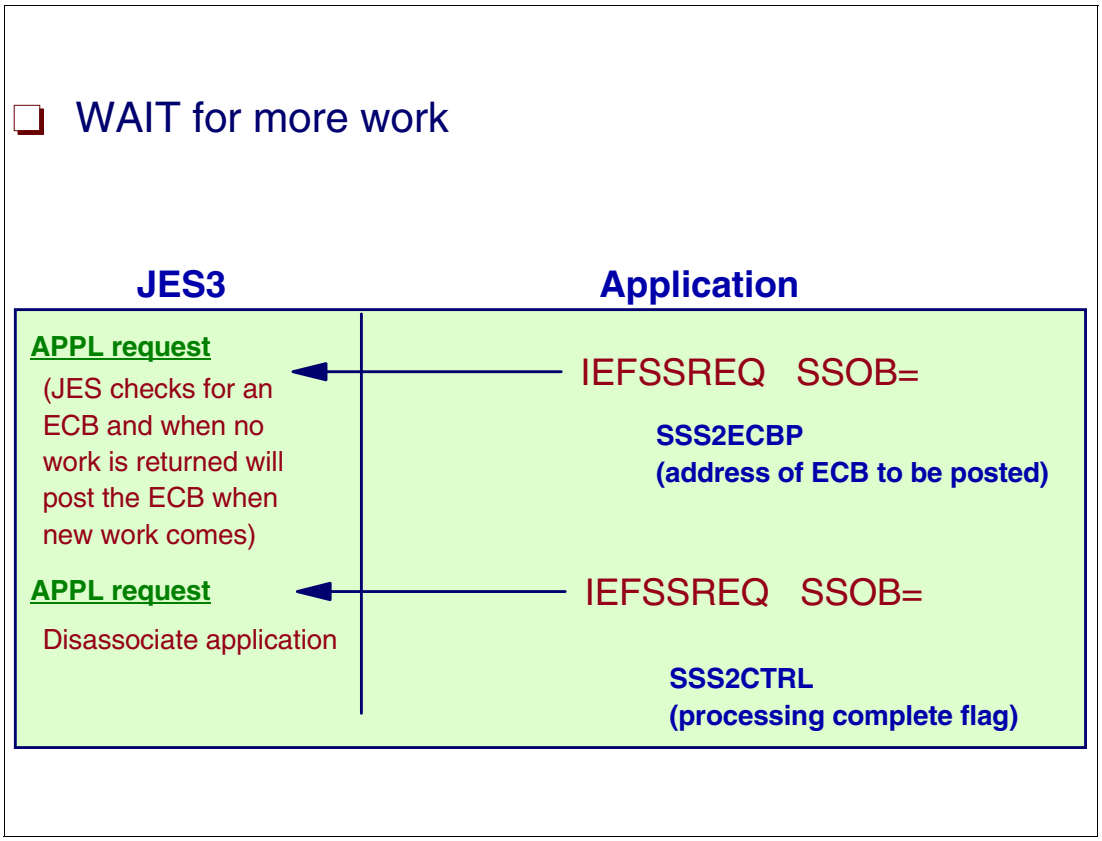
Unallocate SYSOUT

After your application thread has completed processing of the data set, it then unallocates the data set with the text unit of DUNDDNAM specifying the DDNAME of the returned data set from the original allocation. The allocation/unallocation of the data set must occur once per returned data set.

An EOD (SSOBRETN=SSS2EODS) response is a possible return only for PUT/GET processing. When SAPI returns SSS2EODS to the application thread, the application thread can do one of the following:

- Wait on its supplied ECB for a post from JES3. This post indicates SYSOUT has just been generated that contains characteristics matching the application thread's selection criteria. See “SAPI enhanced functions” on page 600.
- The application can then issue another IEFSSREQ to obtain this data set from the JES3. Since multiple applications can be posted from the single piece of work appearing on the queue, there is no guarantee that once posted, a thread will not receive an immediate SSS2EODS return again (that is, another thread received the work).
- Issue another IEFSSREQ request after changing its selection criteria.
- Issue another IEFSSREQ request with the SSS2CTRL flag set indicating the application thread is terminating. See “SSS2CTRL processing flag” on page 600.
- Issue a COUNT request.
- Issue a BULK MODIFY request.

11.44 SAPI enhanced functions



Enhanced functions with SAPI

Provide a pointer, in field SSS2ECBP, to the ECB to be posted when EOD is returned to the application if you need to be posted when new work becomes available that satisfies the current selection criteria.

The application thread must provide a pointer to an ECB in field SSS2ECBP if the application thread wants JES3 to post it when newly created work has characteristics matching the thread's selection criteria. This occurs after JES3 returns SSS2EODS for a PUT/GET request. If an ECB is not supplied, it is the responsibility of the thread to initiate an IEFSSREQ request.

SSS2CTRL processing flag

JES3 provides a minimum amount of input validity checking of an input SSS2 before a final call (SSS2CTRL) is processed. This validity checking includes:

- Ensures a valid SSS2 eye catcher is present
- Ensures a valid version number is present
- Ensures a valid request type is present
- Ensures a valid length is present
- Ensures a valid disposition, if applicable, is present

11.45 Wildcard supported Fields

- Wildcard values
  - * - Multiple characters
  - ? - For a single character
- Selected fields
  - SSS2JOBN - job name
  - SSS2CREA - Creator userid
  - SSS2PRMO - Process modes (1-4)
  - SSS2DEST - Destination
  - SSS2PGMN - User writer name
  - SSS2FORM - Form number (1-8)

Wildcard support

The SSI request returns information for matching the pattern specified by the selected fields. The selected fields are used as filters to select output data sets that match the specified fields. The pattern can contain the following wildcard characters:

- An asterisk ('*') -- matches zero or more characters
- A question mark ('?') -- matches one character.

Wildcards are valid for the following SSS2 selection fields:

- SSS2JOBN (Job Name)
- SSS2CREA (Owning Userid)
- SSS2PRMO (Process Modes)
- SSS2DEST (Destination)
- SSS2PGMN (User Writer Name)
- SSS2FORM (Form Numbers)

11.46 Output functional subsystem (FSS)

- Functional subsystem - FSS
  - Program residing in address space
    - Separate from JES
    - Started by JES as started task
    - Communicates with JES
  - Current existing IBM writer FSS programs
    - Print Services Facility (PSF)
    - Infoprint Server (Basic mode)

Functional subsystem (FSS)

A functional subsystem (FSS) is a collection of programs residing in an address space separate from JES3 that communicates with JES3 to provide a JES3-related function, such as print processing. An FSS extends the scope of JES3 processing. Because an FSS operates in its own address space, it functions independently of JES3 in several areas. A FSS is responsible for:

- The management of storage resources that it needs during data set processing including print buffers.
- Its own recovery and serviceability.
- Its performance and accounting measurements.
- The security of its own resources.

JES3 FSS support

An FSS is dependent on JES3 for control and services. JES3 manages an FSS in the following ways:

- The FSS is defined during JES3 initialization using JES3 initialization statements and parameters.
- JES3 initiates the FSS address space.
- JES3 provides services for use by the FSS. FSS messages sent to the JES3 operator are in a format chosen by the writer of the FSS.
- JES3 controls its own resources, such as the job queues and spool.

- JES3 controls output scheduling for FSS-controlled devices. The FSS application does not control selection criteria when acquiring data sets for print processing. JES3 uses its own work selection criteria to provide the proper data sets to the FSS application.
- JES3 coordinates the termination and restart of the FSS.

Print Services Facility

Certain printers, such as page printers that are driven by Print Service Facility (PSF), run under an address space called an output writer functional subsystem (FSS). Running a printer under the control of an FSS lets the FSA implement functions that the JES3 global address space does not support. Print Services Facility™ (PSF) for z/OS is an IBM licensed printer-driver program that manages and controls data transmitted to Advanced Function Presentation (AFP) printers that are channel-attached, SNA-attached, or TCP/IP-attached.

Inforprint Server

Infoprint Server is an optional feature of z/OS that uses z/OS UNIX System Services. This feature is the basis for a total print serving solution for the z/OS environment. It lets you consolidate your print workload from many servers onto a central z/OS print server.

Some highlights of Infoprint Server are:

- **IP PrintWay™** - Provides fast access to TCP/IP-connected printers and to Virtual Telecommunications Access Method (VTAM)-controlled printers. You can run either IP PrintWay basic mode or IP PrintWay extended mode:
  - IP PrintWay basic mode, the original mode of operation, uses the z/OS Functional Subsystem Interface (FSI) to obtain output data sets from the JES3 spool. IBM does not plan additional enhancements to IP PrintWay basic mode.
  - IP PrintWay extended mode uses the z/OS Sysout Application Programming Interface (SAPI) to obtain output data sets from the JES3 spool. It provides better performance, improved usability, and more function than IP PrintWay basic mode.
- **NetSpool** - Automatically directs VTAM application data to the job entry subsystem (JES3) spool without requiring application changes.
- **Print Interface** - Receives print requests from clients that run on remote systems, such as Windows and UNIX systems, and directs data to the JES3 spool. Accepts data in a variety of formats, including: Advanced Function Presentation (AFP), plain text, Printer Control Language (PCL), Portable Document Format (PDF), and PostScript formats. Also provides commands that let you print from z/OS UNIX System Services.
- **Infoprint Central** - Lets help desk operators and other authorized users or job submitters work with print jobs, printers, and NetSpool logical units (LUs); display printer definitions; and check system status. Infoprint Central is a Web-based print management system.

Defining an output writer functional subsystem

You can define one or more printers to run under the control of a single output writer FSS or you can allow JES3 to define a unique FSS for each printer by default.

You can also select the main on which you want an output writer FSS to operate by coding the SYSTEM= keyword on the FSSDEF initialization statement.

Include a DEVICE statement for each printer that you want to run under this FSS. You must include the FSSNAME= keyword on the DEVICE statement that specifies the same name that you code on the FSSNAME= keyword of the FSSDEF initialization statement.

11.47 FSS address space implementation

- JES function dependent code - FSS
- Functional subsystem application - FSA
  - Function dependent code to control devices
  - Multiple devices (FSAs) per address space
  - Communicates with JES2/JES3 via - FSI
    - Functional Subsystem Interface - FSI

FSS address space components

The code in the JES3 and functional subsystem address spaces consist of the functional subsystem address space (FSS) services and the functional subsystem interface (FSI) services

Functional subsystem application

A functional subsystem application (FSA) is a collection of programs residing in the FSS address space that control one device. There can be multiple FSAs per FSS. IBM recommends that each of the FSAs for the FSS be a separate task. The FSA can be thought of as a logical subset of the FSS and is the lowest level of connection with JES3. The functional subsystem application (FSA) services

FSS The JES3 code that exists in the FSS address space allows the application code (FSA) to communicate with the JES3 address space on the global.

FSA The functional subsystem application (FSA) is contained within a functional subsystem address space and handles a discrete piece of work within that FSS--such as driving an output device. Since more than one writer FSA can exist in one FSS, one FSS is capable of controlling more than one output device.

FSI The functional subsystem interface (FSI) provides the actual communication facility between JES3 and the functional subsystem application. The control block structure to support the functional subsystem interface is built by the subsystem interface called CONNECT. FSI services are actually JES3 and FSS/FSA supplied routines that allow interaction between JES3 and the FSS/FSA. FSI services fall into three categories; communication services, data access services, and control services.

11.48 Functional subsystem interface - FSI

- Formally defined interface
- Communication between FSS and FSA
  - Between function dependent code and JES code
    - In the FSS address space
- Method of communication
  - Via FSIREQ macro

FSI interface

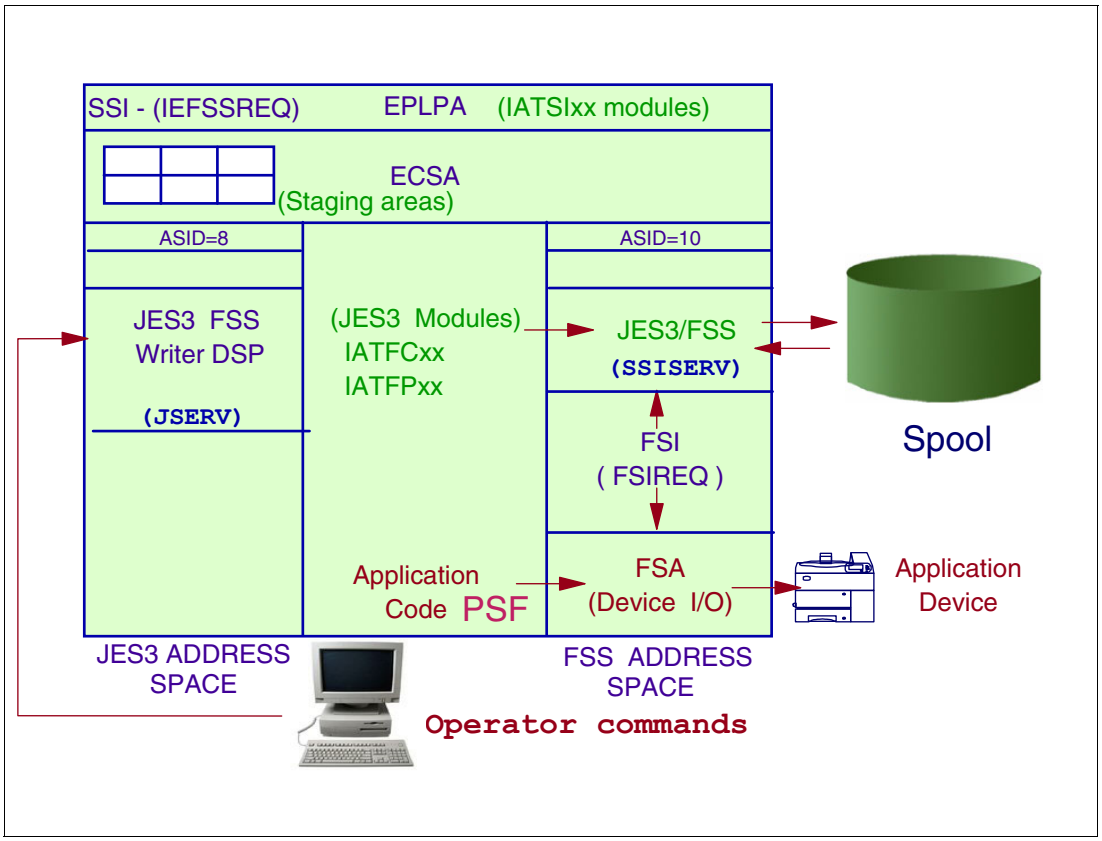
The functional subsystem interface provides the actual communication facility between JES3 and the functional subsystem application. The control block structure to support the functional subsystem interface is built by the subsystem interface called CONNECT. There are three forms of communication that occur between an SSI address space and an FSS address space:

- CONNECT/DISCONNECT requests, which use MVS SSI.
- FSS-to-JES3 communication, which involves communication to the JES3 global address space (through GETDS, RELDS, and SEND) and communication strictly within the functional subsystem FSS and FSA (through GETREC, FREEREC, and CHKPT).
- Global JES3-to-the functional subsystem communication, which involves both the ORDER and POST commands. This is the only instance where a global JES3 address space initiates communication with a non-JES3 address space.

FSIREQ macro

Like the FSS address spaces and JES3 global address space, the function-dependent code and JES3 code in an FSS need a way to request information from each other. The JES3 code and the function-dependent code communicate by using the functional subsystem interface macro, FSIREQ. The FSS and FSA both use the FSIREQ macro to request a service from the JES3 code in the FSS address space. The JES3 code uses the FSIREQ macro to request the FSS or FSA to perform a function. The FSI parameter list (FSIP) is used to communicate the type of request.

11.49 WTR FSS communication



WTR FSS communication

The diagram shows the JES3 address space with the writer FCT and the FSS address space with its components consisting of JES3 modules for the FSS/FSA, PSF code, and the FSI used for communication between PSF and FSS/FSA. Function code 53 via the IEFSSREQ macro is used to connect and disconnect the FSS address space.

JES3 to WTR FSS

Communication between the JES3 address space and the WTR FSS address space takes place via the Subsystem Interface, SSI. There are three levels of communication:

1. Communication that takes place through the MVS SSI.
2. GLOBAL JES3 communication to the FSS address space.
3. FSS address space communication to the JES3 GLOBAL address space.

JES3 communication to the FSS

The WTR FCT communicates to the WTR FSS address space by using the JSERV macro. The request is placed into a staging area and sent to either the FSA or FSS listen task TCB in the WTR FSS address space.

The writer application code, PSF, initiates requests via the FSIREQ macro. This macro request invokes the JES3 code in the FSS address space and when the request needs to be passed to GLOBAL JES3, the SSISERV macro is used to send the request via a staging area to the JES3 address space to the dest queue entry 152 (Dynamic Dest Queue).

FSIREQ communication

JES3 communicates with a FSS address space using the same macros that are used to communicate with any user memory. The FSIREQ macro is used by PSF to initiate requests to JES3. Like the FSS address spaces and JES3 global address space, the function-dependent code and JES3 code in an FSS need a way to request information from each other. The JES3 code and the function-dependent code communicate by using the functional subsystem interface macro, FSIREQ. The FSS and FSA both use the FSIREQ macro to request a service from the JES3 code in the FSS address space. The JES3 code uses the FSIREQ macro to request the FSS or FSA to perform a function. The FSI parameter list (FSIP) is used to communicate the type of request.

FSA communication

The JES3 modules in the WTR FSS address space read the spool from the FSS address space. Pointers to the SYSOUT files are passed to the FSS address space by GLOBAL JES3 when a data set is selected for printing.

PSF writes directly to the printers from the FSS address space.

11.50 WTR FSSDEF statement

```
SSDEF,FSSNAME = fssname
      ,TYPE = WTR
      ,PNAME = procname
      ,SYSTEM = (sysname | sysnamei,sysnamej,...)
      ,TERM = (YES | NO)

DEVICE ,FSSNAME = fssname
      ,DTYPE = PRT1
      ,MODE = (COMP | FSS)
      ,CKPNTPG = nnnnnn
      ,CKPNTSEC = nnnnnn
      ,NPRO = (NO | STANDARD | nnnn)
      ,PAGELIM = nnnnnn | nnnnnn+ | 0+ | NO
      ,PM = (LINE | PAGE | (LINE,PAGE))

 OUTSERV, .... WS = (PM   )
```

FSSDEF statement

An FSS can be defined by an FSSDEF statement during a warm, cold start, and hot start refresh. The FSSDEF statement defines the name and type of FSS. Use the FSSDEF statement to define the characteristics of a functional subsystem (FSS) which operates in its own address space. Use a FSSDEF statement for either of the following:

- To define one or more output writer FSSs for printers that you define to run in FSS mode (via the DEVICE initialization statement). You can define more than one printer to run under the control of a single output writer FSS. If you do not define an output writer FSS for each printer that requires one, JES3 creates an FSS using default values.
- The SYSTEM keyword specifies the JES3 main on which the FSS is to operate. The name(s) must be the same as specified on the NAME parameter of the MAINPROC statement for the main.
- The TERM keyword specifies whether or not the FSS terminates if the JES3 global terminates as the result of an *RETURN or *DUMP operator command.

  Use TERM=YES option after an orderly shutdown of JES3. This will incorporate the changes you make in the DYNALLOC, HWSNAME, CIPARM, RESDSN, and SYSOUT initialization statements. Restarting JES3 will establish the new changes.

DEVICE statement for FSS

FSSNAME Specifies the name of a WTR FSS for which this printer is to be associated.

MODE Specifies which mode the printer operates in, compatibility or FSS WTR. This is the initial setting for the printer and can be changed by the operator.

> **Note:** If the FSSNAME parameter is specified, the default mode is FSS. Otherwise the default is COMP.

DTYPE Specifies the type of printer.

PM Specifies the process mode for any printer. For this parameter to be used when scheduling output to the printer, a new writer selection parameter, WS=PM, has been established and it must also be in use for the printer. This WS=PM parameter is on the OUTSERV initialization statement or can be specified by the operator.

- LINE - specifies that line mode data may be scheduled to the device.
- PAGE - specifies that page mode data may be scheduled to the device.
- (LINE,PAGE) - specifies that either line mode data or page mode data may be scheduled to the device.

PAGELIM This parameter is to be used in the same fashion as the LINELIM parameter. It is used for writer selection of a data set to any type of printer.

- nnnnnn - specifies a maximum number of pages for a data set for selection for output processing.
- nnnnnn+ - specifies a minimum number of pages for a data set for selection for output processing.

CKPNTSEC nnnnnn - specifies an integer from 1 to 32767 as a number of seconds. This is a new way of specifying a data set checkpoint interval. This parameter is for the 3800-3 only.

CKPNTPG nnnnnn - specifies an integer from 1 to 32767 as a number of pages. This is a new way of specifying a data set checkpoint interval. This parameter cannot be specified if CKPNTSEC is specified.

> **Note:** If neither the CKPNTSEC or CKPNTPG parameters are specified, the default is CKPNTPG=200.

NPRO nnnn - specifies a 1 to 4 digit value in seconds. This value will be used as a delay before forcing out pages of data to the data integrity point, (DIP). The value can range from 0 to 9999. Specifying 0 seconds indicates that the run-out is to be immediate.

- STANDARD - indicates to use the value specified on the OUTSERV statement.
- NO - specifies that this device will not use the run-out feature.

Dynamic definitions for FSS and devices

The `*F CONFIG` command makes configuration changes dynamically to JES3. The command is equivalent to adding the following initialization statements to the initialization stream for non-channel attached FSS printers:

- FSSDEF - Functional Subsystem Definition
- DEVICE - I/O Device Definition

11.51 Sample PSF procedure

```
//PRINTW43 PROC
//STEP01   EXEC PGM=APSPPIEP,REGION=4096K,TIME=1440,PARM=(,,,,TCPIP,UNIC
//JOBHDR   OUTPUT PAGEDEF=V06483,      /* JOB HEADER PAGE          */
//        FORMDEF=A10120,CHARS=60D8    /*   FORMDEF: ALTERNATIVE BIN*/
//JOBTLR   OUTPUT PAGEDEF=V06483,      /* JOB TRAILER PAGE         */
//        FORMDEF=A10110,CHARS=60D8    /*   FORMDEF: MAIN BIN      */
//DSHDR    OUTPUT PAGEDEF=V06483,      /* DATA SET SEPARATOR       */
//        FORMDEF=A10110,CHARS=60D8    /*   FORMDEF: MAIN BIN      */
//MSGDS    OUTPUT PAGEDEF=A08682,      /*
//        FORMDEF=A10110,CHARS=60D8    /*
//PRTINFO DD DSN=SAMPPROC.PRTINFO,DISP=SHR
//FONT01  DD DSN=SYS1.FONTLIBB,DISP=SHR  /*
//       DD DSN=SYS1.FONTOLN,DISP=SHR   /*
//       DD DSN=INST.FONTLIB,DISP=SHR   /*
//PRT1     CNTL
//PRT1     PRINTDEV FONTDD=*.FONT01, /* DEFAULT FONT     LIBRARY DD  */
//         OVLYDD=*.OLAY01,        /* DEFAULT OVERLAY  LIBRARY DD  */
//         PSEGDD=*.PSEG01,        /* DEFAULT SEGMENT  LIBRARY DD  */
//         PDEFDD=*.PDEF01,        /* PAGEDEF  LIBRARY DD          */
//         FDEFDD=*.FDEF01,        /* FORMDEF  LIBRARY DD          */
//         FONTPATH=*.DOF1,        /* DATA OBJECT FONT LIBRARY DD  */
//         JOBHDR=*.JOBHDR,        /* JOB HEADER SEPARATOR OUTPUT  */
//         JOBTRLR=*.JOBTLR,       /* JOB TRAILER SEPARATOR OUTPUT */
//         DSHDR=*.DSHDR,          /* DATA SET HEADER SEPARATOR    */
//PRT1       ENDCNTL
```

Sample PSF procedure

Before starting a PSF FSS, you must have a cataloged startup procedure in a system procedure library, such as SYS1.PROCLIB. This procedure specifies PSF initialization parameters and libraries that contain system and installation resources. The name of this procedure can be specified in the FSSDEF statement of the JES3 initialization deck. If the name is omitted, JES3 supplies a default, chosen for the defined printer.

Several PSF startup procedures are supplied with PSF. You can modify the startup procedures supplied with PSF or write your own startup. The startup procedure can also specify defaults that cannot be set with JES initialization statements for printer FSA definitions.

PSF procedure

The figure shows a sample startup procedure for one FSSs, each with a single printer FSA. PRT1 is a channel-attached printer that is defined via a DEVICE statement in the JES3 initialization statements.

11.52 Starting an FSS writer

- Operator calls a hot WTR
  - *V PRT1,ON
  - *X WTR,OUT=PRT1
- JES3 issues an internal MVS start command with token
  - S PROCNAME.FSSNAME,,,(SSID,FSSID,PARMSTRING)
  - S PRINTW43.fssname,,,(JES3,1,,42) 1
- Start messages to operator

IAT6100 (DEMSEL) JOB PRINTW43 (JOB46346), PRTY=15, ID=userID
IAT7035 DEVICE PRT1 ( ) ACTIVE ON SC43 USING FSS fssname

Starting a FSS writer

To start an FSS writer, the operator should call a hot writer using a JES3 command. An output writer FSS, which supports specific printer devices, is started by JES3 in response to one of the following:

- A **`*X WTR,OUT=device`** command is entered that specifies an output device controlled by an FSS (hot writer).
- An output device controlled by an FSS becomes available and there is eligible work (dynamic writer).

MVS internal start command

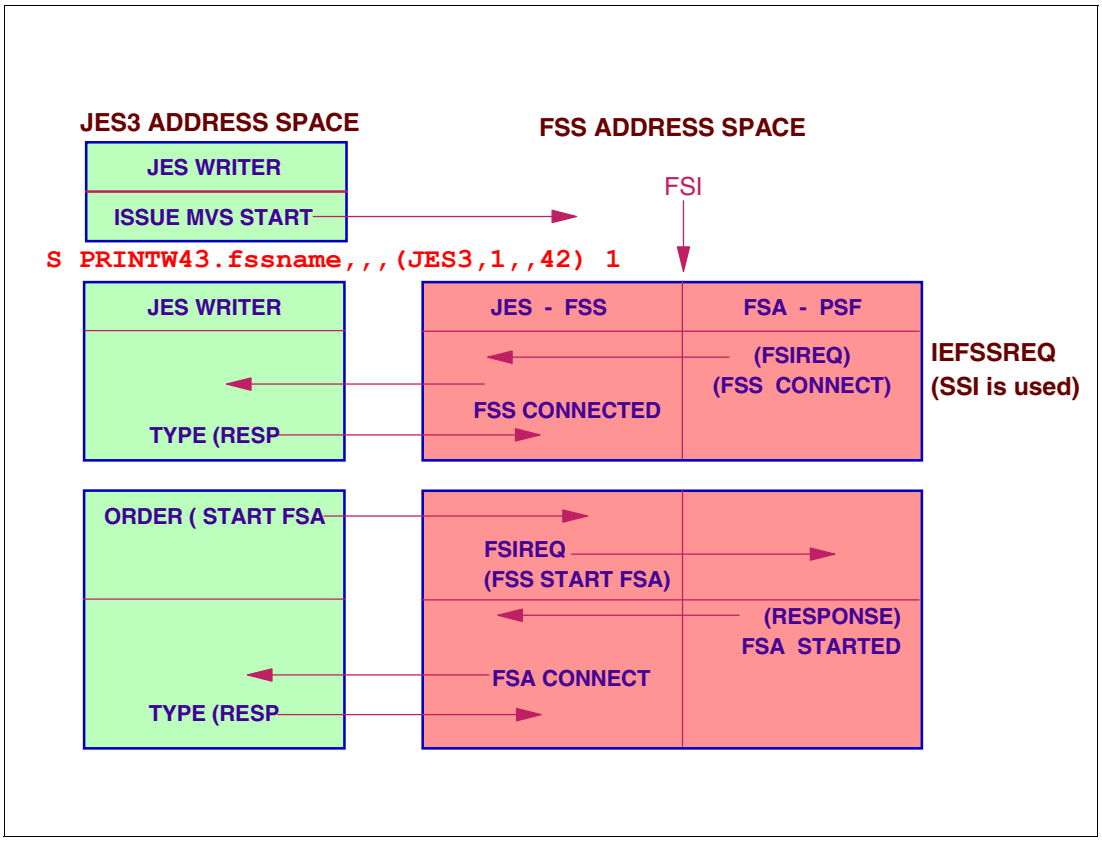
The operator does not see the MVS **`start`** command that JES3 issues to start the FSS address space. However, the operator does see the successful completion of the FSS start. The “1” specified on the start command indicates the number or printers that can execute in the FSS address space.

```
S PROCNAME.FSSNAME,,,(SSID,FSSID,PARMSTRING)
S PRINTW43.fssname,,,(JES3,1,,42)  1

IAT6100 ( DEMSEL ) JOB PRINTW43 (JOB46346), PRTY=15, ID=userID
IAT7035 DEVICE PRT1     (    ) ACTIVE ON SC43 USING FSS fssname
```

No operator dialog is required to initialize an output writer FSS. Special output writer functional subsystem DSPs are initialized to communicate with the FSS driver module using the functional subsystem interface. When initialization completes, the FSS driver is activated.

11.53 Starting an FSS writer address space



Start the FSS address space

The operator issues a ***X WTR,OUT=PRT1** command. This activates an FCT that issues the MVS START command.

```
S PRINTW43.fssname,,,(JES3,1,,42) 1
```

This command starts the FSS address space and the IAZSFSS load module gets control. When the JES operator issues the command to start an FSS printer device, JES determines if the FSS for which the printer is defined is currently active. If that FSS is not currently active, JES starts the FSS. Immediately after JES receives a response for the START FSS order from the FSS, JES issues a START FSA order for each FSA defined to that FSS. If the FSS is currently active, JES converts the command into a START FSA order to start the printer device.

WTR address space

If the printer is successfully allocated and initialization is complete, the FSA issues an FSIREQ CONNECT request to JES to establish the FSA-level functional subsystem interface (FSI). FSA CONNECT processing:

- Notifies JES that the FSA has successfully started.
- Identifies to the FSI the addresses of FSA routines that are to receive control when JES issues the FSIREQ macro.
- Identifies to the FSI the addresses of JES routines that are to receive control when the FSA issues the FSIREQ macro.

An FSIREQ for FSA-CONNECT is issued to the JES3 by module IAZSFSS. The results of the FSA-CONNECT are verified. JES3 then issues an ORDER to start the FSA. In the FSS address space, IAZSFSS attaches a subtask for the application or FSA and module IAZSFSA gets control and initializes the FSA and responds back to the JES3 address space.

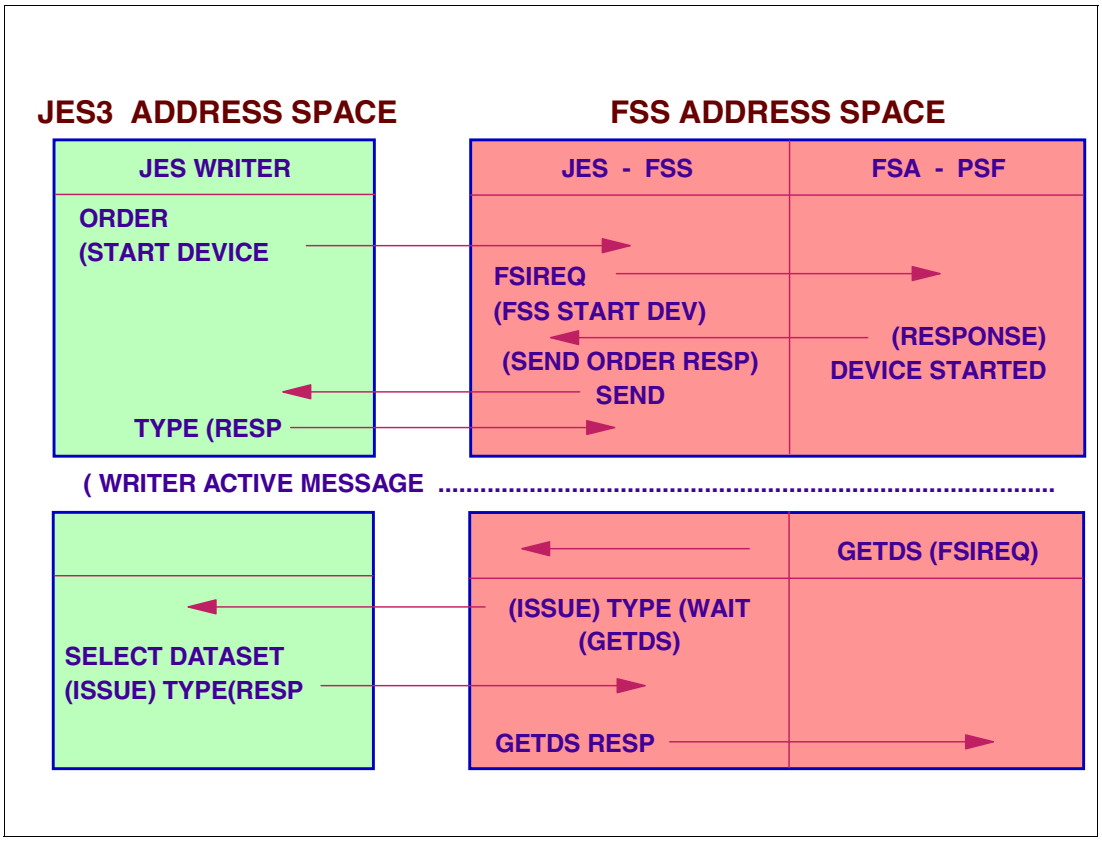
The FSIREQ CONNECT request results in a call to the SSI CONNECT routine of the subsystem specified in the CDFSSID field of the CONNECT parameter list. The CONNECT parameter list is used as the SSOB extension for the SSI call. The SSI CONNECT routine loads the JES functional subsystem support modules into the FSS address space and then passes control to the FSI CONNECT routine in that module.

JES3 address space

When the FSS CONNECT is received, JES3 indicates to that the FSS is connected and to start the FSA. JES3 responds to the FSS connect by sending a start FSA order service request list (SRL) to the FSS address space using a JSERV macro.

The FSS address space then start the FSA and is now waiting for an ORDER for a writer to start.

11.54 Starting an FSS writer



Start FSS writer in FSS address space

Completion of FSA level CONNECT processing signals JES to issue a START device order.

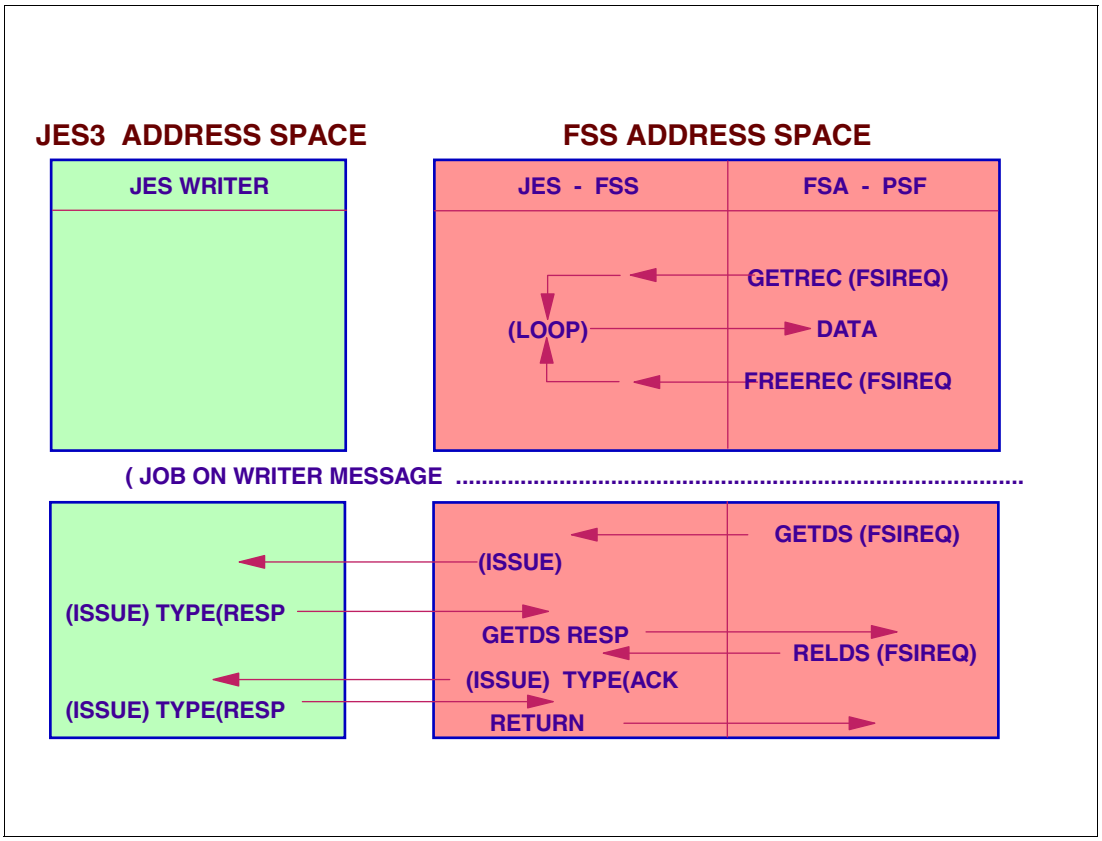
JES3 issues the ORDER to start the device, as shown in the visual. Once the START DEVICE has been responded to from the FSA, the FSA may start selecting output data sets from JES3 using the GETDS. A GETDS request is made from IAZSFSA and JES3 responds with a data set.

To start an FSA, JES issues the START FSA order to the FSS's FSI ORDER routine. During FSS CONNECT processing, the FSS places the address of the FSI ORDER routine into the CDFAD field of the CONNECT parameter list for use by JES. JES passes the address of the START FSA order parameter list in register 1. The parameter list contains the address of the order response area (IAZRESPA).

If the FSA is successfully initialized, the FSA issues the FSA-level FSIREQ CONNECT request. This is the response to the START FSA order.

Therefore, the START device routine is the signal for the FSA to begin issuing GETDS requests. Once the device is started, it can begin requesting data sets from JES for output processing.

11.55 Read data set and release data set



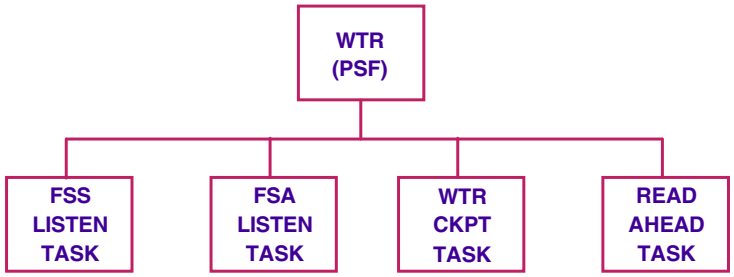
Read the data set

Once the data set is selected, the FSA may start issuing GETREC and FREEREC requests to obtain the data, process it, and free the records. When the data set is completely processed, a RELDS request is issued to mark the data set complete in the JES3 address space.

After an FSA notifies JES3 (using the FSI SEND request) that it successfully started the associated device, it is ready to begin data set processing. As part of data set processing, the FSA invokes the FSI data access services (GETDS, GETREC, FREEREC, RELDS, and CKPT) to:

- Obtain a SYSOUT data set and its characteristics from JES3.
- Obtain logical records of an obtained data set
- Release logical records for a data set to JES3
- Release an obtained data set to JES 3
- Request JES3 to record checkpoint information for a JES spool data set currently being processed by the FSA device

11.56 WTR FSS TCB structure




```
*I F,TYPE=WTR
IAT8701 FSSNAME  TYP  SYSTEM   PROCNAME JOBID     STAT T S MD   RC
IAT8701                                                DSP/DEV  MAXASST
IAT8702 WTRIAZF   WTR SC43      WTRIAZF    JOB26509 FSSC Y   JES  42
IAT8702                                             ACTV IAZFSS
*I,F
IAT8701 FSSNAME  TYP SYSTEM    PROCNAME JOBID      STAT T S MD   RC
IAT8701                                                DSP/DEV  MAXASST
IAT8702 CIFSS1     C/I   SC43       JES3CI       JOB33379  ACTV N Y
IAT8702                                                020,001  00000000
IAT8702 WTRIAZF  WTR SC43       WTRIAZF    JOB34153   FSSC N   JES  42
IAT8702                                             ACTV IAZFSS
```

FSS WTR TCB structure

The FSS address space TCBs are as follows:

- The 'WTR' task is where the I/O to the devices takes place. The modules executing under this task are from the PSF support. These modules issue macro, FSIREQ, to communicate with the FSS/FSA.
- The 'LISTEN MODE' tasks are attached by IATSICD at FSS and FSA CONNECT time respectively. Module IATFCLT listens for requests from the JES3 address space and routes the data to the appropriate processing routine be examining the function identifier in the Service Request List, (SRL). The mapping macro is IATYSRL. The SRL is passed in a staging area. Modules IATFCOR and IATFCPT are called to service the requested functions.
- The 'WTR CHKPT' task is attached at FSA CONNECT time by module IATFPCC. This task is posted by IATFPCP when a checkpoint is requested via, FSIREQ REQUEST=CHKPT, from the WTR task. Module IATFPCW writes the checkpoint to the spool.
- The 'READ AHEAD' task is attached at FSA CONNECT time by module IATFPCC. Module IATFPRA runs under this task and is posted by IATFPGR when a GETREC order is received. This task calls a new module, IATDMBS, which is the access method for obtaining the output data from the JES3 spool. IATDMBS, is called the Block I/O routine and resides in PLPA. In some documentation you may see it referred to as the block spooler.

A display of the WTR FSS shows a job, JOB26509, as active on the FSS writer.

FSS WTR

Use the **`*I F`** command to display:

- The attributes and current status of the FSS
- Information for all FSSs of a particular type (WTR, C/I), whether active or inactive
- Maximum counts and status information for C/I service DSPs running in a C/I FSS.

11.57 Query FSS writers

- Display information for all the inactive FSSs

```
*I,F,INACT
IAT8701 FSSNAME  TYP  SYSTEM PROCNAME JOBID   STAT T S MD   RC
IAT8701                                          DSP/DEV  MAXASST
IAT8702 CIFSS2     C/I   SC43     JES3CI      NONE    INAC N N
IAT8702                                          005,000  00000000
IAT8702 IPDSWAY   WTR SC43    IPDSWAY    NONE    INAC N   JES  42
IAT8702                                          INAC IPDWAY
```

- Display short version of messages for the inactive FSSs:

```
*I,F,INACT,C
IAT8701 FSSNAME  TYP SYSTEM   PROCNAME JOBID   STAT T S MD   RC
IAT8702 IPDSWAY  WTR SC43      IPDSWAY    NONE   INAC Y   JES. 42..
IAT8702 CIFSS2     C/I   SC43      JES3CI       NONE   INAC N N
```

Query FSS writers

The `*I F` command displays:

- The attributes and current status of the FSS
- Information for all FSSs of a particular type (WTR, C/I), whether active or inactive
- Maximum counts and status information for C/I service DSPs running in a C/I FSS.

Syntax:

```
*INQUIRY F,[ACTIVE | ALL | FSS=(fssname,....) | INACT | TYPE=[WTR | CI]] [C | S]
```

Display the short version of messages IAT8701 and IAT8702 for the inactive FSSs which display both types of FSSs:

```
*I,F,INACT,C
  IAT8701 FSSNAME  TYP SYSTEM   PROCNAME JOBID    STAT T S MD   RC
  IAT8702 CIFSS2   C/I SY1      JES3CI   NONE     INAC N N
  IAT8702 CIFSS3   C/I SY1      JES3CI   NONE     INAC Y N
  IAT8702 CIFSS4   C/I SY2      JES3CI   NONE     INAC Y N
  IAT8702 CIFSS5   C/I SY2      JES3CI   NONE     INAC Y N
  IAT8702 CIFSS6   C/I SY6      JES3CI   NONE     INAC N N
  IAT8702 CIFSS7   C/I SY7      JES3CI   NONE     INAC Y N
  IAT8702 CIFSS8   C/I SY3      JES3CI   NONE     INAC Y N
  IAT8702 MF1      WTR SY1      PRTSIMO2 NONE     INAC N   JES  42
  IAT8702 PRT804   WTR SY1      PRT804   NONE     INAC N   JES  42
```

11.58 Writer output multitasking facility

- JES3 WTRs can use "Multitasking"
  - In an Auxiliary Task
- Benefits?
  - Multiprocessor type CPUs
  - Global address space offload
- Current Support in JES3 for
  - Local and remote printers and punches
    - For remotes - (BSC and SNA) devices

Writer output multitasking

When the global is a multiprocessor, the writer output multitasking facility enables JES3 to do more work in parallel. When enabled (turned on), the writer output multitasking facility provides an additional task under which JES3 can do its work. This task, called the JES3 auxiliary task, lets JES3 off-load work that would otherwise be done under the JES3 primary task. JES3 off-loads that part of the output writer's work that actually prints or punches the output. The JES3 primary task can then execute in parallel with the JES3 auxiliary task and other JES3 subtasks. Only traditional (non-FSS) writers are eligible for writer output multitasking.

JES3 off-loads that part of the output writer's work that actually prints or punches the output. The JES3 primary task can then execute in parallel with the JES3 auxiliary task and other JES3 subtask Only traditional (non-FSS) writers are eligible for writer output multitasking.

The writer output multi-tasking facility allows JES3 output writers to do more work in parallel with other JES3 functions on a multiprocessor. To do this, JES3 provides an additional task called the JES3 auxiliary task. This subtask of the JES3 primary task allows writers to execute in parallel with the JES3 primary task or with other JES3 subtasks.

After the system programmer or an operator enables the writer output multi-tasking facility, active output writers execute part of the time under the JES3 primary task and the rest of the time under the JES3 auxiliary task. Writers execute under the auxiliary task while reading the job output files from the spool and while writing these files to an I/O device such as a printer. The rest of the time the writers execute under the JES3 primary task.

11.59 Defining output multitasking

- JES3 initialization
- Operator command

Initialization Stream Parameter

OPTIONS,.....,MT = ON | OFF

Operator Command Control

*F MT = ON or *F MT = OFF

*I MT

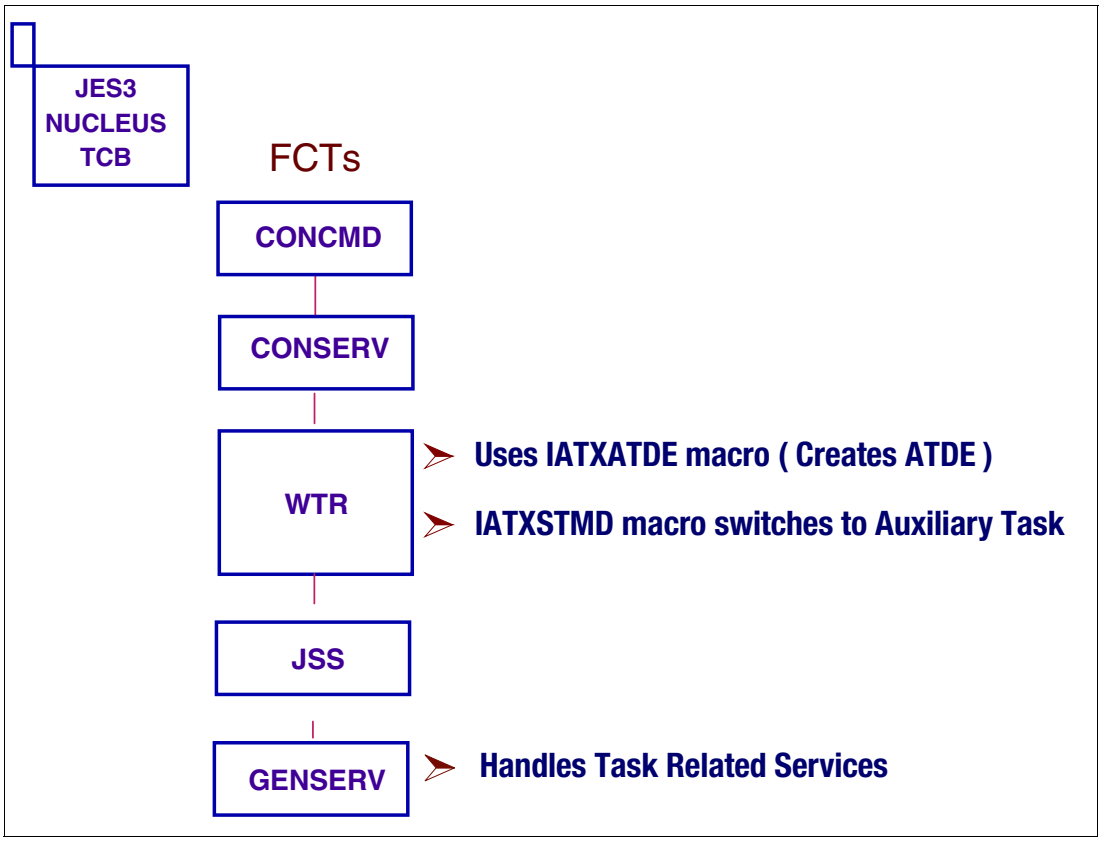
- Auxiliary task does only output to device

Writer multitasking definitions

If the global processor is a multiprocessor, you should turn on the multitasking facility. To turn the facility on through the initialization stream, specify **MT=ON** on the OPTIONS initialization statement. Or you can turn it on while JES3 is running by issuing the command ***MODIFY,MT=ON**.

If the global processor is a uniprocessor, you should turn off the multitasking facility. On a uniprocessor, with the multitasking facility turned on, MVS must do additional task switching for JES3. Also, it may take the output writers longer to print or punch their output because JES3 changes its work sequence. To turn off the multitasking facility during initialization, specify **MT=OFF** on the OPTIONS initialization statement. To turn it off while JES3 is running issue the command ***MODIFY,MT=OFF**.

11.60 WTR multitasking



Writer multitasking FCTs

In order for a function to run under the JES3 auxiliary task (that is, to take advantage of multi-tasking), it must issue the IATXATDE macro, which creates an ATDE. The macro also chains the ATDE to the function's FCT entry and attaches the ATDE to the ATDE queue. The ATDE also contains a ECF mask and address.

When the multifunction monitor receives control from a function running under the JES3 auxiliary task, it scans the ATDE queue to select an FCT for dispatching. For details on how this selection process works, see

If a function issues an AWAIT TYPE=ON or TYPE=OFF or IATXSTMD macro while running under the JES3 auxiliary task, the MFM scans the ATDE chain for the highest priority ATDE for which an AWAITed event has been posted as complete. The MFM determines, from a pointer in the ATDE, which FCT is associated with that ATDE and then dispatches that FCT.

If there are no ATDEs for which an AWAITed event has completed, the MFM reaches a dummy ATDE at the end of the ATDE queue. The dummy ATDE points to a dummy WAIT FCT that causes the JES3 auxiliary task to enter the wait state when the WAIT FCT is dispatched by the MFM.

12

JES3 and multisystem consoles

Consoles are devices that you use to enter commands and receive messages from JES3, MVS and application programs. Consoles have certain capabilities and limitations, depending on how you define them. Consoles in a JES3 environment fall into one of the following classes:

- Multiple console support (MCS) consoles

  MCS consoles are devices that you can physically attach to global or local processors. These consoles allow you to control the sysplex. Refer to *z/OS MVS Planning: Operations*, SA22-7601 for information about MCS consoles in a sysplex environment.

  The characteristics of MCS consoles are defined during MVS initialization with the CONSOLE statement in the CONSOLxx member of SYS1.PARMLIB. These characteristics include:

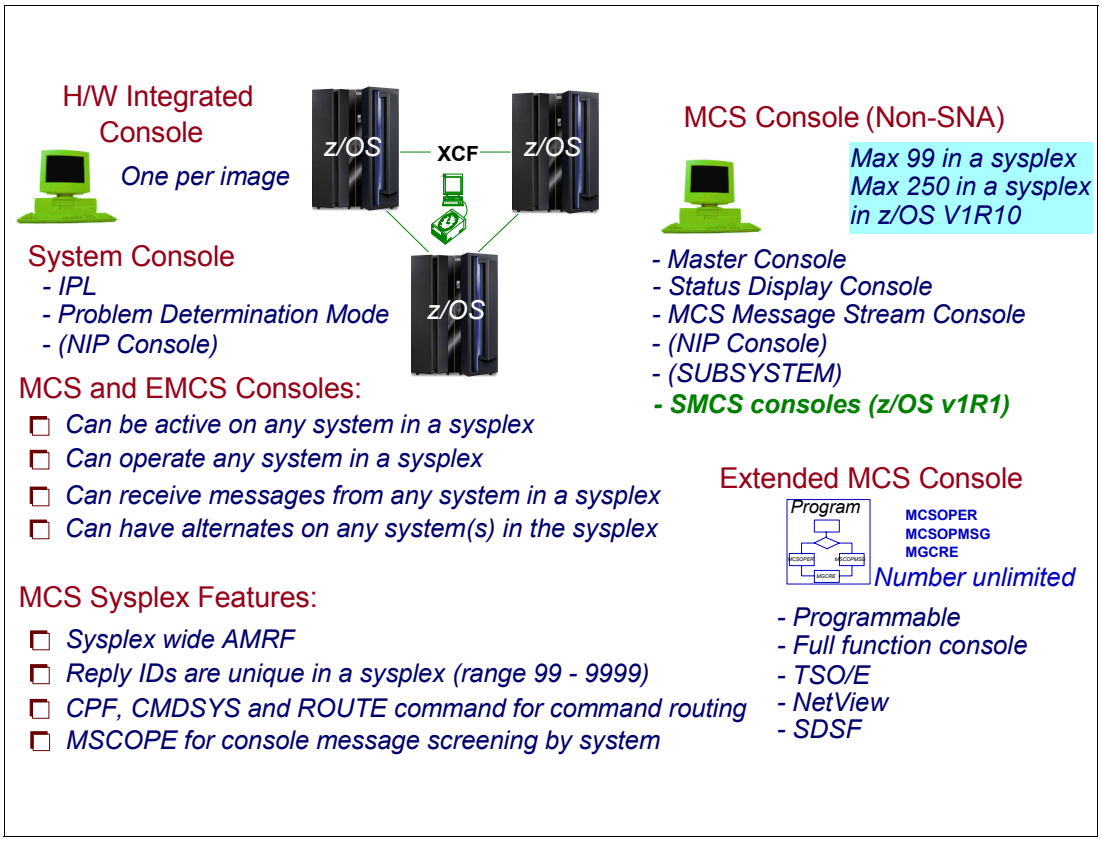
  - The console's authority level which determines what types of commands are allowed to be used at that console.
  - Its message destination which indicates the categories of messages to be routed to that console.

- Remote job processing (RJP) consoles

  RJP consoles are attached to the JES3 global as part of a remote workstation using telecommunications lines. RJP permits you to submit jobs, receive output, and control submitted jobs at workstations that can reside at some distance from your installation.

  The characteristics of JES3 RJP consoles are defined during JES3 initialization with the CONSOLE initialization statement.

12.1 Multisystem consoles in a sysplex



Consoles in a JES3 environment

The visual shows the three types of consoles:

MCS MCS consoles (locally attached and SUBSYSTEM) - up to 99 defined through the CONSOLxx parmlib member.

EMCS Extended MCS (EMCS) consoles - programmable consoles defined and activated through an intended programming interface (the MCSOPER macro with REQUEST=ACTIVATE). The number of active EMCS consoles is not restricted.

System Console integration is supported by MVS and allows the hardware system console to support both hardware functions and the operating system IPL, recovery, and problem analysis.

The system console in problem determination mode is automatically defined by MVS during system initialization.

Consoles in a sysplex

In a sysplex, a console can be active on any system in a sysplex and can provide sysplex-wide control. MCS uses XCF services for command and message transportation between systems and thus provides a single system image for the operators. MCS multisystem support features:

- Sysplex-wide action message retention facility (ARMF)
- Sysplex-wide unique reply IDs
- Sysplex-wide command routing through:

- ROUTE operator command
- Command prefix facility (CPF)
- CMDSYS setting for a console (through CONSOLE statement in the CONSOLxx parmlib member or `CONTROL` **`V,CMDSYS=`** operator command)

Normal message presentation is controlled by a console's ROUTCDE and MSCOPE settings.

Migration to distributed mode with z/OS V1R10

To take advantage of distributed mode, you must perform a sysplex-wide migration to distributed mode. You can do so through a sysplex-wide IPL or dynamically, while you are running. The ability to fallback dynamically to the shared mode of operation is also provided. However, the fallback is potentially disruptive to your operations.

In shared mode, any CONSOLxx CONSOLE definition after the 99th console in the sysplex is rejected.

In distributed mode, each system is allowed to define 250 (or 251 if one is for the system console) consoles. They can be any combination of MCS, SMCS, and SUBSYSTEM consoles. As in shared mode, the SMCS and SUBSYSTEM definitions are available to any system in the sysplex. The MCS definitions now define how the console behaves on that system. Another system may have a unique definition (as long as it is an MCS console) that governs the console’s behavior on that system.

12.2 System symbols overview

- Allow an installation to define symbols for:
  - PARMLIB members
  - JES3 and MVS commands
  - Started task and started job JCL
  - Dynamic allocation
- Static system symbols - system-defined
  - &SYSPLEX - &SYSNAME - &SYSCLONE
  - &SYSCLONE defaults to last two characters of &SYSNAME

System symbols

System symbols act like variables in a program; they can take on different values, based on the input to the program. When you specify a system symbol in a parmlib member the system symbol acts as a "place holder". Each system that shares the definition replaces the system symbol with a unique value during initialization.

System symbolics are also useful in the operations area. A particular application may run on several images. One of the goals of replication is to use a common set of source JCL to initialize the application but retain the uniqueness where required. The most efficient way to do this is to have a single set of source JCL that uses symbolics for those parameters that must be unique per image and initiate the application as a started task or started job.

Support is provided that allows a unique jobname to be associated with a single set of source JCL by adding a JOBNAME= parameter to the start command. Support was also provided that allowed the source JCL for a started task to be a job. Although only a subset of the JOB card parameters were allowed, it provided the level of accounting, security and output control that was required by many customers in order to use started tasks.

Support is provided by allowing the use of system symbolics in:

- The source JCL for started tasks and TSO logon procedures
- MVS commands
- Dynamic allocations

With this support a single set of source JCL can be started on all systems in a sysplex with a single command (either via a common COMMNDxx member or via a ROUTE *ALL command) and resolved to different jobnames on each image.

Static symbols

Static symbols are symbols that do not vary the contents across an IPL. These symbols have their names defined to the system. Your installation defines substitution texts or accepts system default texts for static system symbols. These static system symbols are the following:

- &SYSCLONE
- &SYSNAME
- &SYSPLEX

12.3 JES3 system symbols support

- Use of symbols in:
  - TSO logon procs and started tasks
  - PROCLIB members
- JES3 ensures correct symbol table used
- JES3 use of symbols
  - JES3 commands
  - JES3 PROC

JES3 and system symbols

Static system symbolics in source JCL are supported for demand select jobs (started tasks and TSO logons). JCL rules are used for determining where the symbolics can be used and how they are resolved. Both system symbolics and JCL symbolic parameters are used to resolve the symbolics in the source JCL for demand select jobs. If the JCL symbolic parameters redefines a system symbolic, the value associated with the JCL symbolic parameter is used. For batch jobs, only the JCL symbolic parameters are used.

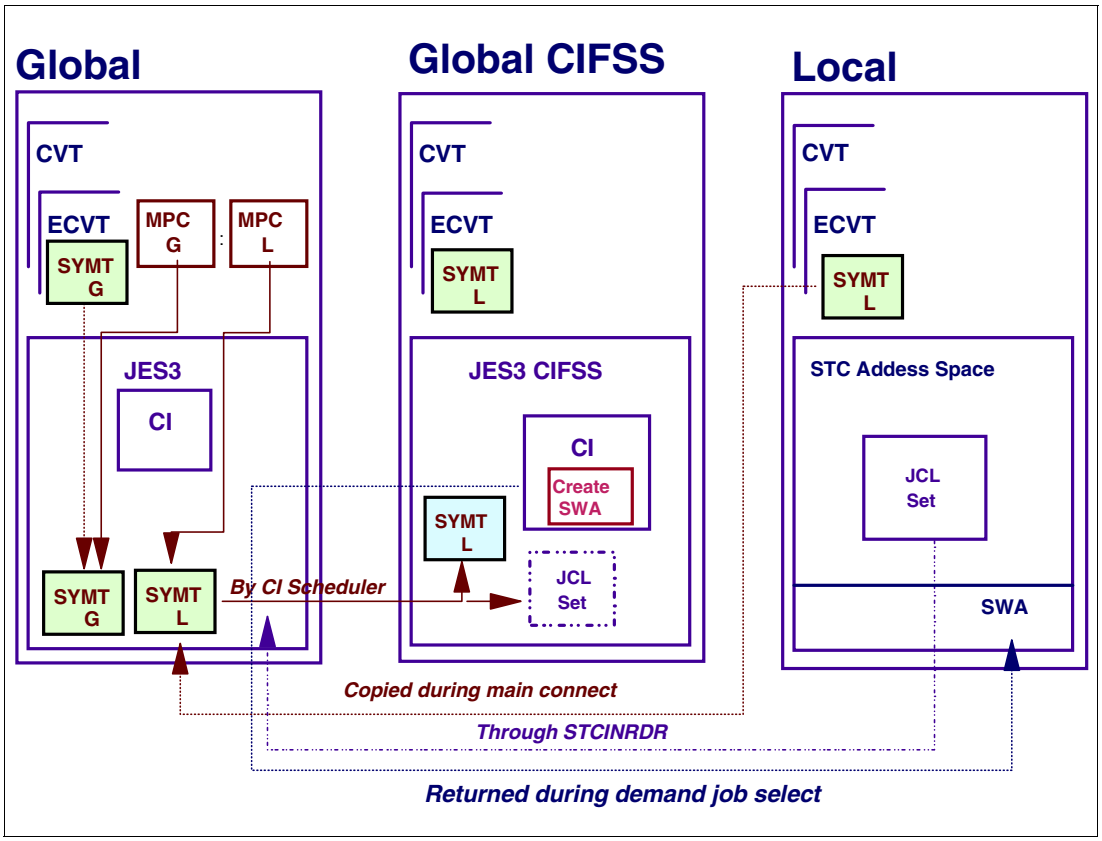
To allow installations to use a single parmlib member, a single procedure or a single command to handle multiple MVS systems in a sysplex, system symbolic substitution allows replication of definitions. Symbolic substitution is supported in parmlib member parameters and operator commands (including JES3 commands as they are processed by MCS), and started task JCL.

JES3 commands

System symbols represent unique values in shared commands. For each system, you can define values for system symbols. When shared commands are processed, each JES3 system replaces the system symbols with the defined values.

Before you use system symbols in JES3 commands, you must understand the types of system symbols, the elements that comprise them, and the general rules for using them. See *z/OS JES3 Initialization and Tuning Reference*, SA22-7550 for details about planning to use system symbols.

12.4 JES3 system symbols support



JES3 symbol support

The visual shows the principles of how the global JES3 receives the system symbol tables from JES3 main processors and how they are passed to Converter/Interpreter processing for demand select jobs.

In order for the replication of installation definitions (cloning) to work for demand select jobs, JES3 must use the correct system symbolics table during the MVS C/I phase even if the conversion occurs on a system other than where the demand select job is to execute. When the local connects, it passes the connecting system's system symbolics to the global. When a demand select job is scheduled for CI, the symbol table of the system, where the job executes, is passed with the JCL to the processor which performs the JCL conversion. For ARM restarts, JES3 uses the symbol table that was used for the original execution of the job.

JES3 allows you to specify system symbols in source JCL for demand select jobs. The symbols allow two or more systems in a JES3 complex to share the source JCL, while retaining unique values in the JCL.

System symbols represent unique values in source JCL for demand select jobs. Each system has its own values for system symbols. When shared demand select jobs are processed, each JES3 system replaces the system symbols with its own values.

When the source JCL for a demand select job is sent to another processor for conversion, the converter uses the system symbols that are defined to the processor on which the started task was submitted (not the processor on which the conversion takes place).

12.5 Consoles in a sysplex

- Single master console for sysplex
  - Assigning a console master authority
- Consoles attached to any system
  - Sysplex-wide control from any console
    - MCS consoles
    - Extended MCS consoles
    - System console
    - Netview consoles
    - TSO CONSOLE mode consoles
    - SMCS consoles controlled by VTAM

Consoles in a sysplex

The introduction of a sysplex into the MVS environment provides a simpler and more flexible way to operate consoles in a multisystem environment. Many changes were introduced into multiple console support (MCS) to support the sysplex environment. These changes began with MVS/ESA Version 4 and have continued with each new MVS release.

In a sysplex, MCS consoles can:

- Be attached to any system
- Receive messages from any system in the sysplex
- Route commands to any system in the sysplex

Therefore, new considerations need to be made when defining MCS consoles in this environment, such as:

- There is no requirement that each system have consoles attached.
- The 99 console limit for the sysplex can be extended with the use of extended MCS consoles. This adds greater flexibility when attaching consoles.
- A sysplex, which can be up to 32 systems, can be operated from a single console.
- Multiple consoles can have master command authority.

With MCS consoles in a sysplex, no matter where they are attached, it is possible to control any system in the sysplex. The ability to assign each console a unique name and unique characteristics greatly eases the task of system management.

Multisystem consoles

The major goals of a sysplex from a systems management point of view are to provide:

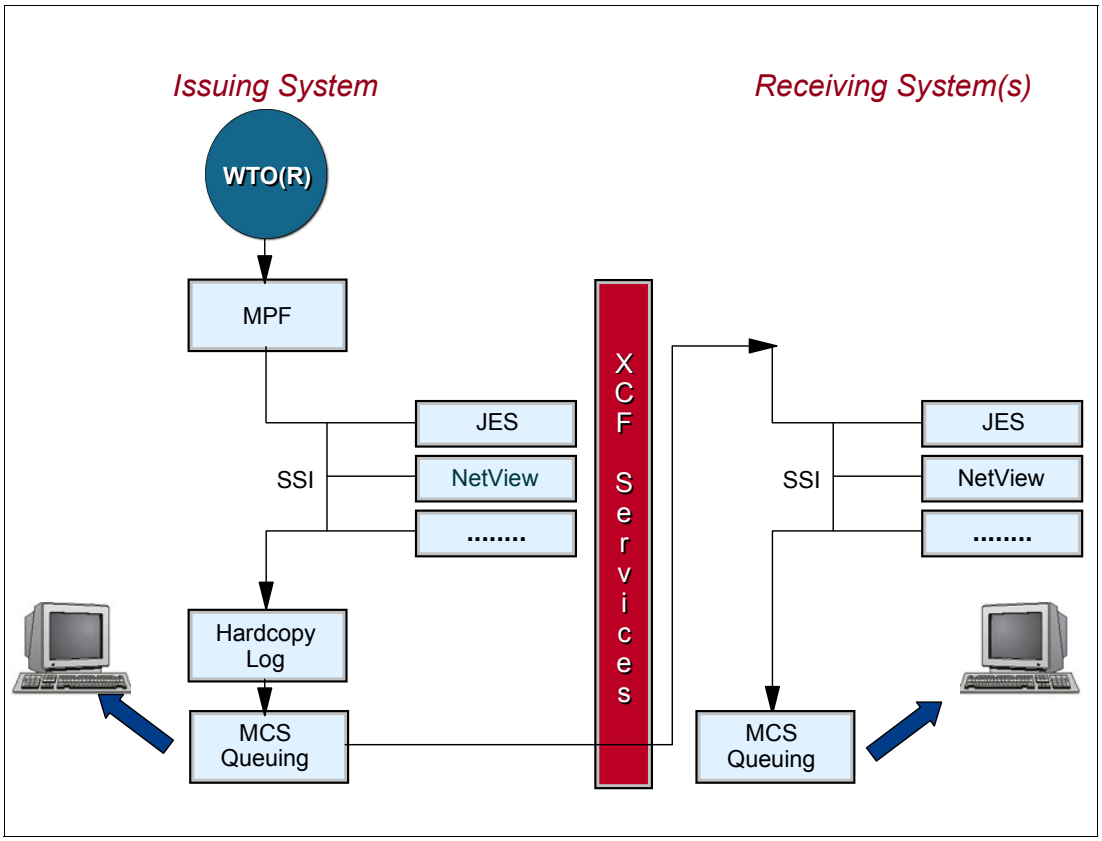
- Single system image
- Single point of control
- Minimize human intervention

Multisystem console support has features to support single system image, single point of control, and minimal human intervention for those sysplex environments with these processors.

In a sysplex, the major tasks of operating an individual MVS image do not change very much. Consoles are used to receive messages and issue commands to accomplish system tasks. With the existence of multiple MVS images and multiple subsystems, the potential exists for the operator to receive an enormous number of messages, since there is the capability to route all messages from all MVS systems to a single console.

Multisystem consoles can be used by applications to be sysplex aware due to the fact that no matter on which system the applications exists, command routing and message routing can be done. The application does not have to exist on the same system as a console used to communicate with it.

12.6 Message flow in a sysplex



Message flow

In a sysplex, a message is routed to all active consoles on all systems that are eligible to receive that particular message.

The visual shows the message flow in a sysplex. It is important to understand this flow since in a sysplex environment, message flow is changed to send messages issued on one system to other systems in the sysplex using XCF services.

WTO(R) The MVS write to operator (WTO) and the write to operator with reply (WTOR) macro services cause messages to be routed to the consoles and the hardcopy log.

MPF First on the issuing system, the message is processed by the message processing facility (MPF). This processing is based on entries in the MPFLSTxx parmlib member.

MPF processing allows an installation to influence how WTO and WTOR messages are to be processed. Through the MPFLSTxx member, you can specify some processing options for a message.

SSI Following MPF processing, the message is broadcast to all active subsystems that request to receive control for the WTO SSI function code 9. The subsystem must use the IEAVG700 interface to indicate that all WTO and WTORs are to be broadcast. The message is presented to each subsystem in turn. Each subsystem may inspect the message and process it as appropriate. A subsystem can alter write-to-operator queue element (WQE) fields, in which case later subsystems on the SSI will see the changed WQE. A WQE is an internal control

block that contains the message text and all related information for that message. The IHAWQE macro maps the WQE fields.

For example, when NetView is using the SSI rather than an extended MCS console for MVS communication, NetView on the SSI inspects all messages to see whether they are marked by MPF as eligible for automation. NetView intercepts automation message text and selected attributes from the WQE and sends the data to the NetView address space for further processing. NetView does not modify the actual WQE.

Log After the message has been inspected by all active subsystems, it is written to the hardcopy log (usually the SYSLOG data set, the operations log (OPERLOG), or both) unless hardcopy logging is suppressed by an exit. OPERLOG is a log stream maintained in a coupling facility that uses the system logger to record and merge communications about programs and system functions from each system in a sysplex. The messages are logged using message data blocks (MDBs), which provide more data than is recorded in the SYSLOG.

Queueing Finally the message is routed for display on the appropriate MCS and extended MCS consoles. The routing may require message transportation using XCF services to other systems in the sysplex because some receiving consoles may not be physically attached to the system where the message was issued.

After the XCF transportation on the receiving system, the message goes through the SSI loop, but it is not logged, and finally the message is processed by the message queueing tasks to be displayed on the consoles.

If a message is destined for a specific console that is not active in the sysplex, it is logged and discarded unless it is an action message or WTOR message, in which case it is processed as an undelivered message. It is sent to all active consoles receiving UD messages. The master console is a UD receiver by default.

Messages that are already "delivered" to an active extended MCS console, but not yet "retrieved", are purged from the MCS queues when the console is deactivated; that is, unprocessed queued messages are not rerouted.

MSCOPE specification

The MSCOPE specification on the CONSOLE statement in the CONSOLxx parmlib member allows you to screen those systems in the sysplex from which a console is to receive messages not explicitly routed to the console.

12.7 CONSOLxx - CONSOLE statement

Specify systems from which messages received

CONSOLE MSCOPE {(sysname|*[,sysname]...)}
{(*ALL) }

VARY CN(name),MSCOPE=SC65

Assigning a console master authority

CONSOLE AUTH(MASTER)

VARY CN(name),AUTH=MASTER

Specifying MSCOPE parameter

MSCOPE allows you to specify those systems in the sysplex from which this console is to receive messages not explicitly routed to the console.

```
MSCOPE  {(sysname|*[,sysname]...)}
        {(*ALL)                  }
```

An asterisk (*) indicates the system on which this CONSOLE is active and *ALL indicates all systems in the sysplex.

The following command adds, deletes, or changes a console's scope of systems from which messages are received:

```
VARY CN(cn),AMSCOPE=(sy..)|DMSCOPE=(sy..)|MSCOPE={(*ALL)|(sy..)}
```

AUTH specifies the group of operator commands that can be entered from the console.

```
AUTH     {(MASTER)           }
         {(INFO)             }
         {([SYS][,IO][,CONS])}
         {(ALL)              }
```

IBM strongly suggests using a security product, such as RACF, to control commands instead of using AUTH, especially with SMCS. For more information about SMCS and console security see *z/OS MVS Planning: Operations,* SA22-7601.

Master authority console

From a console with master authority, you can enter all MVS operator commands. The corresponding authority levels for JES3 are:

```
MASTER: JES3 authority level=15
CONS: JES3 authority level=10
I/O: JES3 authority level=10
SYS: JES3 authority level=5
INFO: JES3 authority level=0
```

Where:

MASTER Indicates that this is a console with master-level authority.

INFO Specifies that any informational commands can be entered from this console. INFO is the default for all consoles except the system console, which is forced to be AUTH(MASTER).

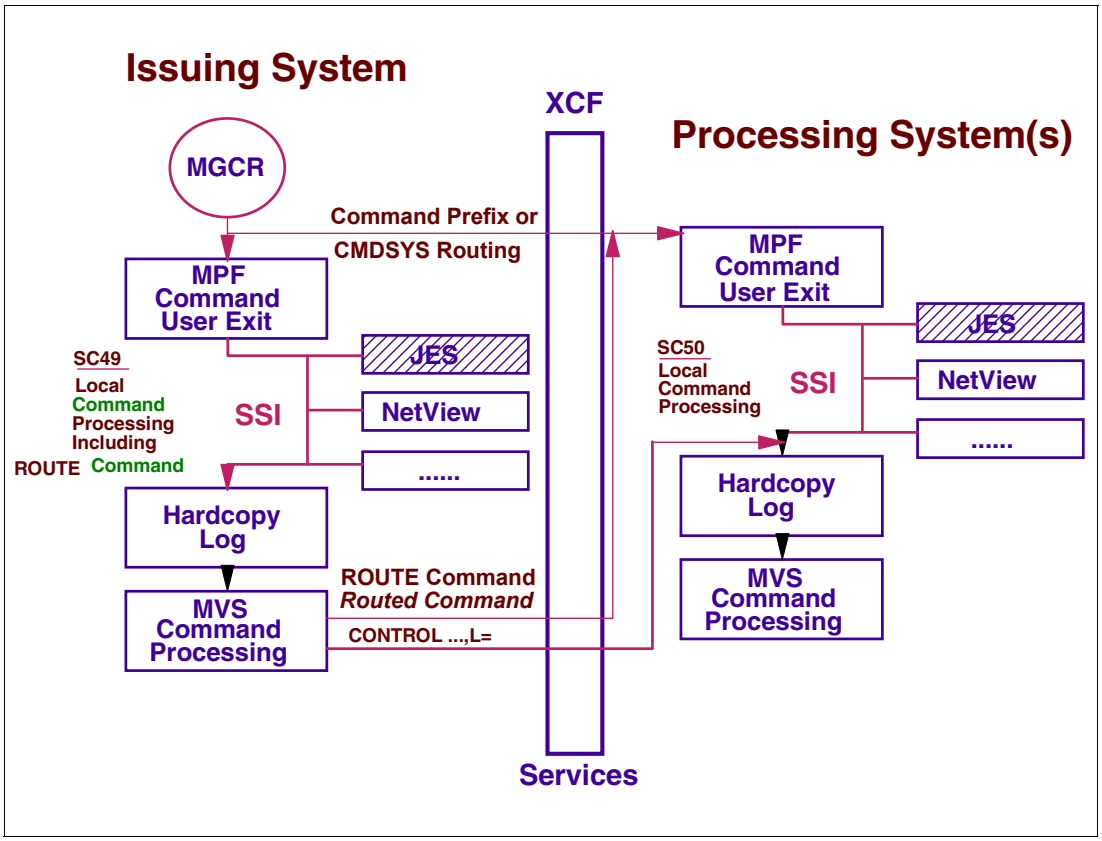
SYS Specifies that system control commands and informational commands may be entered from this console.

IO Specifies that I/O control commands and informational commands may be entered from this console.

CONS Specifies that console control commands and informational commands may be entered from this console.

ALL Specifies that information, system control, I/O control, and console control commands may be entered from this console.

12.8 Command flow in a sysplex



Command flow

When an operator command is entered through the MGCR(E) macro service, the following processing flow takes place:

- Command prefix and CMDSYS routing

  If the command contains a prefix or the console has a CMDSYS specification that directs the command to another system, the command is immediately transmitted using XCF services to the processing system.

  > **Note:** For command prefix and CMDSYS routing, the command is first transported to the receiving system before any system symbol substitution takes place.

  A **REPLY** command is sent to the system where the WTOR was issued. If the REPLY command text contains system symbols, substitution occurs on the receiving system.

- MPF command user exit

  If the command is not transmitted to another processing system, it is processed on the issuing system by the installation MPF command exit routines. The exits are specified using the.CMD statement in the MPFLSTxx parmlib member. These exits can perform authority checking, modify the command text, or the command processing.

  > **Note:** For commands containing system symbols, substitution has occurred before the exit is entered.

- SSI processing

  The command is then broadcast on the subsystem interface (SSI) to all active subsystems. Each subsystem inspects the command and decides whether to process it. The subsystems base the decision on the command prefix characters of the command string. For example, by default, NetView looks for a percent sign (%) and processes the commands starting with the % sign.

  When a subsystem decides to process the command, the command is passed to subsystem processing, and a return code is set to indicate that the command was processed by a subsystem.

  Note: At this point in processing, all system symbol substitution has occurred. The original command text is also available.

- Hardcopy log

  Once the command has been examined by all active subsystems, it is logged to the hardcopy log (usually SYSLOG or OPERLOG).

  Note: The hardcopy log contains the command before any system symbols contained in the command have been substituted and, it also contains the command after substitution has occurred.

- MVS command processing

  If none of the subsystems has marked the command as having been processed, it is assumed to be an MVS command and is passed to the appropriate MVS command processor.

12.9 Command routing in a sysplex

- MVS ROUTE command
- CMDSYS {(sysname) | *)}
  - On CONSOLE statement
- Command prefix facility - CPF
- IEECMDPF program
  - Sample SYS1.SAMPLIB

Command routing

This visual shows the different ways or techniques that be used to route commands in a multisystem sysplex. There are:

- The operator can use the MVS **ROUTE** command to route commands to any MVS image in the sysplex.
- The CMDSYS keyword on the CONSOLE statement in the CONSOLxx member can specify for each console which system, commands issued from the console are to be executed on.
- The command prefix facility can be used to define which MVS image processes command issued with the prefix used.
- The IEECMDPF program in SYS1.SAMPLIB can be used to specify the sysname as a prefix for command routing.

12.10 MVS ROUTE command

```
ROUTE  {sysname,text                          }
       {*ALL,text                             }
       {sysgrpname,text                       }
       {[T=nnn,]                              }
       {*OTHER                                }
       {(sysname,sysgrpname,sysname,..}
```

T=nnn - Optional timeout interval

T -- does not work with a single sysname
-- must be first parameter

ROUTE command

The MVS **ROUTE** command explicitly routes another operator command for execution on another system in a sysplex. It can be issued from both MCS and extended MCS consoles. The response to the command is returned to the issuing console (unless redirected by an L=) operand.

The **ROUTE** command parameters are:

sysname The target system name that receives and processes the command.

text The command and specific operands of the command being routed.

T= Specifies an optional timeout interval. T= is valid with *ALL, *OTHER, sysgrpname, or a list of system names or sysgrpnames. The T= value indicates the maximum number of seconds MVS waits for responses from each system before aggregating the responses.

***ALL** Specifies that the command is to be routed to all systems in the sysplex.

***OTHER** Specifies that the command is to be routed to all systems in a sysplex except the system on which the command is entered.

sysgrpname Specifies that the command will be routed to a subset of systems in the sysplex. System group names are defined by the installation. IBM provides an IEEGSYS member in the SYS1.SAMPLIB to define installation named system groups.

Note: Regardless of the CMDSYS value in effect for the console that issues the `ROUTE` command, the ROUTE command itself is processed on the system on which it was issued.

ROUTE command processing

The `ROUTE` command is processed in two stages; the `ROUTE` command is processed first on the issuing system causing the routed command to be transported to the receiving system, where the routed command is processed as if locally issued. If the routed command is prefixed, the prefix may request rerouting of the command before it can be processed.

12.11 Defining system groups

Sample JCL for IEEGSYS Procedure

```
//IEEGSYS  PROC MEMBER=GSYS00
//IEEGSYS  EXEC PGM=IEEGSYS
//SYSIN    DD   DSN=SYS1.PARMLIB(&MEMBER),DISP=SHR
```

System Group Parmlib Member - GSYS00

```
GROUP(P101)  NAMES(SC52,SC53,SC49)
GROUP(P201)  NAMES(SC54,SC55,SC50)
GROUP(P301)  NAMES(SC42,SC43)
GROUP(CMOS)  NAMES(SC52,SC53,SC49,SC54,SC55,SC50,SC42,SC43)
GROUP(CMOS2) NAMES(SC42,SC43)
GROUP(P942)  NAMES(SC47)
GROUP(JES3)  NAMES(SC49,SC50,SC43)
GROUP(JES2)  NAMES(SC47,SC52,SC53,SC54,SC55,SC42)
GROUP(AOC)   NAMES(SC47,SC49,SC55)
```

Issue a ROUTE command to each system in sysplex

```
ROUTE *ALL,S IEEGSYS,MEMBER=GSYS00
```

Defining system groups

IBM provides an IEEGSYS member in the SYS1.SAMPLIB to define installation named system groups. The IEEGSYS program builds groups only on a single system, and must be executed on every system to which the group definitions apply. To create the system groups:

- Create a SYS1.PROCLIB procedure.

  ```
  //IEEGSYS  PROC MEMBER=GSYS00
  //IEEGSYS  EXEC PGM=IEEGSYS
  //SYSIN    DD   DSN=SYS1.PARMLIB(&MEMBER),DISP=SHR
  ```

- Create a SYS1.PARMLIB member that defines the group names and the systems in each group, as shown in the visual as member GSYS00.
- Issue the following ROUTE command to each system in the sysplex to create the system groups on each system in the sysplex: The system group name can be specified on the MVS ROUTE command, as shown in visual.

  ```
  ROUTE *ALL,START IEEGSYS,MEMBER=GSYS00
  ```

  Or, you can place the **START** command in the COMMNDxx parmlib member.

12.12 ROUTE by groups

```
RO ALLSYS,D T
RO T=005,SC65,D T
RO T=005,SC64,D T
RO T=005,SC63,D T
RO T=005,SC70,D T
SYSNAME  RESPONSES
----------------------------------------
SC63    IEE136I LOCAL: TIME=10.45.06 DATE=2008.326  UTC:
      TIME=15.45.06 DATE=2008.326
SC64    IEE136I LOCAL: TIME=10.45.06 DATE=2008.326  UTC:
      TIME=15.45.06 DATE=2008.326
SC65    IEE136I LOCAL: TIME=10.45.06 DATE=2008.326  UTC:
      TIME=15.45.06 DATE=2008.326
SC70    IEE136I LOCAL: TIME=10.45.06 DATE=2008.326  UTC:
      TIME=15.45.06 DATE=2008.326
```

Route by groups

This visual shows an example of the **ROUTE** command using the groupname defined by the program provided in the IEEGSYS member in the SYS1.SAMPLIB to define installation named system groups.

The IEEGSYS program builds groups only on a single system, and this program must be executed on every system to which the group definitions apply.

The IEEGSYS program does for every group defined in the input file:

- Enqueue the group name exclusive (to serialize with programs that use groups and enqueue shared).
- Build a list of the system names in CSA.
- If the group was previously defined, delete its name/token entry, saving the address of the old list in CSA.
- Create a name/token entry naming the group and pointing to the list of system names.
- If the group was previously defined, free the CSA storage used for the old list.
- Dequeue the group name.

MVS console service routines locate, for the route-by-group command, the name/token data for the group and issues a **RO T=005,system,cmd** for each system listed in the group.

12.13 CMDSYS definition

- Consoles specified in CONSOLxx
  - MCS consoles
  - SMCS consoles
  - Subsystem-allocatable consoles
- EMCS
  - Attributes specified in RACF OPERPARM
- Command routing is done:
  - Using CMDSYS on the CONSOLE statement
  - Using the ROUTE command
  - Using the Command Prefix Facility
  - Using the L=operand on certain commands
- CPF processing is independent of CMDSYS
  - Taken before the CMDSYS routing

CMDSYS definitions

Consoles, both MCS and extended MCS, can have an associated CMDSYS value, which specifies the system to which commands issued on the console are to be sent. This provides an implicit command routing allowing a console that is physically attached to one system to be logically associated with another system.

The CMDSYS is specified for:

MCS consoles Use the CMDSYS parameter on the CONSOLE statement in CONSOLxx parmlib member. CMDSYS can also be specified using the `CONTROL V,CMDSYS=sysname,L=name` operator command.

EMCS consoles For extended MCS consoles, use the OPERPARM segment of the user's RACF profile or specify the OPERPARM data in the MCSOPER macro request. CMDSYS can also be specified using the `CONTROL V,CMDSYS=sysname,L=name` operator command.

Note: There are some commands, LOGON/LOGOFF, TRACK/STOPTR, ROUTE and some variations of CONTROL, which are not affected by the CMDSYS values. These commands are always processed on the issuing system.

The `REPLY` command is always processed on the system where the WTOR was issued.

Command exits

The installation command exits and the active subsystems are invoked only on the target system and the command is logged on the target system. The command response is sent back to the originating console.

Command prefix routing

If a command contains a prefix or the console has a CMDSYS specification that directs the command to another system, the command is immediately transmitted using XCF services to the processing system.

12.14 Extended MCS consoles (EMCS)

- Programming interface for authorized users
  - Issue MVS and subsystem commands
  - Receive messages
  - Used by :
    - TSO/E user
    - Batch job
    - STC address space
  - Exploited by TSO/E commands:
    - CONSOLE
    - CONSPROF
  - No 99 console limit
  - Can be an alternate or have an alternate console

EMCS consoles

The extended MCS console (EMCS) is a set of programmable interfaces introduced in MVS/ESA SP Version 4 that makes it possible for a program to enter commands and receive messages as though it were an MCS console.

The programmable interface can be used by a:

- TSO/E user
- Batch job
- Started task address space

Examples of extended MCS consoles supporters:

- TSO/E users with the appropriate RACF authorization can use the CONSOLE command to establish an extended MCS console session. TSO/E extended MCS consoles can be authorized to issue commands requiring master authority.

  TSO/E users can write REXX programs that act as an operator. The CONSOLE command is used to activate the EMCS console and to issue operator commands. The CONSPROF TSO/E command can be used to control message delivery, and the GETMSG REXX function is used to retrieve queued messages.

It is recommended that you use extended MCS consoles when you write an authorized program that acts as an operator. Extended MCS consoles also provide relief for the number of consoles that can be used in an MVS system as they are not included in the 99 MCS and subsystem consoles in a sysplex limit.

Programmable interface

Extended MCS consoles are not pre-defined. When a program wants to establish an extended MCS console, it uses a set of programmable interfaces. Three authorized macros are available with this programmable interface:

- MCSOPER
- MCSOPMSG
- MGCRE

The MCSOPER macro is used to activate an EMCS console. Once the extended MCS console is activated, the program can receive messages and command responses by invoking the MCSOPMSG macro service, and can issue commands by issuing the MGCRE macro.

The TSO/E CONSOLE and CONSPROF commands make use of the programmable interfaces and can be used by a TSO/E user or applications written in REXX.

The MCSOPER macro service allows you to activate and manage extended MCS consoles. You can specify only one of the following functions each time you invoke the MCSOPER macro:

REQUEST=ACTIVATE Initializes an extended MCS console session. The MCSOPER macro with REQUEST=ACTIVATE defines and activates an extended MCS console to the system. When you activate an extended MCS console, MCS creates a dataspace to store messages and DOM requests. There is one dataspace for every address space with an active extended MCS console. Therefore, if an address space has two active extended MCS consoles, both share the same message dataspace. Note that the system deletes this dataspace upon deactivation of all the extended MCS consoles in the address space.

REQUEST=DEACTIVATE Terminates the session. Through MCSOPER with REQUEST=DEACTIVATE and ABTERM=YES, you can deactivate an active console and switch processing from an extended MCS console to another active MCS or extended MCS console. Before deactivating a console, you must define a valid alternate group for the console through operator attributes. Switching an extended MCS console to an alternate console allows processing to continue without interruption, and is useful if a program representing an extended MCS console abnormally ends, and your installation needs to have its processing taken over by another console.

REQUEST=RELEASE Releases a migration ID from an extended MCS console that has been deactivated by issuing REQUEST=DEACTIVATE.

You need to specify the extended MCS console's attributes when it is activated. You specify the operator parameters in the OPERPARM segment of the user profile of a security product, such as RACF.

Note: You can override the console attributes specified in the user profile of the security product by turning on the MCSOVRDY bit in the MCSOP data area. Therefore, if you specify the extended MCS console attributes in the MCSOP data area and request security product override, the MCSOP data area specifications are used (even if the RACF OPERPARM data is also specified for the user).

12.15 Activating WRAP mode

- Activate WRAP mode by specifying:
  - CONSOLE DEL(W)
- Operator can activate
  - K S,DEL=W
- Change RTME value to 1/4 second for WRAP mode
  - K S,DEL=W,RTME=1/4

WRAP mode

Message presentation features on MCS consoles include:

- Messages are displayed in either ROLL mode or WRAP mode on locally attached MCS consoles.
- The HOLDMODE specification on the DEFAULT statement of the CONSOLxx member allows an operator to temporarily suspend or hold screen updates when in roll, roll-deletable, or wrap mode.

The WRAP mode display allows new messages to overlay the oldest messages on the console screen. A separator line appears between the new and the old messages when messages are being overlaid. WTOR and action messages are also overlaid if WRAP mode is in effect.

You can activate WRAP mode by specifying DEL(W) on the CONSOLE statement of the CONSOLxx parmlib member.

```
CONSOLE  DEL(W)
```

If this option is not specified in parmlib, activate WRAP mode with the following command:

```
CONTROL S,DEL=W
```

Note: You might want to change the RTIME value, which is the interval for message roll, to 1/4 second. If so, enter:

```
CONTROL S,DEL=W,RTIME=1/4
```

12.16 CONSOLxx - CONSOLE statement

- MFORM - Displaying messages
  - Specify the format messages are displayed
  - S - to include name of issuing system
  - X - to suppress jobname/sysname
    - For JES3 messages

```
MFORM  {(M)                 }
       {([J][,S][,T][,X])   }

CONTROL S,MFORM
```

MFORM keyword

MFORM specifies the display format of the messages.

M M indicates that the system is to display the message text only. The message display does not include a time stamp, job ID, or job name information, or the system name. M is the default.

J J specifies that the display is to include the job ID or name.

S S specifies that the display is to include the name of the system originating the message.

T T specifies that the display is to include a time stamp.

X X specifies that the system suppress the job name and system name for JES3 messages issued from the global processor.

You can use the following command to change a console's message display format:

```
CONTROL S,MFORM
```

12.17 JES3 CONSTD initialization statement

```
CONSTD,EDIT=({escape,bkspace,,linedl}),
        GLOBMPF={YES|NO},
        SYN={(syn1,...syn6)|8},
        PLEXSYN={(syn1,...syn6)|*},
        CIFSS={FSSDEF|MSGROUTE},
        DLOG={ON|OFF}
```

Command stacking:

- EDIT=(,,newline,) deleted
- CONSOLxx INIT CMDDELIM{;}
- Example:
  - *I Q;*I A;*I Q,S

CONSTD statement

Some JES3 installations have come to rely on the JES3 global processor to be a "focal point" for message processing and have implemented extensive MPF processing on the global for messages that originate on local processors. To accommodate "global-oriented" MPF processing that an installation may have, an option (GLOBMPF=) is provided on the CONSTD initialization statement.

GLOBMPF=YES Indicates that all messages routed by MCS to the global processor should also be made available to MPF processing on the global processor in addition to MPF processing on the originating system.

GLOBMPF=NO The default. Indicates that messages routed to the global should not be presented to MPF on the global processor. In this case, MPF processing is only possible on the system that a message originates from.

It should be noted that the GLOBMPF option does not influence the routing of messages to the global processor. Installations wishing to use the GLOBMPF option must ensure that routing mechanisms are in place to guarantee the global is presented the proper set of messages. This could include the activation of DLOG which will result in the hardcopy message set being presented to the global, or the definition of a physical console or extended MCS console on the global that receives the proper set of routing codes (or all routing codes).

Prefix registration

JES3 registers the SYSPLEX (PLEXSYN=) and SYSTEM (SYN=) scope command prefixes that are defined on the JES3 CONSTD initialization statement, as shown in the visual.

New line delimiter and command stacking

JES3 no longer uses the delimiter defined by the EDIT keyword in the CONSTD initialization statement to unstack the commands as *newline* has been removed.

```
CONSTD,EDIT=({escape,bkspace,newline,linedl})
```

Instead, it uses the MCS command delimiter that is defined by the CMDDELIM parameter on the INIT statement in the CONSOLxx parmlib member.

```
INIT  CMDDELIM(c)
```

The processing for commands entered from an MCS console or an automated operator interface through MGCR(E) remains the same. When multiple commands are entered through a single input from an MCS console, MCS unstacks commands and sends one command at a time to the SSI. When multiple commands are entered through a single MGCR(E) by an automated operator, the whole stack of commands is sent by MCS to the SSI. If the first command is an MVS command, the JES3 SSI routine returns the whole stack of commands back to MCS for processing. If the first command is a JES3 command, the JES3 SSI routine sends the whole stack of commands to the JES3 address space for processing. The JES3 command processing continues to provide the unstacking service as in prior releases of JES3.

12.18 Command prefixes

- JES3 uses command prefix facility
  - Prefix registration and display services
  - System and sysplex scoped prefixes
  - JES3 sysplex-scoped prefixes
    - Communicate with global from any console
  - JES3 system-scoped prefixes
    - Communicate with JES3 on issuing system

Command prefixes

A new keyword, PLEXSYN, defines the sysplex command prefixes.

PLEXSYN PLEXSYN={(syn1,...syn6)|:us.*:eus.} - Specifies a sysplex scope for the command prefix. The sysplex scope means that any command issued with this prefix from any system in the sysplex executes on the global processor. The default is *. This keyword is used together with the SYN keyword to determine the prefix to be used. If a prefix is defined on both keywords, the prefix is used as a system-scoped prefix.

JES3 registers with CPF sysplex-wide command prefixes that are defined with the PLEXSYN= parameter on the JES3 CONSTD initialization statement. Up to six sysplex-wide JES3 prefixes (also known as synonyms) can be specified. The sysplex-wide prefixes are always registered with CPF on the global processor and are re-registered during the dynamic system interchange. In general, commands routed explicitly with the MVS **ROUTE** command and with sysplex-wide prefixes are rerouted by MCS as required according to the sysplex-wide prefix definition after the **ROUTE** command routing completes. If a command is routed using a sysplex-wide prefix, MCS routes it only *once*, even if the command has multiple sysplex-wide prefixes.

JES3 defines both SYSTEM and SYSPLEX scope prefixes with the option FAILDISP=PURGE. This means that a command prefix is deleted when the defining system is removed from the sysplex or the defining address space terminates. The CPF REMOVE=NO option is used by JES3 for the prefix removal. This implies that both the

command prefix and the command are presented on the receiving system. The length of both SYSTEM and SYSPLEX scope prefixes can be one to eight characters.

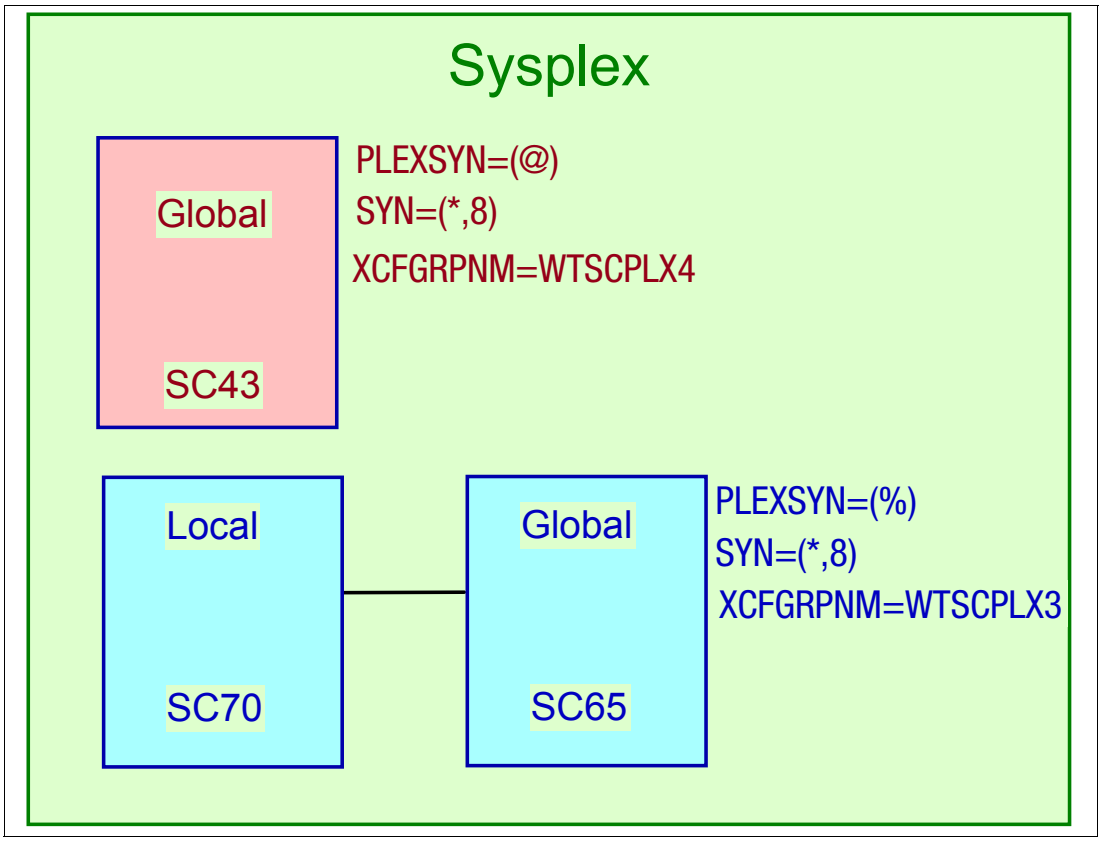
If a prefix is specified on both the SYN= and PLEXSYN= parameter, JES3 defines it as a SYSTEM-wide (SYN=) prefix. If the JES3 global fails to define one or more of the prefixes specified on the PLEXSYN= parameter, a warning message is issued and the prefix is ignored. When every PLEXSYN= prefix registration fails, JES3 uses the default SYSPLEX-scope prefix * and issues a warning message stating that the default sysplex-wide prefix is being used.

Note: Even if * is the default SYSPLEX-wide prefix, it is recognized as a *valid JES3 command prefix on a local* when it is passed to command processing on a local. This happens, for example, if a * JES3 command is routed to a local using double prefixing. The * has preserved the *guaranteed-to-work* JES3 command prefix role for compatibility reasons. This way, the * can be used as the fixed JES3 command prefix by automation products. In this case, the command transportation should be implemented using double command prefixing.

If JES3 fails to register the * prefix, there will *not* be any JES3 sysplex-wide prefixes, and all JES3 commands from locals must be explicitly routed to the global using either the MVS ROUTE command or double command prefixes. The first of the double prefixes must be defined as "REMOVE=YES", to route the command to the global. The second prefix should be the JES3 "*" or any of the system-scope prefixes registered by the global.

Note: Double prefixes are allowed for commands. MCS uses the first prefix to transport the command. Once the command is transported, MCS then presents the command with the remaining prefix to the command processors on that system.

12.19 Multiple JES3 complexes in the same sysplex



Multiple globals in a sysplex

The existing SYN= parameter on the CONSTD initialization statement continues to define up to six SYSTEM scope JES3 prefixes. The SYSTEM scope prefixes are registered on *each* system in the JES3 complex. Commands that begin with SYSTEM scope prefixes are processed on the processor on which the command is entered. Explicitly routed commands with SYSTEM-wide prefixes are processed on the target processor.

When JES3 is unable to register one or more of the SYSTEM-scope prefixes specified on the SYN= parameter on any processors, a warning message is issued. If all prefixes specified on the SYN= parameter fail to be defined to CPF on a processor, JES3 uses the default prefix value 8 on that processor and issues a warning message stating the fact.

Note: The 8 prefix is implemented as a SYSTEM-scope prefix but is not registered with CPF to avoid conflicts with short-form WTOR replies.

Multiple globals in a sysplex considerations

The following considerations should be followed when running more than one global in the same sysplex:

- The XCF groupnames must be unique. This name can be specified on the OPTIONS initialization statement or can default to the home nodename. The XCF group names for the sample sysplex configuration shown in the visual, and are WTSCPLX3 and WTSCPLX4.
- The command prefix has to be selected in the following way:

- JES3 internally issues commands using the * prefix. Thus, if more than one global is active in the sysplex, the * must be explicitly defined in each initialization stream as SYN=*, a SYSTEM-scope prefix, as shown in the visual. Otherwise, any internally JES3 issued commands may be processed by the wrong global.
- You should also specify a sysplex-scoped prefix for each global. This allows any console in the sysplex to send commands to the global and receive the responses.
- If the use of the character 8 as a prefix is required, then specify more than one prefix. The visual shows the prefixes for a sysplex with more than one global.

12.20 Display command prefixes

D OPDATA

```
IEE603I  14.42.58  OPDATA  DISPLAY  840
PREFIX    OWNER     SYSTEM   SCOPE    REMOVE  FAILDSP
 %        JES3      SC70     SYSPLEX  NO      PURGE
 *        JES3      SC70     SYSTEM   NO      PURGE
 *        JES3      SC65     SYSTEM   NO      PURGE
 @        JES3      SC43     SYSPLEX  NO      PURGE
 *        JES3      SC43     SYSTEM   NO      PURGE
 SC43     IEECMDPF  SC43     SYSPLEX  YES     SYSPURGE
 SC65     IEECMDPF  SC65     SYSPLEX  YES     SYSPURGE
 SC70     IEECMDPF  SC70     SYSPLEX  YES     SYSPURGE
 #        RACF      SC43     SYSTEM   NO      PURGE
 #        RACF      SC70     SYSTEM   NO      PURGE
 #        RACF      SC65     SYSTEM   NO      PURGE
```

Display OPDATA command

The **DISPLAY OPDATA** command displays operator information (OPDATA). Depending on the operands specified, the display may represent either sysplex-wide data or system-unique data.

```
D {OPDATA|O}[,PREFIX       ][,L={a|name|name-a}]
            [,{TRACKING|TR}]
            [,{MONITOR|MN}[,FULL]]
```

PREFIX The system displays (message IEE603I) sysplex-wide information about the command prefixes defined for the subsystems in the sysplex. This is the default if no other operands are specified.

TRACKING or TR The status of the Console ID Tracking facility is displayed (message CNZ1001I), along with any recorded instances.

MONITOR or MN The system is to display through message CNZ1100I the enablement status of the monitoring facility for all message types supported, including whether each of these monitor message types are sent to the system log/operlog. The system also displays the number of consoles and, if applicable, TSO/E users that have requested to receive specific message types.

FULL Instead of displaying the number of consoles and TSO/E users that have requested to receive specific message types, the system lists the names of those consoles. If there is any TSO/E user information to display, an additional section listing the user names will be included.

12.21 System name as a command prefix

- IEECMDPF
  - IBM-supplied sample program in SYS1.SAMPLIB
  - Defines the executing system's name as a command prefix
  - For example
    - Run IEECMDPF on system SC65
    - ROUTE SC65,command - valid
    - SC65 command - valid
    - SC65command - valid
    - SC65,command - invalid
- ROUTE *ALL,S IEECMDPF

Command prefix program

You can run IEECMDPF (an IBM-supplied sample program in SYS1.SAMPLIB) to define system names as a command prefix. This program is intended to be executed on each system in a sysplex (for example, through a **START** command in a common COMMNDxx PARMLIB member) to create a command prefix for each system equal to its system name. This allows an installation to direct a single-system command to a given system by simply preceding the command with the system name.

The **ROUTE *ALL,S IEECMDPF** command can be used to register the system name as a prefix on each system in the sysplex.

For example, if you run IEECMDPF on system SC65, then the following have the same effect on each system in the sysplex:

```
ROUTE SC65,command
SC65 command
SC65command
```

12.22 Global MPF processing

- MCS controls message flow in sysplex
  - MPF processing on issuing system
- Use CONSTD GLOBMPF=YES|NO
  - Passes messages routed to global to MPF
- Options for global MPF processing
  - Activate DLOG
  - MCS console or extended MCS console
    - Correct route codes (all route codes)

Global MPF processing

MCS controls all message flow in a sysplex and a message is routed to all active consoles on all systems that are eligible to receive that particular message.

GLOBMPF=YES indicates that all messages routed by MCS to the global processor should also be made available to MPF processing on the global processor in addition to MPF processing on the originating system. Messages that originate from the global processor are eligible for MPF on the global regardless of the GLOBMPF value.

The GLOBMPF option does not influence the routing of messages to the global processor. Installations wishing to use the GLOBMPF option must ensure that routing mechanisms are in place to direct the proper set of messages to the global. This could include the activation of DLOG which will result in the hardcopy message set being presented to the global, the definition of a physical console or extended MCS console on the global which receives the proper set of routing codes (or all routing codes), or the marking of target messages with the 'AUTO' attribute together with an extended console on the global receiving 'AUTO' messages.

When using the GLOBMPF function, remember that MPF processing may have to be adjusted as part of a DSI. Ensure that the MPF options on the old and new global are set up correctly. The SET MPF operator command can be used to change the MPF options for a particular system.

There are two options that force all messages to be routed to the global, they are:

- Activating the JES3 DLOG
- Having an EMCS or MCS console that has all route codes

12.23 JES3 message processing

- Exits 69 and 70 - global message processing
  - MVS dynamic exit facility eligible
  - Exit points IAT_EXIT69 and IAT_EXIT70
- Installing dynamic exits
  - PROGxx parmlib EXIT statement
  - SETPROG EXIT operator command
- MSGROUTE statement no longer required
  - If kept - "J" parameter has new meaning

Message exits

Because JES3 does not transport messages to the JES3 global address space for display and logging purposes, two JES3 installation exits, exits IATUX69 (Determine If a Message is to be Sent to the JES3 Global Address Space) and IATUX70 (Perform Additional Message Processing), are provided to accommodate special message processing that is dependent on running in the JES3 address space and having access to JES3-maintained information. Exits IATUX69 and IATUX70 allow you to implement JES3 environment specific message processing in the global JES3 address space.

Exit 69

Exit IATUX69 allows a message to be examined and optionally sent to the global for further processing in user exit IATUX70. IATUX69 is called from module IATSIWO during WTO SSI processing.

Exit 70

Exit IATUX70 receives messages that have been examined by user exit IATUX69 and sent for further processing in the JES3 global address space. IATUX70 is called from module IATCNSV. A parameter list (mapped by IATYUX70) provides all the information needed to process the message.

Message exits

Exit IATUX69 and exit IATUX70 are managed through the MVS dynamic exit facility. JES3 defines exit points to the MVS dynamic exit facility with the name IAT_EXIT69 and

IAT_EXIT70. By default, JES3 does not define any exit routines to this exit. A sample exit module IATUX69 is provided in the SYS1.SIATSRC data set.

MSGROUTE statement

The MSGROUTE (MVS Message Route Table) statement controls the routing of subsystem modifiable messages (such as most MVS-issued messages). If you do not include a MSGROUTE statement, the routing attributes of the messages that originate from that processor are not modified by JES3 MSGROUTE processing. Even though MSGROUTE processing may not make modifications, a message is still eligible for other forms of JES3 message routing.

```
MSGROUTE,MAIN=main,routecode=(destclass,console,J),...,routecode=(destclass,console,J)
   routecode =
      Specifies the MVS routing code and a console name or destination class to which
      you want messages sent.
   routecode
      Specifies an MVS routing code which between 1 and 128 inclusive.
   destclass
      Specifies a JES3 console destination class to which you want messages with the
      designated MVS routing code mapped.
   console
      Specifies the name of an MCS console or an extended MCS console.
   J
      Specifies that the routing code equivalent of the destination class is to be used
      for the message instead of the message's original routing code(s). If you do not
      specify "J", the routing code equivalent will be merged with the message's
      original routing code(s).
```

When a message is issued with multiple routing codes, JES3 selects a single routing code to use for MSGROUTE processing.

Recommendation: Remove the MSGROUTE initialization statements from your initialization stream if you no longer need this JES3 function. If you retain these statements, update the MSGROUTE J specification if appropriate for your environment due to its changed function in sysplex message routing. Consider using MVS routing codes in the initialization stream and in JES3 commands in addition to, or in place of, JES3 destination classes.

12.24 JES3 console dynamic exits

- Exit IATUX69 called from SSI routine
- Flags message for global processing
  - Flags message for Exit IATUX70
- Exit IATUX70 examines messages from exit IATUX69
  - Runs under CONSERV FCT
- Sample exits provided
- Multiple exit routines allowed

Using message exit IATUX69

You may infrequently need to perform some processing related to a message when running in the JES3 global processor's address space rather than an MPF exit on the issuing processor. Installation exits IATUX69 and IATUX70 provide this function. Exit IATUX69 can be used to examine every message on the issuing processor and to instruct JES3 to deliver selected messages to exit IATUX70. JES3 will then transport the selected messages to the global processor and present them to exit IATUX70, which runs in the JES3 global address space. IBM recommends that you perform message processing on the issuing system whenever possible to avoid the system overhead associated with use of these exits.

Exit IATUX69 indicates the message needs further processing.

Exit IATUX70

Exit IATUX70 allows you to implement JES3-specific processing in the JES3 global address space under the CONSERV FCT.

You can use the EXIT statement of the PROGxx parmlib member, the SETPROG EXIT operator command, or the CSVDYNEX macro to control this exit and its exit routines. JES3 allows multiple exit routines to exist for this exit.

12.25 Installing dynamic exits

❑ SYS1.PARMLIB in the PROG member

```
EXIT  ADD EXITNAME(IAT_EXIT69) MODNAME(IATUX69) STATE(ACTIVE)

SETPROG  EXIT,ADD,EXITNAME=IAT_EXIT69,MODNAME=IATUX69,STATE=ACTIVE

 CSV420I  MODULE  IATUX69  HAS  BEEN  ADDED  TO  EXIT  IAT_EXIT69

D  PROG,EXIT,EXITNAME=IAT_EXIT69

 CSV461I  15.41.05  PROG,EXIT    DISPLAY 938
 EXIT                MODULE      STATE  MODULE   STATE MODULE   STATE
 IAT_EXIT69          IATUX69       A

D PROG,EXIT,EXITNAME=IAT_EXIT69,DIAG
CSV464I 16.00.27 PROG,EXIT DISPLAY
EXIT IAT_EXIT69
MODULE    STATE EPADDR     LOADPT     LENGTH     JOBNAME
IATUX69     A     A4B02660  24B02660  00000008  *
```

PROG member in parmlib

You can use the EXIT statement of the PROGxx parmlib member, the **`SETPROG EXIT`** operator command, or the CSVDYNEX macro to control this exit and its exit routines.

Exit IATUX69

Exit IATUX69 is called from the JES3 WTO SSI processing and allows you to examine a message and decide whether it should be sent to the JES3 global address space for additional processing. If the exit indicates that the message should be sent to the global, the WTO SSI sends the message through the JES3 SSISERV service. This exit is called on the system where the message is issued for all messages regardless of the origin of the message (for example, messages that originate on a local, messages that originate on the global, and messages issued by JES3 DSPs through the JES3 MESSAGE macro). In addition to calling exit IATUX69, the WTO SSI processing continues to call IATUX57 to allow the installation to select a single routing code for JES3 message routing processing.

The following is a summary of the types of WTOs and WTORs that are presented to exit IATUX69 and any special considerations that exist for the type of message. Flags in the exit parameter list indicate the type of request being passed to the exit:

Multi-Line WTO The first line of a multi-line WTO (the major line). Subsequent minor lines are *not* passed to the exit. Using the major line, the exit determines whether the message should be sent to the global.

If the exit determines that the message should be sent to the global, the major line and all minor lines are presented to exit IATUX70 on the global. A separate call to exit IATUX70 is made for each minor line, as follows:

Commands The text of an operator command.

WTORs The text of a WTOR message.

WTOR responses The full text of a WTOR response.

Exit IATUX70

Exit IATUX70 is called in the global JES3 address space when the installation exit IATUX69 has sent a message that requires further processing in the JES3 global address space. This exit cannot influence any routing, presentation or retention attributes of the message.

JES3 defines this exit to the MVS dynamic exit facility with the name IAT_EXIT70. By default, JES3 does not define any exit routines to this exit point. JES3 allows multiple exit routines to exist for this exit. Note that the exit routines are *not* invoked with ASAVE linkage. A sample exit module IATUX70 is provided in the SYS1.AJES3SRC data set.

Displaying dynamic exits

This visual shows the definitions in parmlib for the new JES3 dynamic exits and all the other dynamic exits currently in the system.

The exits can be added by using the SETPROG command as shown in the previous visual.

The MVS **DISPLAY** command can be used to display the status of a JES3 dynamic exit, as follows:

```
D PROG,EXIT,{{EXITNAME|EX|EN}=exitname  }[,DIAG]
```

12.26 JES3 macro processing considerations

- INTERCOM macro
  - Simulate input of operator message
  - CNDB= (optional keyword)
    - Address of console destination block
- MESSAGE macro
  - Issue a message
  - Use CNDB= when a CNDB is available
    - CNDB information merged; Console ID, Command Response Indicator, Routing Code Information

CNDB control block

The console destination block (CNDB) is a control block that encapsulates console routing information for a command or a message. JES3 intercepts JES3 commands from the SSI and saves the command origin including 4-byte console ID and related response processing information into a CNDB. CNDBs are used in MESSAGE macros to direct messages to a specific destination. CNDBs are also used in INTERCOM macros to identify the origin of an internally issued command.

INTERCOM macro

The JES3 INTERCOM macro simulates operator input of a console command. The INTERCOM service is used by DSPs to internally issue commands. The INTERCOM macro processing enters the command into the system with an MCS MGCRE macro. As a result, all commands go through the MCS interface, which serves as the single command entry into the MCS sysplex.

The INTERCOM macro continues to be the primary means for JES3 DSPs to enter commands into the MCS sysplex. Some changes have been made to the INTERCOM macro:

- An optional CNDB= parameter is added to the INTERCOM macro. It addresses a CNDB that contains a 4-byte console ID, a console name, and a CART of the command issuer. If the CNDB= parameter is not specified, the INTERCOM macro uses the default dummy CNDB in the TVTX.
- Since the console ID is included in the CNDB, the CONS= parameter is deleted. The BUFFER= parameter is also deleted.

MCS sysplex introduced the command and response token (CART) for extended MCS consoles to allow a command response to be associated with the command. The CART is supplied by the command issuer. The command processor includes the CART in the command response message(s), thus allowing the issuing program to uniquely associate a command instance with its response.

JES3 also uses the CART to insure accurate presentation of command responses to the NJE and RJP extended MCS console. The CART is carried in the CNDB throughout the command processing.

Using the MESSAGE macro

When a command with a CART is issued from an EMCS console, JES3 obtains the CART from the SVC 34 subsystem interface and saves it in a CNDB. Later, when JES3 command processors issue command responses through the MESSAGE macro, the macro points to the CNDB that contains the CART that associates the command response with the command. The MESSAGE macro service routine then issues a WTO macro with CART= parameter to present the message for MCS processing.

12.27 DSP console appendage

```
          LOGIN   ENTER=CONAPNDG
         ...
          IATXCNS TYPE=GET              GET PARAMETERS
         ...
         ...
CONAPNDG DS    0H
          USING CONSMESS,R6             COMMAND BUFFER BASE
          USING CONAPNDG,R10            ROUTINE BASE
          LR    R6,R1                   CONSMESS ADDRESSABILITY
          LR    R10,R15                 ROUTINE ADDRESSABILITY

          IATXCNDB OPERATION=TRANSFER,  SAVE CALLING CONSOLE
                   CNDB=CONCNDB,    DESTINATION BLOCK
                   OUTCNDB=MYCNDB
         ...
          identify command type (User decision)
         ...
          ARETURN
```

Console appendage for a DSP

JES3 uses Console Destination Blocks (CNDBs) to save console identification and message routing information. JES3 uses the IATXCNDB macro service to work with CNDBs. IATXCNDB includes services for creation, update, copy, and extraction. You should use the IATXCNDB service whenever you work with CNDBs.

When an operator issues a command, JES3 creates a CNDB containing information about the source of the command, such as the console identifier and console type. This CNDB is passed to the DSP's console appendage along with the command text. When your DSP needs to communicate with the operator issuing the command, you may use the MESSAGE macro service, specifying the CNDB that was saved by your appendage. In addition, any commands that your DSP may internally issue on behalf of an operator may be issued using the INTERCOM service specifying a CNDB.

CNDBs may also be used for routing of unsolicited messages. For example, JES3 creates CNDBs for much of the message destination information specified in your initialization stream. When you need to issue a message to this destination, such as a message about a particular main processor, you use the MESSAGE service specifying the appropriate CNDB.

12.28 DSP Processing Examples for CNDBs

DSP Messages

```
...
 LA    R0,MYCNDB          GET REQUESTING CONSOLE
 LA    R1,MYMSG           GET MESSAGE TEXT
 MESSAGE CNDB=(R0),  ISSUE MESSAGE
 TEXT=(R1)
...
```

Issuing Commands

```
...
 LA    R1,MYCMD                GET COMMAND TEXT
 INTERCOM CNDB=MYCNDB,  INTERCOM THE COMMAND
 TEXT=(R1)
...
```

Using CNDBs in DSPs

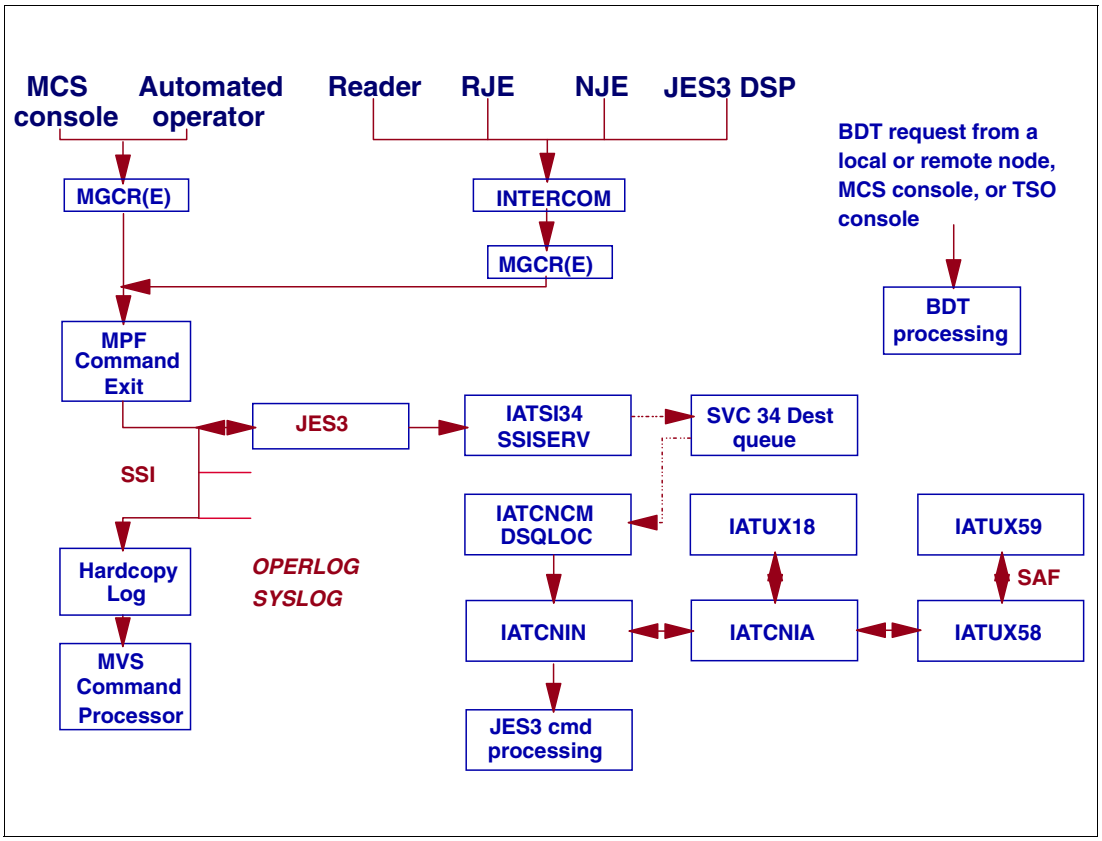
In installation-written DSPs, exits, and other modifications must use CNDBs and the IATXCNDB service when appropriate.

INTERCOM You may need to specify a CNDB on the INTERCOM macro. Use the IATXCNDB service to set up the CNDB if necessary. If a CNDB is not provided, the dummy CNDB is used.

The JES3 INTERCOM service uses MGCRE to issue all commands. MGCRE translates any non-printable data found in the command text as part of its processing. If it is not practical to eliminate the passing of this data with a command, the non-printable data can be enclosed in quotation marks. MGCRE then passes the data unchanged.

MESSAGE You may need to specify a CNDB on the MESSAGE macro. Use the IATXCNDB service to set up the CNDB if necessary.

12.29 JES3 command processing flow



Command processing flow

The visual illustrates the command processing flow. All commands go through the MCS MGCR(E) interface, which serves as the single point of the command entry and all commands are presented to MPF command exits. The MCS MGCR(E) processing invokes the MPF command exits prior to entering the SSI loop for the command processing. JES3 SSI processing (IATSI34) examines input commands to see if they are valid JES3 commands. Valid JES3 command are sent to JES3 through the SSISERV service and further MGCR(E) processing is terminated.

Support for JES3 command entry through a BDT session is deleted. The JES3 commands authorization exit (IATUX56) for commands entered through BDT is also deleted.

MGCR(E) processing

The processing for commands entered from an MCS console or an automated operator interface through MGCR(E) remains the same. When multiple commands are entered through a single input from an MCS console, MCS unstacks commands and sends one command at a time to the SSI. When multiple commands are entered through a single MGCR(E) by an automated operator, the whole stack of commands is sent by MCS to the SSI. If the first command is an MVS command, the JES3 SSI routine returns the whole stack of commands back to MCS for processing. If the first command is a JES3 command, the JES3 SSI routine sends the whole stack of commands to the JES3 address space for processing. The JES3 command processing continues to provide the unstacking service as in prior releases of JES3.

A JES3 DSP can internally issue a JES3 command through the INTERCOM macro (to simulate input of operator command). The INTERCOM macro is converted into an MCS MGCRE.

JES3 command processing (IATCNCM) processes staging areas on the SVC 34 destination queue and enters the commands contained in the staging areas for execution. IATCNIN receives control from IATCNCM to perform input command processing.

JES3 enters all commands, independent of their source, to the system through an internally issued MGCRE macro. The MGCRE processing may invoke one or more MPF command installation exits to affect command processing. The command exits receive control every time a command is entered.

Exit 18

Exit IATUX18 is *not* entered for MVS commands issued from a JES3 source; it is entered only for JES3 commands. In the visual, for JES3 commands, IATCNIN calls the JES3 console authorization checking module (IATCNIA) to validate command authority. IATCNIA calls user exit IATUX18. This exit routine allows you to modify a JES3 command and validate the console's authority to enter the command. If the operator enters a JES3 command at a console that has been defined as not having a high enough authority level for that command, the command is rejected. You could use this exit to allow a particular command to be issued from a console whose definition would reject the command. A new return code (16) is provided which may be used when a command is completely processed within the exit. When this return code is used, JES3 performs no further processing for the command.

Exit 58

The JES3 command authorization process includes invocation to the security authorization facility. Before JES3 calls SAF to perform security processing, exit IATUX58 is given control. This exit allows you to modify security checks or to make your own security decisions for JES3. After the SAF call, JES3 calls exit IATUX59. Also this exit gives you the opportunity to modify security checks or to make security decisions for JES3. After successful completion of the command authorization checks, the command is passed to the appropriate command executor.

Exit IATUX18 continues to be called for *all* JES3 commands independent of their source. In summary, it is recommended to continue to have all installation JES3 commands processing in the JES3 command exit, and move all installation non-JES3 commands processed in to MPF command installation exits (or other facilities).

MGCR(E) processing allows one or more MPF command exits to affect command processing. You specify on the MPFLSTxx parmlib member the MPF command exits. SET MPF operator command lets you dynamically change the MPFLSTxx member configuration and the exit specifications. Up to six command installation exits can be defined. These exits can:

- Change the text of commands.
- In a sysplex, change the destination of commands by routing them to a different system for execution.
- Modify a console's authority to use a particular command. That is:
  - Authorize the command from a console that normally would not have the authority to issue the command
  - Reject the command from a console that normally would have the authority to issue the command
- Execute commands.
- Suppress commands.

12.30 Command processing

- Command length of 126 characters only for:
  - *I U
  - *F U
  - *X JMF
- JES3 command processors support a CART
  - CART kept in CNDB
- BDT no longer supports JES3 commands
  - IATUX56 deleted
- INTERCOM macro still supported

JES3 command processing

JES3 5.2.1 allows the *I U, *F U, and *X JMF commands to be up to 126 characters long. In addition, commands may be stacked (multiple commands on a single line) up to 126 characters, provided each command in the stack adheres to its length restriction.

Subsystems and applications can exploit without restrictions all MCS facilities (for example, CART and 4-byte console ID). The JES3 command processors support 4-byte console IDs and the MCS command-and-response token (CART).

- JES3 supports the CART to insure accurate presentation of command responses to the extended MCS console. The CART is carried in the CNDB throughout the command processing.

Delete any installation-written code contained in the JES3 BDT command authorization exit IATUX56. JES3 5.2.1 no longer calls IATUX56.

You may need to specify a CNDB on the INTERCOM macro. Use the IATXCNDB service to set up the CNDB if necessary. If a CNDB is not provided, the dummy CNDB is used.

The JES3 INTERCOM service uses MGCRE to issue all commands beginning with JES3 5.2.1. Previously, JES3 commands were queued directly to JES3 console services. MGCRE translates any non-printable data found in the command text as part of its processing. If it is not practical to eliminate the passing of this data with a command, the non-printable data can be enclosed in quotation marks. MGCRE then passes the data unchanged.

12.31 Command and message considerations

- Summary end lines for commands
  - *I A - *I G - *I P - *I Q
- Message destinations
  - Can use JES3 destination class - or -
  - MCS routing code
- IATXMLWO macro for multi-line messages

Message and command considerations

Some JES3 inquiry command responses are changed to include a summary message, so that the end of the command response can be easily identified by automation applications. These commands are: **`*I A`**, **`*I G`**, **`*I P`**, and **`*I Q`**.

Consider using MVS routing codes in the initialization stream and in JES3 commands in addition to, or in place of, JES3 destination classes.

Review your user written DSPs and exits to determine if they can benefit from the JES3 multi-line message service, IATXMLWO. The IATXMLWO and MESSAGE macro services allow JES3 functions to issue true multi-line WTOs. The IATXMLWO macro creates one line of a multi-line message. Each line of a multi-line message is stored in its own copy (IATYMLWO token). These lines (tokens) are chained together and sent to the MESSAGE macro to issue the multi-line WTO message. A multi-line WTO message is limited to a maximum of 999 lines.

12.32 Command processing exit IATUX18

- User Exit IATUX18
  - Command modification and authority validation
  - Only sees JES3 commands
  - MVS commands must use MVS exits
  - Command buffer contains a CNDB
    - Console name and ID
    - IATXCNDB macro must be used to access CNDB
- IAT7107 COMMAND CHANGED BY EXIT..
  - Message issued when exit 18 changes a command

Using exit IATUX18 (Command modification and authority validation)
With the removal of the TYPE=MCS and automation console definitions from the JES3 initialization stream, it is no longer possible to define JES3 command authority levels for these consoles in the initialization stream for use by IATUX18.

In addition, installation exit code related to non-JES3 commands entered from JES3 sources must be moved to an MVS command exit. MVS command exits run in the address space from which the command is issued. Therefore, any code you move to an MVS command exit will still run in the JES3 address space, and JES3 address space data such as device group will continue to be available. JES3 provides a new return code for IATUX18 to allow the exit to indicate that it has completely processed a command. Previously, JES3 would issue an error message for any unrecognized command processed within your exit code.

- Move any IATUX18 code related to non-JES3 commands to an MVS command exit.
- Update your IATUX18 code due to the JES3 5.2.1 changes in the exit interface.

When processing commands use information on from the command buffer which is mapped by the IATYCNS macro with the TYPE=INPUT parameter. Use the IATXCNDB service to obtain information from the console destination block which is provided in IATYCNS. The console destination block identifies the console issuing the command.

The IAT7107 message indicates that a JES3 command is modified by user exit 18.

12.33 Displaying JES3 action messages

- MVS Action Message Retention Facility (AMRF)
  - Action messages are saved
  - MVS D R command displays outstanding actions

D R,L
D R,L,SYS=main
D R,L,KEY=dspname
D R,I,KEY=MOUNT
D R,I,KEY=MOUNT,JOB=jobname

Displaying JES3 action messages

The MVS message retention means that action messages are saved by the action message retention facility (AMRF) on the action message retention queue. Retention allows the operator to view the message later. If you choose not to retain a message, the system will not add it to the action message retention queue.

You change the status of the action message retention facility by:

- Use the **`CONTROL M,AMRF`** command to turn the action message retention facility on or off.

Following forms of the DISPLAY command displays outstanding messages requiring operator action. These messages include WTOR messages, action messages saved by AMRF, action messages issued by the communications task, and action messages that were not displayed on all necessary consoles:

D R,L	SETUP-related messages are displayed by the D R command. The D R command also displays messages that were issued before JES3 was fully initialized.
D R,L,SYS=main	All messages issued from the named system are displayed by the D R command.
D R,L,KEY=dspname	No major differences.
D R,I,KEY=MOUNT	No major content differences. Change in the KEY name from prior releases.

D R,I,KEY=MOUNT,JOB=jobname	The D R command uses job name rather than job number.
D R,I,ROUT=nnn	The route code equivalent of the JES3 destination class must be used on the Display command.

JES3 use of the MVS action message retention facilities

MVS retains action messages and WTOR messages that did not receive a response. JES3 depends on MVS outstanding operator action messages retention.

12.34 Displaying action messages

```
IAT5110 JOB VAINITA  (JOB19898) GET      T IGNORE ,SL NOT.REAL.DATA.SET
IAT5200 JOB VAINITA  (JOB19898) IN SETUP ON MAIN=SC50
*IAT5210 JOB VAINITA  (JOB19898) SC50     MOUNT T IGNORE ON 0B3F ,SL,
 NORING NOT.REAL.DATA.SET
D R,L,KEY=MOUNT
IEE112I 17.35.50 PENDING REQUESTS 774
RM=7    IM=8     CEM=3     EM=31    RU=0    IR=0    AMRF
ID:R/K     T SYSNAME  JOB ID   MESSAGE TEXT
    29032 I SC50     VAINITA  *IAT5210 JOB VAINITA  (JOB19898) SC50
                      MOUNT T IGNORE ON 0B3F ,SL,NORING  NOT.REAL.DATA.SET
D R,L,KEY=MOUNT,JOB=VAINITA
IEE112I 17.37.05 PENDING REQUESTS 776
RM=7    IM=8     CEM=3     EM=31    RU=0    IR=0    AMRF
ID:R/K     T SYSNAME  JOB ID   MESSAGE TEXT
    29032 I SC50     VAINITA  *IAT5210 JOB VAINITA  (JOB19898) SC50
                       MOUNT T IGNORE ON 0B3F ,SL,NORING  NOT.REAL.DATA.SET
```

JES3 action messages

This visual shows the displaying of JES3 action messages using the MVS commands. All JES3 action messages have an asterisk (*) before the message ID.

After the JES3 action messages are issued, operator enters:

- **D R,L,Key=MOUNT**

  This is for JES3 issued mount messages.

- **D R,L,Key=MOUNT,JOB=VAINITA**

  This is for JES3 mount messages for a specific jobname or the job number could have been used.

12.35 Sysplex-Wide WTOR Replies

- Single WTOR reply ID for sysplex
- Reply ID range up to 9999
  - CONSOLxx DEFAULT RMAX=9999
  - K M,RMAX=9999
- JES3 WTORs
  - 32,U or 8,U
    - Prefix 8 not registered with CPF

WTOR replies in a sysplex

In the MCS sysplex environment, a single range of WTOR reply ids is shared by all systems.

You want to change the maximum value of the WTOR reply id from its default of 99. Use the RMAX keyword on the DEFAULT statement.

RMAX specifies the maximum value of a reply ID in the sysplex. RMAX also determines the size of the reply ID displayed in the message text. For example, specifying an RMAX value of 9999 results in all messages having a four character reply ID.

Use the following command to dynamically increase the maximum number of reply IDs:

```
K M,RMAX=nnnn
```

13

MVS System Logger/JES3 DLOG

The hardcopy log records command and message traffic for your systems. MVS and JES3 provide three forms of the hardcopy log:

OPERLOG OPERLOG centrally records command and message traffic for systems in a sysplex in Message Data Block (MDB) format. The OPERLOG is created using the MVS System Logger.

JES3 DLOG JES3 DLOG centrally records command and message traffic for systems in a JES3 complex in JES3 format. The JES3 DLOG is written to SYSLOG on the global processor.

SYSLOG SYSLOG individually records command and message traffic for each system in MVS format.

You must have at least one active hardcopy log on each of your systems.

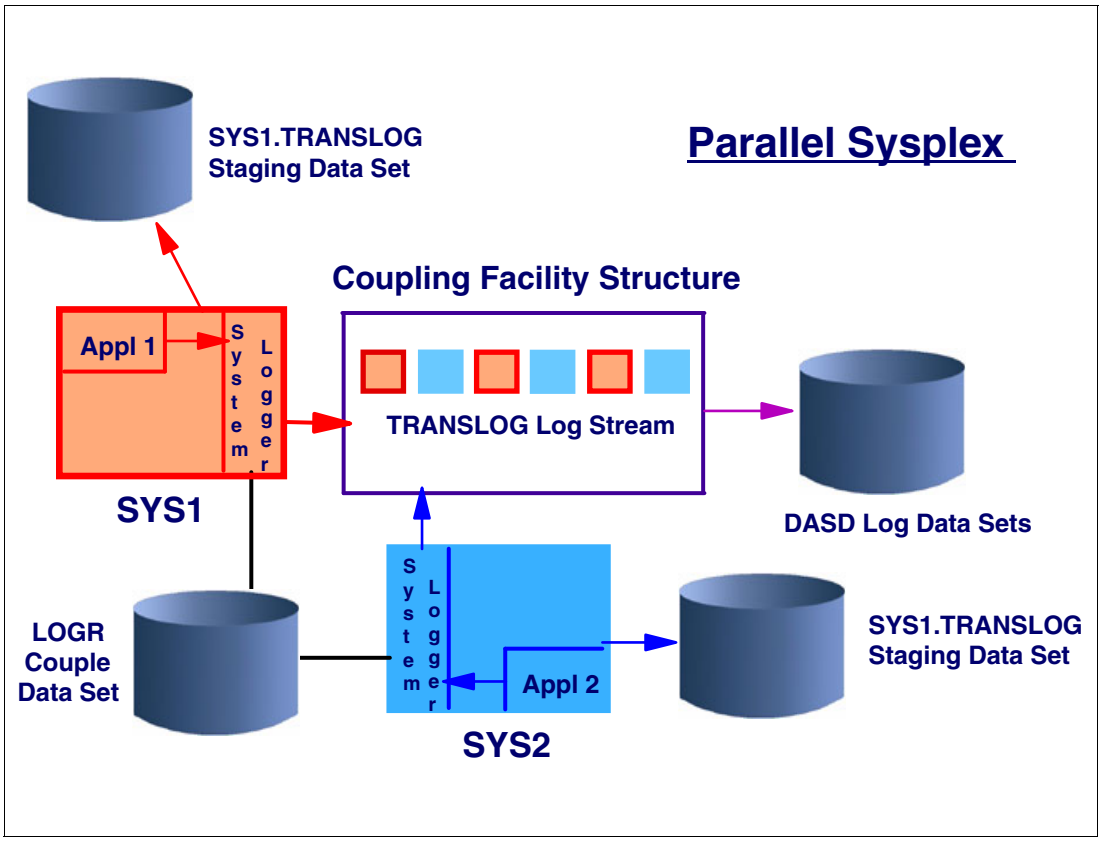
OPERLOG is controlled on each processor using the **VARY OPERLOG** command.

When you use the disk log facility, the log is spun off at installation-defined intervals and processed by JES3 output service according to the DLOG output class. If DLOG is active, you can force the log to be spun off before the installation-defined threshold occurs by entering the MVS **WRITELOG** command on the global. For example, to spin off the DLOG and direct it to output class D, enter **WRITELOG D**.

The JES3 DLOG is controlled using the ***MODIFY O,DLOG=ON|OFF** command.

SYSLOG on the global processor must be active when DLOG is active.

13.1 MVS System Logger



System Logger

System Logger is an MVS component that allows an application to log data from a sysplex environment. Data can be logged from one system or from multiple systems across the sysplex.

The z/OS System Logger is a set of services that allows an application to write, browse, and delete log data. You can merge the log data from applications in a sysplex into a log stream, which is simply a collection of data in log blocks residing in the coupling facility. Log blocks in the coupling facility can be backed up either in storage in each system or on DASD in staging data sets. When the log blocks in the coupling facility reach an installation-defined threshold value, they are offloaded to DASD log data sets. Therefore, at any point in time the log stream consists of records on the DASD log data sets and the log blocks currently in the coupling facility.

System Logger components

A z/OS System Logger configuration includes the System Logger address space in each system, the LOGR couple data set, a log stream structure in a coupling facility, DASD log data sets for offloaded data from the coupling facility log stream, and optionally staging data sets for a backup copy of the log blocks resident in the log stream structure. All these components are shown in the figure.

System Logger address space

The System Logger address space provides the application with services and connections to the coupling facility. An installation must plan and predefine the coupling facility structures,

format and define a policy for the LOGR couple data set, define the DASD log data sets, and optionally create staging data sets. An application can then issue System Logger services.

Log streams

A log stream is a collection of one or more log records (also referred to as log blocks) written by an application using services provided by the MVS System Logger. The application using MVS System Logger services may or may not have multiple instances of itself executing in a sysplex. In the case of a multi-instance application where each instance of the application writes log blocks to the same log stream, the result is a sysplex-wide merged log stream.

System Logger services

You can use System Logger services to merge data from different systems in a sysplex, as shown in the figure where APPL1 and APPL2 are multiple instances of an application writing log blocks into the coupling facility into the TRANSLOG log stream. The System Logger provides a set of system services that allow an application to:

- Connect to and disconnect from a log stream (IXGCONN)
- Browse a log stream (IXGBRWSE)
- Write to a log stream (IXGWRITE)
- Delete data from a log stream (IXGDELET)
- Define, update, and delete log stream and coupling facility list structure definitions in the LOGR policy, (IXGINVNT service or IXCMIAPU service)

13.2 MVS System Logger services

- Executes in own address space
- Provides MVS services
  - Connect to a log stream (IXGCONN)
  - Write data to a log stream (IXGWRITE)
  - Browse data from a log stream (IXGBRWSE)
  - Delete data from a log stream (IXGDELET)
  - Disconnect from a log stream (IXGCONN)
  - Maintain an inventory of log streams (IXGINVNT)

System Logger services

The System Logger component resides in its own address space on each system in a sysplex. The System Logger component provides a set of system services that allow an application to:

- Connect to and disconnect from a log stream (IXGCONN)

  IXGCONN connects and disconnects an application to and from a log stream.
- Browse a log stream (IXGBRWSE)

  IXGBRWSE browses (reads) data from a log stream.
- Write to a log stream (IXGWRITE)

  IXGWRITE writes user-defined log data to a log stream.
- Delete data from a log stream (IXGDELET)

  IXGDELET deletes data from a log stream.
- Define, update and delete log stream and coupling facility list structure definitions in the LOGR policy, (IXGINVNT service or IXCMIAPU service)

  IXGINVBT defines and maintains log stream and coupling facility structure information in the LOGR policy dynamically.

  You can also use the IXCMIAPU utility to specify log stream and structure definitions in the LOGR policy. IXCMIAPU also enables you to request a report of current log stream definitions.

13.3 MVS operations log (OPERLOG)

- MVS console services function providing:
  - Sysplex-wide merged and ordered message log
  - Messages kept in message data blocks (MDBs)
    - MDB - message text and control information
- Independent of SYSLOG
  - SYSLOG records are MVS format
- Contains records for all active systems
- Requires a coupling facility or DASD-only log stream

OPERLOG processing

The operations log is a log stream that uses the System Logger to record and merge communications about programs and system functions from each system in the sysplex. The OPERLOG provides a sysplex-wide merged and chronologically ordered message log.

IBM recommends that JES3 customers with a multisystem sysplex use an OPERLOG coupling facility log stream and turn off JES3 DLOG and SYSLOG.

You can also use OPERLOG as a DASD-only log stream. This method is only suitable for a single system sysplex, because a DASD-only log stream is single-sysplex in scope and you can only have one OPERLOG log stream per sysplex. This means that if you make OPERLOG a DASD-only log stream, only one system can access it.

Message data blocks

The messages are logged using message data blocks (MDB), which provide more data than is recorded in the SYSLOG. You can use the sample program IEAMDBLG, in SYS1.SAMPLIB, to convert OPERLOG records into SYSLOG format.

OPERLOG and syslog

The operations log is operationally independent of the system log. An installation can choose to run with either or both of the logs. If you choose to use the operations log as a replacement for SYSLOG, you can prevent the future use of SYSLOG; once the operations log is started with the SYSLOG not active, enter the WRITELOG CLOSE command.

13.4 JES3 DLOG

- DLOG started as address space on global
  - Establishes EMCS console
  - Receives messages from all systems in sysplex
  - Formats messages in JES3 format
    - Writes to SYSLOG
- CONSTD,.....DLOG=ON|OFF
  - *F O,DLOG=ON|OFF

JES3 DLOG

The hardcopy medium (also known as the hardcopy log) records command and message traffic for your systems. MVS and JES3 provide three forms of the hardcopy medium:

OPERLOG Centrally records command and message traffic for systems in a sysplex in Message Data Block (MDB) format.

JES3 DLOG Centrally records command and message traffic for systems in a JES3 complex in JES3 format. The JES3 DLOG is written to SYSLOG on the global processor.

SYSLOG Individually records command and message traffic for each system in MVS format.

DLOG EMCS console

The DLOG message traffic is managed by MCS. On the global, JES3 activates an extended MCS console with the HARDCOPY=YES attribute to receive the hardcopy message set. The messages received by the DLOG EMCS console are in the MDB format and are converted to the JES3 DLOG format and then written using the WTL macro service to the global JES3 system's SYSLOG (alias DLOG).

The initial state for DLOG is defined on the CONSTD initialization statement with the parameter DLOG=ON or DLOG=OFF, as shown in the figure. The operator can turn the DLOG on or off with a command.

13.5 JES3 DLOG function

- Implemented as a migration tool
- Sysplex-wide log written to spool
  - Contains command and message traffic for all systems in the JES3 complex
- MPF message suppression
- DLOG message format

```
743584 -S008= VAINI    D T
743587  S008= VAINI    IEE136I LOCAL  TIME=17.43.58 DATE=1994.271 GMT TIME=21.43.58  DATE=
743587  S008= VAINI    1994.271
```

JES3 DLOG

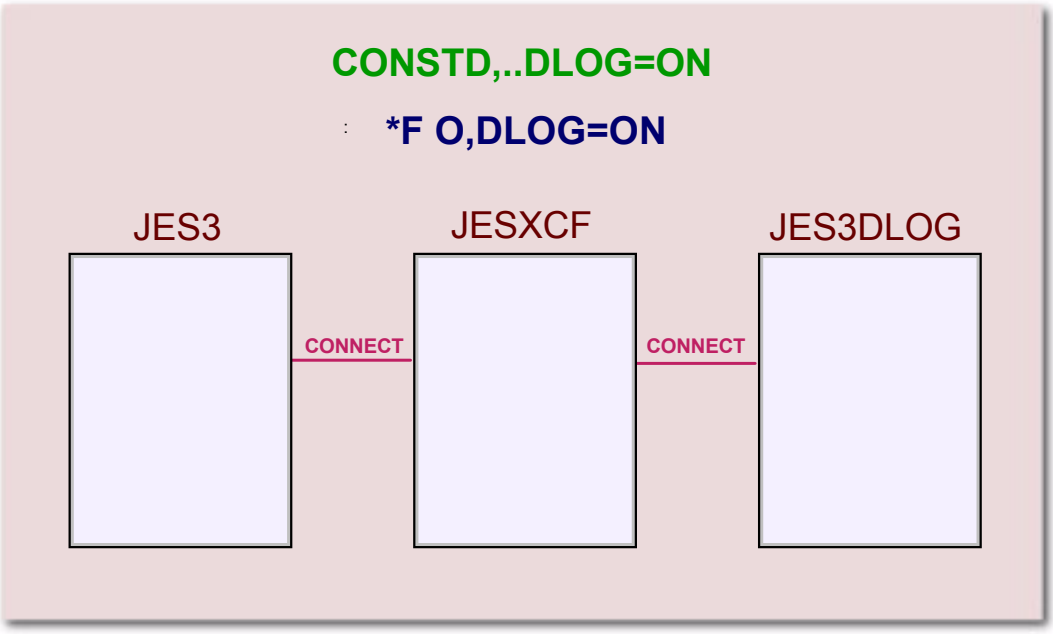
JES3 provides a migration accommodation (DLOG) for customers who are unable to activate the MVS OPERLOG across the JES3 complex. A sysplex-wide log is written from the global processor in the JES3 DLOG format as shown in the figure. The JES3 DLOG may be used as part of a staged migration to OPERLOG. The JES3 DLOG, when active, contains command and message traffic for all systems in the JES3 complex. OPERLOG, on the other hand, may be activated on a system by system basis during the migration period.

For DLOG, the system log is spooled and periodically can be printed by JES3 output service. By default, the log is printed every 500 lines to output class A. To change these defaults, code the LOGLMT and LOGCLS parameters of member IEASYSnn of the MVS SYS1.PARMLIB data set. You can also print the disk log by entering the MVS **WRITELOG** command from the global.

MPF message suppression

For messages suppressed by MPF processing, the MPF suppression character in the log record will be the global processor's suppression character, rather than the character for the originating system. MPF suppression characters are defined in parmlib member MPFLSTxx using the MPFHCF= keyword. The default is an ampersand (&).

13.6 JES3 related address spaces



IAT7114 DLOG INITIALIZATION SUCCESSFUL

IAT7124 DLOG IS NOW ACTIVE - (Local processors)

Activating the DLOG

The JES3 DLOG runs in its own address space, as shown in the figure. This address space is started under the MSTR subsystem through an internally issued start command. Once initialized, the JES3 DLOG address space activates an extended MCS console (NAME=SYSJ3Dxx KEY=xcfgroupname) that is set to receive the hardcopy message set. The figure shows console messages for a DLOG activation/deactivation.

DLOG address space

The JES3DLOG address space tracks all message activity in a sysplex in the JES3 format. It is a separate address space that uses an MCS extended console. Installations can choose JES3DLOG or OPERLOG. JES3DLOG problems are indicated by:

- External symptoms similar to JES3 such as:
  - Performance - high CPU usage
  - Hangs

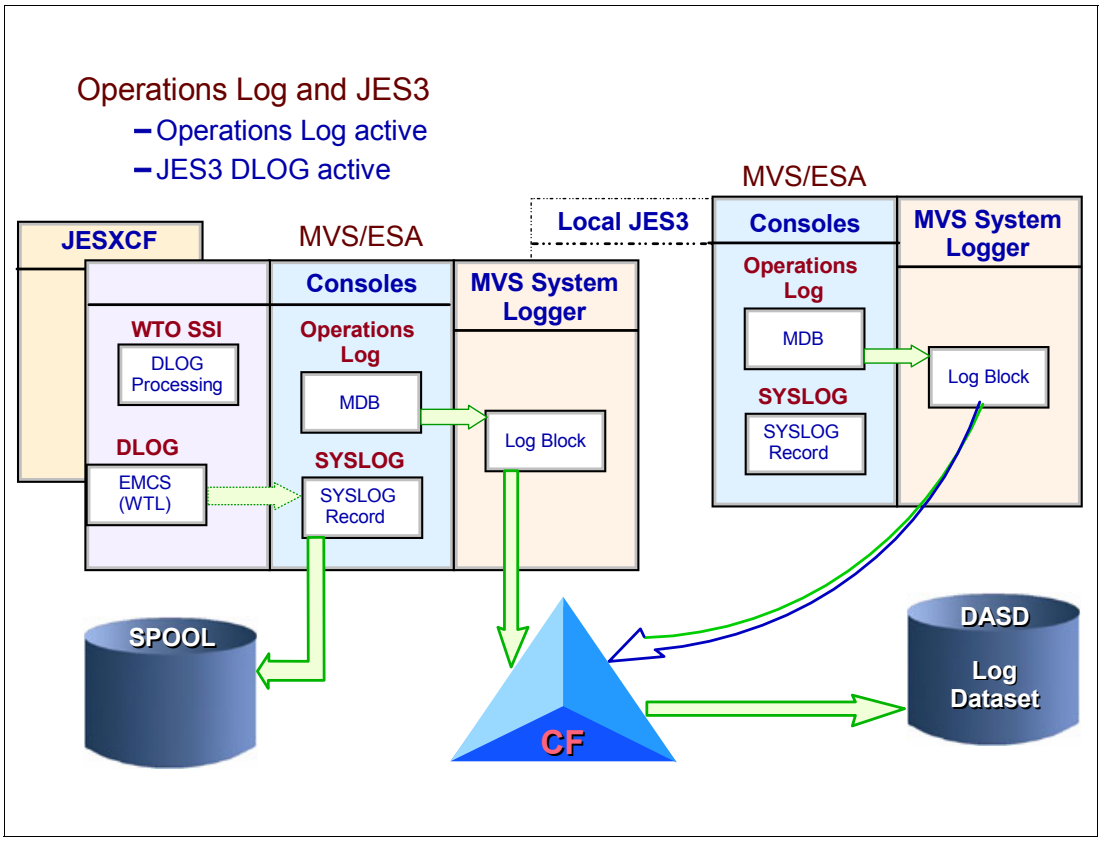
DLOG problems

The DLOG will obtain MDBs (message data blocks) from the consoles data space to create the messages it will put in the log.The following documentation is required for diagnosing JES3DLOG problems:

- A dump of JES3DLOG address space and the data space associated with the Consoles ASID that is created for DLOG.

```
JOBNAME=(CONSOLE,JES3DLOG),DSPNAME=('console'.ieam*)
```

13.7 MVS OPERLOG and DLOG



OPERLOG and JES3

An installation can use MVS Operations log (OPERLOG), the JES3 DLOG facility, the system log facilities (SYSLOG). Messages, commands, and command responses are recorded in the log. The system uses routing codes associated with a message to determine which messages are written to the log.

Message flow to the DLOG

The DLOG message traffic is managed by MCS. On the global, JES3 activates an extended MCS console with the HARDCOPY=YES attribute to receive the hardcopy message set, as shown in the figure. The messages received by the DLOG EMCS console are in the MDB format and are converted to the JES3 DLOG format and then written using the WTL macro service to the global JES3 system's SYSLOG (alias DLOG).

The JES3 WTO subsystem interface routines receive control as part of the MCS WTO or WTL processing. If the DLOG is active, the JES3 WTO SSI processing suppresses the MCS logging of operator commands and WTO messages into the SYSLOG. JES3 never suppresses the MCS logging into the OPERLOG log stream.

14

RJP and NJE

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

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

JES3 offers installations two kinds of protocols:

- Binary synchronous communication (BSC) multi-leaving protocols, where a separate communication line is needed for each device at a workstation.
- IBM systems network architecture (SNA) protocols, where many devices can share a communication line.

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

BSC remote job processing

Features of BSC remote job processing include:

- Password protection for an RJP line, a device, or both.
- Message routing. JES3 sends job-started, job-ended, and abnormal-ending messages.
- Support for programmable and non-programmable devices.
- Output suspension for non-programmable devices.
- Compressed (removal of repeated characters) data support for inbound data.
- Connection through leased or dial-up communication lines.
- Remote console support.
- Work station operator inquiry by data set origin or destination.
- Error recovery and error statistics.
- Message queuing for signed-off consoles.
- System management facilities (SMF) recording.

- Interface with remote terminal processor (RMT) programs for managing devices connected to processors at workstations.

SNA remote job processing

Features of SNA remote job processing include:

- Password protection for accessing an RJP workstation.
- Message routing. JES3 sends job-started, job-ended, and abnormal-ending messages.
- SNA data transmission protocols.
- Connection through leased or dial-up communication lines.
- Multiple logical unit support for concurrent data transmission between JES3 and the console, readers, and printers at a workstation.
- Compression and compaction (substitution of characters according to rules).
- ASCII support. JES3 can handle ASCII or EBCDIC character sets.
- Block size selection.
- Remote console support.
- Work station operator inquiry by data set origin or destination.
- Error recovery and error statistics.
- Message queuing for signed-off consoles.
- System management facilities (SMF) recording.

JES3 network job entry (NJE)

The JES3 network job entry (NJE) facility is similar to remote job entry (RJP) in that they both provide extensions to a computer system. In its simplest terms, NJE is "networking" between systems that interact as peers, whereas RJP is networking between JES3 and workstations. The main difference between them is one of overall compute power and processor location. Remember, RJP is an extension of a single JES3 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 processor complexes into a processing network. Each complex 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.

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

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

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

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

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

14.1 Defining BSC remote job processing

- RJPLINE (BSC Remote Job Processing Line)
  - **RJPLINE,N=linename,A={[/]devnum | (main,devnum,[main,devnum]...)} [,F=DIAL] [,F=NTRS] [,G=grpname] [,I={A | B}] [,I={Y | N}] [,O=AUTO] [,P=password] [,S={2400 | linespeed} [,T=termname]**
- RJPTERM (BSC Remote Job Processing Terminal)
  - **RJPTERM,N=linename,T=termtype [,B={400/512 | buffsiz} [,C={R | S}] [,F=HTAB] [F=NTRS] [,F=PRES] [,F=XBUF] [,G=grpname] [,O=AUTR] [,O=FBIX] [,O= {DC2 | DC3}] [,O=CCNT] [,PR={0 | numofprt}] [,PRW={132 | recsiz,NOTRUNC}] [,PU={0 | numofpun}] [,PUW={80|recsiz}] [,RD={0 | numofrdr}] [,P=password] [,CS={Y|B|N}] SET={N|Y|V|B}]**
- Use the CONSOLE statement to define console support at a remote work station.
- Use the DEVICE statement to define remot printers and punches.

RJPLINE (BSC remote job processing line)

Use the RJPLINE initialization statement to define the characteristics of a single BSC line (and its respective adapter) that will be used by the JES3 global for remote job processing. You can also use this statement to assign a specific RJP work station, defined by the N parameter of an RJPTERM statement, to this line. Considerations:

- One RJPLINE statement is required for every adapter to be accessed by RJP.
- The T parameter (terminal to be assigned to this line) must match the N parameter on an RJPTERM statement and the JNAME parameter on a CONSOLE statement.
- Use the password parameter (P) to specify a e password used to protect the RJP line.

RJPTERM (BSC remote job processing terminal)

Use the RJPTERM initialization statement to define a single remote BSC work station to the JES3 system. This statement causes a default description to be provided for each work station device (printer, punch, or card reader) indicated by the PR, PU, or RD parameters along with the operating characteristics of the work station. If the JES3 default characteristics for a remote printer or punch device are not acceptable, a DEVICE statement should be coded to indicate desired characteristics. If a work station is to have the facilities of a JES3 operator console, then a CONSOLE statement must be coded.

Modifications with a command (*F T) or JES3 restart with cold, warm, or hot start with refresh for all parameters.

14.2 Defining SNA remote job processing

- COMMDEFN (Communication Definition Records)
  - **COMMDEFN [,APPLID={JES3 | applname}] [,P=password] [,LU={255 | nnnn}]**
- RJPWS (SNA Work Station Characteristics)
  - **RJPWS,N=name [,RD={1 | nn}] [,PR={1 | nn}] [,PU={0 | nn} [,C={R | S}] [,G=grpname] [,COMPACT={NO | comptbl} [,P=password] [,PL={2 | n}] [,LU=luname [,luname]...] [,AUTO={N | (Y,luname,modetab)}] [,TRACE=ON] [,SETUP={NO | YES | VARY | BOTH}]**
- Use the CONSOLE statement to define console support at a remote work station.
- Use the DEVICE statement to define remote printers and punches.

COMMDEFN (communication SSI definition)

Use the COMMDEFN statement to specify the optional user communication subsystem interface (VTAM) parameters. Considerations:

- Ensure that the value you specify on the P (password) parameter is the same as the password specified on the PRTCT parameter of the VTAM APPL application definition.

RJPWS (SNA work station characteristics)

Use the RJPWS initialization statement to describe each SNA work station's characteristics to the JES3 system. This statement causes a default description to be provided for each work station device (printer, punch, or card reader) indicated by the PR, PU, or RD parameter along with the operating characteristics of the work station. Considerations:

- Use the CONSOLE statement to define console support at a remote work station. You can use the **`*F CONFIG`** command to add a console.
- Use the DEVICE statement to define remote printers and punches. You can use the **`*F CONFIG`** command to add additional devices to the remote work station.
- If the JES3 default characteristics for a remote printer or punch device are not acceptable, you should code a DEVICE statement to indicate desired characteristics.

Modifications with a command (**`*F T`** and **`*F CONFIG`**) or JES3 restart with cold, warm, or hot start with refresh for all parameters.

See z/OS *JES3 Initialization and Tuning Reference*, SA22-7550 for keyword parameter details.

14.3 Controlling RJP

- Activating RJP
  - BSC - *X RJP
  - SNA - *X SNARJP
- Restarting RJP
  - BSC - *R RJP,L= lname [,I]
  - SNA - *R SNARJP,T= wsname [,I]
- Stopping RJP
  - BSC - *C RJP [,L={lname | ALL}] [,I]
  - SNA - *C SNARJP,T={wsname | ALL} [,I]
- Changing RJP
  - *F T...
  - SNA - *F CONFIG,ADD=mem [,LOG={YES|NO|ERR}] [,P=xxxxxxxx]
- Displaying RJP
  - BSC - *I T,L={(lname[,lname]...) | ALL[,P | ,STAT[,R]}
  - SNA - *I D,WS={(wsname[,wsname]...) | ALL [,ONLINE | TRACEON]}
- Signing on or off at a BSC RJP workstation
  - Workstation operator must use the /*SIGNON card to sign on and off

Controlling JES3 RJP

Activating RJP:

- Use the **`*X RJP`** command to activate BSC RJP. When BSC RJP is operational, you will receive message IAT7500. BSC RJP immediately starts any lines for which automatic start was specified during initialization.
- Use the **`*X SNARJP`** command to activate SNA RJP. When SNA RJP is operational, you will receive message IAT2801.

  If AUTO=(Y,luname) is coded on the RJPWS initialization statement for a workstation, the indicated logical units (LUs) at that workstation are automatically logged on when SNA RJP becomes active. Before the SNARJP DSP can be called, VTAM must be active and the required VTAM networks must be varied online.

Restarting RJP:

- Use the **`*R RJP`** command to end a BSC RJP session or activity on any line and then start it again. The command can be used to end activity immediately or as though the normal workstation sign-off occurred. This command has the same effect as entering an **`*C RJP`** command followed by an **`*S RJP`** command for the same line(s). After the line is restarted, communication with the workstation must be reestablished through the workstation start-up procedure.
- Use the **`*R SNARJP`** command to end a SNA RJP workstation and then start it again. This command has the same effect as entering an **`*C SNARJP`** command followed by an **`*S SNARJP`** command for the same workstation. It can be used to end activity immediately or

conditionally. After the workstation is restarted, communication with the workstation must be reestablished by using the workstation start-up procedure.

Stopping RJP:

- Use the ***C RJP** command to stop a BSC RJP session or activity on any line. The command can be used to stop activity immediately or as though a normal workstation sign-off occurred.
- Use the ***C** command to halt the SNA RJP network, a SNA RJP workstation, or processing on a SNA RJP device.

Changing RJP:

- Use the ***F T** command to:
  - Specify the action to be taken if a remote printer or punch becomes "not ready".
  - Assign a password to a line or to specify that no password is required.
  - Hold or release jobs that are being submitted from BSC RJP workstations.
  - Hold or release jobs on the JES3 job queue that are being submitted from a SNA RJP.
  - Specify whether a line will be started automatically when BSC RJP is reinitialized.
  - Control the RJPSNPS facility.
  - Disable or enable an automatic reader at a SNA RJP workstation.
  - Disable or enable SNA RJP tracing.
  - Change the number of times an incorrect password is allowed from a SNA RJP workstation before logons are inhibited.
  - Change the group name on a SNA or BSC RJP workstation.
- You use the ***F CONFIG** command to make configuration dynamically changes to SNA RJP definitions. The command is equivalent to adding the following SNA RJP initialization statements to the initialization stream:
  - RJPWS - SNA RJP Workstation Characteristics
  - CONSOLE - SNA RJP Consoles
  - DEVICE - SNA RJP Devices

Signing on or off at a BSC RJP workstation:

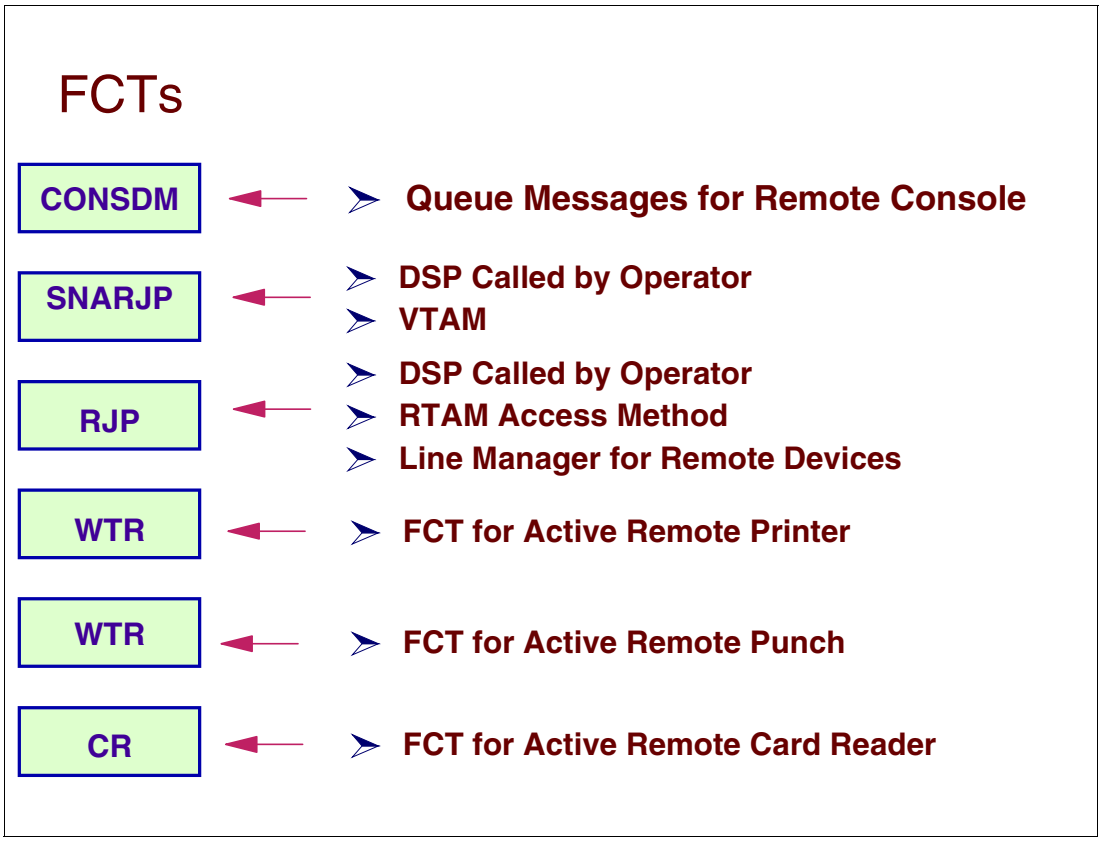
- Before data transmission to a BSC remote workstation can begin, the workstation operator must sign on to JES3. When BSC RJP is active and the line started, use the /*SIGNON card to sign on to JES3. Format:

```
Column    Description
1-8       /*SIGNON
16-20     Workstation name: a five-character name for the remote
          workstation requesting sign-on.
22        A (or blank): A, for programmable workstations only, specifies
          automatic reader.
23        R (or blank): R, for nonprogrammable workstations only,
          specifies the output suspension feature.
25-32     Line password: a one- to eight-character password which must
          have been defined on the RJPLINE initialization statement.
35-42     Workstation password: a one- to eight-character password which
          must have been defined to RACF if RACF password protection for
          RJP is active.
44-51     New workstation password: You can use these columns to provide a
          new password if RACF password protection for RJP is active and
          you want to change the workstation password.
```

  The workstation can end the use of the BSC RJP line by submitting a /*SIGNOFF card.

  Systems network architecture (SNA) remote work stations must use the LOGON command instead of the /*SIGNON statement to notify JES of a connection request.

14.4 Remote job processing (RJP)



RJP FCTs

This figure shows the FCTs used for processing of RJP functions, both SNA and BSC.

CONSDM The CONSDM DSP processes JESMSG and RJP message spooling.

SNARJP Synchronous data link control (SDLC) protocols within the network architecture (SNA) is used. Remote devices using SDLC protocols are managed by the SNARJP DSP. With SNA RJP, line protocols are managed VTAM.

RJP BSC RJP (binary synchronous communication remote job processing) include two logical sections: the RJP line manager DSP, which controls all the line activities; and the remote terminal access method (RTAM), which blocks data into and deblocks data out of the appropriate transmittal buffers. The RJP line manager and the RTAM RJPPUT processing routines can execute simultaneously under different tasks; but the line manager must run under the primary task (IATNUC) because it interacts with other non-multi-tasked JES3 DSPs.

WTR Output can be sent to a variety of devices. For RJP attached devices that are not driven by the JES3 global address space, the WTR DSP passes output to the appropriate interface. This can be RJP, SNARJP, or NJE services.

CR Reader processing takes place in the JES3 global address space. The reader phase reads jobs from any of the sources mentioned above and place the jobs on JES3 spool in batches for later processing. Jobs submitted from BSC RJP remote stations are processed as if they come from a local card reader. Jobs from SNA RJP use a special logical record (LR) interface.

14.5 RJP/NJE console support

- RJP/NJE console support part of JESXCF
- JESXCF uses a EMCS console to manage
  - RJP message receipt and distribution
    - SYSJ3Rxx
  - NJE message receipt and distribution
    - SYSJ3Nxx
- RJP FCT (RJPCONS)
  - Transfers messages from JESXCF to RJP w/s
- NJE FCT (NJECONS)
  - Invokes JESXCF services to issue network commands and retieve command responses

RJP/NJE console support

JESXCF uses extended MCS consoles to implement the JES3 RJP and NJE console support. The JES3 subsystem communications services makes use of a set of MVS services, JESXCF, in order to perform its function. The JESXCF component provides two important features used extensively by the JES3 subsystem communications services:

- Message transport services

  When JSERV or SSISERV is called, JES3 subsystem communications takes the information provided by the requester and builds a staging area. Then, based on the request type, it calls a JESXCF service to build a JESXCF message out of the staging area and then transport the message to its destination.

- Message mailboxes

  When SSISERV calls JESXCF to transport a staging area, it provides to JESXCF the name of the mailbox in which to place the message. When the DSQLOC service is called to process a DSQ, it receives all outstanding staging areas from the JESXCF mailbox associated with the DSQ. These staging areas are queued to the DSQ staging area queue before control is returned to the caller.

RJP consoles

The RJP console services processor (RJPCONS) processes all messages from JESXCF destined for an RJP workstation by invoking JESXCF to obtain messages destined for RJP workstations. When JESXCF returns messages, process under RJPCONS FCT searches the messages and chains the messages to be sent to the workstation's RJP console.

NJE consoles

NJE consoles provide a remote node the capability to inquire on and control work that has arrived from the NJE network. There are no physical JES3 NJE consoles, but instead the NJE console support provides a way for performing command association between a requestor on a remote node and console operations on the JES3 node.

The NJE console services processor (NJECONS) maintains a queue for pending network commands. A console is established for the pending NJE command instance by invoking JESXCF. The command is then issued. When a response is issued for the command, it is routed to JESXCF which in turn notifies this DSP that a command response is available. The responses are retrieved from JESXCF, NJE command response entries are created and routed back to their origin.

14.6 RJP console support

- JES3 RJP console support still needed
  - RJP consoles connected as EMCS
  - RJP consoles still defined in inish deck
    - JES3 passes console information to JESXCF
    - JESXCF has knowledge of RJP console status
  - JESXCF initiates messages to consoles
- JESXCF creates an extended MCS console
  - Console name: SYSJ3Rxx

RJP console support

JES3 internally creates an extended MCS console for RJP console implementation. The RJP extended MCS console name is SYSJ3Rxx with a KEY=xcfgroupname. SNARJP and BSCRJP consoles that are attached to remote job entry workstations can be either real console devices, or logical devices that are being simulated by an application. These consoles are usually used to control work that originates or is associated with a location where a group of print devices reside.

SYSJ3Rxx consoles

The extended MCS console named SYSJ3Rxx, where xx is a number starting with 01, is activated by JESXCF. This console is used to deliver messages to *all* RJP consoles. JES3 messages issued in response to commands entered from RJP consoles are directed to the SYSJ3R01 console. JES3 retrieves messages queued on the SYSJ3R01 console for a specific RJP workstation by using the RJP terminal name as the CART value and delivers these messages to the remote console.

14.7 Defining RJP consoles

CONSOLE,JNAME=name,TYPE=RJP,DEST=(msgdest,..msgdest),
LL=nnn,LEVEL=nn,SAVEMSG={YES|NO}

- CONSOLE TYPE=RJP supported
- SAVEMSG= keyword
- Messages spooled if console logged off
- DEST= accepts MVS route codes
- *INQUIRY O=*

```
IAT8589 CONSOLE DISPLAY 735
NAME      COUNT     SWITCH      LL     AUTH     SAVEMSG
RM002     00000000                    0120     15       YES
  ROUTE CODE=(NONE)
  DEST CLASS=(NONE)
```

Defining RJP consoles

The CONSOLE statement TYPE=RJP defines RJP consoles. The RJP CONSOLE initialization statement is as follows:

```
CONSOLE,JNAME=name,TYPE=RJP,DEST=(msgdest,...msgdest),LL=nnn,
LEVEL=nn,SAVEMSG={YES|NO}
```

The parameters are:

SAVEMSG The SAVEMSG= parameter specifies whether messages should be received and spooled while the console is logged off.

DEST DEST= accepts MVS routing codes in addition to JES3 destination classes. DEST= specifies one or a series of message destinations which represent classes of messages you want sent to the specified RJP console. *msgdest* can be any set of MCS routing codes or JES3 destination classes. Single routing codes and JES3 destination classes are specified as a single value with commas separating the individual values. Ranges of MCS routing codes can be specified by including a dash between the start of the range and the end of the range. The JES3 destination class and MCS routing codes will be combined into a single set of message destinations used to determine what messages will be displayed on the console.

The workstation name is derived from the RJPWS initialization statement for an SNA workstation or the RJPTERM initialization statement for BSC. This workstation name serves as the userID for the workstation console. Users of the RJP console have to log on using this terminal ID and supply the same password.

14.8 RJP console commands

- *SWITCH frjp,trjp - Redirect console messages
- *SWITCH frjp,NONE - Stop message queueing
- *FREE con - Free messages currently queued
- Following commands apply to RJP consoles only
  - *I O command
  - *F O command
- *Z,{ALL | con | msgdest},text - Send a message
- *I D,T={wsname | ALL} - Display RJP w/s
  - .. IAT8618 ..... SPOOLED MSG=xx
  - .. IAT8622 ..... CHAINED MSG=xx

RJP console commands

The following commands may be used to affect message queueing to RJP consoles:

***SWITCH RJP01,RJP02**	Redirects message traffic from one RJP console to another. This command must be issued from an operator (non-RJP) console. Only a single level of switch is allowed.
***SWITCH RJP01,NONE**	Stops message queueing to the target RJP console. This command must be issued from an operator (non-RJP) console.
***FREE RJP01**	Frees all messages currently queued to the target RJP console.
***FREE**	Frees all messages currently queued to the RJP console from which the command is entered.

The following commands apply only to RJP consoles and the DLOG function:

- ***I O** command displays RJP console and DLOG status
- ***F O** command modifies RJP console and DLOG status

The ***Z** command sends a message to another RJP console or to a message destination.

Message IAT8618

To display the number of spooled messages queued to a signed off RJP console, enter:

```
*I D,T=RJP01
  IAT8618 .. gives  SPOOLED MSG=xx
```

For signed on workstations, this command displays the number of messages queued in JES3 storage for the workstation. JES3 issues this message in response to an ***INQUIRY** command that requested the status of a BSC RJP line or workstation or a SNA RJP workstation. The designated line or workstation is not signed on. If SNA RJP is included in the message text, the specified workstation is an SNA RJP workstation; otherwise, the workstation or line is BSC RJP.

Message IAT8622

This message is issued in response to an ***INQUIRY** command that requested the status of a BSC RJP line or workstation or a SNA RJP workstation. The designated line or workstation is signed on.

```
*I D,T=RJP02
  IAT8622 .. gives  CHAINED MSG=xx
```

14.9 Displaying RJP consoles

```
*I O=*
 IAT8589 CONSOLE DISPLAY
 NAME       COUNT     SWITCH       LL      AUTH     SAVEMSG
 RJP01     00000010              0120      15        YES
   ROUTE CODE=(BROADCAST)
   DEST CLASS=(ALL)
   SWITCHED CONSOLES=(RJP06)
*I O=RJP06
 IAT8589 CONSOLE DISPLAY
 NAME       COUNT     SWITCH       LL      AUTH     SAVEMSG
 RJP06     00000000   RJP01      0120      15        YES
   ROUTE CODE=(HARDCOPY,1-128)
   DEST CLASS=(TOTAL)
 RJP01     00000010              0120      15        YES
   ROUTE CODE=(BROADCAST)
   DEST CLASS=(ALL)
```

Commands to display RJP consoles

To display a RJP console status:

```
*I O=*
 IAT8589 CONSOLE DISPLAY
 NAME       COUNT     SWITCH       LL      AUTH     SAVEMSG
 RJP01     00000010              0120      15        YES
   ROUTE CODE=(BROADCAST)
   DEST CLASS=(ALL)
   SWITCHED CONSOLES=(RJP06)
```

When displaying console RJP06 whose messages have been switched to console RJP01, RJP01 console status is also displayed:

```
*I O=RJP06
 IAT8589 CONSOLE DISPLAY
 NAME       COUNT     SWITCH       LL      AUTH     SAVEMSG
 RJP06     00000000   RJP01      0120      15        YES
   ROUTE CODE=(HARDCOPY,1-128)
   DEST CLASS=(TOTAL)
 RJP01     00000010              0120      15        YES
   ROUTE CODE=(BROADCAST)
   DEST CLASS=(ALL)
```

14.10 NJE functions

- Network Job Entry (NJE)
  - In JES, a facility which transmits and recives jobs, operator commands, messages, SYSOUT data, and accounting data between communicating job entry nodes
- NJE Network
  - A group of job entry complexes that communicate with each other
- NJE Units of Work
  - An NJE transfer unit is a unit of work transmitted across the network
    - An NJE job or a nodal message record
- Transmit
  - The node packages the NJE transfer unit and transmits it
- Receive
  - The node recognizes the NJE transfer unit, receives, and stores it
- Store-and-Forward
  - The node accepts the NJE transfer unit, stores it, and schedules it to be forwarded to another node

Network job entry (NJE)

In JES, a facility which provides for the transmission of selected jobs, operator commands, messages, SYSOUT data, and accounting information between communicating job entry nodes that are connected in a network either by binary synchronous communication (BSC) lines, channel-to-channel (CTC) adapters, by System Network Architecture (SNA) connections, or by Transmission Control Protocol/Internet Protocol (TCP/IP or TCPIP).

Unit of work

An NJE transfer unit is a unit of work that is transmitted across the network. An NJE transfer unit can be either an NJE job or a nodal message record (NMR).

An **job** is a transfer unit that contains data to be processed at another node in the NJE network. It begins with a *job header*, is followed by data, and ends with a *job trailer*. The type of data contained in the NJE job further defines the type of the NJE job. The data between the job header and job trailer can be either SYSIN or SYSOUT data. An NJE SYSIN job is an NJE job that contains JCL for a job and may have one or more SYSIN data sets. An NJE SYSOUT job is an NJE job that contains one or more SYSOUT data sets. Each SYSOUT data set is preceded by a *data set header*.

A **nodal message record** (NMR) is a unit of work that begins with an *NMR header* and is followed by message text. The message text can be either a message or system command.

Functions of a node

A node is a system or complex that is defined to an installation. A node in the network can be another complex or system within a single location or it can be a complex that resides in a remote location. Each node that a complex can access must be identified to other complexes by a unique NJE node name.

> **Note:** If a node uses SNA protocols, the node has two names:
>
> – An LU name as defined to VTAM
> – An NJE node name created during JES initialization processing
>
> The NJE node name appears in job headers, data set headers, and NMRs.

Each node in the network can do the following with an NJE transfer unit:

- **Transmit** - The node packages the NJE transfer unit and transmits it to another node.
- **Receive -** The node recognizes the NJE transfer unit, receives, and stores it.
- **Store-and-forward -** The node accepts the NJE transfer unit, stores it, and schedules it to be forwarded to another node.

Types of nodes

NJE uses the following terminology for the nodes that comprise an NJE network.

- Originating Node is the node where the user submitted the request to transmit the data.
- Intermediate Node is a node that lies in the path of either the:

  Originating node and execution node

  Execution node and the destination node

  It *receives* and *transmits* the NJE transfer unit to the next node in the path of the target node.
- Target Node is the node where a NJE job or NMR is received and will either be executed or be processed. The target node can be either a:

  Destination Node - a node that receives and processes:

  - An NJE SYSOUT job.
  - A message contained in an NMR.

  Execution Node - a node where:

  - JCL contained in an NJE SYSIN job executes.
  - A command contained in an NMR is processed.

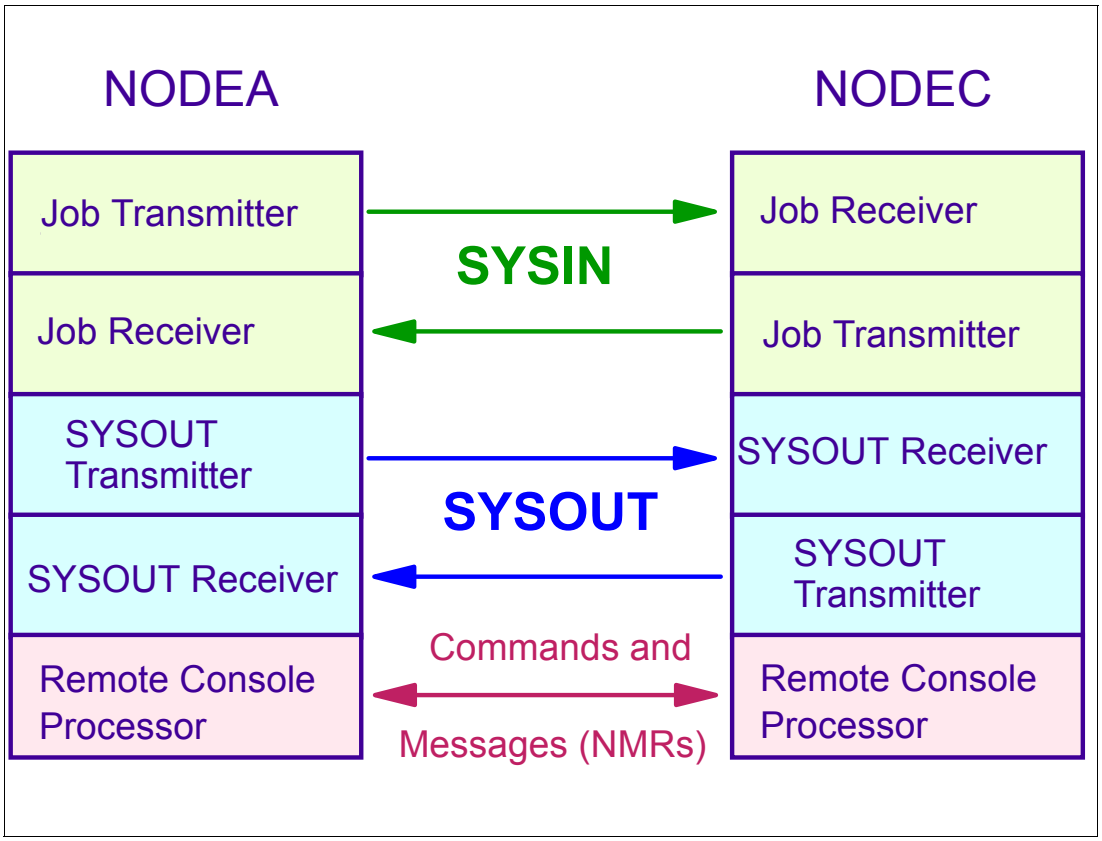
Transmitting and receiving

Each node in the network has the capability of transmitting and receiving a network units of work. The node that sends the network unit of work can use either BSC, SNA, or TCPIP protocols. The type of networking protocol the home node uses to transfer the network stream is determined by the JES3 NJERMT initialization statement that defines the directly connected remote node. The receiving node can process the network job locally or can transmit the network stream to another node in the network.

Store and forward

If the originating node and the destination node are not directly connected nodes, path nodes are used to route the network stream to the destination node. JES3 adds the NJESF scheduler element to the network job to pass the network stream to the next node in the path of the destination node. The store and forward DSP prepares the network stream so that it can be sent to the next destination node using the proper networking protocol.

14.11 NJE units of work



NJE networking

Each node has the capability to:

- Send job streams to the other node.
- Send SYSOUT streams to the other node.
- Send commands to the other node.

Networking examples

If a user at NODEA submits a request to transmit a job to execute at NODEC, NODEA is the originating node. Input service at NODEA processes the user's JCL for the request and creates a network job.

Networking creates separate data sets to contain the different portions of the network stream and creates JDS entries to identify the data sets. Networking assigns the following names to each data set:

```
IATYNCN       Defines the Nodal Message records (NMR).
IATYNJH       Defines the job header information for the network stream.
IATYNDH       Defines the JCL and/or SYSIN data for the job.
IATYNJT       Defines the job trailer information for the network stream.
```

See *Network Job Entry Formats and Protocols*, SA22-7539 for additional information.

14.12 Transport protocols

- BSC
  - JES2, JES3, VM RSCS, VSE/POWER support BSC
  - No AS/400 BSC NJE support
- SNA
  - JES2
  - JES3 with BDT
  - VM RSCS, AS/400, VSE/POWER support SNA
- TCP/IP
  - VM RSCS, AS/400, VSE/POWER support TCP/IP
  - JES2 Version 1 Release 7
  - JES3 Version 1 Release 8

Transport protocols

A network stream is transmitted across a connection that is initialized by a networking protocol. JES3 supports two networking protocols:

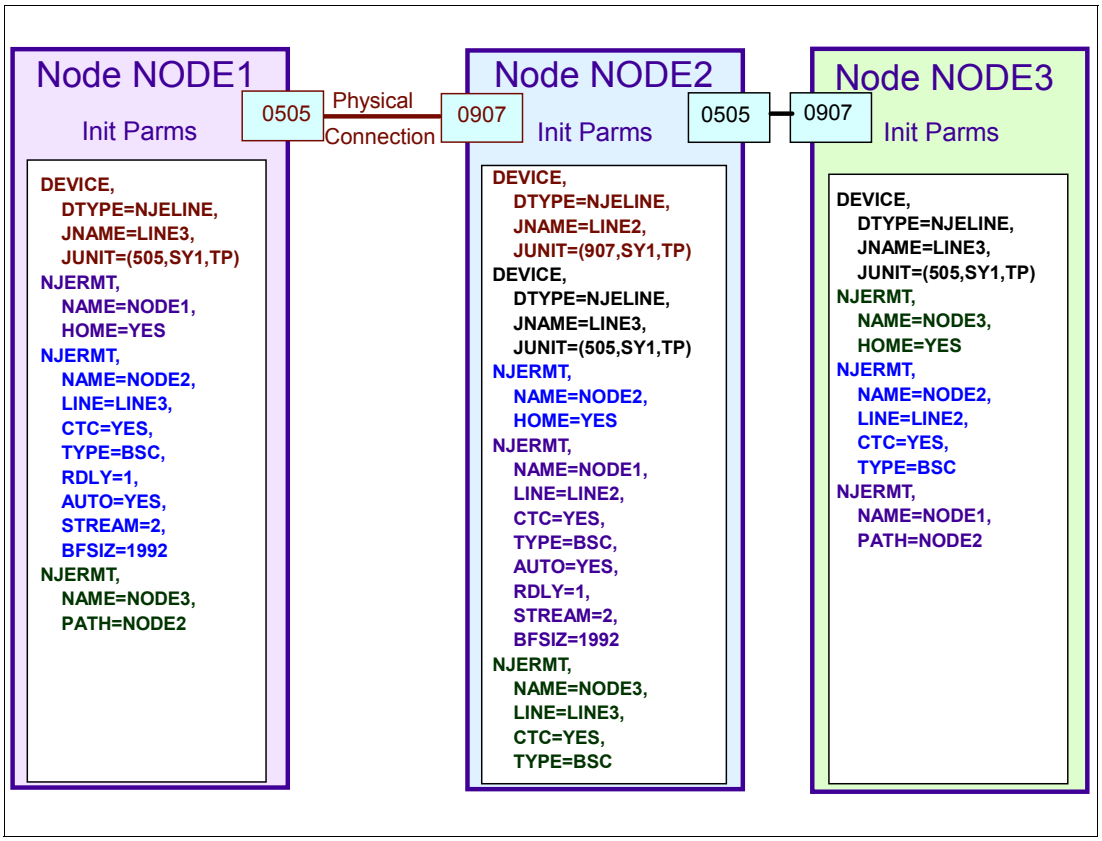
- SNA and BSC

The network is comprised of the home node and all of the remote nodes defined to the home node. Each node in the network is identified by a unique node name assigned to the node during JES3 initialization. A node can be a JES3 complex networking for both SNA and BSC to:

- Another JES3 complex
- JES2 complex
- VM RSCS complex
- VSE/POWER complex
- AS/400®

The type of protocol used to transmit the network stream is determined by the TYPE= parameter on the NJERMT statement that defines the directly connected remote node. If TYPE=BSC is specified on the NJERMT statement that defines the remote node, BSC protocols are used to transmit the network stream. If TYPE=SNA is specified on the NJERMT statement, SNA protocols are used to transmit the network stream. TYPE=TCPIP specifies a TCP/IP networking protocol. Since a node can be directly connected to more than one node, each JES3 node in the network has the capability of transmitting a network stream by using BSC, SNA, or TCPIP protocols.

14.13 BSC initialization statements



BSC definitions

You must code a NJERMT statement for the home node (your node) and one for each remote node that will communicate with the home node.

All DEVICE statements with the parameter DTYPE=SYSMAIN specified must precede the NJERMT statement.

The JNAME parameter on a DEVICE statement that also includes a DTYPE=NJELINE parameter must match the LINE parameter on the NJERMT statement.

To transmit an NJE transfer unit to a complex other than the user's installation (a remote node), the user issues a command or submits a job specifying a destination node name. The destination node can be either *directly-* or *indirectly-connected* to the originating node. In the network depicted in the figure, if NODE1 is the originating node, NODE2 is a directly-connected node to NODE1, and NODE3 is an indirectly-connected node to NODE1.

To transmit an NJE transfer unit from NODE1 to NODE3, NODE2 is required to do a store-and-forward function since there is not a direct physical connection defined between NODE1 and NODE3.

14.14 BSC line or CTC DEVICE statement for NJE

- **DEVICE** - Defines a BSC line or CTC connection
  - **DTYPE** - defines device as BSC line or CTC connection
  - **JNAME** - defines the name of BSC line or CTC connection
  - **JUNIT** - specifies info about line or CTC connection
    - **devnum** - specifies the device number
    - **main** - specifies the name of the main that has access to the line when the main is global
    - **TP** - specifies the destination class that is to receive messages about this line

```
DEVICE,DTYPE=NJELINE,JNAME=LINE,JUNIT=(A123,*ALL,TP,ON)
```

BSC line or CTC DEVICE statement for NJE

The following statements are the primary definitions for BSC lines and devices:

DTYPE= Specifies that a BSC line or a CTC connection for a network of nodes.

JNAME= Specifies a name of a BSC line or CTC connection. The first character of the 1-8 character name cannot be a slash (/). The variable *linename* must match the name you specify on the LINE parameter of the JES3 NJERMT initialization statement that defines your node.

JUNIT= Specifies information about the line or CTC connection. You must code the following parameters for the global. You must repeat these parameters for each local main that has access to the line and could become the global.

devnum - Specifies the device number (by 3-digit or 4-digit hexadecimal number). A slash (/) preceding the device number is not required.

main - Specifies the name of a main that has access to the line when the main is global. The variable *main* must match the name of the main that you define on a MAINPROC statement. Alternatively, a main name of *ALL can be used. Use of *ALL indicates that all mains in the complex have access to this line when the main is the global.

TP - Specifies the destination class that is to receive messages about this line. or connection.

14.15 NJERMT statement for NJE

- **NJERMT** - Defines a node in the JES3 network
  - **NAME** - defines a 1 to 8 character node name
  - **HOME** - specifies if node is home node or remote node
  - **LINE** - specifies 1 to 8 character name of the line that connects the home node to the directly connected remote BSC node defined by this statement
  - **CTC** - specifies type of connection
  - **TYPE=SNA** - specifies networking protocol
  - **RDLY** - specifies restart delay time (in minutes)
  - **STREAM** - specifies number of concuurent data streams that JES3 NJE is to transmit on one line
  - **AUTO** - specifies whether JES3 is to automatically restart line if remote node interrupts transmission
  - **PATH** - specifies name of first node in indirectly connected node
  - **BFSIZ** - specifies buffer size used for communication

BSC NJERMT statement

For BSC networking, the NJERMT initialization statement contains the following definitions:

NAME= Specifies a 1-to-8 character node name. No two nodes should have the same node name. The names that can be specified are:

- Node name
- Home name

HOME= Specifies whether this NJERMT statement defines the home node or a remote node. If it defines the home node, code YES. For all other nodes, omit this parameter or code NO.

LINE= Specifies the 1-8 character name of the line that connects the home node to the directly-connected remote BSC node defined by this statement. This name must match the name specified via the JNAME parameter on the DEVICE statement that defines the line. When the operator starts the line, he can override the name you have specified with this parameter. If you omit the LINE parameter, the operator must specify a line name when starting the line.

CTC= Specifies the type of connection between the home node and the directly-connected remote BSC node defined by this statement. For a node that is connected via a channel-to-channel (CTC) adapter, specify YES. For a node that is connected via a leased line or a dial-up line, omit this parameter or specify NO. This parameter is ignored when TYPE=SNA is specified.

RDLY= Specifies, in minutes, the amount of time that JES3 networking is to wait before it automatically restarts an interrupted line. Allowable values for mm are 0-99. Code this parameter on an NJERMT statement that defines a directly-connected remote BSC node.

STREAM= Specifies the number of concurrent data streams that JES3 networking is to transmit on one line between the home node and the directly-connected remote BSC node defined by this statement. STREAM=2 is not allowed for directly connected, remote BSC nodes, running RSCS under VM.

TYPE= Specifies the networking protocol to be used for communicating with a directly-connected remote node. Include this parameter only when defining a directly-connected remote node.

- BSC Specifies a BSC networking protocol.
- SNA Specifies a SNA networking protocol.

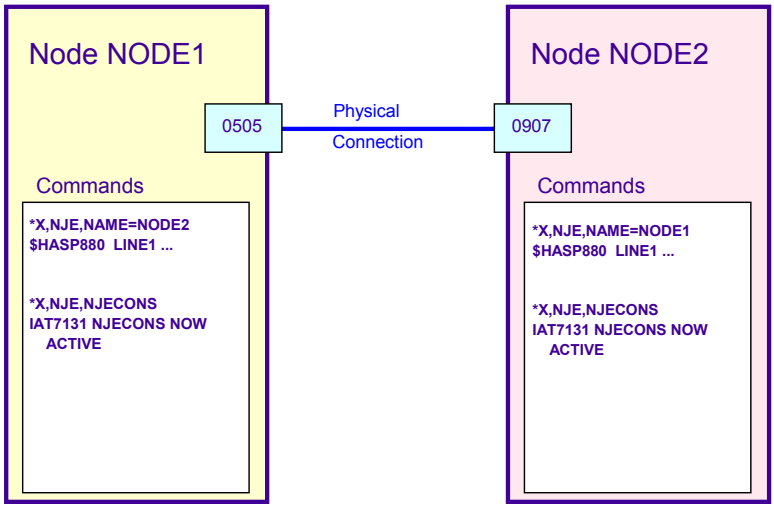
AUTO= Specifies, for a directly-connected remote BSC node, whether JES3 is to automatically restart the line to the node if the remote node interrupts transmission. Specifying YES causes JES3 to automatically restart the line. If you specify NO or omit this parameter and the remote node interrupts transmission, the operator must restart the line. Code this parameter on the NJERMT statement that defines a directly-connected remote BSC node.

PATH= Specifies the name of the first node in the path to an indirectly-connected node. Code this parameter on the NJERMT statement that defines the indirectly-connected node. If you omit this parameter, JES3networking assumes the home node and the remote node are directly connected.

BFSIZ= Specifies the buffer size to be used for communication with the directly-connected remote BSC node defined by this statement. The system programmer at the remote node must specify the same buffer size as you specify. Do not specify a buffer size of less than 400 bytes. The maximum buffer size allowed is the size of the spool buffers (specified via the BUFSIZE parameter on the BUFFER statement) minus 46.

14.16 BSC - NJE commands

- ***X,NJE,NAME=**
  - Starts communication on a networking BSC line
  - BSC line could have been defined as directly connected or indirectly connected
- ***X,NJECONS**
  - Starts networking console support



BSC commands

Use the ***X NJE** command to start communication on a networking line that directly connects your complex to a remote node.

You must use this command after JES3 initialization to start communication on a line that directly connects your node to a remote node before you can transmit to or receive data from that node or from indirectly connected nodes whose path is through that node.

If more than one line connects your node with a remote node, you can also use this command to start the additional lines as they are needed.

Use the **`*X,NJE`** command to specify I/O operations across the BSC NJE line and only for BSC/NJE networking.

NJECONS DSP

Use the **`*X NJECONS`** command to start networking console support DSP. After JES3 initialization and before you can send or receive commands or messages from nodes in your network, you must use this command to start networking console support.

JES3 starts networking console support. You can now send and receive messages and commands to or from any other node in the network.

```
IAT7131  NJECONS NOW ACTIVE
```

14.17 NJE command and message flow

- NJECONS DSP receives responses from JESXCF
- ***T** command used for operator commands - command response expected
  - *T nodename,command - Send a command
  - *Z nodename,message - Send a message
- An extended MCS console created by JESXCF
  - Console name: SYSJ3Nxx

NJE consoles

JES3 networking console support is handled by the NJECONS DSP (module IATCNNJ). This DSP is called and canceled by the operator.

NJECONS DSP

The NJECONS DSP processes nodal message records (NMRs), which contain network commands or messages. The NMRs may be received from or sent to other nodes in the network.

The NJECONS DSP mainline performs initialization, routing and termination functions. The DSP initializes the *network console queue* (NCQ) and anchors it from the TVT. Routines in NJECONS DSP maintain also the *network pending command* queue (NPC). When a remote node sends a command to this node, the command NMR is chained from an NPC entry which will represent the NJE command. Console support is established for the NJE command instance by invoking JESXCF with a unique console name associated with the new NJE command instance. The command is issued. When a response is issued for the command, it is routed to JESXCF which in turn will notify the NJECONS DSP that a command response is available. The responses are retrieved from JESXCF. NJE command response entries are created and routed back to their origin for each message received from JESXCF.

NJE EMCS console

The name JESXCF assigns to the NJE extended MCS console is SYSJ3Nxx with a KEY=*xcfgroupname*. The NAME parameter on the NJERMT statement that defines HOME=YES is the default for the *xcfgroupname* parameter on the OPTIONS statement.

The xx in the SYSJ3Nxx console name is an internally generated number that starts with 01. For example, suppose a command, **CP SM RSCS CMD WTSCPLX3 *I Q**, is sent from VM to JES3. When the command is received at node WTSCPLX3, the following occurs:

- The command NMR is chained from an NPC entry which will represent the NJE command. Console support is established for the NJE command instance by invoking JESXCF with a unique *console name* associated with this NJE command instance. The same *console name* is used to build Console Destination Block (CNDB) for use on the INTERCOM macro. The command is then issued with the INTERCOM macro. When a response is issued for the command, it is routed to JESXCF which in turn will notify this DSP that a command response is available.
- The CNDB (the console destination block or CNDB encapsulates console routing information) for the INTERCOM has a unique *command and response token* (CART) value assigned identifying the command instance and SYSJ3N01 as the console. Later, the CART value is used to filter the command response from the SYSJ3N01 console's message queue. Once the response is retrieved from the console, it is sent back through the network to the remote node.
- DSI processing switches the console name from SYSJ3N01 to the next available console name (e.g. SYSJ3N02).

*SEND (*T) command

Use the ***T** command to send selected commands to other for processing.

You can use this command to modify or display the status of jobs submitted at your node and sent to another node for processing.

```
Syntax: *T nodename,system-commands
Note: The comma between nodename and system-commands is required.
```

*MESSAGE (*Z) command

Use the ***Z** command to send a message to one or more consoles on the system or the NJE network. You can direct messages to a specific console or to a group of consoles designated by a common MCS routing code or JES3 destination class.

```
Syntax:*Z {ALL | con | node | msgdest},text
```

Using JES3 to authorize commands from NJE

JES3 allows only *INQUIRY and *MODIFY commands to be entered through NJE unless an installation overrides the JES3 authorization processing by using IATUX35 exit or RACF.

The ***I** commands allowed from NJE are:

- ***I Q,N**
- ***I B**
- ***I J,E**

The ***F** commands allowed from NJE are:

- ***F J,C**
- ***F J,CP**
- ***F J,CO**
- ***F J,H**
- ***F J,R**

14.18 Dynamic SNA NJE node definitions

- Can dynamically add: TCPIP and SNA nodes
  - Cannot ADD or DELETE home node
  - Adjacent - PATH= for non-adjacent
- Keywords: ADD= DEL=
  - *F NJE,ADD=nodename,TYPE=[TCPIP | SNA] [,PATH=nodename]
  - *F NJE,DEL=nodename
- Commands not checkpointed

Adding SNA nodes dynamically

Use the `*F NJE` command to:

- Dynamically add a directly-connected node to your network at your home node.
- Dynamically add an indirectly-connected node to your network.
- Dynamically add an alias of your home node to your network.
- Dynamically delete a node or alias from your network.
- Reset the lines connecting your node to a specific remote node when your node has abnormally ended communication on the line
- Place jobs scheduled for transmission to a specific remote node in operator hold status
- Release a remote node from hold status

Note: You **cannot** dynamically **add** or **delete** your installation (the **home node**) to the network.

Do not use the `*F NJE` command to change the type of protocol, from SNA to BSC, for dynamically added nodes. Be careful when using the `*F NJE` command to change the type of protocol from SNA or TCP/IP to BSC. In particular, a device with DTYPE of NJELINE must be available for this BSC node to use. If such a device does not exist, it must be added with a hot start with refresh before you can use the modified BSC node. You cannot delete an active adjacent BSC node from your installation. An active node is a node currently transmitting to or receiving data.

14.19 Symbols in commands in exits

- Substitution before commands are passed:
  - User exit 18
    - Command Modification and Authority Validation
  - User exit 58
    - Security Information Before JES3 Security Processing
  - User exit 59
    - Security Information Before JES3 Security Processing
- Substitution occurs after exit is entered:
  - User exit 35 - Validity Check Network Commands
    - Incoming NJE commands
- NODEA operator issues: *T NODEB,*I B,&&SYSNAME
- NODEA exit 18 sees: *T NODEB,*I B,&SYSNAME
- NODEB exit 35 sees: *T NODEB,*I B,&SYSNAME
- NODEB exit 18 sees: *I B,NODEA

Using system symbols with NJE commands

When using JES3 exits for command processing with symbols in the command, symbolic substitution takes place as follows:

- For JES3 commands, substitution takes place before exits 18, 58 and 59 are entered.
- For NJE commands, substitution takes place after exit 35 is entered.

In the example, for sending commands via NJE you must use a double ampersand with symbolic symbols. When the system finds two consecutive ampersands at the beginning of a valid symbol, it removes the first ampersand and keeps the second ampersand in place. A subsequent process can then substitute text for the symbol in later processing, or the second ampersand can remain as a literal character.

So if you route a command to another node in a network, use double ampersand (&&) notation to cause substitution to occur on the receiving node.

However, since exit 35 is entered before substitution occurs, be aware that the command may have the symbol still there.

14.20 BSC NJE commands to Other Nodes

BSC NJE between SC43 and SC50

```
SC50    *I  NJE
SC50     IAT8653   -   SC43C      -   WTSCPLX4            ACTIVE
                      line                node                status
SC50    RO SC43,*I  NJE
SC43     IAT8653   -   PLX3C      -   WTSCPLX3            ACTIVE
SC50    RO SC43,*I  J=NJECONS
SC43     IAT8674  JOB  NJECONS    (JOB02539)  P=15  NJECONS(ACTIVE)
SC50    *Z  WTSCPLX4,HELLO, HOW ARE YOU
SC43     IAT7146  WTSCPLX3  HELLO,  HOW ARE YOU
SC50    *T  WTSCPLX4,*I  NJE
SC50     IAT8653   -   PLX3C      -   WTSCPLX3            ACTIVE
SC50    *T  WTSCPLX1,$DQ
SC50     IAT7146  WTSCPLX1    15.24.29     $HASP646  23 PERCENT SPOOL
SC47     $HASP249  COMMAND RECEIVED FROM *CONSOLE AT WTSCPLX3
SC47     $DQ
```

NJE commands to other nodes

This figure has some examples of NJE commands and NJE commands to other nodes in the NJE network and the responses to the commands.

- Use the JES3 **`*I NJE`** command to display the status of the networking nodes and communication lines.
- Use the MCS **`ROUTE`** command to direct a command to one or more systems in a sysplex for processing. You can direct a command to:
  - All systems in the sysplex
  - A subset of the systems in the sysplex
  - One system in the sysplex.

Note: In the example, the MCS **`ROUTE`** command could be used to send operator commands from JES3 global running on SC50 main to another JES3 global main running on SC43 because the two JES3 complexes were running in the same sysplex.

14.21 z/OS Bulk Data Transfer (BDT)

Bulk Data Transfer (BDT) SNA NJE Feature

- JES3 SNA NJE support to other network systems
  - z/OS with z/OS BDT
  - z/OS JES2 SNA networking
  - RSCS networking
  - VSE advanced functions with VSE/POWER
  - AS/400 Communication Utilities

User JCL and JES3 Control Statements Required

- JCL for networking jobs require NO changes if node names do not change when the NJE protocl to adjacent nodes is changed
- XMIT JCL statement

z/OS BDT and JES3 SNA NJE

The BDT systems network architecture (SNA) NJE feature (a licensed program) allows z/OS JES3 users to transmit jobs, output (SYSOUT), commands, and messages from one computer system to another within a SNA network. Any of the following systems can participate in the network: z/OS JES2, z/OS JES3 (with BDT), VM/SP (with RSCS Networking), VSE/Advanced Functions (with VSE/POWER) and AS/400 Communication Utilities.

BDT runs in its own address space. A JES complex may have one or more BDT address spaces. Each processor in the complex may have one or more BDT address spaces. A complex with multiple BDT address spaces is called a poly-BDT complex. You must decide whether you want to set up such a complex. A poly-BDT complex is beneficial during testing, as a way to separate testing from production work.

Defining BDT to MVS

BDT operates under the control of MVS and therefore requires the following MVS definitions:

1. You must define BDT as a secondary MVS subsystem by creating an entry in an IEFSSNxx member of SYS1.PARMLIB.
2. You must specify MVS system parameters in an IEASYSxx member of SYS1.PARMLIB.
3. You must give authorized program facility (APF) authorization to SYS1.BDTLIB, the BDT module library, by updating SYS1.PARMLIB member IEAAPFxx.
4. If you are installing BDT in a JES3 complex, you will have to add several statements containing BDT-related information to the JES3 initialization stream. The statements are CONSOLE, SYSID, and, for SNA NJE customers, NJERMT.

Defining BDT to VTAM

BDT runs as a VTAM application program, requiring several VTAM definitions:

1. You must define BDT as a VTAM application program on the APPL definition statement. One APPL statement is required for each type of node your system will have, that is, one APPL statement for a file-to-file node and one APPL statement for a SNA NJE node.
2. You must define each remote node as a VTAM cross-domain resource.
3. You must define session parameters to VTAM by creating VTAM logon mode table entries.

Allocating BDT SNA NJE data sets

With MVS and VTAM definitions complete you can start work on BDT itself. Your first task is to allocate data sets that are used during BDT SNA NJE operation:

1. You must allocate a data set to contain the BDT initialization stream. Later you will put initialization statements into this data set and then specify its name on the procedure that the operator invokes to start BDT.
2. You must allocate a data set to contain the BDT work queue, in which BDT places jobs waiting to be processed. Sample BDT SNA NJE feature only JCL procedure:

```
//BDT43    PROC
//* ************************************************************
//*  THIS PROCEDURE MAY BE USED TO INITIALIZE MVS/BDT WITH    *
//*  THE FILE TO FILE FEATURE (FTF), NJE FEATURE, OR BOTH.    *
//*  IT IS MEANT AS AN EXAMPLE ONLY.  THE DATA SET NAMES      *
//*  MAY BE MODIFIED AS REQUIRED.  ALL DATA SETS MUST BE      *
//*  CATALOGED PRIOR TO INVOCATION.                           *
//* ************************************************************
//BDT      EXEC PGM=BDTINTK,REGION=5000K,TIME=1440
//* STEPLIB  DD DISP=SHR,DSN=SYS1.SBDTLIB
//BDSPOOL  DD DISP=OLD,DSN=SYS1.SC43.BDTSPACE
//CRSPOOL  DD DISP=OLD,DSN=SYS1.SC43.BDTSPACE
//BDTOUT   DD SYSOUT=A
//SYSABEND DD SYSOUT=A
//BDTABEND DD SYSOUT=A
//BDTIN    DD DISP=SHR,DSN=SYS1.PARMLIB(BDTSNAT9)
```

Creating a BDT initialization stream

To define the environment in which BDT will operate each time it is started you must code initialization statements. You can specify the amount of main storage BDT will use, the names of nodes in the network, the pacing rate for communication between nodes, and many more parameters. You identify the data set containing the initialization statements in the BDT start procedure.

The IBM-provided initialization statements are in SYS1.SBDTSAMP data set.

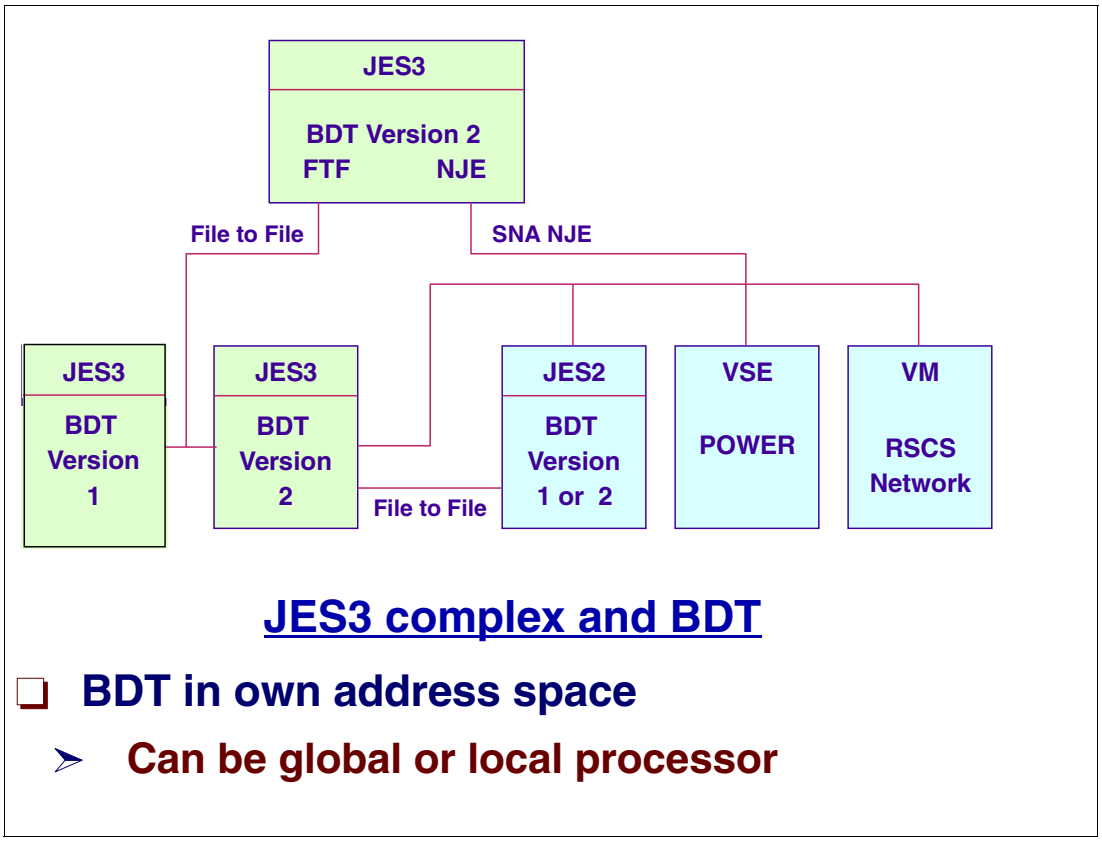
XMIT JCL statement

Use the XMIT JCL statement to transmit records from an MVS node to a JES3 node, a JES2 node, a VSE/POWER node, a VM/RSCS node, or an AS/400 node.

JES3 passes the jobs to BDT, which have to enter the network to adjacent SNA NJE node. If necessary, the network jobs can be routed through intermediate systems (store and forward) to reach their destination.

When writing new JCL, IBM recommends using the XMIT JCL (//name XMIT form) since this statement is not dependent on using a particular JES subsystem. In addition, the XMIT JCL is preferred because it allow transmission of records that a //*ROUTE XEQ, /*ROUTE XEQ or a /*XEQ statement does not allow.

14.22 z/OS BDT options



JES3 BDT and networking nodes

This figure shows the possible combinations of communication from JES3 SNA NJE and BDT file-to-file (FTF) to other networking systems. JES3 BDT runs in its own address space which can be on either the global or local processor.

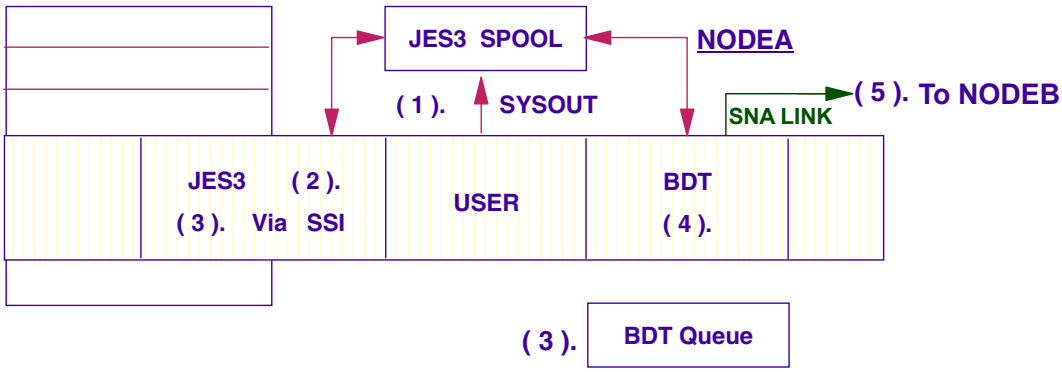
BDT address space

The MVS/Bulk Data Transfer (BDT) Facility is a subsystem of MVS that runs under its own address space on any main in the complex. A complex can have one or more z/OS BDT subsystems in its configuration.

BDT file-to-file

The z/OS BDT Facility is available as a program product of the MVS system product. z/OS BDT is available in two releases. z/OS BDT Version 1 allows the installation to transfer (copy) data sets from its original location to a new data set in the same or a new complex. This is the file-to-file function of z/OS BDT.

14.23 BDT SNA NJE processing



- 1 - User creates SYSOUT for NJE -> JES3 spool
- 2 - JES3 creates a BDT transaction for SYSOUT
- 3 - Transaction placed on BDT transaction queue
- 4 - BDT user process sysout (PSO) interface for SYSOUT
  - Allocate, Open, Read, Close, Unallocate
- 5 - Transmit file to NODEB

JES3 BDT node to node transaction

In a JES3 complex, an z/OS BDT subsystem can run on a global or a local. JES3 uses two function control tables (BDT FCT and BDTCOMM FCT) that provide the interface between JES3 and z/OS BDT. Collectively, these FCTs are called the JES3/BDT communications interface.

This figure is an overview of how a BDT transaction takes place from the creation of the SYSOUT file to the transmission by BDT to another node.

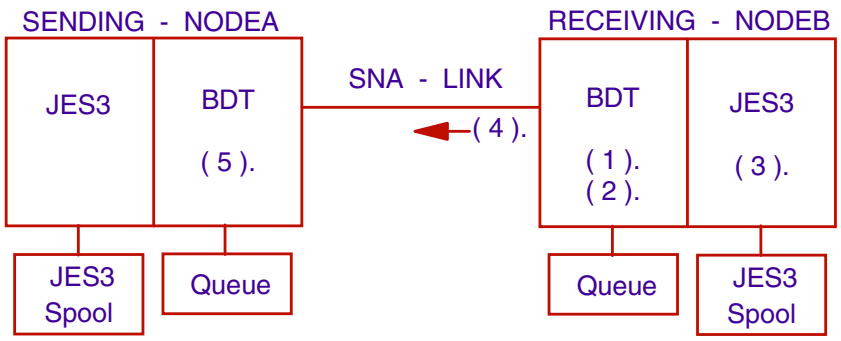
User SYSOUT

A user can create a network job to send one or more SYSOUT data sets (DEST=node) to one or more nodes. Each SYSOUT data set is represented by one or more data set headers. The data set headers indicate:

- The node where the SYSOUT data set should be processed
- Processing options for the SYSOUT data set

A shuttle *BDT Subsystem Interface Data* (BSID) area is associated with each z/OS BDT address space that is defined in the complex. JES3 uses the shuttle to send requests to the specified z/OS BDT subsystem. The BSID contains a data area where JES3 places the request. To send a BSID to z/OS BDT, JES3 issues a JSERV macro TYPE=RESP. The subsystem communication routines that process the JSERV macro, place the BSID in the shuttle staging area and pass it through the subsystem interface to the specified z/OS BDT subsystem.

14.24 BDT - SYSOUT received at destination node



- 1 - BDT receives sysout stream
- 2 - BDT creates data set to JES3 spool
  - Allocate, open, write, close unallocate
  - Unallocate - specify external - writer name - NJERDR
- 3 - NJERDR DSP Selects the spool data (OSE)
  - "NJERDR" is DSP like "INTRDR"
- 4 - BDT NODEB notifies sender - "when store complete"
- 5 - BDT removes transaction from queue at NODEA

BDT - SYSOUT received at destination node

This figure shows what happens when the SYSOUT arrives at the receiving node.

NJERDR DSP

The data set containing the stream is received by the z/OS BDT subsystem. z/OS BDT decompresses the data and writes the data to spool as a single data set. z/OS BDT spins off the data set with a destination of NJERDR. The data set is queued to a NJERDR DSP or a NJERDR DSP that is waiting is posted to indicate a data set was received and is ready to be processed.

The NJERDR DSP, module IATNTNR, passes records containing a Record Identifier (RID e.g. SYSIN or SYSOUT) and data to module IATNTJS. Module IATNTJS removes the RID from the spool record. For a network SYSOUT stream, module IATNTJS creates at least four data sets. Each data set contains either:

- Job header information
- Data set header information
- File description blocks that contain the addresses of the SYSOUT data sets on spool
- SYSOUT data
- Job trailer information

To process the data, module IATNTNR obtains buffers (JDS, JDAB, JCT, AND JMR) that will be used to create a network job at this node to process the network stream.

14.25 BDT transmission streams

NJE Transfer Units

- For jobs and sysout
  - Data stream identifier - Job header
  - Data stream identifier - Data set header
  - Data stream identifier - (Either job or sysout)
  - Data stream identifier - Job trailer
  - Data stream identifier - Stream end
- Job header, data set header, and job trailer data sets
  - Are distinct spool data sets (DSISO)
  - Data streams in transmission buffers are compressed
  - *Network Job Entry Formats and Protocols (SA22-7539)*

Contents of an NJE Job

An NJE job contains either SYSIN or SYSOUT data and the control records used to identify the data being transmitted A node uses these control records to transmit an NJE job:

- **A job header**: An NJE job must contain a job header to indicate the start of the job. The NJE job can contain either SYSIN or SYSOUT data.
- The job header may include several sub-sections depending on the NJE product where the NJE job originated. All NJE products require a prefix section and a general section in the job header.
- **A data set header**: A data set header is required if the data is SYSOUT or optionally SYSIN data. If the NJE job contains a job and a SYSIN data set, a record characteristics change section (RCCS) may be included in the NJE job to indicate a change in the length of the records.
- **A job trailer**: A job trailer indicates the end of an NJE job. The job trailer is divided into several sections depending on the networking facility where the NJE job originated. All networking facilities require a job trailer prefix section and a job trailer general section.

The home node packages the data to be transmitted in a network stream. JES3 recognizes two types of network streams: a network job stream and a network SYSOUT stream. A network job stream and a network SYSOUT stream both include a job header and trailer. The data sets for the data streams have their own spool space (DSISO) and can be purged following transmission.

The data streams are compressed and their format follows *Network Job Entry Formats and Protocols,* SA22-7539 definitions.

14.26 XMIT JCL statement

//[name] XMIT parameter[,parameter]... [comments]

- Parameters:
  - DEST=nodename[.vmuserid]
  - DLM=delimiter
  - SUBCHARS=substitute
- Allows network transmission from JES3 node
  - To any node in network
  - //*ROUTE Differences
    - Delimiters may be used - End of the transmission JCL
    - All data from (XMIT to delimiter) not examined

XMIT JCL statement

Use the XMIT JCL statement to transmit records from an MVS node to a JES3 node, a JES2 node, a VSE/POWER node, a VM/RSCS node, or an AS/400 node.

Syntax:

```
//[name]  XMIT  parameter[,parameter]...  [comments]
```

Parameters:

- **DEST=nodename[.vmuserid]** - Identifies the destination for all following records until a delimiter stops transmission of the records.
- **DLM=delimiter** - Specifies a delimiter to stop the transmission of records.
- **SUBCHARS=substitute** - Specifies a substitute for internal reader control statements. (JES3 only)

Use of XMIT JCL Statement in a JES3 System allows network transmission of the job. A //*ROUTE XEQ statement can also be used to transmit records from a JES3 node. Because an XMIT JCL statement allows transmission of records that the //*ROUTE XEQ statement does not allow, use XMIT JCL statements rather than //*ROUTE XEQ statements.

The sending system does not process or check the records for validity except when the JCL is processed by an internal reader (such as with TSO/E submit processing). In this case, the system recognizes /*EOF and /*DEL as internal reader control statements and errors can occur on the sending system if /*EOF or /*DEL are included in the XMIT JCL stream.

14.27 Using XMIT statement

- Allows VSE Jobs to be Transmitted

```
//VSESUB   JOB ('acct info'),..........      'submitting job'
//    XMIT  DEST=NODEVSE,DLM=DD
* $$POWER ............
* $$POWER ............
//VSEJOB      JOB  ...
        ..............
/*
DD
```

- Multiple Jobs in Single Transmission

```
//JOBSUB        JOB  ('acct info'),......      'submitting job'
//       XMIT        DEST=NODEA,DLM=DD
//JOB1            JOB ('acct  info'),..........
      .............
//JOB2            JOB ....
       ..........
/*
DD
```

Using XMIT JCL

For example, a JOB statement for the receiving node must immediately follow a //*ROUTE XEQ statement. This requirement means that a //*ROUTE XEQ statement cannot be used to transmit records beginning with $$POWER control statements to a VSE node; however, an XMIT JCL statement can transmit such records.

NJE jobs contain two JOB statements. The first JOB statement is used to route the work to the remote node. The second JOB statement is the statement used to process the work. The JES3 //*ROUTE XEQ or //XMIT statement have their first JOB statement verified at the submitting node and their second JOB statement verified at the execution node.

The system builds network job header and trailer records from information on the JOB statement and any //*NETACCT statements, if included, preceding the XMIT JCL statement. Then the system transmits all the records between the XMIT JCL statement and a delimiter statement.

The records can consist of a job input stream, an in-stream DD * or DD DATA data set, or any job definition statements recognized by the destination node. If the records are a job input stream, and the destination node can process the JCL, the transmitted input stream is executed at the destination node. The records must be 80 characters long.

The records end when the system finds one of the following delimiters:

- /* in the input stream, if a DLM parameter is not coded on this XMIT JCL statement.

14.28 XMIT JCL statement rules

- Must Follow Job Statement and //*NETACCT
- Error on DEST= - All Flushed until Delimiter
- If Delimiter not Found - All JCL Flushed from XMIT on
- Error on XMIT - Everything Flushed
- No Special Characters for Delimiter Except -
  - Specified in Quotes: '$$'

XMIT rules

This figure shows the rules that apply when using the XMIT JCL statement.

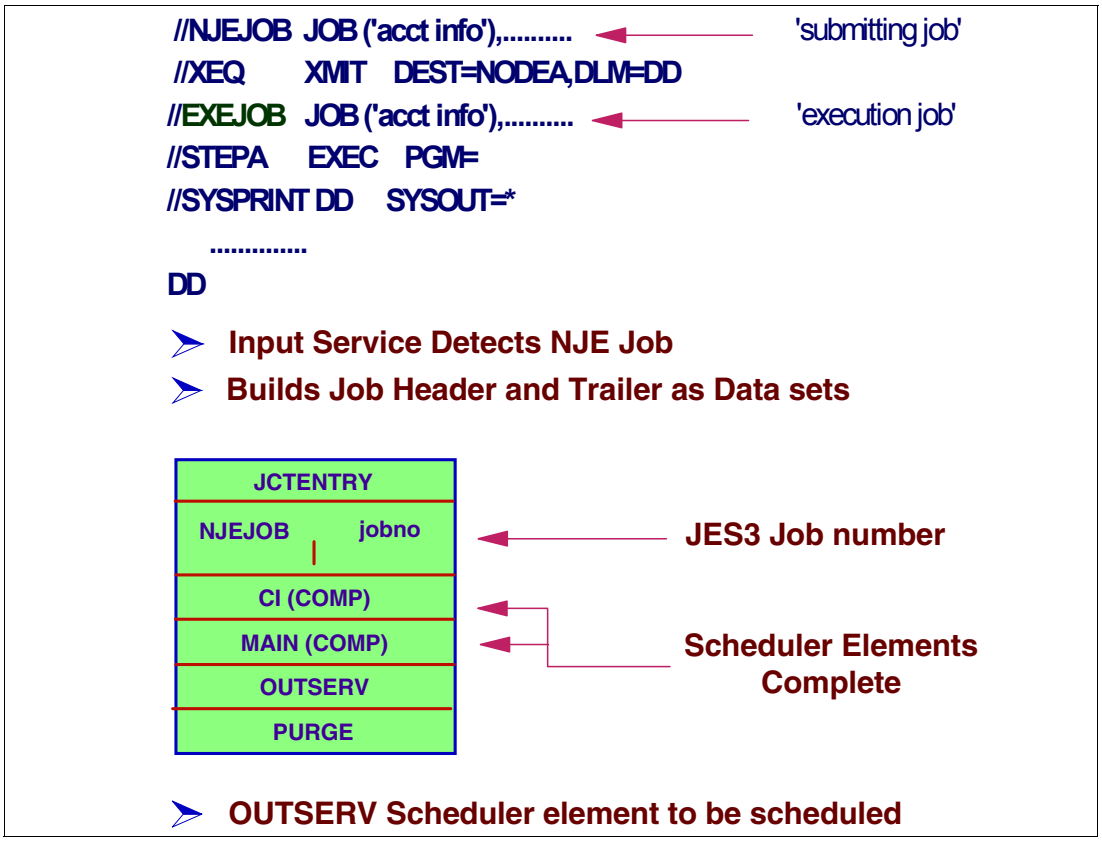
Use the DEST parameter to specify a destination for the following input stream records. The DEST parameter can send the records to a node or, for a node that is a VM system, to a guest system running on the virtual machine.

If the delimiter is not two characters, the system terminates the job and does not transmit any records. If the specified delimiter contains any special characters, enclose the delimiter in apostrophes. In this case, a special character is any character that is neither alphanumeric nor national ($, #, @).

If the delimiter contains an ampersand or an apostrophe, code each ampersand or apostrophe as two consecutive ampersands or apostrophes and enclose the delimiter in apostrophes. Each pair of consecutive ampersands or apostrophes counts as one character.

If the system finds an error on the XMIT JCL statement before a specified DLM parameter, all jobs in the batch are flushed.

14.29 NJE JOB received at originating node



NJE job received at originating node

When a user's job requested a network job to be transmitted, JES3 input service creates for the submitting job a JCT with the CI, MAIN, OUTSERV, and PURGE scheduler elements. The CI and MAIN scheduler elements are marked complete so that OUTSERV and PURGE are the scheduler elements that will processed.

For a network SYSOUT stream, the CI and MAIN SEs have already been marked complete (due to the execution of the job) and the OUTSERV and PURGE scheduler elements will complete the processing required to transmit the network stream. If the network stream will be forwarded to another node, JES3 creates a NJESF, OUTSERV and PURGE scheduler element.

JSS scheduling

The JES3 job segment scheduler (JSS) adds a job to the output service queue when it determines all previous SEs for the job are complete. JES3 places the job on the appropriate output service queue and:

- Indicates that normal or spin-off processing should be performed for the network job
- Posts the OUTSERV FCT

14.30 JES3/BDT NJE outbound job processing

OUTSERV FCT

- OSEs for network job stream
  - JES3 output service has "queue" Q = BDT
    - Q=BDT - Q=WTR - Q=HOLD
  - Post BDT FCT - (Work to do)

BDT FCT

- Create BDT transaction for transmission - (SSI)
  - Transaction now on BDT queue - (BDT jobno)

MVS/BDT on NODEA

- BDT transmits job

JES3/BDT SNA NJE outbound job processing

When the multifunction monitor (MFM) dispatches the OUTSERV FCT for processing, the output service driver module IATOSDR is invoked. The output service driver locates a job and creates output service elements (OSEs) to represent the data sets in the network stream.

An OSE must be built for each part of the network stream to be transmitted across the network. If necessary, an OSE for the job header, an OSE for the job trailer and an OSE for the data set header, if the data is SYSOUT data. Output Service groups the OSEs that represent the network stream together. Since the OSE that represents the data was built before the OSEs for the job header, data set header and job trailer, IATOSBP makes a copy of the OSE created by IATOSDO so that all the OSEs for the network stream are grouped together.

BDTCOMM FCT

After building the necessary control blocks needed for the SNA networking job, the BDTCOMM FCT is posted. The JES3/BDT communications interface notifies the z/OS BDT subsystem at the home node that the JES3 job queue contains a network job that should be transmitted to the destination node by using SNA protocols. Module IATOSBM processes a TYPE=GET request to obtain the OSEs.

Module IATBDCI creates a transaction to send to the z/OS BDT subsystem. Module IATBDCI issues a JSERV to send the data set information in a staging to the BDT queue.

NODEA BDT transmits the job.

14.31 SNA NJE job received at execution node

- BDT receives job — **MVS/BDT on destination node**
- Creates sysout data set
  - Allocate, open write, close, unallocate
- JES3 at unallocation — **JES3 on destination node**
  - "SPINOFF" data set
- Post NJERDR FCT
- Selects sysout data set - IATXOSPC — **NJERDR FCT**
- Creates utility job - ISDRVR - PURGE
  - Uses same modules and logic as BSC NJE
- PURGEs sysout data set

SNA NJE job received at execution node

The transmitting node sends each record in the network stream. BDT in the receiving node reads the record and writes the decompressed record to JES3 spool. All the transmitted records are placed in a single spool data set. The data set contains all the components of the network stream: the job header, the data set header if the data is SYSOUT, the data and the job trailer. Each record in the data set contains a record identifier (RID) which is followed by the data in the record. A RID is used by networking to identify the contents of the record.

Spinoff data set

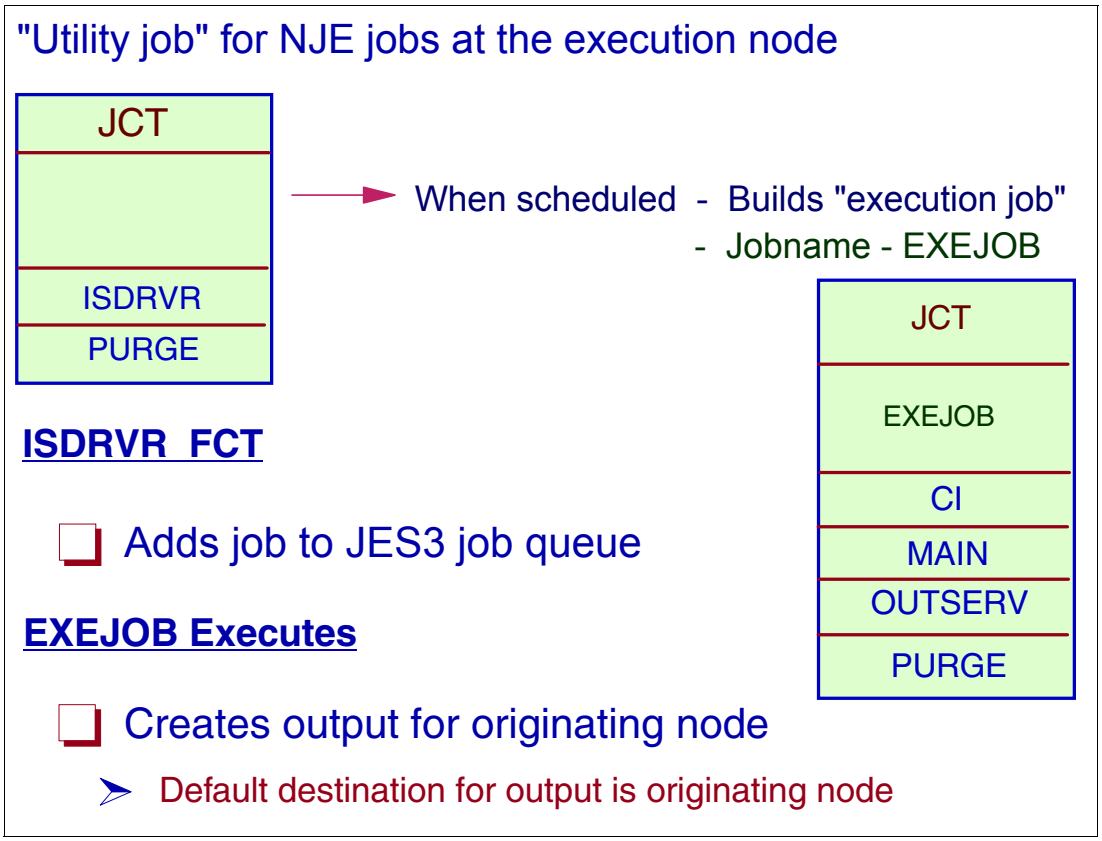
When the entire network job stream is processed, BDT spins off the data set with a destination of NJERDR. The data set is queued to a NJERDR DSP or a available NJERDR DSP that is waiting is posted to indicate a data set was received and is ready to be processed.

Module IATNTNR, the NJERDR DSP driver, requests work from output service by indicating in a writer selection parameter area (WSP) that a SYSOUT data set destined for an external writer named NJERDR is required. The NJERDR uses the IATXOSPC macro service to request a data set from output services and calls 'Networking Job/SYSOUT Receive Module' IATNTJS. IATNTJS creates a “utility” job to process the “spin off” network stream at this node.

NJE utility job

The “utility” job’s scheduler elements are ISDRVR and PURGE for jobs at destination node and NJESF, OUTSERV, and PURGE for SNA and TCPIP remote destination jobs. Remote BSC jobs have NJESND scheduler element instead of OUTSERV.

14.32 JES3 processing for a received NJE job



JES3 processing for a received SNA NJE job

When the “utility” job’s ISDRVR scheduler element job is scheduled, the transmitted network job is read and passed through input service. Input service adds the job as a “normal” job to the execution node’s JES3 job queue with scheduler elements CI, MAIN, OUTSERV, and PURGE.

EXEJOB job

When the job finally executes and creates SYSOUT output, the default destination for the SYSOUT data is the originating node. The OUTSERV scheduler element of the EXEJOB job will then send the output to the originating node.

14.33 OUTSERV for NJE job at execution node

OUTSERV FCT

- Builds OSE for sysprint - DEST=originating_node
- Module IATOSBP (at OSE construct)
  - Builds NJE header trailer
  - Place OSEs on Q=BDT Queue for SNA NJE jobs (No utility job)
  - BDT FCT posted

BDT FCT

- Select OSEs
- Create transaction for MVS/BDT via SSI

z/OS BDT A/S

- Select transaction - transmit output to originatin node
  - Record identifiers are added to each record.

OUTSERV for an NJE job at execution node

When the OUTSERV FCT is scheduled to build OSEs, the output service driver, IATOSDR o completes building the OSEs for network jobs (both “job stream” and “SYSOUT stream” work).

OSE construction

An OSE must be built for each part of the network stream to be transmitted across the network. If necessary, module IATOSBP builds an OSE for the job header, an OSE for the job trailer and an OSE for the data set header, if the data is SYSOUT data. Output Service groups the OSEs that represent the network stream together.

After module IATOSDR has built the necessary control blocks needed for the SNA networking job, the BDTCOMM FCT is posted. The JES3/BDT communications interface routine notifies the z/OS BDT subsystem that the JES3 job queue contains a network job that should be transmitted to the destination node by using SNA protocols.

BDT FCT

A routine in module IATBDCI (BDT FCT) is invoked by the JES3 dispatcher (multifunction monitor - MFM) when the FCT is posted. Processing in module IATBDCI uses the IATXOSBM TYPE=GET macro to search the output service writer RESQUEUE chain for a network job that requires SNA protocols. Module IATBDCI creates a transaction to send to the z/OS BDT subsystem and issues a JSERV to send the information in a staging area to BDT address space. BDT then transmits the network job by selecting the transaction from its queue.

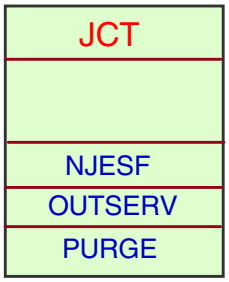
14.34 Output received at originating node

MVS/BDT Address Space at Originatin Node

- BDT receives transmission
- Creates SYSOUT data set on JES3 spool
  - Allocate, open, write, close, unallocate
  - Unallocate - specify external writer name - NJERDR
  - Output is compressed in transmission buffers

NJERDR FCT

- Select OSE and use IATNTJS to
  - Create utility job
  - NJESF - OUTSERV - PURGE



Output received at originating node

At the originating Node, BDT receives the SYSOUT network job and writes the received data to JES3 spool. The spool data set for the received network job is created with an external writer name of NJERDR. JES3 output service OSE construction posts the NJERDR FCT once an OSE is build for the network job's spool data.

NJERDR FCT

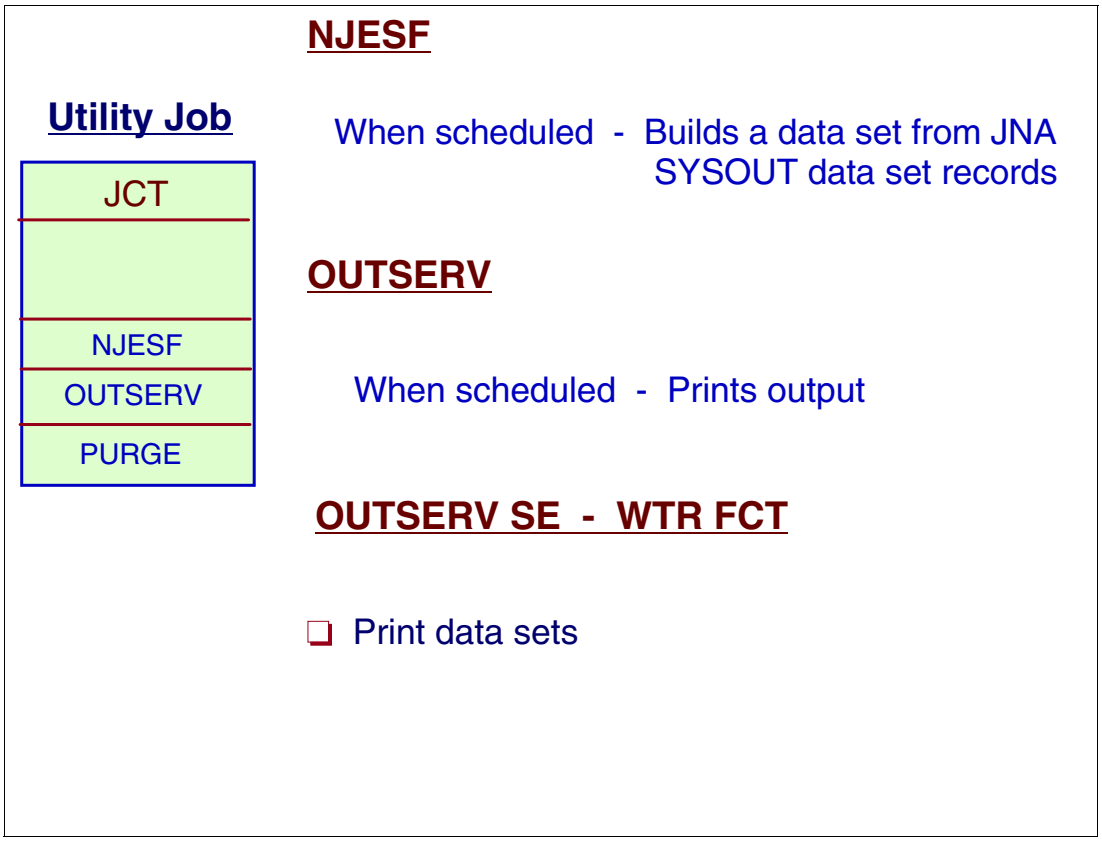
The NJERDR FCT selects the OSE and creates a utility job. When a network stream that contains a SYSOUT data set is received, module IATNTJS builds a job that will process the SYSOUT data. This job has NJESF, OUTSERV, and PURGE scheduler elements.

NJESF DSP

The NJESF DSP prepares a SYSOUT stream for local printing or store-and-forward through a communication link. It also prepares a job stream for store and forward via a BSC or SNA link.

According to the networking specifications, SNA data transmission is required to use compression to reduce the length of records for transmission by removing blanks and duplicate characters. Compaction (reducing the length of records by representing certain 8-bit characters with only 4 bits) is optional and is controlled by NJE and SNA protocols.

14.35 Output processing at originating node



Process output at originating node

After receiving a network job's SYSOUT stream from the execution node JES3 has built a "utility" job with NJESF, OUTSERV, and PURGE scheduler elements to process the network job's output.

JES3 networking creates "utility" jobs with the NJESF scheduler element for the following types of NJE streams received from other nodes:

- job streams to be forwarded through BSC, SNA or TCPIP nodes.
- SYSOUT streams to be forwarded through BSC, SNA or TCPIP nodes.
- SYSOUT streams to be processed locally.

NJESF DSP

The NJEFS DSP, once scheduled, prepares the network job for the next scheduler element in the "utility" job. For SYSOUT streams for local processing it updates the JDS in preparation for the OUTSERV DSP.

All of the data set headers for this stream are in a single spool file. Any SYSOUT data set within the stream may be preceded by one or more data set headers. the destinations for these multiple data set headers can be mixed, i.e. local, BSC, SNA and TCPIP destinations can all exist in a single received stream.

NJESF DSP processing for the various destinations is as follows:

- Data sets for local processing at this node - A JDS entry is built from the information in the data set header. if a data stream section exists in the data set header, and it contains scheduler work block (SWB) data, the SWB data are written to spool.
- Data sets destined for a remote node through a BSC link - First, the data set header is compressed and written to spool. The SYSOUT data set itself is compressed and written to spool in the same file as the data set header. A skeleton JDS entry is then built to represent the compressed data, and added to the JDS chain.
- Data sets destined for a remote node through a SNA or TCP/IP link - SNA NJE streams are processed through the z/OS BDT product where as TCP/IP/NJE streams are processed through NETSERV interface. First, the data set header is written to spool in its own file. A skeleton JDS entry is created for the data set header and added to the JDS chain. An entity name is built to represent the sysout data set. If a JDS entry for the SYSOUT data set has not previously been built, it is built at this point and added to the JDS chain.
- Store-and forward job streams processing - The JDS entries for the “utility” job's input files are read. The job's scheduler element structure is changed, if necessary, to match the destination type (i.e. NJESF/NJESND for BSC and NJESF/OUTSERV for SNA and TCPIP). The DSP cleans up and returns if the destination is through SNA/TCPIP node. For BSC, the job header, JCL data, and job trailer are compressed and written to spool in separate files. Skeleton JDS entries are built for these files and added to the JDS chain. control is then passed back.

14.36 JES3/BDT SNA NJE transmission summary

Originating Node		Execution Node	
JES3	BDT	JES3	BDT
NJE Job Submitted Build JES3 "Utility" Job OUTSERV - Build OSEs BDT FCT - Transaction			
	Get Transact. Transmit Job		Receive Job Create Data set
		NJERDR FCT - Get OSE Create Utility Job -ISDRVR - PURGE ISDRVR FCT - Add Job (Job executes - Output for originating node) OSE for originating node IATOSBP - OSEs BDT FCT - Transaction	
	Receive sysout - Create NJE Data set		Get Transact. Transmit sysout
NJERDR FCT - IATNTJS Create Utility Job (njesf-outserv-purge) WTR - Prints Data set			

Transmission summary

This figure is a summary of the process of sending a SNA NJE job from an originating node to execute on the execution node and the SYSOUT coming back to the originating node to print that was discussed in the previous visuals.

14.37 z/OS BDT network streams and BDT group identifier

- GROUPID
  - Number assigned sequentially at OSE build
    - BDTnnnnn Where nnnnn=00000 to 65535
    - OSEGRPID Field (for BDT)
- To odentiry netwrok streams
  - GROUPID, jobname, and jobno

```
*I U Q=BDT J=?
IAT8131 JOB PCPRTJV (JOB22144), D=WTSCPLX9, L=8, PG=0, SR=8,
IAT8131 JOB PCPRTJV (JOB22144), BY=4084, BG=BDT00000.
IAT8131 JOB PCPRTJK (JOB22145), D=WTSCPLX9, L=8, PG=0, SR=8,
IAT8131 JOB PCPRTJK (JOB22145), BY=4084, BG=BDT00000.
IAT8131 JOB PCPRTJK (JOB22145), D=WTSCPLX9, L=8, PG=0, SR=8,
IAT8131 JOB PCPRTJK (JOB22145), BY=4084, BG=BDT00001.
IAT8131 JOB PCPRTJK (JOB22145), D=WTSCPLX9, L=8, PG=0, SR=8,
IAT8131 JOB PCPRTJK (JOB22145), BY=4084, BG=BDT00002.
IAT8131 JOB PCPRTJK (JOB22145), D=WTSCPLX9, L=8, PG=0, SR=8,
IAT8131 JOB PCPRTJK (JOB22145), BY=4084, BG=BDT00003.
IAT8119 NUMBER OF JOBS FOUND : 2
IAT8141 NUMBER OF BDT JOBS FOUND : 5
```

BDT network streams and BDT group identifier

Each JES3 job is identified by a JES3 number and a groupid. In the box below,JES3 JOB13 has one networking job (groupid BDT00000) and JOB14 has four networking jobs (groupids BDT00000, BDT00001, BDT00002, and BDT00003). Each JES3 networking job/SYSOUT stream running in BDT has a different z/OS BDT group number. In the table below, JOB13 and JOB14 indicates each networking job/SYSOUT stream as it is identified in z/OS BDT.

JOB13	JOB14
BDT00000	BDT00000
	BDT00001
	BDT00002
	BDT00003

After the job is placed on the z/OS BDT job queue, z/OS BDT sends a transaction notification BSID to the *JES3/BDT Communications Interface* (IATBDCI). This BSID is a queued response BSID. After receiving that transaction notification BSID, IATBDCI issues an *Output Service BDT/TCP Manager* IATXOSBM macro PUT request. This PUT request, containing indicators in the *Output Service BDT/TCP Parameter List* (IATYOSB) indicates that the file work has been **queued** on z/OS BDT (field OSEBFLG2 is set to OSEBQUED). T he IATXOSBM macro processing in IATOSBM module updates the OSEs pertaining to that job to indicate that the job has been placed on the z/OS BDT queue. The OSE is also updated with the BDT job number. IATOSBM uses the jobname and groupid to retrieve the Resident Job Queue Table entry (RQ) for the job. IATOSBM then returns control to IATBDCI.

14.38 Using JES3 DSISO for SNA NJE output data sets

- Can be used to 'Break-up' Transmission Streams
  - Keep Streams Smaller
- Sample Job Transmission from NODEA

```
//JOBA        JOB ( 'acct info' ),MSGCLASS=X
// STEPA      EXEC PGM =. . . . . .
//SYSPRINT    DD  SYSOUT = X, DEST = NODEB
//PRINTB      DD  SYSOUT = X, DEST = NODEB
```

  SYSOUT,CLASS = X,TYPE = (PRINT,DSISO)

- Unique networking Streams - Separate Transactions
- Separate MVS/BDT Job Numbers
- OUTSERV SNAGROUP={ YES | NO }
  - Specifies whether JES3 groups DSISO data sets destined for SNA/NJE nodes.

Using JES3 DSISO for SNA NJE

To make transmission streams smaller, use DSISO SYSOUT classes for processing of SYSOUT to other nodes. On JES3 (Output Service Defaults and Standards) initialization statement you can specify:

SNAGROUP= {YES | NO}

Specifies whether JES3 groups DSISO data sets destined for SNA/NJE nodes.

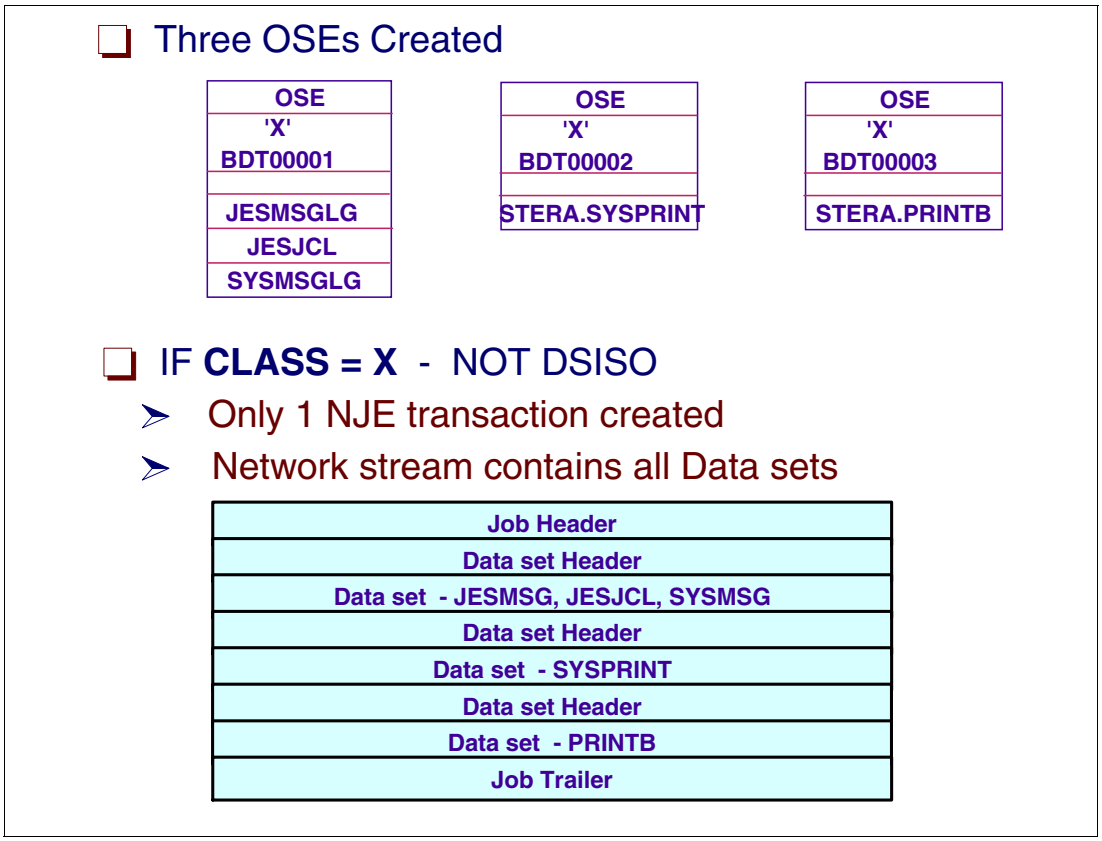
YES - Specifies that JES3 group DSISO data sets, if possible, which are destined for SNA/NJE nodes. Specifying YES allows the receiving node to assign a single job number to a group of data sets that were produced by one job at the originating node.

Specifying SNAGROUP=YES also causes JES3 to always group non-spin-off DSISO data sets (TYPE=DSISO specified on the SYSOUT initialization statement for the SYSOUT class). However, spin-off data sets, such as those produced by specifying FREE=CLOSE in the JCL, are sent separately because they are sent when the data set is closed rather than at the end of the job.

> **Note:** Specifying SNAGROUP=YES prevents JES3 from releasing spool space used for DSISO data sets until all of the job's data sets have been sent to the specified node.

NO - Specifies that JES3 send DSISO data sets destined for SNA/NJE nodes as separate data sets. Specifying SNAGROUP= NO allows JES3 to release spool space as soon as a data set is sent.

14.39 Using DSISO for SNA NJE



Using DSISO

This is the example from the last figure after the OSEs have been created. The 3 OSEs would be processed separately. After all SYSOUT data sets have had their SYSOUT OSEs built for a job, module IATOSDR calls IATOSBP, supplying a pointer to a RESQUEUE indicating the job that is being processed and a pointer to the chain of SYSOUT OSEs (built by IATOSDO) for the current job. The head pointer to the chain of in-storage SYSOUT OSEs for this job is in IATODDR (location OSDRSECH). IATOSBP obtains working storage for building the job header, data set header and job trailer.

Note: If CLASS=X was not a DSISO class, then there would be one transaction created and one transmission stream.

14.40 BDT initialization definitions

SYSID	– Specify z/OS BDT for Other Nodes
BDTNODE	– Define Nodes to z/OS BDT
CELLPOOL	– Pool for z/OS BDT use SP= 21
OPTIONS	– z/OS BDT Installation Options
SNABUF	– Allocate storage for data buffers

```
CELLPOOL,ID=.. ,CNUM=.. ,SCNUM=.. ,MAXET=. ,AUTODEL= .. ,PGRLSE=..
OPTIONS,JES3 = YES,.........(for NJE support)
ENDRBAM
SYSID,NJEAPPL = .... ,NJENAME = .... ,NJEAPSWD = password,...
BDTNODE,N = .... ,APPL = .... , TYPE = NJE, LU = 5,....
SNABUF,SIZE=....,PRI=....,SEC=(....,...),AUTODEL={YES | NO}
ENDINIT

NJEAPPL APPL AUTH = ACQ,  ........
```

- Define LU as Multiple of 4 Plus 1
- LU = 5 is Default for Number of VLUs betweem Nodes
- Specify either 5,9,13,17,21,25, or 29
- NOTE: Maximum Nodes is 100

BDT initialization definitions

To define the environment in which BDT will operate each time it is started, the system programmer must code initialization statements. The system programmer can specify the amount of main storage BDT will use, the names of nodes in the network, the pacing rate for communication between nodes, and many more operational characteristics. BDT initialization statements are similar in function to JES3 initialization statements. BDT reads these initialization statements during each warm or cold start of BDT.

Initialization statements:

CELLPOOL A cell (sometimes called a storage cell) is another name for the main storage that is defined by a CELLPOOL statement. A cell pool is a group of cells that are related by the fact that BDT uses them for the same purpose.

A subsystem with only a SNA NJE node requires 12 different cell pools. CELLPOOL statements must be first in an initialization stream.

OPTIONS You can use the OPTIONS statement to specify operating characteristics of the home BDT subsystem.

SYSID The SYSID statement specifies the name of the home SNA NJE node. It also provides the application name and password that identify each node to VTAM.

BDTNODE BDTNODE TYPE=NJE specifies that the **home and remote** BDT nodes may send and receive only SNA NJE jobs and SYSOUT. This parameter corresponds to the BDT SNA NJE feature for JES3 installations.

1-100 BDTNODE statements (home and remote combined) per initialization stream.

SNABUF Use the SNABUF statement to allocate storage for data buffers. Each BDT subsystem uses data buffers to send data to and receive data from other BDT subsystems. You must define the size of these buffers and the number that BDT is to allocate. You may define several different sizes. You must ensure that each buffer size you define matches the size of an VTAM request unit (RU). BDT allocates these buffers from subpools 2, 3, 4, or 5.

SNA NJE transfers-- VTAM APPL definition statement

BDT is an VTAM application program. If BDT is to handle SNA NJE transfers, you must define BDT to VTAM as a SNA NJE node by coding an VTAM APPL definition statement:

name The label on the APPL statement must match the name on the NJEAPPL parameter of the BDT SYSID statement.

AUTH=ACQ You must code AUTH=ACQ to allow the BDT application to establish sessions using the VTAM SIMLOGON macro.

EAS=n Specify a value for n that is equal to the anticipated number of concurrent sessions that this node will have with all other BDT nodes. SYS1.SBDTSAMP contains a samples.

Virtual logical units

To achieve concurrent data transfer, BDT multileaves, over a single session, the data for each active transfer by splitting the session into "subsessions" known as virtual logical units (VLUs). A VLU is a logical path between two nodes that represents one user of a session between BDT nodes. Each active transfer is considered to be a user of one VLU.

For each session, BDT always requires one VLU for communicating control information. The other VLUs transfer the data.

For each JES3 SNA NJE session, 28 data transfer VLUs are possible. Of the 28 possible VLUs in a session, one node can schedule 14 (7 for sending jobs and 7 for sending output) and the other node can schedule 14 (7 for sending jobs and 7 for sending output). They are specified in groups of four.

14.41 BDT VLU usage

- Communication VLU - Dedicated to Messages - Commands - Control
  - VLU - Outbound JOB Streams
  - VLU - Outbound SYSOUT Streams
  - VLU - Inbound JOB Streams
  - VLU - Inbound SYSOUT Streams
- 9 VLUs - 2 each of Streams - 1 Dedicated .. etc.

BDT VLU usage

Virtual logical unit (VLU): In BDT, data and program logic that represents one user of a SNA session. The virtual logical unit enables more than one user to concurrently use a session.

A single set of VLUs between node A and B would consist of:

- Communication VLU (for control information)
- VLU for job stream 1 from node A to node B
- VLU for output stream 1 from node A to node B
- VLU for job stream 1 from node B to node A
- VLU for output stream 1 from node B to node A

A VLU can transfer data in two directions--either to or from a node. During the time a VLU is allocated to a specific transfer, however, that VLU can transfer data in only one direction. Once that transfer completes and BDT deallocates the VLU, the VLU will again be able to transfer data in either direction.

14.42 SNA NJE commands

```
*I NJE N=WTSCPLX9
 IAT8659 - NODE  WTSCPLX9  NOHOLD  PATH  WTSCPLX9  SNA PROTOCOL
*S BDT I NODE=WTSCPLX9
 IAT9514 COMMAND/TRANSACTION ACCEPTED, TQI DISABLED
 BDT8646 WTSCPLX9 IS ONLINE
 BDT8645 NJE NODE WTSCPLX9 VLU COM TYPE COM VLU STATUS ONLINE / ALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU OJ1 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU OS1 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU IJ1 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU IS1 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU OJ2 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU OS2 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU IJ2 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
 BDT8645 NJE NODE WTSCPLX9 VLU IS2 TYPE XFR VLU STATUS ONLINE / UNALLOCATED / CLOSED /
*S BDT C SNA NODE=WTSCPLX9
 IAT9514 COMMAND/TRANSACTION ACCEPTED, TQI DISABLED
 BDT2862 SESSION QUIESCE REQUESTED FOR WTSCPLX9 (WTSCPLX9), TYPE=NJE
 BDT2803 SESSION TERMINATED WITH WTSCPLX9 (WTSCPLX9)
/S SNA NODE=WTSCPLX9
 BDT2822 LOGON IN PROGRESS FOR WTSCPLX9 (WTSCPLX9)
 BDT2802 SESSION ESTABLISHED WITH WTSCPLX9 (WTSCPLX9), TYPE=NJE
 BDT2877 - WTSCPLX9 (WTSCPLX9) - BUFSZ = 01024, SNA TERMINATION EXTENSION = YES
```

SNA NJE commands

You can submit commands by typing them at your console, following the rules of command syntax.

Frequently used commands can be assigned to the program function keys (PF keys) on the console's keyboard. The PF keys can be set up to issue a command immediately when you press them, or to produce skeletal commands with blank spaces for you to fill in.

BDT command prefix

The prefix tells the system that the command is a BDT command. The prefix you use depends on the type of console you are using (MCS, JES3, or TSO):

- From MCS consoles:
  - **bdt-char** - The IEFSSNxx parmlib member BDT subsystem definition keyword parameter **C=bdt-char** defines the BDT command character
- From JES3:
  - `*S,BDT, bdt_command`

14.43 JES3 networking over TCP/IP

- Network Job Entry (NJE) using Binary Synchronous (BSC) or System Networking Architecture (SNA) protocol requires hardware that is out of service
- TCP/IP/NJE sessions take advantage of TCP/IP hardware independent layered stack to establish connections over a number of existing hardware protocols.
- Other NJE networking components (RSCS, VSE, JES2) use NJE over TCP/IP
- NJE communication driven through **Netserv** address space
  - Netserv functionality in common JES component
  - TCP/IP/NJE sessions support IPv6
  - TCP/IP/NJE sessions support TLS/SSL
  - Performance enhacements over SNA NJE

JES3 networking with TCP/IP

Starting with z/OS V1R8, JES3 provides support for NJE over TCP/IP. JES2 has been providing NJE over TCP/IP support since z/OS V1R7. VM (RSCS), VSE, and AS/400 have been providing NJE over TCP/IP support for several releases.

TCP/IP/NJE sessions take advantage of TCP/IP hardware independent layered stack to establish connections over a number of existing hardware protocols such as Ethernet and token ring. To use TCP/IP sessions, z/OS Communications Server TCP/IP requires z/OS UNIX System Services and ACF/VTAM to be configured and active. TCP/IP/NJE sessions support IPv6 and TLS/SSL, assuming that there is support on the node being connected to.

In JES3, TCP/IP/NJE can be thought of as a hybrid between BSC and SNA. NJE over TCP/IP uses the same record structure and data streams as BSC NJE. But like SNA NJE, in JES3, the NJE communication is driven through a separate *Netserv* address space that is analogous, although not functionally identical, to BDT.

Netserv address space

The role of the Netserv address space is to make all the communication interface calls to TCP/IP. The Netserv and JES cooperate to despool from JES data that is to be sent and to spool to JES data that is received. The Netserv functionality is in the JES Common component; therefore, with limited operational differences, it is the same in both JES2 and JES3.

IPv6

IPv6 uses a 128-bit address space, which has no practical limit on global addressability and provides 3.4 ¶ 10(50) unique addresses. This gives enough addresses so that every person could have a single IPv6 network with many nodes, and still the address space would be almost completely unused.

IPv4 addresses are represented in dotted-decimal format. The 32-bit address is divided along 8-bit boundaries. Each set of 8 bits is converted to its decimal equivalent and separated by periods. In contrast, IPv6 addresses are 128 bits divided along 16-bit boundaries. Each 16-bit block is converted to a 4-digit hexadecimal number and separated by colons. The resulting representation is called colon-hexadecimal.

The following are the three conventional forms for representing IPv6 addresses as text strings:

- The preferred form is x:x:x:x:x:x:x:x, where the x's are the hexadecimal values of the eight 16-bit pieces of the address. For example:

  ```
  2001:DB8:7654:3210:FEDC:BA98:7654:3210
  2001:DB8:0:0:8:800:200C:417A
  ```

  It is not necessary to write the leading zeros in an individual field, but there must be at least one numeral in every field (except for the case described in the following bullet).

SSL and TLS

The Secure Socket Layer (SSL) protocol provides data encryption, data origin authentication, and message integrity. It also provides server and client authentication using X.509 certificates. SSL begins with a handshake during which the server is authenticated to the client using X.509 certificates. Also, the client can optionally be authenticated to the server. During the handshake, security session parameters, such as cryptographic algorithms, are negotiated and session keys are created. After the handshake, the data is protected during transmission with data origin authentication and optional encryption using the session keys.

SSL is not defined by the IETF. TLS is based on SSL and is defined by the IETF as RFC 2246.

The transport layer security (TLS) provides endpoint authentication and communications privacy over the Internet using cryptography. Typically, only the server is authenticated (i.e., its identity is ensured) while the client remains unauthenticated; this means that the end user (whether an individual or an application, such as a Web browser) can be sure with whom they are communicating. The next level of security in which both ends of the "conversation" are sure with whom they are communicating is known as mutual authentication. Mutual authentication requires public key infrastructure (PKI) deployment to clients unless TLS-PSK or TLS-SRP are used, which provide strong mutual authentication without needing to deploy a PKI.

TCP/IP/NJE performance enhancements

The TCP/IP network job receiver does not use double spooling like BDT does, the data is written to spool using USAM on the server side. When the job reaches the global, which builds the required job structures for the incoming job. The ownership of the spooled data is changed, but the job is not respooled.

14.44 Networking over TCP/IP

- To transport data between NJE nodes using TCP/IP
  - A virtual circuit (TCP connection) is established between the two nodes
    - A virtual circuit is a path between two applications over which TCP packets may be sent
    - In an IP network each system is assigned an IP address
    - Port numbers indicate which packets correspond to which applications

Networking over TCP/IP

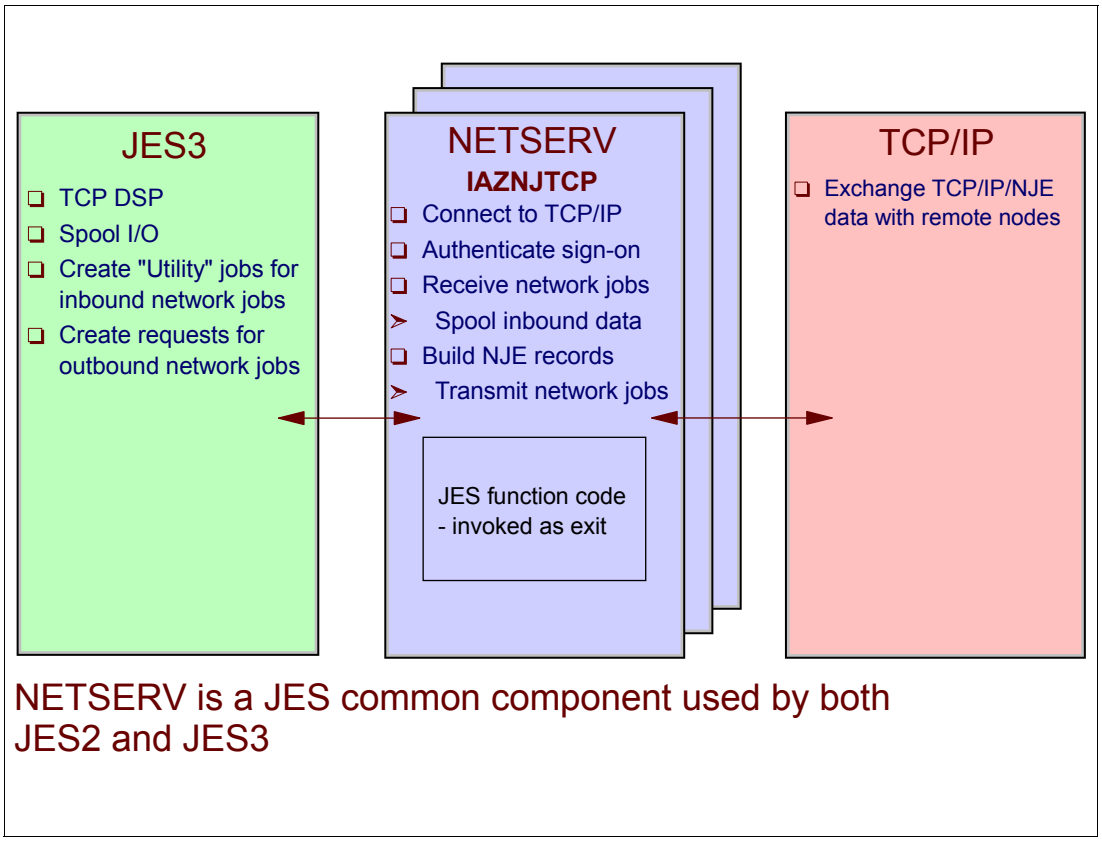
Before data can be sent from one network job entry (NJE) node to another using TCP/IP, a virtual circuit must be established between the two nodes. A *virtual circuit* is a path between two applications over which TCP packets can be sent.

TCP sets up a connection between two endpoints (nodes), identified by the respective *IP addresses* and a *port number* on each. A port is the abstraction used by Internet transport protocols to distinguish among multiple simultaneous connections to a single destination host.

Each system in a network is assigned an IP address. Each TCP/IP service machine will have an IP address. Many applications on a system can use the TCP/IP service machine.

To enable the TCP/IP service machine to separate incoming packets, the applications use *port* numbers to indicate which packets correspond to which applications. TCP allows an application to open a virtual circuit in either *passive* mode (waiting for incoming requests to open, also known as *server mode*) or *active* mode (sending requests to open, also known as *client mode*). In general, one TCP application (the server) issues a passive open for a port number (known as the *well known port*) and the other TCP application (the client) will normally issue an active open for the well known port on the system (IP address) where the first (server) application is located. Either side can attempt to connect to the other side's port. The TCP connection or virtual circuit path between the two applications will be completed and data can be exchanged over the path.

14.45 Address spaces for JES3 TCP/IP/NJE



Address spaces for TCP/IP/NJE

In order to minimize the amount of duplicate implementation work in JES2 and JES3, all data communication in TCP/IP is done using a server address space, Netserv, which is one an additional part of the JES common component. JES code running in the Netserv server address space is required to control the definition and startup of the server. The JES code, invoked as exits in the server address space, moves data back and forth between the server and JES.

Netserv address space

The role of the Netserv address space is to make all the communication interface calls to TCP/IP on behalf of the JES. The Netserv listens for connections, incoming and outgoing data and cooperates with JES3 to read data from the spool or write data to the spool.

In the JES3 TCP/IP/NJE implementation, the Netserv address space can run either on the global or a local main processor. There is no limit to the number of Netserv address spaces that can run in the JES3 complex simultaneously.

A Netserv address space is a started task that requires authority to communicate with JES in order to spool, despool jobs and data sets. The common TCP/IP/NJE component, IAZNJTCP, uses TCP/IP services and, thus, must operate in the z/OS UNIX System Services environment. Therefore, the Netserv must have the proper authority to function. The required authorities consists of:

- A definition in the STARTED class to associate the started task with a userid.
- An OMVS segment in the security definition for the userid associated with the started task.

14.46 Defining JES3 TCP/IP/NJE

- Any TCP/IP definitions to be used by JES3 must be defined in TCP/IP
- Netserv definitions (NETSERV):
  - A host name and port of a local socket over which the Netserv listens incoming information from TCP/IP.
    - Host name and Port
  - A system name where the Netserv is to run
  - The name of the TCP/IP stack.
- Socket definition (SOCKET):
  - Foreign socket used to connect to TCP/IP and a unique name representing the JES3 global view of the:
    - Socket - Host name - Port number - Netserv under which the socket task will run
- JES3 network node definition (NJERMT):
  - TYPE=TCPIP specifies a TCP/IP networking protocol

Defining JES3 TCP/IP/NJE

Any TCP/IP definitions to be used by JES3 must be defined in TCP/IP. This includes IP addresses, host names for the TCP/IP resolver, port names, and special service names. A JES3 node can use NJE protocol over TCP/IP to exchange jobs and data sets with a remote node. TCP/IP/NJE consists of the following TCP/IP definitions:

Netserv definition

A **network server**, typically abbreviated Netserv, which runs as a separate address space from JES3 on the global or any local that is at a sufficient software level. A NETSERV definition consists of the following information:

- A host name and port of a local socket over which the Netserv listen for incoming information from TCP/IP. The host name can be left unspecified in order to indicate to TCP/IP that it can use any IP address that is defined for the home node.

  The port can be any number from 1 to 65,535. It can also be specified as, or allowed to default to, the special value of 0. This value indicates that a default service name should be used. The service name is VMNET for non-secure connections, and NJENET-SSL for secure connections, if TLS=YES is specified.

  More than one Netserv can use the default IP address by omitting the HOSTNAME parameter, but they must specify unique ports.

- A system name where the Netserv is to run. If you do not specify a system name, the Netserv will run on the current JES3 global. If a DSI is performed while a Netserv is active, the Netserv will remain active on the same system on which it was active before the DSI.

However, if the Netserv is subsequently brought down and back up, it will start on the new global.

- The name of the TCP/IP stack (i.e the name of TCP/IP address space) which contains the IP addresses over which the Netserv listens for incoming information.

> **Note:** The Internet protocol suite (commonly TCP/IP) is the set of communications protocols that implement the protocol *stack* on which most networks run.
>
> A protocol stack (sometimes communications stack) is a particular software implementation of a networking protocol suite. The terms are often used interchangeably. Strictly speaking, the suite is the definition of the protocols, and the stack is the software implementation of them.

Socket definition

A **socket definition**, defining a foreign socket that will be used to connect to TCP/IP. Each socket runs as a subtask under a Netserv address space. The SOCKET definition consists of the following information:

- A unique name representing the view of the socket by JES3 global, and used in inquiry and modify commands as well as internal JES3 processing of outbound and inbound TCP/IP data.
- A host name.
- A port number, handled the same way as the Netserv port number.
- The Netserv under which the socket task will run.

The concept of *foreign* and *local* sockets exist in TCP/IP. A JES3 socket defines JES3's usage of a foreign socket only. The local socket is implicitly defined by the NETSERV statement.

> **Note:** A socket is an abstraction that represents an endpoint of communication. Most applications that consciously use TCP and UDP do so by creating a socket of the appropriate type and then performing a series of operations on that socket. The operations that can be performed on a socket include control operations (such as associating a port number with the socket, initiating or accepting a connection on the socket, or destroying the socket) data transfer operations (such as writing data through the socket to some other application, or reading data from some other application through the socket) and status operations (such as finding the IP address associated with the socket).

JES3 network node definition (NJERMT)

The NJERMT initialization statement to define a node in the JES3 job entry network. TYPE=TCPIP specifies the TCP/IP networking protocol to be used for communicating with a directly-connected remote node.

You may use the SECSIGNON parameter (valid for TYPE=TCPIP only) to indicate that the signon procedure includes additional checking using the encryption of a random string to confirm the identity of the node.

14.47 JES3 TCP/IP/NJE compared with BSC and SNA

- NJE definitions by TYPE=
  - NJERMT TYPE=BSC - DEVICE with DTYPE=NJELINE
  - NJERMT TYPE=SNA - VTAM communication with BDT
  - NJERMT TYPE=TCPIP
    - NETSERV and SOCKET
- All definitions are made in JES3
  - The Netserv definition for a socket plays the same role as the SYSID definition with BDT
  - The socket definition within the Netserv plays the same role as a VTAM LU definition inside BDT
- Netserv uses SAPI to select outbound data, BDT uses PSO
- Netserv and BDT uses USAM to spool inbound data
- JES3 TCP/IP network job receiver does not use double spooling like is done for SNA NJE jobs

TCP/IP networking

The Netserv address space plays the same role in the TCP/IP/NJE function as BDT does in SNA. Certain definitions in one world have analogous definitions in the other. More specifically, the Netserv definition plays the same role as the SYSID definition in the SNA world, it defines the server address space. The socket definition within the Netserv plays the same role as a VTAM LU definition inside BDT. But a major difference is that while BDT has its own initialization stream and definitions, the Netserv and Socket definitions and modify commands to add, change, or delete these definitions all live in JES3.

Netserv and SAPI

The Netserv is similar to BDT in another way, too. When an outbound job stream is to be sent, both address spaces use a SYSOUT retrieval subsystem interfaces to despool this data from JES3 and then send it to the destination node. BDT uses the Process Sysout (PSO) SSI and USAM to despool data, the Netserv uses the Sysout Application Programming Interface (SAPI) and the block spooler.

SNA NJE with BDT

When a SNA NJE node gets an incoming network job, BDT receives the incoming network job (job header, data set header, data, and job trailer). It writes each of these to separate files on spool using DYNALLOC, OPEN, PUT, and CLOSE. This is the first stage of spooling. Then the NJERDR FCT, under control of module NJE Reader Driver (IATNTNR), reads the data off spool and feeds the resulting buffers to Networking Job/SYSOUT Receive module (IATNTJS), which builds the job structure and rewrites the data. This is the second stage of spooling.

BDT's copy of the data and the job under construction both exist until the job structure is completely built at which point the BDT copy is purged.

Netserv and TCP/IP

The Netserv, like z/OS BDT, receives jobs or sysout streams from remote nodes and writes the data to the JES3 spool. To accomplish this, a modified SNA NJE like processing is used. When the Netserv server receives inbound network job records (these could include, either data or a job header, data set header, or job trailer) it calls, as an exit, a JES3 TCP (TCP/NJE) Address Space Exit Routine (in IATNTTXR module). IATNTTXR's functional routine allocates and opens a dummy spool data sets with external writer name TCPNJE for the job header, data set header, data, and job trailer.

Open and close processing for SYSOUT is data involves IATSIOR and IATDMJA and the communication data area JDS Interface Block (JIB). A JIB represents a single Job Data Set Control Block (JDS) that is being created in a user address space and is being sent over to the global to "register" as a global control block.

To avoid double spooling, TCP/IP/NJE requires multiple JDSes to have spool space allocated from the same Job/Data set Track Allocation Table (DSTAT), instead of individual DSTATs. Instead of individual JIBs, a Multiple JDS Interface Block (MJIB) is used by the JDS access SSISERV request. The MJIB consists of multiple JDS entries.

When Netserv has received each piece of the inbound network job's data, JDS entries have been constructed, and placed into the MJIB, the MJIB is sent (SSISERV) to the global. If there are too many JDS entries to fit into one staging area, a multi-segmented staging area is sent.

JES3 global processing

On the global processor, IATDMJA, the JDS Access Interface module, invokes IATDMJA is IATDMJAM to process the multiple JDS interface block created in the Netserv address space. IATDMJAM converts the MJIB into the multiple JDS entries under a single TAT and prepare the incoming data for Networking Job/SYSOUT Receive Module (IATNTJS). IATNTJS creates a "utility" job to process the received data. The ownership of the JDS entry is converted from the Netserv job server to the “utility” job.

When an inbound network job is received, the NJERDR DSP acts as the front end to the networking receiver routine (IATNTJS) for SNA NJE processing (the NJE DSP driver (IATNTDR) is the front end for BSC/NJE processing).

JES3 output processing with TCP/IP

JES3 output service queues outgoing TCP NJE data on a special TCP queue. This results in a post of the TCP (TCP/NJE) DSP Driver (IATNTTDR) which builds a TCP NJE transaction and sends the it to the Netserv server. If Netserv is running on local, a JSERV is used to communicate the TCP NJE transaction to the local. On the global or the local, the TCP (TCP/NJE) Server Request module (IATNTTSR) is called to places the TCP NJE transaction on a Netserv's request queue and the communication task, IATNTTCT, in the Netserv address space is posted.

IATNTTCT's functional routine initiates a SAPI request to retrieve outgoing data for the job. To make the spool access faster, instead of using SAPI GET to retrieve the data, IATNTTCT then uses the block spooler (IATDMBS) to read the data.

14.48 JES3/TCP/IP/NJE transmission summary

Originating Node		Execution Node	
JES3	Netserv	JES3	Netserv
NJE Job Submitted Create TCP output Build TCP NJE transaction Queue TCP NJE transaction Post Netserv			
	SAPI Get Transact Transmit Job		Receive Job Create Data set SSISERV MJIB
		MSGCFCT FCT Create Utility Job -ISDRVR - PURGE ISDRVR FCT - Add Job (Job executes - Output for originating node) OSE for originating node IATOSBP - OSEs TCP FCT - Transaction	
	Receive sysout - Create NJE Data set		Get Transact. Transmit sysout
MSGCFCT FCT - IATNTJS Create Utility Job (njesf-outserv-purge) WTR - Prints Data set			

Transmission summary

This figure is a summary of the process of sending a TCP/IP/NJE job from an originating node to execute on the execution node and the SYSOUT coming back to the originating node to print.

14.49 Defining TCP/IP/NJE NETSERV

NETSERV initialization statement

```
NETSERV,NAME=nsvname [,HOSTNAME=hostname]
[,PORT= {0 | port} ] [,STACK=stackname]
[,SYSTEM=sysname] [,ITRACE={YES | NO}]
[,JTRACE={YES | NO}] [,VTRACE={YES | NO}]
```

*F NETSERV,ADD operator command

```
*F NETSERV,ADD=nsvname [,HOSTNAME=hostname]
[,PORT= {0 | port} ] [,STACK=stackname]
[,SYSTEM=sysname] [,ITRACE={YES | NO}]
[,JTRACE={YES | NO}] [,VTRACE={YES | NO}]
```

Defining TCP/IP/NJE NETSERV

The NETSERV initialization statement describes the NETSERV address space. Its characteristics are the address space name, the TCP/IP stack it uses, the system where it runs, and various tracing information.

The **`*F NETSERV,ADD=`** operator command can dynamically add a new TCP/IP/NJE network server.

Netserv initialization statement parameters

Parameters on NETSERV initialization statement or command:

NAME= Specifies a 1-8 character name that uniquely identifies this Netserv and is also used to start the Netserv address space. The name must contain only A-Z, @, $, #, or 0-9 and cannot start with a digit.

HOSTNAME= Specifies the IP host name or IP address that the Netserv uses to listen over TCP/IP for incoming data. This name can be omitted to use the default, which tells TCP/IP to listen over any IP address that is defined **for** this processor. Multiple Netserv address spaces can listen over the same host name or use the default, if different ports are used.

The host name can consist of up to 60 characters and must be a legal TCP/IP address or host name defined to the TCP/IP resolver. The host name can lbe in IPV4 or IPV6 format.

PORT= Specifies a port number that the Netserv will use, in combination with the host name, to listen over TCP/IP for incoming data. The port must be a number from 0 to 65,535. Port 0 indicates that when the Netserv starts a standard service name of VMNET. VMNET will be used instead of a numeric port. The service of VMNET corresponds to port 175. Although there is a corresponding port for the service NJENET-SSL, it is suggested that you use the TLS=YES parameter on the NJERMT statement or a **`*F NJE`** command instead.

SYSTEM= Specifies the name of the system that the Netserv will run on. If the SYSTEM= parameter is omitted, the Netserv will run on the global. After a DSI, an active Netserv that defaulted to the global will remain running on the old global. However, the Netserv will run on the new global if it is subsequently brought down and back up.

STACK= Specifies the name of the stack that TCP/IP uses to get its definitions. If the STACK= parameter is omitted, TCP/IP will use its default stack. The stack must contain only A-Z, @, $, #, or 0-9 and cannot start with a digit except when removing an existing assignment.

ITRACE= Indicates that internal tracing in the Netserv address space will be active.

JTRACE= Indicates that JES tracing will be active. JES tracing occurs during JES specific exit points in the Netserv when data records, headers, and trailers are transmitted and received.

14.50 Netserv address space

- Netserv is a started task
  - *X TCP command to start TCP/IP/NJE Netserv
  - ASCRE macro used to create Netserv addrss space
    - No JCL required
  - Netserv address space
    - IAZNJTCP, common component, uses TCP/IP services
    - Must operate in z/OS UNIX environment
    - Supports multiple stacks, VIPA, Sysplex Distributor
    - TLS/SSL support is available
  - Netserv needs proper authority to function:
    - RACF STARTED class definition with a user ID
    - An OMVS segment for the user ID associated with the started task

Netserv address space

The role of the Netserv address space is to make all the communication interface calls to TCP/IP. Netserv and JES3 cooperate to despool from JES3 data that is to be sent and to spool to JES data that is received. The Netserv functionality is in the JES common component (IAZNJTCP); therefore, with limited operational differences, it is the same in both JES2 and JES3.

The NETSERV address space can run either on the global or a local. There is no limit to the number of NETSERV address spaces that can run in the JES3 complex at a single time. You specify the system on which a NETSERV is to run when you define it to JES3.

Starting a TCP/IP/NJE network server *X TCP

Use the **`*X TCP`** command to start a network server address space (Netserv) for TCP/IP/NJE.

You must use the **`*X TCP`** command to start a Netserv before you can start communication to a node through any sockets that are defined under that Netserv. Following any restart of JES3, other than a cold start, if the Netserv was not canceled before the restart, it is automatically called again. Syntax:

```
*X TCP,NETSERV=ntsvname
```

The Netserv address space is started by using the ASCRE macro. (This is a common JES implementation.) The purpose for this, rather than a START command for a procedure, is so that a Netserv can work with no PROCLIB needed to be customized, although a RACF started task table is still needed.

When the ***X,TCP,NETSERV=netserv** command is issued, the TCP/IP DSP driver IATNTTDR gets control. A TCP DSP is attached and placed on the FCT chain. IATNTTDR calls TCP (TCP/NJE) Address Space Create (IATNTTAC) which will either handle the start on the global, if that is where the server is to run, or else JSERV a staging area to the correct system. The TCP (TCP/NJE) Address Space Initialization module (IATNTTAS) gets control in the newly created address space and performs all initialization activities before giving control to the common program, IAZNJTCP.

JES3 job name for the Netserv address space is IEESYSAS, MVS address space name for the Netserv is *ntsvname*.

z/OS Communications Server

The TCP/IP/NJE implementation supports virtual IP addresses (VIPA addresses), and the use of sysplex distributor to share IP addresses throughout a sysplex. There is no additional JES3 setup to take advantage of either of these features other than to specify the appropriate IP address or host name on the SOCKET initialization statement.

TLS/SSL support

Use of SSL and TLS by NJE/TCP is through Application Transparent TLS (AT-TLS). All of the SSL/TLS definitions are defined in the TCP/IP and security profiles, rather than in JES3 parameter statements. The TLS= parameter on the NJERMT initialization statement indicates whether the Transport Layer Secure facility will be used by any socket that is used to communicate with this node.

Netserv security product STARTED class definition

The NETSERV address space must be defined to the security product STARTED class. To minimize the number of STARTED profiles, it is suggested that you define all of your NETSERVs with a common name pattern so that you can cover them all with one generic profile.

```
RDEFINE STARTED JES3*.* STDATA( USER(JES3ID) GROUP(SYS1) TRUSTED(YES))
```

Security product user profile OMVS segment

The Netserv address space common component uses TCP/IP services and requires the z/OS UNIX System Services environment to operate. An OMVS segment in the security definition for the user ID associated with the started task.

```
ALTUSER JES3ID OMVS(UID(0) HOME('/') PROGRAM('/'))
```

14.51 NJERMT statement (node)

NJERMT initialization statement

```
NJERMT,{NAME=rmtnm | ALIAS =aliasnm} [,DEFCLASS={YES | NO}]
[,EXPWD=recpassword1] [,NETHOLD= {YES | NO}]
[,PWD=sendpassword1] [,RETRYCT={20 | nnn} [,SPART=spart]
[,PWCNTLE={SENDCLR | SENDENC |  LOCALCHK}]
[,SECSIGNON={YES | NO}] [,TLS={YES | NO}] ,TYPE=TCPIP
[,JOBTRANS=jt] [,OUTTRANS=ot] [,JOBRECV=jr] [,OUTRECV=or]
```

*F NJE operator command

```
*F NJE,ADD=rmtnm,TYPE=TCPIP

*F NJE,N=rmtnm,TYPE=TCPIP [,JT=n] [,JR=n] [,OT=n] [,OR=n]
[,TLS={YES | NO}] [,,SP=spart] [,SS={YES | NO}]
[,PWCNTL={ SENDCLR | SENDENC |  LOCALCHK}]
```

NJERMT statement

The NJERMT statement has some new parameters that define a TCP/IP connection. The TYPE=TCPIP parameter defines the protocol as TCP/IP. The JOBTRANS, OUTTRANS, JOBRECV, and OUTRECV define the number of transmitters and receivers. Each of these can be from 1 to 7. According to the NJE protocol, the sum of the transmitters can be a maximum of 8, and likewise for the sum of receivers.

The TYPE=TCPIP parameter defines the node as an NJE over TCP/IP node. Following parameters on the NJERMT statement define:

- JOBTRANS, OUTTRANS, JOBRECV, and OUTRECV define the number of transmitters and receivers for the node. Each of these is a number between 1 to 7. According to the NJE protocol, the sum of the transmitters can be a maximum of 8, and likewise for the sum of receivers.
- SECSIGNON=YES defines that nodes signing on to each other verify their identity using session keys defined to the security product's APPCLU class. JES3 allows the SECSIGNON=YES only for TYPE=TCPIP nodes.
- TLS=YES indicates that the TCP/IP Transport Layer Secure (TLS) facility will be used to encrypt transmissions.

*F NJE command

The `*F NJE` command can dynamically add a directly-connected node, delete a node or alias, change the type of networking protocol you are using between your node and a remote node, and change the usage of secure signon protocol and the Transport Layer Secure facility.

14.52 Define TCP/IP SOCKET

SOCKET initialization statement

SOCKET,NAME=sockname [,NETSERV=ntsvname]
[,NODE=nodename] [,HOSTNAME=hostname]
[,PORT=port] [,ITRACE={NO | YES}] [,JTRACE={NO | YES}]
[,VTRACE= {NO | YES}]

***F SOCKET,ADD operator command**

*F SOCKET,ADD=sockname
[,HOSTNAME=hostname] [,PORT= {0 | nnnnn}]
[,NETSERV=ntsvname] [,NODE=nodename]
[,ITRACE={NO | YES}] [,JTRACE={NO | YES}
[,VTRACE={NO | YES}]

***START,TCP,SOCKET=sockname command**

Starts a socket under a Netserv

SOCKET initialization statement

A socket defines the IP address or host name of a TCP/IP/NJE remote node, and a port. In order for two nodes to communicate using the TCP/IP protocol, a socket must be started on one node. This node is referred to as a client. The corresponding node at the other end of communication is referred to as a server. If JES3 is a server node, a server socket will be defined automatically. If JES3 is a client node, a socket must be define using the SOCKET initialization statement or JES3 commands.

NAME= Name is a 1-8 character name that uniquely identifies this socket. The name must contain only A-Z, @, $, #, or 0-9 and cannot start with a digit.

HOSTNAME= The hostname specifies the IP host name or IP address that this socket will use to communicate with TCP/IP.

The host name can consist of up to 60 characters and must be a legal TCP/IP address or host name defined to the TCP/IP resolver. The host name can be in IPV4 or IPV6 format.

PORT= Specifies a port number that the socket will use, in combination with the host name, to communicate with TCP/IP. The port must be a number from 0 to 65,535. A port of 0 indicates that when the socket starts a standard service name of VMNET, VMNET will be used instead of a numeric port. The service of VMNET corresponds to port 175. Although there is a corresponding port for the service NJENET-SSL, it is recommended that you use the TLS=YES parameter on the NJERMT statement or **`*F NJE`** command instead.

NETSERV= Specifies the name of the Netserv on which the socket will be activated. This parameter does not need to be specified during initialization, but if the parameter is omitted, it must be modified later. Otherwise, the socket cannot be started.

NODE= Specifies the name of the node that will use this socket. This parameter does not need to be specified during initialization, but if the parameter is omitted, it must be modified later. Otherwise, the socket cannot be started.

ITRACE= Indicates that internal tracing in the socket address space will be active.

JTRACE= Indicates that JES3 tracing will be active. JES3 tracing occurs during JES3 specific exit points at the socket level in the Netserv address space during which data records, headers, and trailers are transmitted and received.

VTRACE= Indicates that verbose tracing will be active at the socket level. Verbose tracing involves various diagnostic messages being written to the console and job log from the Netserv address space.

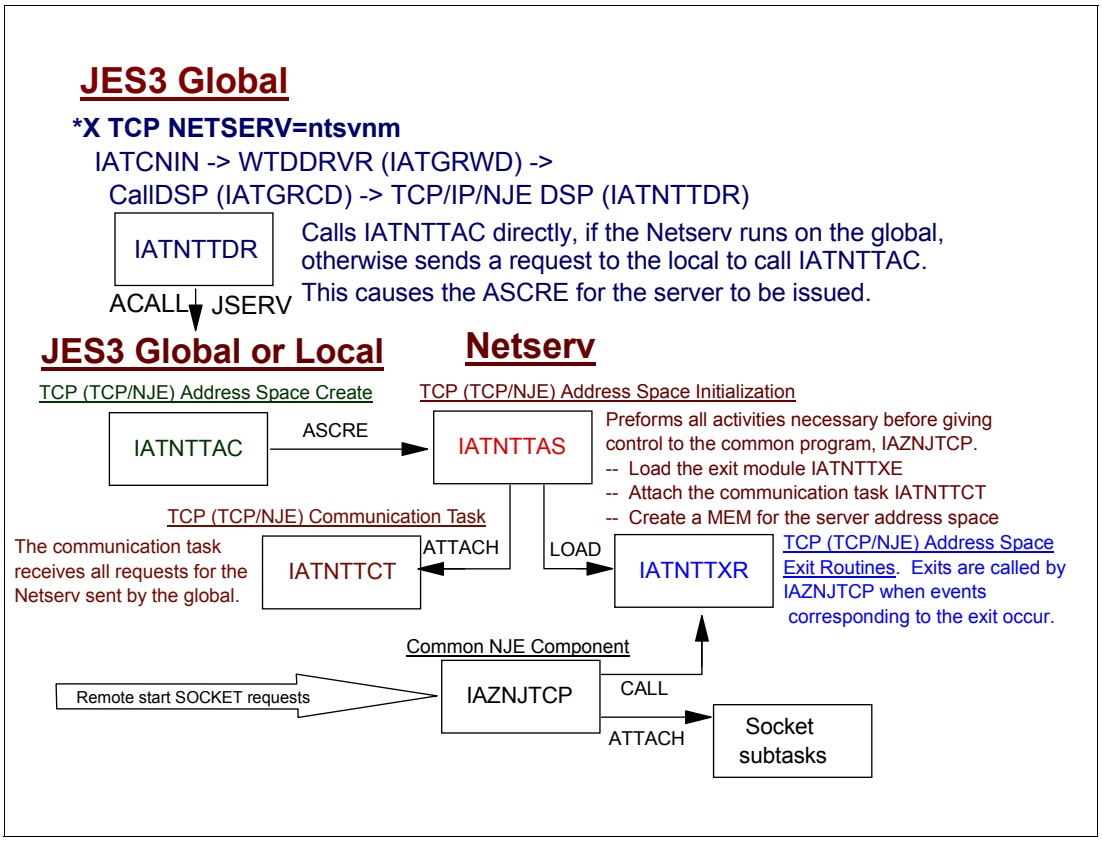
*F SOCKET,ADD operator command

The **`*F SOCKET,ADD=`** command to dynamically adds a new TCP/IP/NJE socket connection that is used to communicate with a remote NJE node using the TCP/IP protocol.

*S TCP,SOCKET=sockname operator command

The **`*S TCP,SOCKET=`** command starts a socket under a Netserv, in order to begin communication with a remote node with TCP/IP protocol.

14.53 *X TCP - How It's Done



*X TCP - how it's done

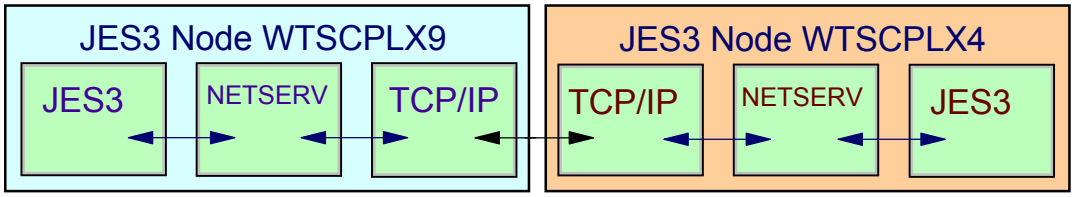
The visual shows JES3 processing steps and modules for the **`*X TCP`** command.

*S TCP SOCKET - how it's done

Processing steps by the load modules:

- IATCNIN - Validate **`*S TCP SOCKET=`** command
- TCP/IP NJE DSP console appendix (in IATNTTDT Data CSECT) - Save the START command text and post the TCP FCT ECF for 'START' and 'CANCEL' commands.
- IATNTTDR - Validate *S TCP SOCKET= request, call TCP (TCP/NJE) Server Request (IATNTTSR) to queue the start socket request and post the communication task (IATNTTCT) in the Netserv address space.
- IATNTTCT - Receives the requests for the Netserv sent by the global. IATNTTCT determines which function is needed in order to send the request on to IAZNJTCP and calls TCP Server/Socket Services (IATNTTSS).
- IATNTTSS - Posts the appropriate IAZNJTCP work ECB to tell IAZNJTCP there is incoming work.

14.54 TCP/IP/NJE example environment



JES3 node definitions

```
NETSERV,NAME=J3TCPIP,HOSTNAME=WTSC43.IBM.COM,PORT=1234
NJERMT,NAME=WTSCPLX9,HOME=YES
NJERMT,NAME=WTSCPLX4,TYPE=TCPIP
SOCKET,NAME=WTSCPLX4,HOSTNAME=WTSC65.IBM.COM,PORT=1234,
NETSERV=J3TCPIP,NODE=WTSCPLX4

NETSERV,NAME=J3TCPIP,HOSTNAME=WTSC65.IBM.COM,PORT=1234
NJERMT,NAME=WTSCPLX4,HOME=YES
NJERMT,NAME=WTSCPLX9,TYPE=TCPIP
```

A socket statement for WTSCPLX4 to WTSCPLX9 is optional. One socket statement is necessary.
A “server” socket @0000001 is dynamically created on WTSCPLX4 when a *S TCP SOCKET=WTSCPLX4 command is issued on WTSCPLX9. If more sockets are needed, the next socket is @0000002, if @0000001 is still active, then @0000003 and so forth.

JES3 node definitions

The visual shows a scenario where two nodes, one in WTSCPLX9 and one inWTSCPLC4, wish to communicate using NJE over TCP/IP. Each node defines itself as HOME=YES and the remote node as TYPE=TCPIP. Each node also defines a NETSERV address space. Note that both nodes do not specify the SYSTEM parameter on the NETSERV statement, this defaults to the global processor. The WTSCPLX9 node also defines a SOCKET for the WTSCPLX4 remote node, and a port.

The HOSTNAME and PORT parameters on the NETSERV statement are optional. The default is that the NETSERV listens to TCP/IP over any host and port that it can for the node. The HOSTNAME and PORT parameters shown illustrates the relationship between the NETSERV definition on one node and the SOCKET definition on the other.

DSI and Netserv address space

There is an optional parameter, SYSTEM, on the NETSERV definition that is not shown here. It defines which system the NETSERV address space is to be active. The default is the JES3 global. If a DSI is done, the default moves to the new global, but only when the Netserv is inactive. If a Netserv defined to default to the global is active while a DSI is done, it will remain up where it is, but if it is brought down and back up, it will come up on the new global.

14.55 Example - on WTSCPLX9 start TCP/IP/NJE

```
*I NETSERV=J3TCPIP
IAT8707 NETSERV INQUIRY RESPONSE 791
INFORMATION FOR NETSERV J3TCPIP
  SYSTEM=, HOST=WTSC43.IBM.COM, PORT= 1234, STACK=TCPIP,
  JTRACE=NO, VTRACE=NO, ITRACE=NO, ACTIVE=NO
  SOCKETS DEFINED IN THIS NETSERV
  SOCKET   ACTIVE   NODE     SERVER
  WTSCPLX4 NO       WTSCPLX4 NO
END OF NETSERV INQUIRY RESPONSE
*X TCP NETSERV=J3TCPIP
IAT6306 JOB (JOB44000) IS TCP     , CALLED BY VAINI
IRR812I PROFILE J3*.* (G) IN THE STARTED CLASS WAS USED 822
       TO START J3TCPIP WITH JOBNAME J3TCPIP.
IAT6100 ( DEMSEL ) JOB IEESYSAS (JOB44001), PRTY=15, ID=STC
ICH70001I STC      LAST ACCESS AT 08:48:16 ON THURSDAY, MAY 29, 2008
IAT9301 TCP START SUCCESSFUL FOR SERVER J3TCPIP
IAZ0542I J3TCPIP IAZNJTCP for HBB7740 compiled Nov 16 2007 07:35:48
IAZ0537I J3TCPIP NJETCP SERVER WAITING FOR WORK
*S TCP NODE=WTSCPLX4
IAT7100 (CONCMD  ) *S,TCP,SOCKET=WTSCPLX4
*S,TCP,SOCKET=WTSCPLX4
IAZ0543I J3TCPIP TCP/IP connection with IP Addr: WTSC65.IBM.COM Port: 1234 Initiated
IAZ0543I J3TCPIP TCP/IP connection with IP Addr: WTSC65.IBM.COM Port: 1234 Successful
IAT9305 NODE WTSCPLX4 SIGNED ON  NETSERV J3TCPIP  SOCKET WTSCPLX4
IAZ0544I J3TCPIP WTSCPLX4 NJE connection with IP Addr: wtsc65.ibm.com Port: 1234 Successful
```

Example - on WTSCPLX9 start TCP/IP/NJE

You must use the **`*X TCP`** command to start a Netserv address space before you can start communication to a node through any sockets that are defined under that Netserv. Following any restart of JES3, other than a cold start, if the Netserv was not canceled before the restart, it is automatically called again.

The sockets that are defined for the requested node need to be started by an internally generated **`*S TCP,SOCKET=`** command.

`*I NETSERV=netserv`
 	
`*X,TCP,NETSERV=netserv`
	Displaying a TCP/IP/NJE Network Server information.
	Starts the NETSERV address space. All the sockets under this NETSERV are also activated.

`*S,TCP,NODE=nodename`
 	
`*S,TCP,SOCKET=socket`
	Starts communication with all sockets assigned to the specified node.
	Starts communication with the specified socket.

Note: JES3 converts internally **`*S,TCP,NODE=nodename`** to **`*S,TCP,SOCKET=sockname`** commands. The *sockname* is the name of the socket that will used for this node.

14.56 Example - on WTSCPLX9 convert networking protocol

```
*I NJE N=NOGO
IAT8659 - NODE NOGO     NOHOLD PATH NOGO     SNA PROTOCOL
*F NJE N=NOGO TYPE=TCPIP
IAT8460 NJERMT UPDATE COMPLETE. REQUEST HONORED.
*I NJE N=NOGO
IAT8711 NODE INQUIRY RESPONSE 883
INFORMATION FOR NODE NOGO
  TYPE=TCPIP, JT=1, JR=1, OT=1, OR=1, SS=NO, TLS=NO, ACTIVE=NO,
  PWCNTL=SENDCLR
  SOCKETS DEFINED FOR THIS NODE
  NONE
END OF NODE INQUIRY RESPONSE
*F SOCKET ADD=NOGO HOSTNAME=WTSC70.IBM.COM PORT=2345 NODE=NOGO NETSERV=J3TCPIP
IAT8160 ADD    COMPLETE FOR SOCKET NOGO
*S TCP SOCKET=NOGO
IAZ0543I J3TCPIP TCP/IP connection with IP Addr: WTSC70.IBM.COM Port: 2345 Initiated
IAZ0545I J3TCPIP Error encountered in function connect() - EDC8128I Connection refused.
IAZ0543I J3TCPIP TCP/IP connection with IP Addr: WTSC70.IBM.COM Port: 2345 ended due to TCP/IP error, rc: 1128
IAZ0528I J3TCPIP IP address of remote peer could not be resolved, TCP/IP rc: 1128
*I NJE N=WTSCPLX9
IAT8659 - HOME WTSCPLX9 PRTDEF A PRTTSO A PRTXWTR A PUNDEF B
```

Convert networking protocol

It is possible to convert nodes from one networking protocol to the other. In other words, nodes that are currently using BSC, SNA, or TCP/IP protocol can be converted to one of the others. The following scenarios describe converting an existing SNA node to TCP/IP.

Before converting to TCP/IP, you must:

- Define a Netserv using the NETSERV statement or the *F,NETSERV,ADD command.
- Define profiles for the Netserv in the STARTED security class.
- Define a socket describing the IP address or host name of the remote node, and a port.
- Define any IP addresses and host names that nodes use to TCP/IP.
- You can dynamically add nodes; restarting JES3 is necessary only to make the additions permanent when the nodes are defined in the JES3 initialization stream.

Before you can establish any TCP/IP network connections, z/OS Communications Server TCP/IP must be active, which in turn requires z/OS UNIX System Services and ACF/VTAM to be active. A TCP DSP can be started by issuing *CALL,TCP. This can start a Netserv address space. If the TCP DSP is started by issuing *CALL,TCP before TCP/IP is active, the Netserv address space waits until TCP/IP is active. If the remote node is JES2 or JES3, you must start the appropriate Netserv there as well. The sockets can then be activated from either side. The operator can migrate any jobs that remain waiting for BDT to despool the output by using the NJEROUT DSP. Jobs are transmitted when the socket is activated.

The visual shows the messages issued during the conversion of an existing SNA node to TCP/IP.

14.57 Example - on WTSCPLX9 reroute network jobs

```
*I NJE N=WTSCPLX9
IAT8659 - HOME WTSCPLX9 PRTDEF A PRTTSO A PRTXWTR A PUNDEF B

IRR010I USERID VAINI IS ASSIGNED TO THIS JOB.
IAT6100 (JOB06809) JOB VAINIXMIT (JOB44003), PRTY=01, ID=VAINI NET-ID=*NONE SUB=JOB43866
IAT9370 JOB VAINIXMIT (JOB44003) GROUPID (TCP00000) TRANSMISSION TO NODE WTSCPLX4 SUCCESSFUL
IAT7450 JOB VAINIXMIT (JOB44003) PURGED

IRR010I USERID VAINI IS ASSIGNED TO THIS JOB.
IAT6100 (JOB06809) JOB VAINIXMIT (JOB44004), PRTY=01, ID=VAINI NET-ID=*NONE SUB=JOB43866
*IAT9352 NO SOCKET ACTIVE TO NODE NOGO FOR TCP NJE TRANSACTION
*I U Q=TCP J=? N=ALL
IAT8131 JOB VAINIXMIT (JOB44004), D=NOGO, L=10, PG=0, SR=10, BY=0,
IAT8131 JOB VAINIXMIT (JOB44004), TG=TCP00000.
IAT8119 NUMBER OF JOBS FOUND : 1
IAT8141 NUMBER OF TCP JOBS FOUND : 1
*X NJEROUT TO=WTSCPLX4 JG=(44004,TCP00000)
IAT6306 JOB (JOB44005) IS NJEROUT , CALLED BY VAINI
IAT9200 NJEROUT (RRT44005) IS ACTIVE
IAT9201 JOB VAINIXMIT (JOB44004) GROUPID (TCP00000) REROUTED TO (WTSCPLX4), NEW JOB (JOB44006)
IAT9219 NJEROUT (RRT44005) WAITING FOR START/RESTART/CANCEL REQUEST
IAT7450 JOB VAINIXMIT (JOB44004) PURGED
IAT9370 JOB VAINIXMIT (JOB44006) GROUPID (TCP00000) TRANSMISSION TO NODE WTSCPLX4 SUCCESSFUL
IAT7450 JOB VAINIXMIT (JOB44006) PURGED
SE 'WTSCPLX4: IAT9140 JOB VAINIXMIT (JOB22212) ADDED TO JES3 JOB QUEUE',USER=(VAINI),LOGON
*C J=0044005
IAT8082 JOB NJEROUT (JOB44005) CANCEL ACCEPTED (PRINT=NO )
```

Example - On WTSCPLX9 reroute network jobs

The NJE reroute DSP (NJEROUT) reroutes a network job to a remote location or back to its original destination. You can also reroute the output to a different VM userid or RJP workstation. The NJEROUT DSP is valid for rerouting work to both TCP/IP/NJE, BSC/NJE and SNA/NJE nodes.

Rerouting NJE work

The NJEROUT DSP allows you to reroute:

- A job that was sent to your node from a remote node
- A job that was submitted locally
- SYSOUT streams that are being routed through your node
- SYSOUT streams (from JES3 jobs run at your node) that were originally destined for a remote node

Job and SYSOUT streams, however, cannot be rerouted *from* the home node using this facility.

Job and SYSOUT streams

The networking protocol that a node is using does not affect the rerouting of job or SYSOUT streams. Job or SYSOUT streams that are currently destined to a node that uses one networking protocol of BSC, SNA or TCP/IP can be rerouted to another node using the same

protocol or a node using a different protocol. Rerouting is especially useful if you are converting from one protocol to the other, because some jobs might have been queued for one resource or the other during the conversion period.

To begin the rerouting process, the operator must ***X** the NJEROUT DSP. The ***S** and ***R** commands can then be used to reroute specific jobs.

Message in the visual

The visual shows the messages issued for job transmission to execution node and a rerouting operation:

- JES3 message IAT9352 informs of a failed NJE transmission
- ***I U Q=TCP J=? N=ALL** output service command with the appropriate parameters displays information about TCP/IP/NJE network jobs. TCP/IP/NJE network jobs might be either network job streams or network SYSOUT streams.
- Message IAT8131 identifies jobs found in output service Q=TCP
- The ***X NJEROUT TO=node JG=(jes3_job_nbr,tcp_group_nbr)** command requests the specified job's TCP output data to be rerouted

JCL for the example job

The following JCL was used to create the messages shown in the visual.

```
//VAINIXMT JOB (999,POK),EXPERT,MSGLEVEL=1,MSGCLASS=A,
//         NOTIFY=VAINI,REGION=0M
//XMIT    XMIT DEST=NOGO,DLM=$$
//VAINIEX  JOB (999,POK),EXPERT,MSGLEVEL=1,MSGCLASS=A,
//         NOTIFY=WTSCPLX9.VAINI,REGION=0M
//OPUT    OUTPUT JESDS=ALL,DEST=WTSCPLX9
//TMPROC  PROC
//TMP     EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSTSPRT DD SYSOUT=*,DEST=(WTSCPLX9,VAINI)
//        PEND
//EXEC    EXEC TMPROC
//SYSTSIN  DD *
  LISTC LEVEL(VAINI)
$$
```

14.58 Example - WTSCPLX4 messages

```
*I NJE N=WTSCPLX4
IAT8659 - HOME  WTSCPLX4  PRTDEF A PRTTSO A PRTXWTR A PUNDEF B
IAT8707 NETSERV INQUIRY RESPONSE
IAZ0543I J3TCPIP TCP/IP connection with IP Addr: wtsc43.ibm.com Port: 1030 Successful
IAT8160 ADD    COMPLETE FOR SOCKET @0000001 (INTERNAL)
IAT9305 NODE WTSCPLX9 SIGNED ON  NETSERV J3TCPIP  SOCKET @0000001
IAZ0544I J3TCPIP  NJE connection with IP Addr: wtsc43.ibm.com Port: 1030 Successful
*I SOCKET=@0000001
IAT8709 SOCKET INQUIRY RESPONSE 937
INFORMATION FOR SOCKET @0000001
  NETSERV=J3TCPIP, HOST=, PORT=    0, NODE=WTSCPLX9, JTRACE=NO,
  VTRACE=NO, ITRACE=NO, ACTIVE=YES, SERVER=YES
END OF SOCKET INQUIRY RESPONSE
*I NETSERV=J3TCPIP
INFORMATION FOR NETSERV J3TCPIP
  SYSTEM=SC65, HOST=WTSC65.IBM.COM, PORT= 1234, STACK=TCPIP,
  JTRACE=NO, VTRACE=NO, ITRACE=NO, ACTIVE=YES
  SOCKETS DEFINED IN THIS NETSERV
  SOCKET   ACTIVE   NODE     SERVER
  @0000001 YES      WTSCPLX9 YES
END OF NETSERV INQUIRY RESPONSE
Rerouted job is received and executed
IAT9127 JOB (JOB22212) IS VAINIXMT FROM WTSCPLX9(VAINI   )
IAT6100 (JOB22212) JOB VAINIEX  (JOB22213), PRTY=01, ID=VAINI
IAT7450 JOB VAINIXMT (JOB22212) PURGED
IAT9370 JOB VAINIEX  (JOB22213) GROUPID (TCP00000) TRANSMISSION TO NODE WTSCPLX9 SUCCESSFUL
IAT7450 JOB VAINIEX  (JOB22213) PURGED
```

Example - WTSCPLX4 messages

The visual shows the messages issued on node WTSCPLX4 during the execution of the example operations.

When WTSCPLX9 connects to WTSCPLX4, a “server” socket @0000001 is created. The **`*I,SOCKET=@0000001`** display shows that the socket is assigned to node WTSCPLX9.

When a job was submitted on WTSCPLX4 to execute on WTSCPLX9, the following messages were issued on WTSCPLX4:

```
IRR010I  USERID VAINI    IS ASSIGNED TO THIS JOB.
IAT6100 (JOB19166) JOB VAINITS  (JOB22217), PRTY=01, ID=VAINI NET-ID=*NONE SUB=JOB22069
IAT9370 JOB VAINITS  (JOB22217) GROUPID (TCP00000) TRANSMISSION TO NODE WTSCPLX9
SUCCESSFUL
SE 'WTSCPLX9: IAT9140 JOB VAINITS  (JOB44011) ADDED TO JES3 JOB
QUEUE',USER=(VAINI),LOGON
IAT7450 JOB VAINITS  (JOB22217) PURGED
SE 'WTSCPLX9: IAT6108 JOB VAINIEX  (JOB...ID) ENDED, MAXCC=0000
',USER=(VAINI),LOGON
IAT9127 JOB (JOB22217) IS VAINITS  FROM WTSCPLX4(VAINI   )
SE 'WTSCPLX4: IAT9141 JOB VAINITS  (JOB22217) ADDED TO JES3 OUTPUT
QUEUE',USER=(VAINI),LOGON
```

14.59 Summary of JES3 TCP/IP/NJE commands

*I D D=netserv	Display SUPUNIT entry for a Network Server
*INQUIRY,NETSERV=	Display a TCP/IP/NJE Network Server
*MODIFY,NETSERV=	Modify a TCP/IP/NJE Network Server
*MODIFY,NETSERV,ADD=	Add a TCP/IP/NJE Network Server
*MODIFY,NETSERV,DELETE=	Delete a TCP/IP/NJE Network Server
*CALL,TCP,NETSERV=	Start a TCP/IP/NJE Network Server
*INQUIRY,SOCKET=	Display a TCP/IP/NJE Socket
*MODIFY,SOCKET=	Modify a TCP/IP/NJE Socket
*MODIFY,SOCKET,ADD=	Add a TCP/IP/NJE Socket
*MODIFY,SOCKET,DELETE=	Deleting a TCP/IP/NJE Socket
*START,TCP,SOCKET=	Start a TCP/IP/NJE Socket
*START,TCP,NODE=	Start all sockets to a node
*CANCEL,TCP,SOCKET=	Stop a TCP/IP/NJE Socket
*INQUIRY,NJE,N=	Display the status of nodes
*MODIFY,NJE,N=	Change nodes and network settings
*MODIFY,NJE,ADD=	Add a node
*MODIFY,NJE,DEL=	Delete a node
*CALL,NJECONS	Start networking console support
*CANCEL,NJECONS	Stop networking console support

Examples of JES3 NJE commands

Use the `*I SOCKET=` command to display information about a socket or all sockets.

The `*I NETSERV=ALL` displays information about all Netserv address spaces. Up to 999 address spaces may be defined.

The existing `*I NJE` command can be used to display TCP/IP/NJE nodes.

14.60 Secure signon

- SECSIGNON=YES/NO parameter on NJERMT statement
- Uses profiles in RACF **APPCLU** class for the encryption key
  - RACF profile name should be:
    - **NJE.homenode.remotenode**
  - A node participating in secure signon supplies a random string to its NJE partner
  - The other node encrypts the string and sends the result back to the first node
  - At the same time, it supplies its own random string
    - First node encrypts that string and sends the result back
    - If both nodes have the same encryption key defined to the RACF APPCLU class, the encryptions yields the same result and the nodes are allowed to sign on

Secure signon

With TCP/IP, there is a greater potential for a hacker to pretend to be an NJE over TCP/IP node than with BSC or SNA nodes. It is much harder for someone to hack into BSC or SNA and pretend to be an NJE node. Generally, this can happen if hardware equipment is connected the wrong way. To cope with this potential risk, JES3 provides a mechanisms to secure the NJE over TCP/IP communications. These are secure signon and the transport layer secure (TLS) protocol.

Secure signon involves an encryption key, which is on a node by node basis (and only directly connected nodes). The encryption key is stored in the RACF APPCLU class as a profile which the JES code can extract.

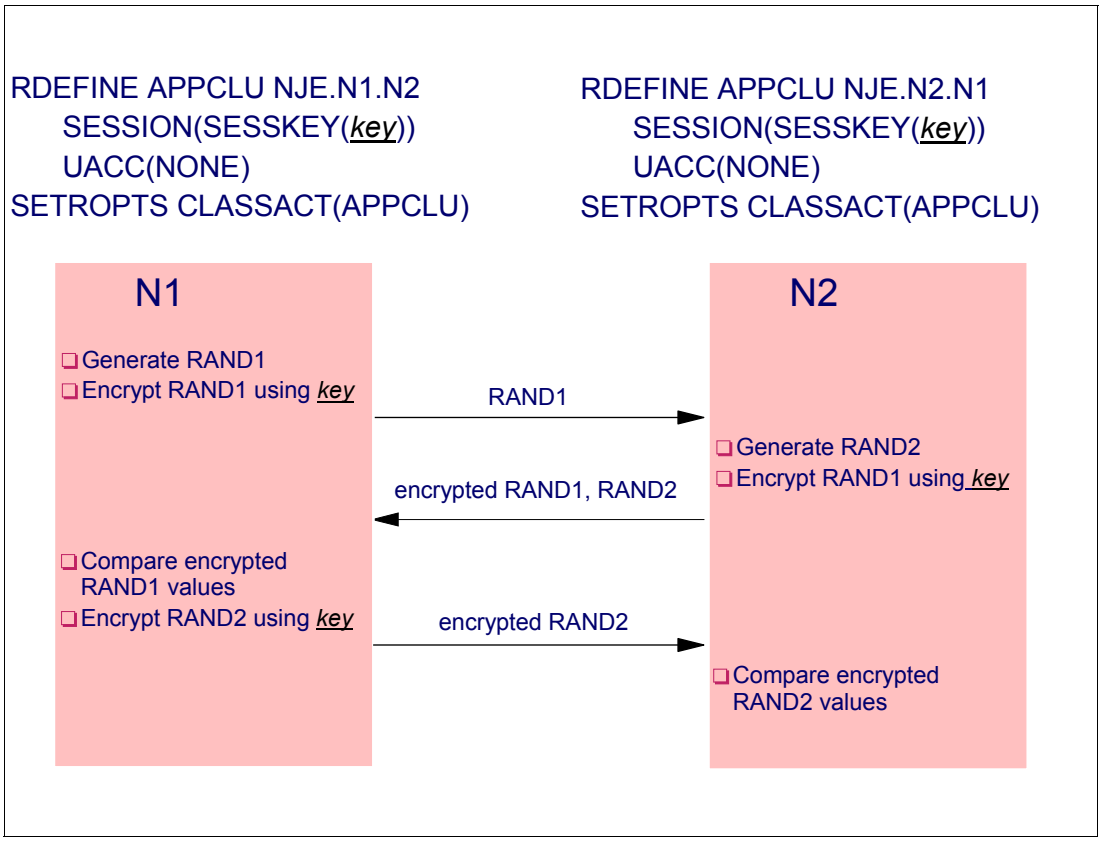
NJERMT statement

Secure signon is selected by using the SECSIGNON=YES/NO parameter on the NJERMT statement.

The RACF profile name should be **`NJE.homenode.remotenode`**. A node participating in secure signon supplies a random string to its NJE partner. The other node encrypts the string and sends the result back to the first node. At the same time, it supplies its own random string.

The first node must then encrypt that string and send the result back. If both nodes have the same encryption key defined to the RACF APPCLU class, the encryptions will yield the same result and the nodes are allowed to sign on. Otherwise the signon fails. This is a means for two nodes to authenticate one another.

14.61 Secure signon



Secure signon

Secure signon is an optional security mechanism to ensure that two nodes prove their identities to each other during NJE signon. Secure signon is provided only in JES2 and JES3.

Secure signon has nothing to do with Transport Layer Secure (TLS) in TCP/IP. TLS is a separate security feature.

Secure signon uses the APPCLU security class to provide an encryption key. A node participating in secure signon supplies a random string to its NJE partner. The other node encrypts the string and sends the result back to the first node. At the same time, it supplies its own random string. The first node must then encrypt that string and send the result back.

If both nodes have the same encryption key defined to the APPCLU class, the encryptions will yield the same result and the nodes are allowed to sign on. Otherwise the secure signon fails.

Note: The APPCLU security class must be active on both nodes.

The SESSION parameter in the APPCLU security class on both nodes must specify a session key and both session keys must be the same. The name of the APPCLU security profile is NJE.homenode.rmtnode.

14.62 Coexistence considerations

- User Exit points
  - same exit points for TCP/IP as for BSC and SNA
- NJERMT, NETSERV and SOCKET definitions
  - definitions are checkpointed
- For a fallback to a lower level release of JES3:
  - (IPL the global on a lower level release or perform a DSI to a lower level local processor)
    - Any NJERMT definition previously made with TYPE=TCPIP appears to be TYPE=BSC
    - The node is not usable
    - The SOCKET and NETSERV statements are ignored
    - When you return to JES3 V1R8 or later the definitions are reinstated.

User exits

JES3 job networking user exits:

- IATUX35 (Validity Check Network Commands)
- IATUX36 (Collect Accounting Information)
- IATUX37 (Modify the JES3 Networking Data Set Header for Local Execution)
- IATUX37 - Changed installation exit for TCP/IP.
- IATUX38 (Change the SYSOUT Class and Destination for Networking Data Sets)
- IATUX39 (Modify the Data Set Header for a SYSOUT Data Set)
- IATUX40 (Modify Job Header for a Network Stream Containing a Job)
- IATUX42 (TSO Interactive Data Transmission Facility Screening and Notification)
- IATUX43 (Modify Job Header for a Network Stream Containing SYSOUT Data)
- IATUX50 (JES3/BDT Unknown BSID Modifier Exit)
- IATUX58 (Modify Security Information Before JES3 Security Processing)
- IATUX59 (Modify Security Information After JES3 Security Processing)
- IATUX66 (Determine Transmission Priority for a SNA/NJE Stream)
- IATUX67 (Determine Action When Remote Data Set Is Rejected by RACF)
- IATUX68 (Modify Local NJE Job Trailers)

Coexistence considerations

The IATXNTS macro is used to search the networking node table to verify the validity of a destination. Since JES3 z/OS V1R8 adds a new TCP/IP type of node, a new keyword is required for IATXNTS to support TYPE=TCPIP nodes. The new keyword is TYPETCP. If you have a user modification or installation exit that calls IATXNTS and uses the TYPEBSC and the TYPESNA keywords, you must also specify the TYPETCP keyword. The three keywords must all be specified or all omitted.

This change must be made before starting JES3 level HJS7730. The change is required even if you do not plan to make use of NJE over TCP/IP support. If you do not have any intention of using the NJE over TCP/IP support yet, or if you do not know what TCP/IP action to take for a node, you can specify the same label on TYPETCP as you do for TYPESNA.

Falling back to a previous release

When you use the new NJERMT, NETSERV and SOCKET definitions on JES3 level HJS7730, the new definitions are saved on the spool. After falling back to a lower level release of JES3 (IPL the global on a lower level release or perform a DSI to a lower level local processor), any NJERMT definition previously made with TYPE=TCPIP appears to be TYPE=BSC. The node is not usable. The SOCKET and NETSERV statements are ignored. When you return to JES3 release HJS7730, the definitions are reinstated.

14.63 Coexistence messages

During fallback to a lower release, the following messages are issued:

```
IAT4131 SPOOL RECORD ERROR DETECTED (JCT ) FOR JOB TCP (JOBxxxxx)
IAT4174 CONFIRM DELETION OF JOB TCP (JOBxxxxx) DUE TO SPOOL RECORD
ERROR(S) (CONTINUE, SNAP(,ALL) OR TERMINATE)
```

When returning to a JES3 V1R8, the following messages are issued:

```
IAT4159 ERROR RESTORING TCP/IP CHECKPOINT
IAT4103 ERROR(S) ENCOUNTERED RECOVERING JES3 STATUS CHECKPOINT
DATA (CONTINUE OR CANCEL)
```

When you enter the *X DSI command on the global, if any NETSERV address spaces are active, the following message are issued:

```
IAT0921 DSI - WARNING: SYSTEM sysname HAS nnnnn ACTIVE NETSERVS
```

Coexistence messages

During fallback to a lower release, the following messages will be issued:

```
IAT4131 SPOOL RECORD ERROR DETECTED (JCT) FOR JOB TCP (JOBxxxxx)
IAT4174 CONFIRM DELETION OF JOB TCP (JOBxxxxx) DUE TO SPOOL RECORD ERROR(S)
(CONTINUE, SNAP(,ALL) OR TERMINATE)
```

When returning to a JES3 level HJS7730, the following messages will be issued:

```
IAT4159 ERROR RESTORING TCP/IP CHECKPOINT
IAT4103 ERROR(S) ENCOUNTERED RECOVERING JES3 STATUS CHECKPOINT DATA (CONTINUE
OR CANCEL)
```

When you enter the ***X DSI** command on the global, if any NETSERV address spaces are active, the following message will be issued:

```
IAT0921 DSI - WARNING: SYSTEM sysname HAS nnnnn ACTIVE NETSERVS
```

15

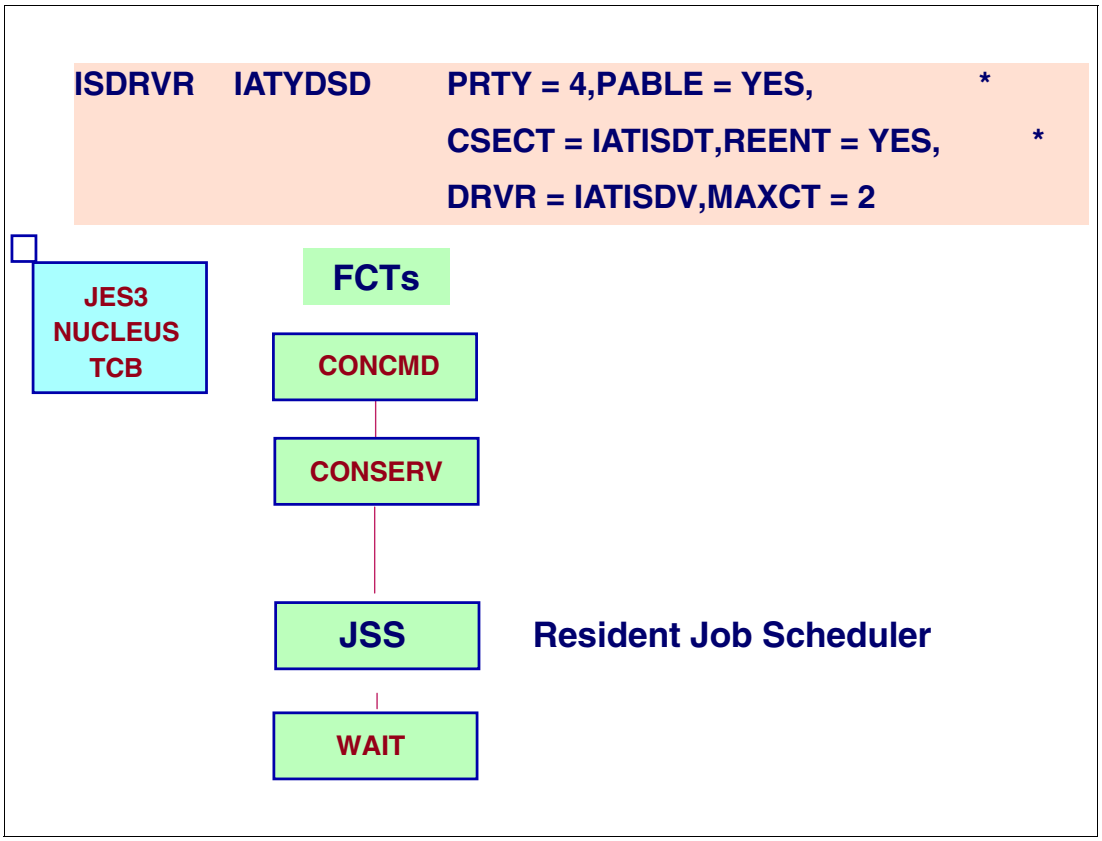
JES3 dynamic support programs

JES3 is composed of routines called dynamic support programs, (DSPs). With the same mechanisms it uses for processing jobs, JES3 schedules and processes DSPs. JES3 users can write their own dynamic support programs to customize JES3 processing in their system.

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

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

15.1 Scheduling and DSP dictionary



Scheduling DSPs

Each small piece of work that JES3 performs when processing a job is accomplished with a JES3 program called a dynamic support program, or DSP. Each DSP is represented on the FCT chain by one or more FCT entries or elements. The elements on the FCT chain are executed according to their priority, and are placed on the FCT chain with the high priority element first. The higher priority elements are executed before the lower priority elements.

DSP dictionary entry

The IATYDSD macro generates an entry for a dynamic support program (DSP) in the DSP dictionary, (module IATGRPT), or in a C/I FSS address space, (module IATGRPTF). An entry in the table is required for each DSP in order for it to be recognized as part of JES3. The following are parameters on the macro that define the DSP characteristics:

ISDRVR Specifies the 1- to 8-character name of the DSP whose entry is being created by this macro. If the DSP is to be callable, this name will appear as the argument of an **`*X,dspname`** command. If the DSP is to be processable, this name will appear in the //*PROCESS dspname JCL statements. The label is required.

PRTY Specifies the priority to be assigned to the DSP, in the range from 1 through 255. This priority becomes the FCT priority when the DSP is activated.

REENT Specifies that the DSP is reenterable.

DRVR Specifies the name of the DSP driver module to be loaded, if necessary, for each use of the DSP.

CSECT Specifies the name of the data CSECT to be loaded by the job segment scheduler driver (module IATGRJR) for each use of the DSP.

MAXCT Specifies the maximum number of copies of this DSP that may be concurrently active. The number must be within the range 1 through 65535. If this parameter is not specified for a DSP specifying REENT=YES, no MAXCT limit is imposed. This parameter is dynamically alterable by using the *MODIFY command unless you specify MUCC=NO.

JSS scheduling

JSS schedules all DSPs into execution. The function of the job segment scheduler (JSS) is to select scheduler elements (SEs) and prepare them for processing by JES3. Every SE denotes a unit of work JES3 must perform to process the job. Those units of work re processed sequentially. Each SE in the job control table (JCT) represents one or more dynamic support programs (DSPs), and each DSP is represented by one function control table (FCT) entry on the FCT chain, the master JES3 dispatching queue.

15.2 Writing DSPs

- Add DSP to System
  - IATYDSD Macro in IATGRPT
  - Resident FCT? - IATYFCD in IATGRPT
    - Assemble and Link in IATNUC
- Define DSECTs
  - IATYFCT, IATYTVT, IATYJDA, IATYFDB,
  - IATYEQU, IATYREG, IATYxxx, etc...

Creating DSPs

Before your DSP can be executed (after you've assembled it and link-edited it), you need to include the name of the DSP in the DSP dictionary. To do so, add one or more of each of the following two macros in module IATGRPT:

- IATYDSD - Generates one entry for the DSP in the DSP dictionary.
- IATYFCD - Generates an entry in the permanent function control table (FCT). You need one IATYFCD macro for each FCT entry you want.

Note: If you want the DSP to have a resident FCT, include the IATYFCD macro in IATGRPT, but do not include IATYDSD.

Update of DSP dictionary

If you are updating the DSP dictionary, add the new names to the end of the dictionary and perform a warm start or a hot start. If you add new DSP names to some position other than the end of the DSP dictionary, you must cold start the JES3 system because the DSP numbers reside in control blocks in spool data sets. All routines should reference JES3 control blocks using named fields in JES3 DSECTs. Routines should not make references using an absolute displacement because the displacements of fields are subject to change.

All JES3 tables should be referred to by using DSECTs generated by mapping macros. For ease of reference, place these macros at the beginning of a program after the prolog but before any executable instructions. Use of the mapping macros insulates the user-written DSPs from changes to system control blocks.

15.3 Steps for scheduling DSPs

- JSS schedules DSP
- DSP initialization
- DSP housekeeping
- DSP termination
- Register conventions

R0, R1	Parameter Information
R2 - R9	DSP Scratch Registers
R10	DSP Base Register
R11	FCT Address
R12	TVT Address
R13	CSECT Address
R14	Return Address
R15	Entry Point or Return Code

DSP processing after scheduling

The following represents the recommended standard sequence of required tasks for the initialization phase of a user-written DSP, which the job segment scheduler (JSS) will schedule. Use the list only as a general guide since, at times, you can omit steps or vary the sequence to reflect particular processing requirements.

As a minimum, perform step 1. The other steps depend on whether the DSP needs a job description accounting block (JDAB) and job data set (JDS) information.

1. Establish a base register for the DSP. By convention, DSPs use register 10 as the standard base register. Although another register can be used, using the standard base register convention will ease program analysis.
2. Issue the LOGIN macro to establish the means by which the operator can communicate with the DSP.
3. Establish the JESTAE environment.
4. Issue the IATXCNS macro specifying TYPE=GET. On return, register 1 contains the address of the parameter buffer for the job.
5. Extract the parameter information from the parameter buffer in the previous step.
6. Issue the IATXCNS macro specifying TYPE=RELEASE to free the parameter buffer.
7. Issue the GETUNIT macro to request any required JES3 support devices. If the required devices are unavailable, cancel or request specialized rescheduling for the unavailable devices. The system programmer must specify the type of devices required by the DSP in the device requirements table (DRT). The DRT is located within the DSP dictionary and is

built using the REQ parameter on the IATYDSD macro. The system programmer must fill in the GETUNIT LIST (GLIST) before the DSP issues the GETUNIT macro.

8. Issue the JDSGET macro to get the JDS, and extract FDBs for the data sets to be processed as input, if any. The JDS contains an FDB and data set name for each SYSOUT data set or data set entered through the input stream (SYSIN) that is associated with the job.
9. Issue the JDSREL macro to release the JDS.
10. I Issue a sign-on message. Once the DSP has verified its parameters and has successfully obtained its devices, it can issue a sign-on message to notify the operator that the DSP is active.

Register conventions

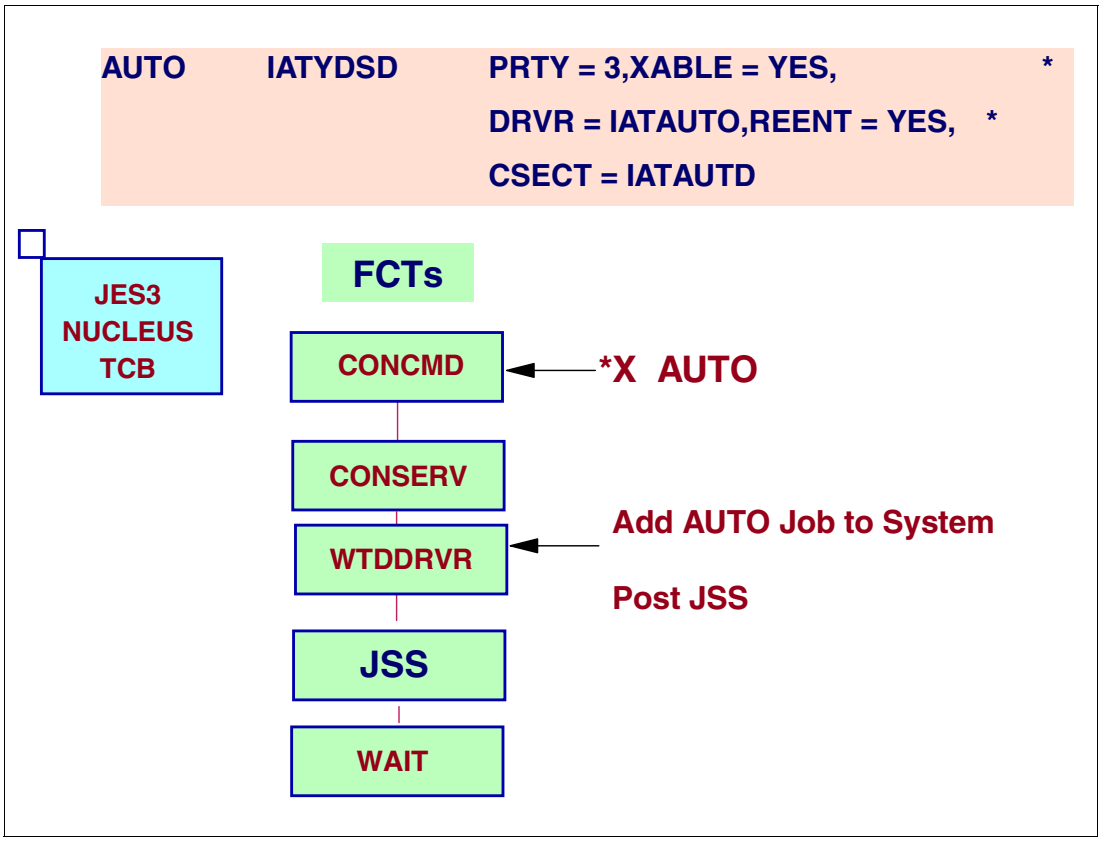
Register usage for installation exits is defined for each exit in In general, registers 11, 12, 14, and 15 on entry are as defined below but other input registers are also provided. Also, most installation exits use the same register conventions.

JES3 stores the following data in the registers before passing control to a DSP:

- Registers 0 through 10 are undefined.
- Register 11 contains the address of the function control table (FCT) for the DSP.
- Register 12 contains the address of the JES3 transfer vector table (TVT).
- Register 13 is the base register of the DSP's data CSECT, if one is defined for the DSP.
- Register 14 is the address to which a called routine must return.
- Register 15 is the entry point address of the called program.

Once JSS schedules the DSP and it gets control, the DSP should initialize, do some housekeeping, and finally have a termination routine.

15.4 Writing a user DSP



Creating a user DSP

When writing a user DSP, define it in the IATGRPT module and to make it callable by the operator, specify XABLE=YES, XABLE=YES specifies that the DSP can be called by using the `*X` command.

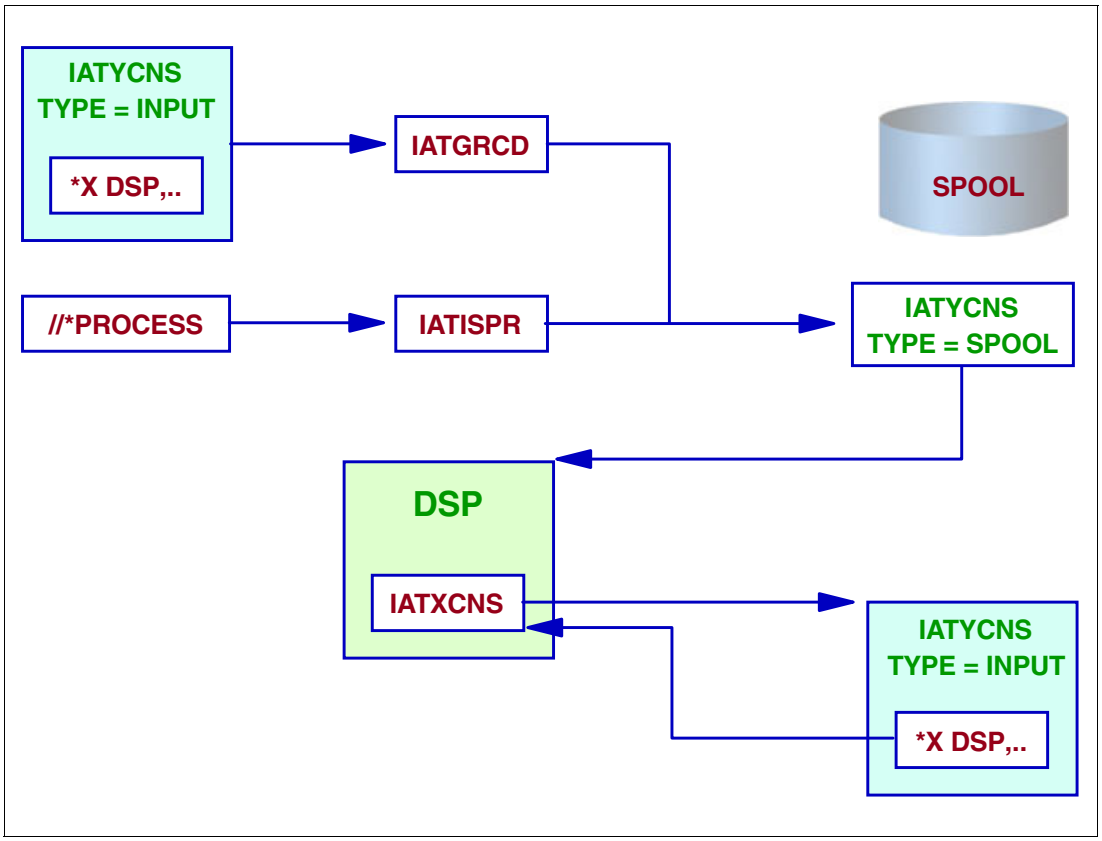
CONCMD DSP

The `*X` command issued by the operator is read by a DSP named CONCMD, which is responsible for command processing. The CONCMD DSP is represented by a resident FCT entry, so it will be dispatched (if there work to do) when the multifunction monitor reaches the entry.

WTDDRVR DSP

The CONCMD DSP simply routes the `*X` command to another DSP, the work-to-do driver (WTDDRVR). Like the CONCMD DSP, the WTDDRVR DSP is represented by a resident entry on the FCT chain. It is the WTDDRVR DSP which actually begins the process of adding the called job to the system.

15.5 DSP initialization



Steps during DSP initialization

The IATYCNS macro generates various tables used to communicate information between console service and other functions within JES3. It is a mapping macro.

Retrieve operator call of DSP

The IATXCNS TYPE=INPUT macro reads a console input buffer record (field CONSMESS of data area IATYCNS) from the JES3 spool. The macro can either:

- Read the job description accounting block (JDAB) associated with the called job, and read and return the address of the console input buffer record, or
- Release the console input buffer record returned by a previous macro call as shown in the last step.

DSP created using //*PROCESS statement

A console spool message buffer is created for the DSP. For the called DSP, module IATGRCD creates the spool message buffer containing the DSP related information.

DSP gets scheduled

When the DSP is scheduled by JSS, the called or PROCESSed DSP information is obtained by the DSP from the spool message buffer. Now the DSP knows what the call parameters are.

15.6 DSP initialization processing

1. ISSUE IATXCNS TYPE = GET
2. PROCESS PARMS

- REG 1 points to a console cell
- This cell contains PARMS
  - USING CONSMESS,R1

3. ISSUE IATXCNS TYPE = RELEASE

DSP processing

Issue the IATXCNS macro specifying TYPE=GET. On return, register 1 contains the address of the parameter buffer for the job. You now have access to the operator `*X` command and its parameters.

When finished processing the `*X` command parameters, issue the IATXCNS TYPE=RELEASE to free the JDAB parameter buffer that contained the parameters. When doing this RELEASE, you must specify that the console input buffer record returned by the previous IATXCNS TYPE=GET macro call is to be released. The address of the console input buffer record address must be specified by the BUFFER= parameter when issuing the macro.You can use register 2.

CONSMESS data area

The IATXCNS macro reads a console input buffer record (field CONSMESS of data area IATYCNS) from the JES3 spool. The macro can either:

- Read the job description accounting block (JDAB) associated with the job, and read and return the address of the console input buffer record, or
- Release the console input buffer record returned by a previous macro call.

Note: The console input buffer is sometimes called an input message buffer or an input parameter buffer. Issuing the IATYCNS macro with TYPE=INPUT generates the buffer table, as shown in the previous visual.

15.7 DSP housekeeping

- Operator communication
- Use of subtasking
- Unit record I/O
  - Use JESOPEN, JESEXCP, JESCLOSE
- Subroutines or other load modules
  - Use ALOAD, ADELETE
- Command input simulation
- General facilities use

DSP functions during start-up

DSPs, except DSPs running in a C/I FSS address space, must issue a LOGIN macro for operator communication and must always respond to console messages. DSPs running in a C/I FSS address space must issue a LOGIN macro to define the entry point to a console appendage routine but do not communicate directly with the operator.

MVS macro usage

Do not use MVS macros that either implicitly or explicitly use the MVS WAIT function, except under the control of a JES3 subtask (see the IATXCSF macro). These macros include:

```
ATTACH      FIND      PURGE
BLDL        LINK      WAIT
CLOSE       LOAD      WTO
DOM         LOCATE    WTOR
DYNALLOC    OPEN      XCTL
```

DSPs should not use these macros. Use of these macros disrupts the flow of processing on the global main and can cause a degradation in system performance by possibly causing JES3 itself to wait. The queued access methods, QSAM and QISAM, use these macros and, therefore, should not be invoked. If your program needs to use one of the above macros, you must first establish a JES3 subtask environment in which to use the macro. To do so, include the IATXCSF macro in your program. IATXCSF passes control to a JES3 subtask.

Unit record I/O

DSPs must perform all unit record and tape I/O using the JESOPEN, JESEXCP, and JESCLOSE macros. DSPs and installation exit routines operating in a C/I FSS address space cannot perform unit record and tape I/O.

Loading modules

If your program consists of more than one module, you should organize it so that the first module is reenterable and all other parts are at least serially reusable. If you use multiple modules in your program, use the ALOAD and ADELETE macros for dynamic loading and unloading of the subprograms. In case of a failure in your DSP, be sure to set up a JESTAE environment that includes the ADELETE macro.

Console communication

All DSP-to-operator communication is accomplished by the console service routines. Each DSP that requires two-way communication with the operator must define a console appendage routine for accepting asynchronous entries from console service. The DSP must use the LOGIN macro to define the console appendage routine's entry point. For reentrant DSPs, therefore, appendages are typically located in the DSP data CSECT. See next visual.

General facilities use

The IATXCSF macro provides the ability to execute code containing implicit or explicit MVS WAIT macros (for example: OPEN, CLOSE, FIND). This is done by passing control from one of the JES3 subtasks provided for this purpose to an appendage routine specified by the ENTER parameter.

The two types of JES3 subtasks that can be invoked by the IATXCSF macro are:

General subtasks Any caller can use these subtasks.

Specific subtasks Callers specifying the subtask ID can use these subtasks.

IATXCSF uses one of the four general subtasks when the ID parameter is not specified. The caller receives control under one of the subtasks and has no control over which subtask is used. When running under a general subtask's control, you must free (before relinquishing control) all resources that were acquired while running under control of the subtask.

15.8 Operator communication with DSPs

- Establish console appendage
- Use Macro - LOGIN
  - Establishes address in FCT
- Where do **YOU** put it ?

LOGIN ENTER = DJAPPEND,FAIL = YES

LOGIN macro

DSPs, except DSPs running in a C/I FSS address space, must issue a LOGIN macro for operator communication and must always respond to console messages. DSPs running in a C/I FSS address space must issue a LOGIN macro to define the entry point to a console appendage routine but do not communicate directly with the operator. Instead, the CIDRVR FCT receives the operator commands and routes them to the C/I FSS address space using the MVS functional subsystem intercommunication (FSI).

Console appendage

All DSP-to-operator communication is accomplished by the console service routines. Each DSP that requires two-way communication with the operator must define a console appendage routine for accepting asynchronous entries from console service. The DSP must use the LOGIN macro to define the console appendage routine's entry point. For reentrant DSPs, therefore, appendages are typically located in the DSP data CSECT.

The console appendage is entered when a `*START`, `*RESTART`, or `*CANCEL` command is received for the DSP, and may be entered when a `*FAIL` command is received.

Because DSP console appendages run under the CONCMD FCT, which is the highest priority FCT, their processing time should be as short as possible. You should limit processing to deciding whether to accept the command, and, if necessary, saving the command and posting the command processing routine of the DSP.

15.9 DSP initialization and messages

00	DSP Accepts command - save in work area
04	Queue command - Use DEQMSG macro
08	Reject command - Consoles issues IAT7130
12	Reject command - add text to IAT7130

- Issue "Login" message - using calling console

IAT7902 JES3 SPOOL MAINTENANCE DSP STARTED

- Issue AWAIT macro
  - ECF is operator message post

NI DJDFLAG2,FF - DJNOMSG - DJNOSTRT allow messages

AWAIT ECFMASK = X'40',ECFADD = DJDFLAG2

Operator communication with DSP

The LOGIN macro establishes communication and transfer of data between console service and the dynamic support program (DSP) using the macro. This macro must be executed by each DSP that allows the receipt of messages and responses from the consoles.

The return codes, shown in the visual, specify what to reply is sent back to the operator or that the command is queued to be processed by the DSP.

Then set your ECF to allow new messages from the operator and issue an AWAIT waiting for the next communication from the operator.

```
NI   DJDFLAG2,FF - DJNOMSG - DJNOSTRTallow messages
AWAIT    ECFMASK = X'40',ECFADD = DJDFLAG2
```

Command from operator

On entry to the console appendage, register 1 contains the address of the console message buffer (which is mapped by the TYPE=INPUT expansion of the IATYCNS macro). The buffer includes the console destination block (CNDB), verb code, message length, and flags used by the JES3 command processing routines. Register 0 contains the FCT address of the function that owns the console appendage. Once your DSP processes the operator command, the DSP issues a message to let the operator know everything is okay.

15.10 Use of subtasking

- Any coding that has WAITs
  - Execute this code under a JES3 Subtask
- Consider using JESTAE

```
         TM    DCBOFLGS,DCBOFOPN          Is DCB open?
         BC    ALLOFF,DRTTB347            No, avoid closing it
         IATXCSF ENTER=DRCLSBTK           Call subtask to close it
         .
* THE FOLLOWING CODE EXECUTES
* AS A SUBTASK OF JES3
DRCLSBTK DS    0H                         Subtask to close DCB
         LR    R4,R14                     Save return address
         CLOSE (ICTPRDCB),MF=(E,ICTMFLOP)  Issue MVS CLOSE macro
         LR    R14,R4                     Restore return address
         SR    R15,R15                    Clear return code
         BR    R14                        Return to caller
```

- Writing DSPs
  - Documented in JES3 Customization

Use of subtasking

As stated in a previous figure, use the IATXCSF macro to obtain a subtask to execute code that would cause a WAIT of the JES3 task. The macro specifies the address of the code to execute under the subtask (MYSUB).

JESTAE macro

If your DSP uses an MVS function that could cause an abnormal termination, be sure that you set up a JESTAE recovery environment. The purpose of the recovery environment is to insure that a failing DSP does not also bring down the JES3 primary task (IATNUC).

The JESTAE macro defines a DSP abnormal exit routine. The JESTAE exit routine must be resident throughout the life of the JESTAE request. A DSP can issue more than one JESTAE macro. All JESTAE requests issued by programs running under the same FCT are queued so that the exit established by the most recent JESTAE request will be the first to get control. If this exit fails or requests that the abnormal termination continue, the exit established by the previous JESTAE request will get control. This process is called JESTAE percolation.

Coding example

The coding example in the visual illustrates the use of the IATXCSF macro. The code to be executed under control of the subtask begins at label DRCLSBTK.

See *z/OS JES3 Customization*, SA22-7542document for more information.

15.11 Dump job DSP

- Operator callable DSP
- Dumps JES3 job queue to tape - OUT mode
- Dumps jobs back to spool - IN Mode
- When to use DJ?
  - Migration to new release
  - Need to do COLDSTART - but: Save jobs
  - Spool is becoming FULL - Offload some jobs
  - Put jobs on tape - Take to another global

Dump job DSP

Dump job is a specialized JES3 utility designed to dump jobs from the JES3 job queue to tape, and later, reintroduce those jobs back into the job queue at the point of processing at which they had been dumped.

This facility is used when a system requires a software change, when a test is to be conducted on a dedicated system, when the user shuts down the complex but requires the integrity of the job queue to be maintained or when job data is being archived. Use dump job for:

- Archive. Some installations regularly dump jobs to tape to save them for a given period of time.
- Provide additional spool space. Jobs can be dumped to tape when the current workload is heavy, and restored when the workload lessens.
- Perform preventive maintenance. Jobs not complete at a time when preventive maintenance is scheduled can be dumped and subsequently restored.
- Migrate. Installations can save and restore jobs when migrating from one level of JES3 to another.

15.12 Dump job DSP - (spool offload)

- *X DJ,OUT=(TA33490)
  - Job selection criteria
    - "ALL" for input and output modes

*S DJ,

J = Jobid
P = priority
N = netname
ALL
C = class
JOBS
NETS
RANGE=(jobno1-jobno2)

Calling dump job

When calling dump job to dump to tape, specify OUT= to some tape device. As many as eight DJ DSPs can be invoked concurrently, permitting you to dump and restore jobs simultaneously.

You can specify the range of jobs to be selected for dumping or restoring. Jobs can be in any non active stage of processing and, with the exception of DJC networks and jobs waiting to be transmitted across networking lines, do not need to be in hold status.

The selection criteria allows dumping by:

- Job number - J=12
- Priority - P=2
- DJC network name N=PAYR
- All jobs in the queue
- By job class
- All jobs that are not part of a DJC network
- All DJC networks
- All jobs within a specific range of job numbers that are not part of a DJC network

15.13 Dump job reset option

- DJ looks at each job only once
- Prevents dumping a job twice
- Prevents recursive DJ failures
  - *S DJ,RESET,selection - criteria
  - Resets job DJ indicators
  - Allows DJ to look at those jobs again
- Restrictions
  - Output mode DJ only
  - Only one output mode DJ may be active

Dump Job reset option

Dump job looks at every job once as it makes a pass through the job queue. Jobs are flagged as being looked at to avoid dumping the same job twice.

When a DJ DSP is invoked to dump a job or net, the DJ facility sets dump control flags for each job processed. Because of these flags, any jobs remaining on the JES3 job queue after DJ has processed them cannot be dumped again until the flags are reset.

Use the ***START** command with the RESET parameter to reset DJ dump control flags for all jobs or a selected set of jobs on the JES3 job queue. Enter this command when all other dumping using the DJ facility is complete.

15.14 Dump job server mode

- DJ executes in own address space
  - DJ server address space
- DJ can run in server mode or non-server mode
  - Default mode is SERVER=NO
- Tape library tapes may be used
  - Eliminates need for non-library tapes just to use DJ
- Original DJ non-server mode can be used

Dump job server mode

Dump job can be run in either server or non server mode. When dump job is run in server mode, a dump job server address space is started to allocate the tape device. All tape-related functions such as reading or writing to the tape are done from the dump job server address space.

You can use the dump job facility to dump jobs to tape devices that you have defined through the use of the JUNIT and XUNIT initialization parameters or to the tape devices that include the IBM 3494 and 3495 Tape Library Dataserver.

Using JUNIT and XUNIT to perform the dump job process is the traditional way that maintains that JES3 manages the tape I/O to JES3 defined tape devices. Using the IBM 3494 and 3495 Tape Library Dataserver as well as any tape devices in the JES3 system to perform a dump job exploit a new feature of dump job. This feature is running dump job in server mode In this mode, the tape I/O portion of dump job runs in its own address space. The major MVS I/O facilities are used freeing the installation from having to remember the volumes and their current order when the job restore is performed.

15.15 DJ server mode

- All tape access in DJ server address space
  - Uses standard data management
    - DYNALLOC, OPEN, CLOSE, EXCP, EOV
    - Handles multi-volume mounting
- Server mode data set naming (cataloged)
  - For SL tapes - JESx.DJ.Dyyyyddd.Thhmmss
  - For NL tapes - JESx.DJOUT

Using server mode

In the server address space, all tape handling is done by standard data management macros, which includes the handling of multi-volume tape dumps.

The data set name that is generated is different for standard label tapes versus unlabeled tapes.

- For standard label tapes, the data set name has the following format, where jesn is the JES3 subsystem name.

  ```
  jesn.DJ.Dyyyyddd.Thhmmss
  ```

- For unlabeled tapes, the data set name is not unique and has the following format, where jesn is the JES3 subsystem name.

  ```
  jesn.DJOUT
  ```

To dump jobs to tape, issue the **`*S DJ`** command and specify which jobs you want dumped.

15.16 DJ server mode tape drives

- Tape drives do not need to be defined to JES3
  - Devices defined to JES3:
    - Vary online as execution device - *V 560,ON,SY1
  - Devices not defined to JES3
    - MVS vary online if not already online
- Generic or esoteric names can be used:
  - *X,DJ,OUT=(3490),LABEL=SL,SERVER=YES
  - *X,DJ,OUT=(LDE10435),SERVER=YES

Defining JES3 tape drives

The device used by Dump Job when using server mode does not have to be defined to JES3, but you may do so. Also, a generic device name (e.g. 3490) or an esoteric device name (e.g. TAPE) may be specified on the ***X** command.

If you have tape DEVICE statements in your initialization stream for use only by Dump Job, these statements can be removed once you decide only to run Dump Job in server mode. To remove tape DEVICE and SETNAME statements from the initialization stream requires a JES3 warm start.

If you want JES3 to continue to manage tape devices for jobs in execution but no longer need them for Dump Job, you can remove the DTYPE, JNAME, and JUNIT parameters from the tape DEVICE statements and perform a hot start with refresh. If you change your mind and want to add them back, this can also be accomplished by performing a hot start with refresh.

Using tape drives

The following is an example of dumping jobs to tape when dump job is in server mode. The first thing that needs to be done is to ***X** the dump job DSP. In this case, we are creating a standard labeled tape on device 560. If device 560 is defined as a JES3 managed device, it must be varied online as an execution device to the global processor (SY1):

```
*V,560,ON,SY1
IAT8180 0560     VARIED ONLINE  TO JES3 ON SY1
*X,DJ,OUT=560,LABEL=SL,SERVER=YES
```

Instead of using a specific device, a generic or esoteric device name could have been specified on the *X command. In this case, the device name must be enclosed in parentheses:

```
*X,DJ,OUT=(3490),LABEL=SL,SERVER=YES
```

or

```
*X,DJ,OUT=(LDE10435),SERVER=YES
```

Note: When using non-server mode, the device used by Dump Job must be defined to JES3 via a DEVICE statement and must be defined as a shared device. It must be defined as a JES3 global device via the DTYPE, JNAME, and JUNIT parameters, and as an execution device via the XTYPE and XUNIT parameters.

15.17 Using DJ server mode

- *X DJ,SERVER=YES|NO,.....
  - DSN=dsname
    - Only allowed with SERVER=YES and IN=
    - *X,DJ,IN=560,SERVER=YES,DSN=JES3.DJ.D1998091.T163039
  - VOL=
    - Only allowed with IN= and NL tapes
    - ***X,DJ,IN=560,SERVER=YES,DSN=JES3.DJOUT,VOL=(TAPVOL,TAPVL2,TAPVL3)**

Using serve mode to dump jobs

The following is an example of restoring jobs from tape when dump job is in server mode. When calling the dump job DSP, for standard label, server mode requests, the DSN= parameter must specify the name of the data set that was created when the jobs were dumped to tape:

```
*X,DJ,IN=560,SERVER=YES,DSN=JES3.DJ.D1998091.T163039
```

As a result of the ***X** command, a dump job server address space is started. The dump job server address space initializes and allocates the tape device. The tape device is allocated with deferred mounting so you will not see any IAT5210 messages asking the operator to mount the tape if this is a JES3 managed device. A mount message (IEC501A) will be issued when a ***S DJ** command is issued and the tape data set is opened.

If a unlabeled tape was created when the jobs were dumped to tape, the VOL= parameter must be specified in addition to the DSN= parameter. This is necessary because unlabeled tapes are always created and cataloged with the data set name jesn.DJOUT. If you create multiple unlabeled tapes, JES3 needs to know the volsers to determine which instance of jesn.DJOUT you want to restore. This is not a problem for standard labeled tapes because the data set name that is generated and cataloged is unique.

```
*X,DJ,IN=560,SERVER=YES,DSN=JES3.DJOUT,VOL=(TAPVOL,TAPVL2,TAPVL3)
```

15.18 DJ server mode examples

- Server address space created by ASCRE
  - Procedure name - IEESYSAS
  - MVS identifier id DJnnnnn - DJ00033

*X DJ,OUT=560,LABEL=SL,SERVER=YES

```
IAT6306 JOB (JOB00033) IS DJ      , CALLED BY 01

IAT6100 ( DEMSEL ) JOB IEESYSAS (JOB00034), PRTY=15, ID=*UNKNOWN

SY1 IEESYSAS  IEF403I IEESYSAS - STARTED - TIME=16.11.04

IAT5110 JOB IEESYSAS (JOB00034) GET  255 T SCRTCH ,SL JES3.DJ.D1998091
```

Server mode example

As a result of the ***X** command, a dump job server address space is started. The dump job server address space initializes and allocates the tape device. The tape device is allocated with deferred mounting so you will not see any IAT5210 messages asking the operator to mount the tape this is a JES3 managed device. A mount message (IEC501A) will be issued when a ***S DJ** command is issued and the tape data set is opened. The following messages are seen by the operator:

```
IAT6306 JOB (JOB00033) IS DJ      , CALLED BY 01
 IAT6100 ( DEMSEL ) JOB IEESYSAS (JOB00034), PRTY=15, ID=*UNKNOWN
 SY1 IEESYSAS  IEF403I IEESYSAS - STARTED - TIME=16.11.04
 IAT5110 JOB IEESYSAS (JOB00034) GET 255 T SCRTCH ,SL JES3.DJ.D1998091.
```

Server address space

The job name of the dump job server address space is DJ followed by the job number of the DJ DSP that started the server address space. To display information about the dump job server address space, issue one of the following commands:

D A,DJ* or D A,DJ000033

15.19 DJ server mode command examples

```
D A,DJ*  ...or...  D A,DJ000034

SY1           IEE115I 16.12.03 1998.049 ACTIVITY 351
 JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
00000    00006    00001      00026    00003    00001/00300       00001
 DJ000033 DJ000033 IEFPROC  NSW  *   A=0024   PER=NO   SMC=000
                                     PGN=N/A  DMN=N/A  AFF=NONE
                                     CT=000.093S  ET=058.661S
                                     WUID=JOB00034 USERID=++++++++
                                     WKL=SYSTEM   SCL=SYSTEM   P=1
                                     RGP=N/A      SRVR=NO  QSC=NO
                                     ADDR SPACE ASTE=034BC900

*I J=34

IAT8674 JOB IEESYSAS (JOB00033) P=15 CL=A        R=(512K,512K),
 MAIN(EXECUTING-SY1)
```

Command examples to display active server job

The job name of the dump job server address space is DJ followed by the job number of the DJ DSP that started the server address space.

The job name of the dump job server address space is DJ followed by the job number of the DJ DSP that started the server address space. To display information about the dump job server address space, issue one of the following commands:

D A,DJ* or D A,DJ000033

***I J=34**

15.20 DJ operator messages

```
 IAT7285 DJ0560 (JOB00033): OUTDSN=JES3.DJ.D1998091.T163039
 IAT7213 DJ0560 (JOB00033): UP AND RUNNING; OUTPUT ON UNIT 0560, DEVICE MVS 560
*IAT7228 ISSUE START OR CANCEL FOR DJ (JOB00033) (0560)
```

Start dumping jobs to tape in priority 6

*S DJ,P=6

```
*IEC501A M 0560,TAPVOL,SL,,IEESYSAS,DJ000034

...tape now mounted...

IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB51    (JOB33436
IAT7450 JOB JOB51    (JOB33436) PURGED
IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB33    (JOB33435
IAT7450 JOB JOB33    (JOB33435) PURGED
IAT7230 DJ0560 (JOB00033): DUMP PROCESSING COMPLETE FOR PRIORITY LEVEL 06
IAT7253 DJ0560 (JOB00033): 0000038 JOBS SUCCESSFULLY DUMPED TO TAPE
IAT7220 DJ0560 (JOB00033): FUNCTION COMPLETE ON UNIT 0560
*IAT7228 ISSUE START OR CANCEL FOR DJ (JOB00033) (0560)
```

Operator messages

After the dump job server address space has successfully initialized, dump job issues the messages below to show that it is ready to begin dumping jobs to tape. Message IAT7285 contains the name of the tape data set that contains the jobs that are dumped to tape. This data set name must be specified on the ***X DJ** command when you restore the jobs from tape.

```
IAT7285 DJ0560 (JOB00033): OUTDSN=JES3.DJ.D1998091.T163039
IAT7213 DJ0560 (JOB00033): UP AND RUNNING; OUTPUT ON UNIT 0560, DEVICE MVS 560
*IAT7228 ISSUE START OR CANCEL FOR DJ (JOB00033) (0560)
```

To dump jobs to tape, issue the ***S DJ** command and specify which jobs you want dumped. In the example that follows, jobs in priority 6 will be dumped to tape:

```
*S,DJ,P=4
```

As a result of the ***S** command, dump job dumps the requested jobs to tape:

```
*IEC501A M 0560,TAPVOL,SL,,IEESYSAS,DJ000034
```

Eventually, the following will be seen:

```
tape now mounted
```

15.21 DJ server mode commands

- Operator control of DJ server mode
  - *C DJ - (only if one DJ active)
  - *C DJ,J=jobno -or- *C DJdevnum
  - Use MVS cancel command to cancel server A/S

```
*C,DJ

IEF234E R 0560,TAPVOL,PVT,IEESYSAS,DJ000034
IEF471E FOLLOWING VOLUMES NO LONGER NEEDED BY IEESYSAS
        TAPVOL.
IEF404I IEESYSAS - ENDED - TIME=16.37.00
IAT7200 DJ0560 (JOB00033): DUMP JOB DSP TERMINATING
IAT7450 JOB DJ       (JOB00033) PURGED
```

Server mode commands

You can now issue additional ***S DJ** commands to dump other jobs to tape. When you are finished dumping jobs to tape, you can cancel the DJ DSP. As a result, the dump job server address space will close the tape data set and end, as follows:

```
*C,DJ

IEF234E R 0560,TAPVOL,PVT,IEESYSAS,DJ000034
IEF471E FOLLOWING VOLUMES NO LONGER NEEDED BY IEESYSAS
        TAPVOL.
IEF404I IEESYSAS - ENDED - TIME=16.37.00
IAT7200 DJ0560 (JOB00033): DUMP JOB DSP TERMINATING
IAT7450 JOB DJ       (JOB00033) PURGED
```

15.22 DJ log data set

```
*X DJ,SERVER=YES OUT=560
> OUT DSN=JES3.DJ.D1998091.T163039 - TAPVOL
*S DJ P=6
IAT7226 DJ0560 (JOB00033): JOB JOB999   (JOB32766) CANNOT BE DUMPED -
IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB21    (JOB33414)
IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB12    (JOB33413)
IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB30    (JOB33418)
IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB3     (JOB33420)
IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB6     (JOB33438)
IAT7229 DJ0560 (JOB00033): SUCCESSFULLY DUMPED JOB JOB24    (JOB33434)
IAT7230 DJ0560 (JOB00033): DUMP PROCESSING COMPLETE FOR PRIORITY LEVEL
IAT7230 DJ0560 (JOB00033): DUMP PROCESSING COMPLETE FOR JOBS REQUEST
IAT7253 DJ0560 (JOB00033): 0000038 JOBS SUCCESSFULLY DUMPED TO TAPE
IAT7220 DJ0560 (JOB00033): FUNCTION COMPLETE ON UNIT 0560

*S DJ P=1
IAT7230 DJ0560 (JOB00033): DUMP PROCESSING COMPLETE FOR PRIORITY LEVEL
IAT7253 DJ0560 (JOB00033):      NO JOBS SUCCESSFULLY DUMPED TO TAPE
IAT7220 DJ0560 (JOB00033): FUNCTION COMPLETE ON UNIT 0560
*C DJ
IAT7200 DJ0560 (JOB00033): DUMP JOB DSP TERMINATING
```

DJ log data set

The visual shows the DJ log data set which contains the DJ messages related to what was dumped to the tape.

In addition to writing all messages to the calling console, the DJ facility logs in a separate data set all DJ START commands and all DJ job-related messages that indicate whether a job was successfully dumped or restored. If tracing is specified via the TRACE parameter on the ***S** command, all trace output is also recorded in the data set. You can print the DJ message log data set by specifying the SPIN=YES parameter on the ***S DJ** command. If SPIN=YES is not specified on the **START** command, the DJ message log data set is printed when the DJ DSP is cancelled.

When the first start command completes, a second command can be issued to dump a different set of jobs to the tape.

```
*S DJ,P=1
```

15.23 Restore jobs in server mode

```
*X  DJ,IN=560,SERVER=YES,DSN=JES3.DJOUT,VOL=(TAPVL1,TAPVL2),LABEL=NL

*X  DJ,IN=560,SERVER=YES,DSN=JES3.DJ.D1998091.T163039

 IAT6306 JOB (JOB33530) IS DJ      , CALLED BY 01
 IAT6100 ( DEMSEL ) JOB IEESYSAS (JOB33531), PRTY=15, ID=*UNKNOWN
 IEF403I IEESYSAS - STARTED - TIME=16.51.54
 IAT5110 JOB IEESYSAS (JOB33531) USES     T TAPVOL ,SL JES3.DJ.D199809

 IAT7213 DJ0560 (JOB33530): UP AND RUNNING;  INPUT ON UNIT 0560, DEVICE MVS 0560
*IAT7228 ISSUE START OR CANCEL FOR DJ (JOB33530) (0560)

*S  DJ,P=6

*IEC501A M 0560,TAPVOL,SL,,IEESYSAS,DJ033530

...tape now mounted...

IAT7255 DJ0560 (JOB33530): JOB JOB21    (JOB33414) ENTERED INTO
PRIORITY 04 AS JOB (JOB33525)
```

Dump jobs in with server mode

If a unlabeled tape was created when the jobs were dumped to tape, the VOL= parameter must be specified in addition to the DSN= parameter. This is necessary because unlabeled tapes are always created and cataloged with the data set name jesn.DJOUT. If you create multiple unlabeled tapes, JES3 needs to know the volsers to determine which instance of jesn.DJOUT you want to restore. This is not a problem for standard labeled tapes because the data set name that is generated and cataloged is unique.

```
*X,DJ,IN=560,SERVER=YES,DSN=JES3.DJOUT,VOL=(TAPVOL,TAPVL2,TAPVL3)
```

As a result of the *CALL command, a dump job server address space is started. The dump job server address space initializes and allocates the tape device.

```
IAT6306 JOB (JOB33530) IS DJ      , CALLED BY 01
IAT6100 ( DEMSEL ) JOB IEESYSAS (JOB33531), PRTY=15, ID=*UNKNOWN
IEF403I IEESYSAS - STARTED - TIME=16.51.54
IAT5110 JOB IEESYSAS (JOB33531) USES     T TAPVOL ,SL JES3.DJ.D1998091
```

After the dump job server address space has successfully initialized, dump job issues the following messages to show that it is ready to begin restoring jobs from tape.

```
 IAT7213 DJ0560 (JOB33530): UP AND RUNNING;  INPUT ON UNIT 0560, DEVIC
*IAT7228 ISSUE START OR CANCEL FOR DJ (JOB33530) (0560)
```

To restore dump jobs from tape, issue the *START,DJ command and specify which jobs you want dumped. In this example, we will restore jobs in priority 6:

```
*S,DJ,P=4
```

As a result of the ***S** command, dump job restores the requested jobs from the tape (message IAT7255 is displayed on two lines due to space limitations):

```
*IEC501A M 0560,TAPVOL,SL,,IEESYSAS,DJ033530
```

(tape now mounted)

```
IAT7255 DJ0560 (JOB33530): JOB JOB21    (JOB33414) ENTERED INTO PRIORITY
06 AS JOB (JOB33525)
```

15.24 Creating JESNEWS data set

LCL REP

*X JESNEWS,DS=RJP,TYPE=ADD

TSO DEL

Enter Text via:

*S JESNEWS,text ..... for each line

Activate Data set:

*R JESNEWS

JESNEWS data set

JES3 provides a special utility program that allows you to broadcast information to local, TSO/E, and RJP users. This utility, dynamic support program (DSP), can be run at the same time as the other support functions of JES3, such as input service and the main device scheduler. You can start the DSP at an operator console by using the **`*X`** command.

You can use the JESNEWS DSP to create, to replace, or to delete three special output data sets that can be included as part of a normal output data set burst page. The JESNEWS DSP work on three types of data sets: local, TSO/E and RJP. Use these data sets to send information to JES3 users.

You can end processing of the JESNEWS DSP by entering the following command:

```
*C,JESNEWS
```

The text is entered by the operator using the **`*S JESNEWS`** command and when finished entering the text, the **`*R JESNEWS`** command activates the data set.

15.25 JESNEWS DSP

- Output data set
  - Broadcast information to:
    - TSO Users - TSO
    - Remote Terminal Users - RJP
    - Local Users - LCL
- Create data set(s) via - Operator DSP
- Create data set via - //*PROCESS in a Job

Using the JESNEWS DSP

You can use the JESNEWS DSP to create, to replace, or to delete three special output data sets that can be included as part of a normal output data set burst page. The JESNEWS data set is printed at the end of the output for 3 types of users:

- TSO users jobs
- RJP submitted jobs
- Local users jobs

JESNEWS with //*PROCESS statement

This nonstandard job uses one or more special processing functions in place of or in addition to standard processing or skips one or more of the standard functions. Specify a nonstandard job by following the JOB statement with a JES3 //*PROCESS statement for each processing function.

The JESNEWS data set is created by the operator command or by executing a job with //*PROCESS JESNEWS with the data following the //*PROCESS statement.

```
//JESNEWS JOB ..........
//*PROCESS JESNEWS
TYPE=ADD,DS=TSO
/*
```

15.26 Dump core DSP

- Find Modules in Storage
  - *S DC,F=IATGRCT
  - *S DC,C=2301080,S=80
- Set Traps
  - *S DC,TRAP=7EO,BASE=2314688,START=*
  - *S DC,TRAPON - (activates trap)
  - *S DC,TREGS - (display registers at wait)
- Display and Print Control Blocks
  - *S DC,OPTION=FCT
  - *S DC,OPTION=(SNP=JCT,J=10)

Dump core (DC) DSP

The dump core DSP is used to display and modify data in main storage, to intercept program flow during execution, and to format control blocks for debugging. This facility can be used only on the global processor.

Use Dump Core (DC) to display storage allocated to JES3. Dump core commands allow you to:

- Display and then modify data in main storage
- Intercept program flow during processing
- Format control blocks for debugging purposes
- Find the location of a module in storage
- Display a requested portion of JES3's storage
- Set traps
- Display and print JES3 control blocks

Determine where the output from the dump core DSP should be routed. You specify the destination of the output by using the OUT= parameter on the `*X DC` command when you invoke the dump core facility or any `*S DC` command.

15.27 JCT dataspace problem determination

- DC DSP displays JCT in Dataspace
  - ***S DC,OPTION = (SNP = JCT),J = jobno|ALL** (optionally)**.....SOURCE = DSPACE|ALL|JCTDS**
- JMF enhancements for JCT dataspace statistics
- IPCS models for IATYJQX and IATYJCT
- SYS1.DUMPxx for errors accessing JCT dataspace
  - JES3 A/S and JCT dataspace

Using DC for JCT dataspace

The DC DSP displays a specific JCT entry or all JCT entries. The options to display the JCT entry are:

- JCT entry from Data Space
- JCT entry from spool
- Both the Data Space entry and the spool entry

The operator command is:

```
*S DC,OPTION=(SNP=JCT),J=jobno|ALL
```

The above option allows for display of a specific job using J=jobno or of all jobs using J=ALL.

To specify the location of the JCT, the SOURCE keyword can be used:

```
SOURCE=DSPACE|JCTDS|ALL
```

Where:

DSPACE Display Data Space copy of JCT

JCTDS Display JCT spool data set copy of JCT

JCTDS Display JCT spool data set copy of JCT and JCT Data Space copy.

JMF is changed to support JCT Access Method related information. Also displayed is the JQE table information.

The report name is JQE/JCT Access Method Report.

The three main areas of the report are:

- JCT Data Set Information
- JCT Data Space Information
- JQE Information

New IPCS models

Two new IPCS models for the JCT Data Space changes are included in this release. The two macros are IATYJCT and IATYJQX.

```
SYS1.DUMPxx
```

For errors encountered accessing the JCT Data Space, a dump of the JES3 address space and a dump of the JCT Data Space are taken.

15.28 JCT problem determination

- Enter Dump Command
  - DUMP COMM = (JCT DATA SPACE)
- Reply operands for dump command
  - R xx,DSPNAME = (xxxx. JES3JCT)
    Where: xxxx - ASID of dataspace Owner

JCT problem determination

The JCT dataspace can be dumped using the DUMP command. Specify the ASID of the dataspace and the owner JES3JCT when prompted for a reply to the DUMP command.

15.29 Job related information DSPs

- *X DISPLAY - for Job info
  - Displays detailed information about:
    - A single job or
    - All jobs in the JES3 job queue
  - Obtains diagnostic information from the JES3 control blocks associated with the jobs in the job queue
- *X DISPDJC - for DJC Net info
  - Specify the name of DJC network to be displayed
  - Specify ALL networks in the system to be displayed

DISPLAY DSP

Use the **`*X DISPLAY`** command to display detailed information about a single job or all jobs in the JES3 job queue. The **`*X DISPLAY`** command obtains the diagnostic information from the JES3 control blocks associated with the jobs in the job queue.

OUT=devname on the **`*X`** command returns output to the device type you specify. If you omit this parameter, the output goes to the calling console.

You can specify a job number or name to display or a priority level to display jobs.

DISPDJC DSP

Use the DISPDJC facility to display the status of a dependent job control network on a printer. For each network the DISPDJC facility displays:

- The name of the network.
- The FLAG1 parameters as defined in the job network control block (JNCB).
- The FLAG2 parameters as defined in the JNCB.
- The number of jobs in the designated network.
- The number of jobs in the designated network that have completed.

For each job in the network the DISPDJC facility displays:

- The job name
- The current status of the job (completed, active, inactive, or in network hold)

- The names of the jobs that are successors to the designated job and cannot be processed until its completion
- The name and net-id of a successor in another network
- The number of predecessor jobs that must complete before the designated job can be processed
- The action to be taken when a predecessor job ends normally or abnormally

16

Spool partitioning and spool recovery

Most JES3 installations can benefit from using several spool partitions. If used properly, spool partitioning can provide many advantages. However, if these advantages do not apply to your installation, one spool partition will serve as well. The major advantages to using spool partitions are:

- If a spool data set fails, the failure affects only a subset of the jobs in the JES3 complex. That is, the failure affects only those jobs that have data in the spool partition including the failed spool data set, not jobs that have data in other spool partitions. (The failure may not affect all jobs in that spool partition, however. Some jobs may not have had any data on the failed data set.) Thus, spool partitioning improves spool RAS.
- By spreading the use of spool partitions across jobs, job classes, and SYSOUT classes, you can limit the number of processors that compete for each partition. If processor competition for spool data sets is an actual or potential problem for your installation, spool partitioning could improve system performance.
- By specifying track group size on a partition by partition basis, you can tailor spool space allocation to the requirements of jobs using that partition. Efficient use of spool space minimizes spool access time and can improve performance.
- By isolating the JES3 initialization data in its own spool partition, you can prevent the infrequently-accessed initialization data from occupying the track groups that have the best performance characteristics in the default partition or any other partition.
- By isolating critical work in specific spool partitions, you can ensure that spool space is available for critical jobs and users. At the same time, you can ensure that spool space requirements of noncritical applications do not interfere with spool space requirements of critical applications.
- By isolating certain types of work in specific partitions, you can better determine what action to take if a spool data set fails.

16.1 Spool partitioning

SPART,NAME=partition-name
,DEF={YES|NO}
,INIT={YES|NO}
,SPLIM={min,marg}

,OVRFL={NO|partition-name|NO}
,GRPSZ=nnn

- Spool partitioning by:
  - Execution main - MAINPROC
  - Job class - CLASS
  - SYSOUT class - SYSOUT

Spool partitioning

A partition is a group of spool data sets. The partition is used by spool space management to manage track groups within the data sets.

The SPART initialization statement has a parameter that defines action to be taken when the spool space in that partition is exhausted.

```
SPART,...OVRFL=(yes|no|spart),...
```

Where:

yes Specifies that the spool partition may overflow into the default spool partition. This is the default for this parameter.

no Specifies that the spool partition may not overflow. This only affects allocations requested by USAM. Jobs requesting additional spool space will wait. JSAM requests to the partition will overflow into the default spool partition.

spart Specifies the name of the spool partition into which spool space allocation will overflow.

When implementing spool partitioning:

- If OVRFL=YES is not specified on any SPART statement, all out of space allocation will overflow into the default partition. Overflow into other than the default partition is not permitted.
- The OVRFL parameter is ignored if it is specified on a default spool partition definition.

- Code OVRFL=NO to prevent output data sets designated for one partition to overflow into another partition. If it becomes necessary to increase spool space in an out of space partition, another spool extent can be moved into the partition.

Spool space shortage conditions

The SPART initialization statement has a parameter to support minimum and marginal spool space conditions. Also the specification of this condition via the BUFFER statement has been changed to conform with this specification on the SPART statement.

```
BUFFER,...SPLIM=(min,marg),...
                 10  25                (defaults)
SPART,...SPLIM=(min,marg),...
                10  25                 (defaults)
```

Where:

min Specifies a minimum percentage of spool space in a partition. When this threshold is detected for the default partition, all sysout data set track allocation is suspended.

marg Specifies a marginal percentage of spool space in a partition. When this threshold is detected for the default partition, all job selection is suspended.

Implement this function as follows:

- If SPLIM is not specified on any SPART initialization statement, the partition will default to the specification on the BUFFER statement.
- The specification of the values as percentages is new with this release.
- The suspension of sysout processing or job selection only works for the default partition.
- When either a minimum or marginal condition exists, an action message is issued to inform the operator of the condition.

16.2 Spool partitioning concepts

- Default partition
  - No overflow allowed
  - STT expansion
  - Used when other partitions:
    - Deleted
    - Out of space
  - Contains job control blocks and SYSIN

Partitioning concepts

Using SPART initialization statements, you can define up to 1024 spool partitions. Additionally, you can identify one of the partitions as the default partition by specifying DEF=YES on a SPART statement.

You need not define spool partitions or specify a default partition. If you do not define a spool partition (do not include SPART statements in the initialization stream), JES3 creates a minimum of three spool partitions. JES3 assigns spool data sets to one of the following partitions:

DRAINED	Result of an operator drain, hold or cancel command
UNAVAIL	For data sets unavailable to JES3 during JES3 initialization
DEFAULT	JES3 names the default partition JES3PART

If you define partitions but not a default partition, JES3 uses as the default partition the partition defined on the first SPART statement in the initialization stream.

Default partition

The default spool partition always contains:

- JES3 spool access method (JSAM) single and multi-record files
- Job input (SYSIN) data
- JES3 control blocks created by input service

It may also contain output spool data for:

- Jobs requesting a spool partition with no free space that overflows into the default partition
- Jobs requesting a spool partition that has been deleted
- Jobs that do not request a spool partition and for which the job class and processor have no spool partition designation

16.3 Spool partitioning commands

- Display largest users of spool space

```
*I  Q,SP=[ ALL                    ][ ,U(,N=)  ]
         [ SPART                  ][ ,O       ]
         [ {SPART(,SPART),..}     ][ ,DD      ]
```

*I Q,SP=P1,U,N=1

IAT8527 P1: JOB(JOB00010) TEST USES 80 GROUPS, 10%

Partitioning commands

You can use the **`*I Q`** command to identify the spool data sets assigned to a particular partition and to determine if the partition is an overflow partition, the default partition, or the initialization partition. You can also display the size of a partition, the amount of space that is currently available, and the largest users of spool space.

You might want to use this command to help determine if a performance problem is the result of JES3 using a high percentage of the available spool space in one or more spool partitions.

Use the **`*I Q`** command to display:

- A list of jobs waiting for a DSP
- The names of the spool data sets assigned to a spool partition and whether the partition is defined as the default partition, the initialization partition, or an overflow partition
- The size of the partition and the amount of space currently available
- The status of a partition, and users of the largest spool space
- The status of a spool data set and the name of the spool partition that the data set belongs to
- All the defective tracks currently known to JES3
- The amount of space available on all the JES3 spool data sets in the complex

16.4 Spool partitioning commands

- Display spool partition status

  *I Q,SP=P1

  *I Q,SP=P1,DD

  *I Q,SP=P1,O

- Change spool partition status

  *F Q,SP=spart,O= [NO | YES | SPARTN]

  Examples

  *F Q,SP=P1,O=P2
  *F Q,SP=P1,O=NO
  *F Q,DD=SPOOL4,SP=P3
  *F Q,SP=P1,O=P2

Command examples

You can use the **`*F Q`** command to assign or reassign a spool overflow partition or to reassign a spool data set from one partition to another. You can reassign any spool data set to any spool partition as long as the default partition has a minimum of one data set assigned to it and the number of available track groups in the default partition does not fall below the minimal condition established by your installation. (The changes remain in effect if you restart JES3 using a hot start. They do not remain in effect if you use a warm start.)

The **`*F Q`** command also allows you to reassign a job's spool data from one partition to another. You can do this for all jobs in a specific job class or for all jobs that run on a specific main. These changes might or might not remain in effect after a JES3 restart. Once data is written to a spool data set, the data itself does not move. If you want to use a specific spool partition for the output data from a particular job, it is not necessary to modify JES3 system parameters; use the //*MAIN control statement in the job's JCL to override the partition that JES3 would normally use to write the job's output data.

When you reassign data from one partition to another, keep in mind that the partition assignments for a particular job's spool data can overlap. While a job is running, JES3 uses the following priority scheme to choose the partitions it will use. For each portion of the job's data, JES3 uses the first partition in the list that is assigned for that part of the job's data.

16.5 Spool partitioning commands

- Partitions by MVS image

  *I G,SY1,SP

  *F G,SY1,SP,P3

- Partitions by job class

  *I C=A,SP

  *F C=A,SP=P3

Partitioning commands

Once data sets in a SYSOUT class are assigned to a partition, you cannot change the assignment without restarting JES3. Note that by using JES3 commands, initialization statements, and the //*MAIN control statement, you can assign data from any combination of jobs, classes of jobs, processors, and SYSOUT classes to a partition. You can use the **`*INQUIRY`** command to identify the spool partition assigned for a specific job, for all jobs in a specific job class, and for jobs that run on a specific main.

Using partitions

When submitting a job, the user can request that JES3 write the job's spool data to a specific spool partition. To do this, the user specifies the name of the spool partition on a //*MAIN JES3 control statement. This allows the user to override partition names specified on MAINPROC or CLASS statements. The user, however, cannot override partition names specified on SYSOUT statements. Instructions for coding the //*MAIN statement are in *z/OS MVS JCL User's Guide*.

Command examples

The visual refers to the modifying with a GMS command to specify to which spool partition JES3 writes the jobs' spool data when the jobs execute on processor SY1. The second modify specifies that all class A jobs should use spool partition P3.

16.6 Spool recovery

- Avoid coldstarts due to spool failures
  - Single track errors or large numbers
  - Provide method for suspending volume use
  - Prevent new allocations to volume
  - Replace spool volumes
- No RAS facilities for JCT and Checkpoint
  - Other techniques are possible

Spool volume recovery

Spool recovery procedures and operator commands are an important part of maintaining the JES3 system. When spool volumes errors occur, you should have procedures in place to provide recovery from any type of error.

The `*F Q` command allows you to control activity on a spool data set. For spool volume recovery, you can do the following:

- You can stop JES3 from allocating additional space on a specific spool data set and then restart space allocation processing at a later time. This action does not affect the jobs that already have data on this data set; the jobs continue to run in the normal manner.
- If necessary, you can place a spool data set and all jobs with spool data on the data set in hold status and release both the data set and the jobs at a later point in time.
- Another parameter allows you to place the data set in hold status and cancel all jobs with spool data on the data set. You then can release the data set from hold status and resume allocating space on the data set.

All these changes remain in effect when you restart JES3, using a hot or warm start.

16.7 Spool recovery

- Suspend allocation of track groups
  - *F Q,DD=spool1,DRAIN
    - Jobs only use allocated spool space
- Resume track group allocation
  - *F Q,DD=spool1,USE

Draining spool volumes

Use the **`*F Q`** command to prevent JES3 from allocating additional space on a spool data set or to allow JES3 to resume allocating space on a spool data set.

DRAIN Stops allocating space on the specified spool data set.

USE Resumes allocating space on the specified spool data set. Enter the USE parameter for a spool data set that was previously stopped using the DRAIN parameter.

For the DRAIN parameter, JES3 stops allocating space on the specified spool data set(s). If there are jobs in the system that currently have data on the data set(s), JES3 continues normal processing for these jobs and releases the space as each job completes.

JES3 issues message IAT8091:

```
IAT8091  ALLOCATION FROM SPOOL DATA SET ddname SUSPENDED
```

16.8 Spool recovery

- Holding jobs on volume with errors
  - *F Q,DD=spool1,HOLD
    - Drains volume
    - Jobs with space on volume - "spool hold"
    - Jobs continue executing
    - Jobs printing continue to use volume

Recovery using HOLD and RELEASE

The **`*F Q`** command can be used to HOLD or RELEASE a specific spool data set and to hold, release, or cancel all jobs that have data on the data set.

HOLD Holds the specified spool data set and holds further scheduling for all jobs that have data on the data set (places the data set and the jobs in spool hold status).

JES3 puts the specified spool data set(s) in spool hold status and does not allocate additional space on the data set(s). JES3 does not schedule any additional processing for jobs that have data on the specified spool data set(s). Processing that is in progress at the time you submit the command is allowed to complete.

JES3 issues the following messages:

```
IAT8091 ALLOCATION FROM SPOOL DATA SET ddname SUSPENDED
```

This message is issued for each job affected by the command.

```
IAT8083  JOB QUEUE FOR SPECIFIED EXTENTS HELD
```

RELEASE JES3 releases each of the specified spool data set(s) from spool hold status and resumes allocating space on the data set(s).

For each job that has data on the specified data set(s), JES3 releases the job from spool hold status as long as the job does not also have data on other spool data sets that are being held.If other data sets are being held, JES3 again issues message IAT8091.

All the job's spool data sets must be released before the job can be released from spool hold status.)

JES3 takes the following action on jobs that it releases from spool hold status:

- If you originally used the HOLD parameter to put the jobs in spool hold status, each job resumes processing at the point where it left off.
- If you originally used the STOP parameter to put the jobs into spool hold status.

16.9 Spool recovery

- Cancelling jobs on volume with errors
  - *F Q,DD=spool1,STOP
    - Drains volume
    - Jobs with space on volume - "spool hold"
    - Active jobs with data on volume - cancelled
    - Jobs are re-queued after "spool hold" release
  - *F Q,DD=spool1,RELEASE

Canceling jobs on volumes

The **`*F Q`** command can be used to HOLD or RELEASE a specific spool data set and to hold, release, or cancel all jobs that have data on the data set.

STOP Holds the specified spool data set and holds further scheduling for all jobs that have data on the data set if the jobs are not currently active on a main (places the data set and the jobs in spool hold status). This parameter cancels all jobs with data in the data set if the jobs are currently active on a main.

RELEASE Releases the specified spool data set and all jobs that have data on the spool data set for further scheduling (releases the data set and the jobs from spool hold status).

STOP option

JES3 issues the following messages for the **`STOP`** command:

```
IAT8091  ALLOCATION FROM SPOOL DATA SET ddname SUSPENDED
```

This message is issued for each job put into spool hold status as a result of this command.

```
IAT8083  JOB QUEUE FOR SPECIFIED EXTENTS STOPPED
```

JES3 puts the specified spool data set(s) into spool hold status and does not allocate additional space on the data set(s).

JES3 puts each job that has data on the specified spool data set(s) and is not active on a main into spool hold status and does not schedule any additional activity for the job. If JES3 is currently performing any activity on behalf of the job, the activity terminates.

For each job with data on the specified spool data set(s) that is currently active on a main, JES3 requests that MVS cancel the job. If MVS cannot cancel the job, message IEE841I is issued. To cancel the job, you must enter the MVS FORCE command.

Release option

JES3 issues the following messages:

```
IAT8091  ALLOCATION FROM SPOOL DATA SET ddname RESUMED
```

This message is issued once for each job affected by the command.

```
IAT8083  JOB QUEUE FOR SPECIFIED EXTENTS RELEASED
```

JES3 releases each of the specified spool data set(s) from spool hold status and resumes allocating space on the data set(s).

For each job that has data on the specified data set(s), JES3 releases the job from spool hold status as long as the job does not also have data on other spool data sets that are being held. If other data sets are being held, JES3 again issues message IAT8091. All the job's spool data sets must be released before the job can be released from spool hold status.

16.10 Spool recovery

- Cancelling jobs on volume with errors
  - *F Q,DD=spool1,CANCEL
    - Jobs with space on volume - CANCELed
    - Active jobs with data on volume - CANCELed
    - Jobs are PURGEd from system
    - All output for jobs purged

Using the CANCEL option for recovery

The **`*F Q`** command can be used to HOLD or RELEASE a specific spool data set and to hold, release, or cancel all jobs that have data on the data set.

CANCEL Holds the specified spool data set (places the data set in spool hold status) and cancels all jobs that have data on the data set.

CANCEL option

JES3 issues the following messages:

```
IAT8091  ALLOCATION FROM SPOOL DATA SET ddname SUSPENDED
```

This message is issued for each job affected by the command.

```
IAT8083  JOB QUEUE FOR SPECIFIED EXTENTS CANCELED
```

JES3 puts the specified spool data set(s) in spool hold status and does not allocate any additional space on the data set(s).

JES3 cancels each job that has data on the specified spool data set(s). It also cancels any output from the job that is ready to be printed.

16.11 Restart without a spool volume

- Volume allocated via JES3 PROC
  - Remove DD and hotstart
- Volume allocated via DYNALLOC
  - Dismount volume or Vary offline
    - IAT3341 UNABLE TO OPEN ..... continue or cancel
    - Allocation fails but JES3 continues
- Initialization messages for jobs - IAT3387
  - Replies: hold - hold,all - cancel - cancel,all

Restart without a spool volume

To allocate a spool data set, include a DD statement for the data set in the JES3 start procedure. To dynamically allocate the spool data set, omit the DD statement and include a DYNALLOC statement for the data set the JES3 initialization stream. Dynamic allocation provides an easier method for changing your spool configuration than allocating the spool data sets through the JES3 start procedure with DD statements.

When you restart JES3 on a global processor with either a warm start or a hot start, you can remove a spool data set from the system or reinstate a spool data set that was previously removed from the system.

The ability to remove and reinstate a spool data set is useful when I/O errors occur on the volume containing the spool data set and the error affects JES3 system functions. If the volume is repairable, you can remove the volume for repairs and return it to the system when repairs are complete without jeopardizing all the jobs in the system; only jobs with data on the spool data set on the failed volume are affected.

To restart without a spool volume, use the techniques in the visual, as follows:

16.12 Replace spool volume at restart

- Return same volume removed
  - Hotstart - replace DD or Vary online volume
  - IAT3342 - held jobs are released
- Replace old volume with new volume
  - Can be same or different volume
  - Jobs using old volume are canceled

Replace spool volume during hot start

You can use warm start initialization with the replace function to restart JES3 and replace one or more existing spool data sets, including one of the checkpoint data sets, with new spool data sets. By replacing existing spool data sets with larger spool data sets, you can increase spool capacity. Warm start initialization to replace a spool data set can be used after an orderly shutdown or after JES3 ends abnormally due to an equipment or a system failure.

The effects of warm start processing with the replace function are the same as normal warm start processing except that the replace function also removes one or more spool data sets from the system, replaces them with other, possibly larger, data sets, and cancels each job in the system that has data on the replaced data sets. If the replaced data set contained single track table (STT) records, JES3 might have lost information such as the status of the devices.

To replace the same or a different volume with the same name, a hotstart can be done.

16.13 Spool operator commands

- Status of spool volumes
  - *I Q,DD=spool1
    - A-available D-draining H- spool hold U-missing
- How much space on volume
  - *I Q,DD=spool1,S
- Spool volumes used by a job
  - *I J=500,SD

Operator commands for spool

Use the **`*I Q,DD=spvol`** command to display:

- The names of the spool data sets assigned to a spool partition and whether the partition is defined as the default partition, the initialization partition, or an overflow partition
- The size of the partition and the amount of space currently available
- The status of a partition, and users of the largest spool space
- The status of a spool data set and the name of the spool partition that the data set belongs to
- All the defective tracks currently known to JES3
- The amount of space available on all the JES3 spool data sets in the complex
- A list of jobs of a particular category in the JES3 job queue

16.14 Spool operator commands

- Display jobs with data on volume
  - *X DISPLAY,DD=spool1
- List jobs with data and track groups
  - *X DISPLAY,DD=spool1,LIST

Spool space using DISPLAY DSP

The DISPLAY DSP can be used to display the jobs using a spool volume and list the jobs and their track groups.

The **`*I Q`** command allows you to display the status of any spool data set as well as the size of the data set and the amount of space currently available. Also, for a particular job, you can list the names of either all the spool data sets containing data for the job or only those data sets that are being held.

If you need to know which jobs have space allocated on a particular spool data set and the amount of space allocated to each job, you can use the DISPLAY DSP.

16.15 BADTRACK statements

- BADTRACK table in storage
  - Created from BADTRACK statements
  - BADTRACK,DDNAME=SPOOL1,CYL=00CA,TRK=0001
  - Record contains:
    - Type of error encountered - Time/Date added
    - Location of error track - X.G
- Operator inquiry command - *I Q,BT

IAT8539 SPOOL1 BADTRACK ON VOL001 CYL=00CA TRK=0001 OCCURRED

AT INITIALIZATION
DURING FORMATTING
08/25/96 10.14.00

BADTRACK statements

A BADTRACK table and search algorithm is in storage. The table is called, the "track group bypass table":

- Entries may be added to the table dynamically.
- A BADTRACK checkpoint record is created.
- Operator commands to inquire on tracks in error

The BADTRACK table is constructed during JES3 initialization based on the BADTRACK statements in the initialization stream. There is an entry in the table for each spool extent. The entries in the table describe a track group where the error or bad track occurred.

The JES3 checkpoint contains a record describing the tracks that are in error. When the checkpoint record exists, the BADTRACK table can be constructed by converting the tracks in error to the appropriate track group.

When an I/O error is encountered on a JES3 spool extent, an entry to the BADTRACK table is created dynamically. Entries are added only under the following conditions:

- All permanent write failures that are device related.
- Permanent read errors are ignored on the assumption that when the track group is reassigned and the track is really bad, a write error will occur.

Entries are added to this checkpoint record at initialization time when BADTRACK cards are encountered in the initialization stream. This record is also structured to allow dynamic addition of a new entry when bad tracks are encountered during the running of the system.

Operator command for BADTRACKs

The operator may list the known bad tracks in the system. The information displayed by this command is kept in the JES3 checkpoint record.

```
*I Q,BT

IAT8539 SPOOL1 BADTRACK ON VOL001 CYL=00CA TRK=0001 OCCURRED
```

Where the error occurred:

```
AT INITIALIZATION
DURING FORMATTING
yy/mm/dd hh/mm/ss
```

The text indicates at what time the error occurred.

17

JES3 Monitoring Facility (JMF)

System programmers can use the JES3 Monitoring Facility (JMF) to obtain statistical data of the system.

JES3 tuning and performance diagnosis requires a great deal of JES3 knowledge. JMF can expand the information that is available to the person attempting either of these tasks. It can also provide a great deal of interesting information, some of which has value and some of which is merely information.

You should run JMF regularly and when the system is running normally. When JES3 is running poorly you will have the "historical perspective" necessary to identify deviations from when JES3 is running normally. You must have a base from which you can assess changes, even if the base moves as the configuration and workload evolves over time.

There is workload and capacity information that can be derived from JMF reports:

- Estimates of the total number of jobs in the system and the distribution of this work (for example, CI, MAIN, and OUTSERV)
- Changes in workload (group and initiator use counts)
- Demand for JES3 managed resources (for example, tape drives and spool space)

17.1 JMF analysis

- Things to look for when system runs poorly
  - A clear description of observed changes in behavior
  - When the behavior changed
  - Known changes that have occurred in the system prior to this time, including:
    - Configuration changes (DASD movement, catalog movement)
    - Operational changes (changes in message traffic to different consoles)
    - Known workload changes (additional TSO users)
    - Maintenance or software changes

JMF analysis

When there is classic symptoms of a JES3 problem, the question is:

- How does one determine what change occurred in the operating environment and how does one associate the external change with changes in JES3 internal processing so that the observed "abnormal" JES3 behavior can be explained? What is the first thing that you should do? You need to collect several pieces of information. These should include:
  - A crisp, clear description of the observed changes in behavior
  - When the behavior changed
  - Known changes that have occurred in the system prior to this time including:
    - Configuration changes (for example, DASD movement, Catalog movement)
    - Operational changes (for example, changes in message traffic to different consoles)
    - Known workload changes (for example, additional TSO/E users)

If you are fortunate, you will be able to find the change and then use your knowledge of JES3. Most likely, you will get into the situation where he change is not easily identified, or there were several changes that occurred at the time, or "we didn't change a thing". You are going to have to examine the behavior of the system to find the change that no one remembers.

You probably aren't going to dive into JMF reports first; but after looking through the normal RMF data and other reports, you have to investigate the JMF data. The JMF data must be examined in relationship to what JMF has reported in the past when the system was healthy. Without reference points, diagnosing the problems is very difficult.

17.2 JMF analysis

- Run JMF regularly to get a feel for the normal range of values reported.
- When the system is running poorly, run JMF and look for values that are different from the regular runs.
- Use values from the regular runs when I the system has good performance for comparison with values obtained when you feel the system is running poorly
- Following is a short list of performance problem symptoms:
  - Consoles aren't responsive
  - Inquiry commands are not coming back promptly
  - TSO logons are backed up
  - Output processing is slow

JMF analysis

Finding problems is essentially an exercise in observation. When JES3 performance characteristics change, something in the environment has changed.

So now you have a purported JES3 performance problem. At least all the normal signs are there:

- Consoles aren't responsive
- Inquiry commands are not coming back promptly
- TSO/E logons are backed-up
- Output processing is slow

17.3 JMF analysis

- To get the results you expect from JMF, you will need to determine:
  - The number of reports and samples you need to ensure the results of your analysis are correct
  - The number of functional control tables (FCTs) you need to monitor to either obtain information for the appropriate FCT or give enough information for analysis

JMF analysis

JMF generates one or more reports depending on the values specified in the INTERVAL and TIME parameters on the modify command. To determine the number of reports that will be generated, use the following formula:

Number of reports = TIME/INTERVAL

If a remainder exists, JMF will round up. For example, if a **`*X JMF,TIME=70,INTERVAL=15`** command is entered, the number of reports that JMF generates equals 70 divided by 15; five reports will be generated. The final report is generated even though an entire interval has not been completed.

JMF generates one or more samples for a report depending on the values specified in the INTERVAL and CYCLE parameters on the modify command. IBM suggests generating at least 1000 samples for each report. To determine values to specify on the INTERVAL and CYCLE parameters, use the following formula:

Number of samples = INTERVAL*60/CYCLE

For example, if 1000 samples and 4 reports should be generated in 60 minutes, the operator should enter a **`*X JMF,CYCLE=.9,TIME=60,INTERVAL=15`** command to generate the results.

17.4 Using JMF

- *X JMF,CYCLE=2,INTERVAL=15,TIME=15
  - FCT=50,AWAIT=5,WAIT=100
  - SDM=Y
  - SPOT=ALL,NAME=NUC,WIDTH=nn,HFCT=ALL
  - JOB=50,JSTAT=5
  - SSI=Y
  - WTR=Y
  - OUT=devname
  - DEBUG=N
  - INTRDR=Y
  - SMF=N
  - DESTQ=Y

Using JMF

JMF should be run under normal conditions. A good sample size is 1000 samples in an interval. Sampling once a second for an hour should be sufficient for normal situations.

The default number of FCTs is 250 which may not be sufficient in the following cases:

- If you have a great amount of BSC NJE activity
- When JMF is running with a long interval

The default value for JOB will let you track the first 50 jobs in the RESQUEUE and report scheduling information on them. Some reports are jobs in which you are interested. There are some jobs dependent on this option being in effect, so have JMF track one job.

JMF sampling activity and output can be tailored to your needs by specifying the correct start-up options for your installation.

17.5 JMF parameters

- WLM=Y | N
  - Specifies whether (Y) or not (N) to report Workload Management (WLM) information (such as the backlog of jobs in each service class). WLM=Y is not valid on a local processor. Y is the default on the global, and N is the default on the local
- WTR=Y|modulename|N
  - Specifies that JES3 formats a hardcopy report (Y); that the installation specify its own module to format a hardcopy report (modulename); that no hardcopy report is generated (N). This parameter does not apply to the local

JMF parameters

A new option, WLM, on the `*X JMF` command has been added to allow you to specify whether or not you want the WLM information reported. WLM=Y is the default on the global processor. WLM=N is the default on a local processor. If you attempt to specify WLM=Y on a local, the command will be rejected.

Only jobs in WLM-managed groups are included in the report.

When WLM information is requested, the following information will be reported for each service class that is detected during the sampling interval. Information will be reported only for those jobs in WLM managed groups, as follows:

- The number of jobs waiting to be scheduled for main service.
- The number of jobs is MDS processing.
- The number of jobs waiting to be selected by an initiator (GMS Select).
- The number of jobs that were eligible to execute somewhere in the SYSPLEX.

Specify WTR=Y to get a report

17.6 JMF reports

- JMF generates one or more reports depending on the values specified in the INTERVAL and TIME parameters o
  - To determine the number of reports that will be generated, use the following formula:
    - Number of reports = TIME/INTERVAL
  - If a remainder exists, JMF will round up. For example, if a *X JMF,TIME=70,INTERVAL=15 command is entered
    - The number of reports that JMF generates equals 70 divided by 15; five reports will be generated
    - The final report is generated even though an entire interval has not been completed

JMF reports

The JES3 Monitoring Facility (JMF) is a JES3 diagnostic and performance utility that you can use to help monitor, diagnose, and tune your JES3 complex. You can use JES3 operator commands to produce JMF reports that contain data about:

- JES3 spool data management
- JES3 CPU/storage
- JES3 functions
- JES3 device scheduling
- JES3 job throughput

Since JMF measurements vary from installation to installation, you must first establish your own baseline measurements by running JMF over a period of time under normal operating conditions. See *z/OS JES3 Diagnosis* for sample JMF reports and the commands that you need to generate those reports.

17.7 JMF reports

- There are seven major reports generated by JMF:
  - System report
  - FCT and AWAIT report
  - Spool data management report
  - JES3 control block utilization report
  - Job analysis
  - Hot spot analysis report
  - JES3 function report

JMF reports

Only the system report and the JES3 control block report are created with every run of JMF. The remaining reports are optional and can be eliminated. The reports often have overlapping and complimentary information. You will normally have to use two or more of the reports to analyze what the system is doing.

System report

The system report contains information about IATNUC, IATAUX tasks, and CPU utilization. It also provides information about JES3 storage requirements and configuration data.

FCT and AWAIT report

The FCT and AWAIT report contains information about the activity and location of each FCT at the time it is AWAITing. The spool data management report contains information about the spool rate, space utilization, etc.

Note: You will find that this report does not match the resource management facility (RMF) reports. The spool data management report is the most accurate.

Control block utilization report

The JES3 control block utilization report contains information about the control block pools (for example, RQ's, JSAM buffers, staging areas, and JQE's) and performance information about the JCT data space.

Job analysis report

The job analysis report contains information about job flow through the JES3 scheduler elements, JSS work-to-do queue, allocated JES3 managed devices, etc.

Hot spot analysis report

The hot spot analysis report contains information regarding the utilization of modules and the frequency of use in the JES3 address space.

Function report

The JES3 function report contains information about internal reader activity, Subsystem Interface (SSI) response time, and JES3 DESTQ lengths.

17.8 JMF overhead

```
JMF OVERHEAD
     MINIMUM =    .000704 SEC.
     MAXIMUM =    .105328 SEC.
     AVERAGE =    .001472 SEC.      .73 %
OF JMF CYCLE TIME
MVS OVERHEAD
     MINIMUM =    .000256 SEC.
     MAXIMUM =    .167264 SEC.
     AVERAGE =    .000944 SEC.      .47 %
OF JMF CYCLE TIME
```

JMF overhead

JMF will also report on the number of tasks simultaneously active in the ES3 address space. You will also get a chart showing the percent active for each task. The only tasks that you have control over are the C/I subtasks. JES3's performance is impacted by having a great number of subtasks (there is no TCB ready queue similar to the ASCB ready queue). You should control the number of these tasks. You should consider moving the C/I function to a C/I FSS address space. This will have the added benefit of reducing local lock contention in the JES3 address space.

17.9 IATNUC posted

- The IATNUC POSTED - ACTIVE number is a good approximation of the CPU utilization in the JES3 local address space. If your installation's workload produces large amounts of printed output, you may see a significant amount of CPU utilization for the IATAUX task also

```
IATNUC POSTED - ACTIVE                13.69 %
IATNUC POSTED - NOT ACTIVE             7.37 %
IATNUC NOT POSTED                     72.80 %
IATNUC IN NONSTANDARD WAIT              .04 %
IATNUC SUSPENDED - LOCAL LOCK REQ      5.67%
IATNUC SUSPENDED - OTHER                .40 %
```

IATNUC posted

The example shows the IATNUC task is using the CPU 14% of the time. However, the task is considered to be 20% busy because 6% of the time the task is suspended due to unavailability of the local lock. The task busy time is the sum of the:

- ACTIVE,
- IN NONSTANDARD WAIT
- SUSPENDED - LOCAL LOCK REQ
- SUSPENDED - OTHER

A rule of thumb is to keep the IATNUC CPU utilization below 60%. Going any higher may lead to performance degradation. A few things can be done to reduce CPU utilization. If the global system has multiple engines, use the writer multi-tasking feature. Also, using C/I or writer FSS will off-load some of the CPU processing to other address spaces.

From a performance point of view, the starting and stopping of dynamic writers introduces additional CPU utilization for the IATNUC task. On the other hand, there is overhead associated with using a large number of idle hot writers. This overhead shows up in high multi-function monitor CPU consumption. There are performance and operational trade-offs to be made when making these decisions.

17.10 System report

- System Report consists of the:
  - General Information Section
  - Working Set Section
  - JES3 Subtask Section
  - Global Processor Description Section
- FCT and AWAIT Report consists of the:
  - FCT and AWAIT Report Section
  - DSP Analysis Report
  - FCT and AWAIT Highlight Report Section
  - DMJA FCT Summary Section
  - JES3 WAIT Analysis Section

System report

System Report consists of the:

- General Information Section
- Working Set Section
- JES3 Subtask Section
- Global Processor Description Section

You can use these sections to obtain information on the CPU utilization of the nucleus and auxiliary task. It also describes JES3's real storage requirements and configuration information for your installation.

FCT and AWAIT report

FCT and AWAIT Report consists of the:

- FCT and AWAIT Report Section
- DSP Analysis Report
- FCT and AWAIT Highlight Report Section
- DMJA FCT Summary Section
- JES3 WAIT Analysis Section

You can use these sections to obtain information on the workload distribution in your installation.

JMF can report on its use of FCTs and use an FCT report to identify how the work is distributed across JES3 functions. An FCT report contains information on the first 250 FCTs in the FCT chain. If you require information for a specific FCT, you will need to determine the number of FCTS on the FCT chain that precede the desired FCT. The last entry in the report will be the requested FCT. Use the following formula to approximate the number of FCTs in your installation.

17.11 Reports .....

- Spool data management report
  - Spool Data Set Description Section
  - Spool Partition Description Section
  - Spool Space Utilization Snapshot Section
  - Single Track Table Space Allocation Section
  - Spool I/O Activity Section
  - Buffer Chaining by Spool Data Set Section
  - SDM Exceptional Conditions Section

Spool data management

Spool Data Management Report consists of the:

- SDM Parameters Section
- Spool Data Set Description Section
- Spool Partition Description Section
- Spool Space Utilization Snapshot Section
- Single Track Table Space Allocation Section
- Spool I/O Activity Section
- Buffer Chaining by Spool Data Set Section
- SDM Exceptional Conditions Section

You can use these sections to determine if JES3 is accessing your installation's spool environment efficiently.

18

System Display and Search Facility (SDSF) in the JES3 environment

This chapter is for use with z/OS System Display and Search Facility (SDSF) in the JES3 environment. It is intended primarily for system programmers and operators, and assumes you are familiar with the z/OS operating system, including JES3. It contains information about migration, customization, security, operation, maintenance and problem determination, including explanations of SDSF messages.

18.1 JES3 SDSF

This chapter contains information about the z/OS Version 1 Release 13 System Display and Search Facility (SDSF) in the JES3 environment. It assumes you are familiar with the z/OS operating system, including JES.

SDSF (Program Number 5694-A01), a feature of IBM mainframes running z/OS, enables users and administrators to view and control various aspects of mainframes' operation. These include jobs in execution, job output, status of Unix System Services processes, system information, workload scheduling, and log files.

SDSF displays data on panels. Commands and actions that you enter on the panels let you monitor and control jobs and system resources. The SDSF Primary Option Menu lists the panels that you are authorized to use.

The objects, displayed on the SDSF panels, are initiators, printers and punches, jobs, SYSIN/SYSOUT data sets, and so on. Information for the objects is extracted using formal JES3 or MVS programming interfaces, for example subsystem interface (SSI) calls. Actions against objects are also invoked through formal programming interfaces or operator commands. Most actions generate MVS or JES commands. In a JES3 environment, the MVS system authorization facility (SAF) is required for SDSF security. When a request is made to access a resource, and the profile that protects the resource is not defined, or the associated class is not active, SDSF fails the request. All SAF profiles must be defined and activated in all of the classes that are used for SDSF security.

Most SDSF panels display information in a tabular format. You can scroll the information up, down, right, and left.

History

SDSF was originally known as SPOOL Display and Search Facility when it was a field-developed program offering. The word SPOOL was changed to System when it became a program product in the late 1980s. Starting with z/OS Release 9 SDSF also supports a REXX interface, allowing batch program facilities to use SDSF. The REXX support implementation presents data through stem variables containing SDSF-originated information.

Prior to z/OS Version 1 Release 10 SDSF supported only JES2 environments. z/OS Version 1 Release 10 SDSF included support for the JES3 environment. The JES3 job-related Display Active Users (DA), Input Queue (I) and Status (ST) panels were available for JES3 displays. Other SDSF panels that do not depend on JES were also available in the JES3 environment.

z/OS Version 1 Release 11 expanded function in the JES3 environment to include the SYSLOG, Job Class (JC), Spool Volumes (SP) and JESPLEX (JP) displays. Support was also added to display and modify output descriptors for JES3 jobs through the Output Descriptor (OD) panel and the Job Data Set (JDS) panel. JES3 browse of a job that is running on a system other than the one you are logged on to, shows data from buffers not yet written to the spool.

z/OS Version 1 Release 13 expands SDSF function in the JES3 environment to include Initiator (INIT), Job 0 (J0), Line (LI), Node (NO), Punch (PUN), Reader (RDR), Held Output Queue (H) and Output Queue (O) panels for JES3 objects. Network Connect (NC) and Network Server (NS) panels show information about JES job networking.

18.2 SDSF functions

SDSF is a program that runs under z/OS TSO/E and uses Interactive System Productivity Facility (ISPF) panels to display JES and MVS data. Some of the functions described in this book are specific to JES3. With SDSF, you can do the following:

- Display immediate, up-to-date information about the jobs submitted to JES3 for processing, including:
  - Jobs on the JES3 queues - Status (all queues), Input Queue, Output Queue, Held Output and J0 (JES3 job zero)
  - Job status of a specific job, including the job's priority and input class, the time and date the job was entered in the system, and the time and date the system began processing the job
  - System information about active jobs
  - Spool data sets for a specific job
  - Output from a job
- Monitor and control jobs, output, and resources in a JES3 complex without using JES3-specific command syntax.
- Display and control z/OS UNIX System Services processes.
- Manage system resources, such as main processors in the JES3 complex, job classes, and WLM enclaves.
- Control JES3 managed printers, punches and readers.
- Enter MVS and JES3 system commands from any TSO/E terminal.
- View the system log (SYSLOG), operations log (OPERLOG), or user log (ULOG) online and search for specific information, which can reduce problem management time and eliminate the need for a printed copy of the log.
- View input data sets of jobs that are being processed or waiting to be processed.
- View output data sets online and purge them, which can reduce the system print load.
- View and control BSC NJE lines.
- View and control server-type networking devices.
- View and control Network Job Entry (NJE) connections to an adjacent node.
- Monitor and control the IBM Health Checker for z/OS checks.
- Get online information: help for panels, commands, and messages; an interactive tutorial for ISPF users; and online documentation through BookManager®.

SDSF may be invoked on either a local or global processor running JES3. When SDSF is invoked on a local processor, the global processor must also be at the z/OS V1R10 JES3 or later level.

SDSF information

Information about SDSF and z/OS is available on the Internet. If it is supported by your 3270 emulator, you can click a web address to launch a web browser.

- SDSF home page: usage tips, presentations, as well as a wizard to help you enable the sysplex support can be found at:

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

- The latest edition of *z/OS SDSF Operation and Customization*, SA22-7670 is available at:

  `http://publibz.boulder.ibm.com/epubs/pdf/isf4cs90.pdf`

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

18.3 USING JES3 SDSF

When SDSF is running under TSO ISPF, ISPF provides an SDSF invocation under Option 13.14. To add SDSF to your own customized ISPF selection menu, insert the following text:

```
S SDSF System Display and Search Facility
```

The ZSEL statement in the PROC section should be updated to invoke SDSF with the ISFISP entry point:

```
&ZSEL = TRANS(TRUNC (&ZCMD,'.')
 . . . .
 S,'PANEL(ISFSDOP2) NEWAPPL(ISF) SCRNAME(SDSF)'
 ' ' ' '
   ,
 *,'?' )
IF (&ZCMD = 'S')
  &ZSEL = 'PGM(ISFISP) NOCHECK NEWAPPL(ISF) SCRNAME(SDSF)'
IF (&ZCMD = 'S.')
  &ZSEL = 'PGM(ISFISP) NOCHECK NEWAPPL(ISF) SCRNAME(SDSF)'
```

When you invoke SDSF as an ISPF dialog using the ISFISP entry point, you can specify parameters to specify an initial panel and other values. SDSF may be interactively invoked with TSO commands `SDSF` or `ISF` outside ISPF.

SDSF panel structure in the JES3 environment

Figure 18-1 is a view of the full SDSF panel structure of the JES3 environment.

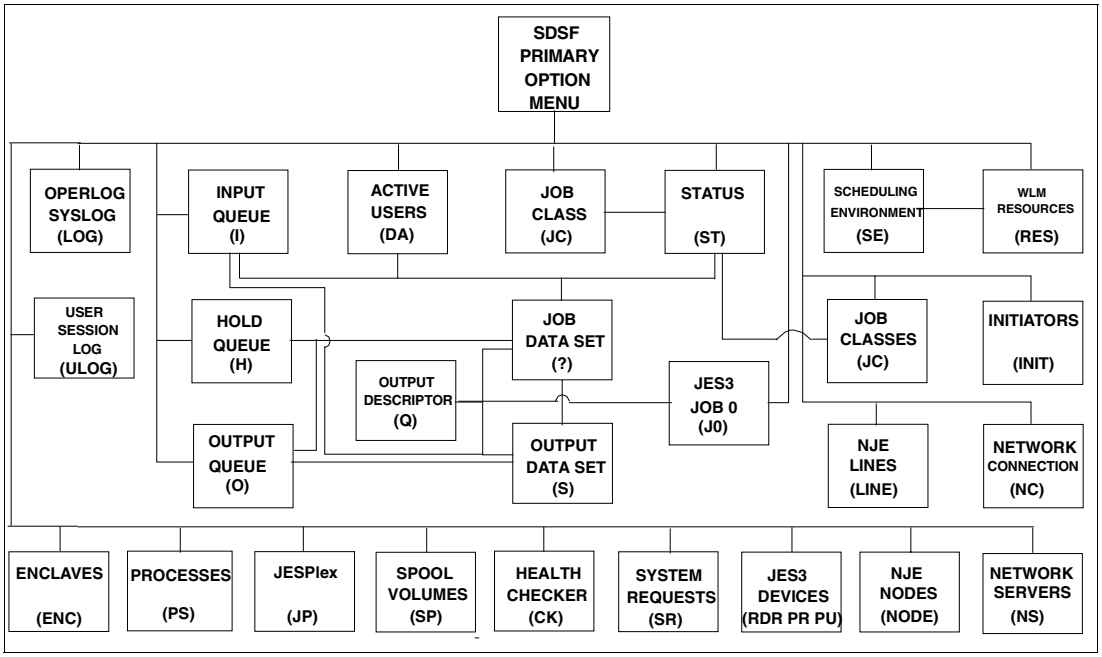


Figure 18-1 Full SDSF panel structure in the JES3 environment

JES3 SDSF primary menu options

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
HQX7780 -----------------  SDSF PRIMARY OPTION MENU
--------------------------
COMMAND INPUT ===>                                          SCROLL ===>
HALF

DA    Active users                     INIT  Initiators
I     Input queue                      PR    Printers
O     Output queue                     PUN   Punches
H     Held output queue                RDR   Readers
ST    Status of jobs                   LINE  Lines
J0    Job zero                         NODE  Nodes
                                       SP    Spool volumes
LOG   System log                       NS    Network servers
SR    System requests                  NC    Network connections
JP    Members in the JESPlex
JC    Job classes                      CK    Health checker
SE    Scheduling environments
RES   WLM resources                    ULOG  User session log
ENC   Enclaves
PS    Processes

END   Exit SDSF
```

Figure 18-2 SDSF primary option menu - JES3

Figure 18-2 shows the SDSF functions available on the authorized primary option menu panel in a JES3 environment. The SDSF commands for the functions are as follows:

DA The **Display Active Users (DA)** selection allows authorized users to display information about jobs, users, and started tasks that are active in the sysplex. It also shows system data, such as CPU usage and paging information. In a JES3 environment, the DA selection also requires RMF.

I The **Input Queue (I)** selection allows authorized users to display information about jobs that are on the JES input queue or that are executing.

O The **Output Queue (O)** selection displays information about SYSOUT data sets for jobs, started tasks, and TSO users on any *nonheld* JES output queue.

H The **Held Output (H)** selection shows the user information about SYSOUT data sets for jobs, started tasks, and TSO users on any *held* JES output queues.

ST The **Status (ST)** selection allows authorized users to display information about jobs, started tasks, and TSO users on the JES queues.

J0 The **Job zero (J0)** selection displays information about SYSOUT data sets for a JES3 job 0.

LOG O The **OPERLOG (LOG [O])** selection allows authorized users to display a sysplex-wide system message log, which contains console messages, operator commands, and responses for the sysplex.

LOG S The **SYSLOG (LOG S)** selection allows authorized users to display the system log. The SYSLOG is a data set residing in the primary job entry subsystem's spool space. If JES3 DLOG is active on the global, system log entries are for the whole

JES3 complex. The DLOG message prefix (IATYCNS TYPE=DLOG) is different from the MVS hardcopy log prefix (IHAHCLOG). The JES3 `*F 0` command enables or disables the DLOG.

SR The **System Requests (SR)** selection allows authorized users to display outstanding operator replies (WTORs) and messages retained by the Action Message Retention Facility (AMRF).

JP The **JESPLEX (JP)** selection allows authorized users to display and control the main processors in a JES3 JESPLEX.

JC The **Job Class (JC)** selection allows authorized users to display and control the job classes defined to JES. Both JES and WLM managed classes are shown.

SE The **Scheduling Environment (SE)** selection allows authorized users to display the sysplex wide scheduling environments. A scheduling environment. is a list of resource names along with their required states. If an MVS image satisfies all of the requirements in the scheduling environment associated with a given unit of work, then that unit of work can be assigned to that MVS image. If any of the requirements are not satisfied, then that unit of work cannot be assigned to that MVS image.

RES The **Resource (RES)** selection allows authorized users to display WLM resources. To display resources in the sysplex, access the panel with the RES command. To display resources for a scheduling environment, access the panel with the `R` action character from the SE panel. When a resource is used as part of a scheduling environment, the resource is an abstract element that can represent an actual physical entity (such as a peripheral device), or an intangible quality (such as a certain time of day). A resource is listed in a scheduling environment along with a required state of ON or OFF. If the corresponding resource state on a given system matches the required state, then the requirement is satisfied for that resource.

ENC The **Enclaves (ENC)** selection allows authorized users to display information about WLM enclaves. An enclave is an anchor for a transaction that can be spread across multiple dispatchable units in multiple address spaces. These multiple address spaces can even span across multiple systems in a parallel sysplex. The value of using an enclave to represent a transaction is that the resources used to process the transaction can be accounted to the transaction itself, rather than to the address space or spaces that the transaction runs in. In addition, you can assign a performance goal to the enclave, which means that as a transaction consumes system resources, it can switch periods to run with a new goal. Any number of tasks and SRBs can be grouped in an enclave.

PS The **Processes (PS)** selection allows authorized users to display information about z/OS UNIX System Services processes. A process is a program or command that is actually running the computer. It consists of a loaded version of the executable file, its data, its stack, and its kernel data structures that represent the process's state within a multitasking environment. The executable file contains the machine instructions (and any calls to shared objects) that will be executed by the hardware. A process can contain multiple threads of execution. A process is created via a fork() system call and ends using an exit() system call. Between fork and exit, the process is known to the system by a unique process identifier (pid).

INIT The **Initiators (INIT)** selection displays information about JES initiators that are defined for the JES3 job class groups. The display shows both mode JES and WLM initiators.

PR The **Printers (PR)** selection displays information about JES printers.

PU The **Punches (PU)** selection displays information about JES punches.

RDR The **Readers (RDR)** selection displays information about JES readers.

LINE The **Lines (LINE)** selection displays information about JES lines and their associated transmitters and receivers.

NODE The **Nodes (NODE)** selection displays information about JES NJE nodes.

SP The **Spool volumes (SP)** selection allows authorized users to display information about JES spool volumes.

NS The **Network servers (NS)** selection displays information about JES server-type networking devices.

NC The **Network connections (NC)** selection displays information about JES server-type networking devices.

CK The **Health Checker (CK)** selection displays information from IBM Health Checker for z/OS.

ULOG The **User Session Log (ULOG)** selection allows authorized users to display the MVS and JES commands and responses issued during the user's session, including commands generated by SDSF and SAF. SDSF deletes the user session log when an SDSF session is ended or when the ULOG CLOSE command is issued. SDSF uses MVS console services to acquire an extended console that is used to issue commands and receive responses.

/ The slash **(/)** system command allows system commands to be issued on the `COMMAND INPUT` line.

Only those SDSF panel commands (such as DA, I, and O) for which the user is authorized are displayed on the SDSF Primary Option Menu.

JES2 SDSF primary panel

Figure 18-3 on page 850 shows the ISPF display of the primary option menu in a JES2 environment for a user with full authority.

The SDSF support in the JES2 environment includes some functions that are not available in the JES3 environment. These are:

- The **Multi-Access Spool (MAS)** selection allows authorized users to display and control the members of a JES2 MAS.

  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.

  The analogous JES3 JESPlex panel simplifies the display and control of members in a JES3 JESPlex. The JES2 MAS panels and JES3 JP panels share a single field list.

- The **Spool Offload (SO) selection** displays information about JES2 spool offloaders and their associated transmitters and receivers.

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

- The **ResourceMonitor (RM)** selection displays information about critical JES2 resources such as JOEs (Job Output Element), JQEs (JES2 Job Queue Element) and BERTs (HASP Block Extension Reuse Table).

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
HQX7780 -----------------  SDSF PRIMARY OPTION MENU
---------------------------
COMMAND INPUT ===>                                            SCROLL ===>
PAGE

DA    Active users                     INIT  Initiators
I     Input queue                      PR    Printers
O     Output queue                     PUN   Punches
H     Held output queue                RDR   Readers
ST    Status of jobs                   LINE  Lines
                                       NODE  Nodes
LOG   System log                       SO    Spool offload
SR    System requests                  SP    Spool volumes
MAS   Members in the MAS               NS    Network servers
JC    Job classes                      NC    Network connections
SE    Scheduling environments
RES   WLM resources                    RM    Resource monitor
ENC   Enclaves                         CK    Health checker
PS    Processes
                                       ULOG  User session log
END   Exit SDSF
```

Figure 18-3 SDSF primary option menu - JES2

JES3 monitoring

The JES3 MONITOR DSP monitors a resource or queue based on information you specify. JES3 starts the MONITOR DSP and monitors various queues and resources automatically.

The monitor DSP makes it possible to monitor how long a job or FCT has been waiting for a specific JES3 function or resource. For example, if you want to know when a job has been waiting for a CI DSP for more than five minutes, you can set the monitor DSP to issue a message when five minutes have elapsed.

The JES3 monitor DSP runs as an FCT under the JES3 nucleus task and monitors unavailable JES3 resources. A JES3 resource is anything that can use an FCT or a job that can become unavailable. The following JES3 resources can be monitored:

- Generalized subtasks - allow to execute code containing implicit or explicit MVS WAITs.
- AENQ resources - obtain use of a JES3 resource.
- JQEs - Job Queue Element.
- Job numbers - JES3 supports as many as 999,999 jobs in your JES3 complex at the same time. However, JES3 limits the maximum number of jobs by choosing the smallest of the following:
  - The value you specify on the job limit parameter on the JOBNO= keyword of the OPTIONS initialization statement
  - The range of job numbers that you define on the JOBNO= keyword of the OPTIONS initialization statement
  - The number of entries in the job control table (JCT)

- File directory entries - The data management FILE DIRECTORY accounts for all opened multirecord files and some single-record files.
- JSAM buffers - The primary of JSAM buffers is defined with the BUFFER initialization statement.
- Spool space.

SDSF panels

When you use SDSF interactively, SDSF displays data on panels. There are panels for active jobs, output groups, printers, initiators and so on. Most SDSF panels are tabular, that is, they display data in rows and columns.

```
Display  Filter  View  Print  Options  Search  Help  1
-------------------------------------------------------------------------------
 SDSF SAMPLE  SYS1     2                                  3  LINE 1-22 (31)
 COMMAND INPUT ===> _    4                                  SCROLL ===> PAGE
 PREFIX=*  DEST=(ALL)  OWNER=IBMUSER  FILTERS=2   5
 NP    JOBNAME  6      ProcStep JobID    Owner    C Pos DP PGN Real Paging
 7     CATALOG  CATALOG  IEFPROC                    NS  FF     3228   0.00
       TAPEPOL  PROC01   M02XF83L TSU19596 TAPEPOL   OT  FF  69    0   0.00
       TANDA    E52TOOL1 M02SA06L TSU18751 TANDA     OT  FF  63 1488   0.00
       KSHEL    PROC01   M02PV317 TSU07739 KSHEL     OT  FF  60    0   0.00
```

Figure 18-4 A sample SDSF tabular panel

The numbered entities on the sample SDSF tabular panel in Figure 18-4 are as follows:

1. **Action bar** - The action bar permits you to select a pull-down menu to SDSF tasks.
2. **Title line** - The title line shows the panel name as well as status information.
3. **Message area** - Short error and confirmation messages appear here.
4. **Command line** - The command line lets you enter SDSF, MVS, or JES commands.
5. **Message and information lines** - Longer messages appear below the command line.
6. **Data area** - The column titles and tabular data columns and rows.
7. **NP column** - Action characters for rows.

Global options and the format of the panels are defined in ISFPARMS. The options include things like the names of SDSF data sets, what generic and wildcard characters to allow in SDSF commands, and whether to display the action bar on SDSF panels. The format of the panels includes the order and titles of the columns.

The ISFPARMS can be defined only in the ISFPRMxx member of PARMLIB in the JES3 environment. The statements in the ISFPRMxx member are processed by an SDSF server. If the SDSF server is not started, defaults for all values are used.

An FLD statement, along with FLDENT statements in the ISFPRMxx member, defines the fields, including column names and titles, for an SDSF panel. FLD statement is associated with the field list for a particular panel by GROUP statement. The group function parameters are used to determine which functions the members of a group can perform. The SAF profiles GROUP.group-name.server-name, in the SDSF class, define the user-to-group associations. SDSF checks for READ access for a users-to-group association.

The source of the panel column data is either readily available from in-storage control blocks (Immed column) or the data comes from the JES spool and requires an I/O operation (Delay column). SDSF maintains an alternate column list for columns requiring I/O operations for data. I/O operations are only done when the columns are visible on the window or are being sorted.

You can define a primary and alternate variable field list for each SDSF panel. The primary field list contains those fields that are shown upon entry into a panel. The alternate field list contains fields that can be displayed with the ? command.

The COLSHELP (COLSH for short) command shows the columns on the SDSF panels. All possible columns are included. The actual columns that are available to you, as well as their titles, may have been customized with field lists in ISFPARMS. In the columns' help display, an X in the Delayed? column indicates that access for the column is delayed.

To switch between primary and alternate field lists display of a panel, use the **?** command or the panel's action bar View pop-up choice 4 (Change field list to ALTERNATE / PRIMARY). To select the View option, press Enter with the cursor on View.

You can overtype columns on any tabular panels. The syntax for overtyping columns on tabular panels is the column title followed by = and the changed value, all within <>. Enclose the column title and value in single quotation marks.

Security in JES3 environment

When processing under JES3, only SAF-based security is used. This is optional in the JES2 environment but required in the JES3 environment. There is no ISFPARMS processing for security purposes. In cases where SAF cannot make a decision (SAF return code 4), the user is denied access to the resource. This is consistent with the general z/OS policy that access to a resource should be denied unless explicitly granted.

Using SDSF in batch

Using batch job processing, you can issue often-repeated SDSF commands by creating a list of the commands as control statements. In the list, you specify the SDSF panel you wish to use and the operation you wish to perform on it.

SDSF in batch is invoked with one of two program names on a JCL EXEC statement:

SDSF Supports commands and action characters.

ISFAFD Supports commands, action characters, and overtyping of fields on tabular and other panels, such as the print panels.

In the JCL for a batch SDSF job the ISFIN DD statement defines the input data, and the ISFOUT DD statement defines the output data set. For example, the JCL for a batch job to invoke program name ISFAFD might use the following statements:

```
//SDSF   EXEC PGM=ISFAFD
//ISFOUT DD SYSOUT=*
//ISFIN  DD *
```

To change panel width and depth of the batch output, specify `PARM='++xxxx,yyyy'` on the EXEC statement, where xxxx is the depth of the panel (number of lines) and yyyy is the width (number of characters). For example, to set the depth to 32 and the width to 1000, use:

```
//SDSF   EXEC PGM=SDSF,PARM='++32,1000'
```

If you do not specify the PARM, the width defaults to 132 and the depth to 60. The maximum for width and depth is 9999.

Note: When you invoke SDSF with either program name SDSF or ISFAFD, SDSF determines whether to process the JES2 or JES3 environment. You can request that SDSF not do that determination and process JES2. For this purpose, use the alternate program name SDSF2 or ISFAFD2.

Using program name SDSF

With program name SDSF to access a panel and display its contents, use the panel command and `++ALL`. For example, to select the H panel and display its contents, use:

```
H
++ALL
```

When ++ALL is specified, anything else on the card is ignored.To move around on the panel, you can use scroll commands (RIGHT, LEFT, UP, DOWN, TOP, BOTTOM).

Notes about SDSF program on commands and actions in batch processing:

- All SDSF commands, as described in the online help, may be used as long as they do not require ISPF pop-ups.
- To use an action character, code ++action-character in your batch job ISFIN input.
- To prevent a confirmation pop-up from being displayed for destructive action characters, use the SET CONFIRM OFF command.
- A successful FIND must be issued prior to issuing an action character. This protects against issuing an action character against wrong rows.

  For example, to find job JOBXYZ on the O panel, browse it with the **S** action character and issue a `RESET` in case the job is not found. `ISFIN` data would be:

  ```
  O
  FIND 'jobxyz'
  ++S
  RESET
  ```

- Attribute bytes present on the SDSF panels are translated out when you invoke SDSF with program name SDSF.

Using program name ISFAFD

When you invoke SDSF with program name ISFAFD, it works the same as when you invoke it with program name SDSF, with some differences:

- Action characters do not require a successful FIND.
- Overtypes and PF keys are supported.
- The contents of a panel are not updated until you explicitly refresh the panel. You do this with the AFD REFRESH command.
- Attribute bytes are present on the SDSF panels.

With program name ISFAFD, you can use the SDSF commands as you would with program name SDSF. You can also use these AFD commands:

AFD LOCATE	Scroll OPERLOG
AFD LOGSTAMP	Control OPERLOG or SYSLOG printing with SDSF's PRINT function
AFD QUERY	Display information about the current data set, the code page that is in use on the message line or columns on the current tabular panel
AFD REFRESH	Request that SDSF refresh the current display
AFD WTOR	Control the display of WTOR messages at the bottom of the Log panel
AFD NP	Controls the width of the NP column

AFD .END	Assigns a label, .END, to the current top line of the SYSLOG or OPERLOG

Notes about ISFAFD program on commands and actions in batch processing:

- Selected PF keys may be used by coding `++AFD PFxx` (where `xx=03` is a request to end the current panel and `xx=05` repeats the previous FIND).
- Columns on tabular panels and on other SDSF panels can be overtyped. The syntax for overtyping columns on tabular panels is the column title followed by = and the new value, all within <>.

  Where it is valid when using SDSF interactively, you can combine an action character and overtypes; the action character must precede the overtypes.

SDSF batch and security

To protect use of SDSF in batch, you control which group of users a user is assigned to. You do this in the JES3 SDSF environment through SAF. SAF is dynamic and it allows you to assign users to the same group regardless of the environment from which they invoke SDSF (interactive, batch, REXX or Java).

Action characters are not SAF-protected as separate resources. However, use of most action characters causes an interaction with two resources, both of which must be protected:

- The object of the action character, such as an initiator, printer, MAS member, or job
- The MVS or JES command that is generated in response to the action character. When these resources are protected, a user requires authority to both resources to enter most action characters.

The objects of action characters are such things as initiators in the SDSF class, printers and punches in the WRITER class, and jobs, output groups, and SYSIN/SYSOUT data sets in the JESSPOOL class. The resource name that protects the object and the access level required varies from panel to panel.

Most action characters generate MVS or JES commands that are protected in the OPERCMDS class. Users can be conditionally permitted to access OPERCMDS resources so they are authorized to use MVS and JES commands only while they are using SDSF.

SDSF in batch example

See Appendix B, “SDSF REXX and SDSF in batch examples” on page 999 for an example of SDSF in batch.

Using SDSF with the REXX programming language

Using REXX with SDSF provides a simpler and more powerful alternative to using SDSF in batch. You invoke SDSF function with the host command environment SDSF. Data and SDSF messages are returned in REXX variables. An authorized user can access SDSF functions in a REXX program using the following:

ISFCALLS() command	Add and delete the SDSF host command environment
ISFEXEC command	Issue SDSF commands to access tabular panels and other information
ISFACT command	Issue action characters and overtype columns
ISFGET command	Get all of the values for a single row
ISFLOG command	Work with the SYSLOG and OPERLOG
ISFSLASH command	Issue system commands
Special REXX variables	Issue SDSF commands for filtering and options, and check messages

ISFACT and special variables	Browsing output
ISFACT and special variables	Printing output
ISFRESET() command	Drop specified special variables

Many of the things you work with in a REXX exec, such as the list of columns on an SDSF panel, the contents of the title line of a panel, and the contents of responses to SDSF commands such as WHO, may change over time. You should design your REXX execs to minimize the impact of those changes.

For an up-to-date description of the REXX SDSF functions and special variables, see *z/OS SDSF Operation and Customization,* SA22-7670.

REXXHELP command

Information about using REXX with SDSF is also available in SDSF's online help. The help includes links to descriptions of commands, action characters and overtypable columns, which are not included here.

To display the online help about using REXX with SDSF:

- Type REXXHELP on any command line in SDSF when using SDSF under ISPF.
- Type COLSHELP, which shows information about columns on various display panels.
- To search SDSF's help, including the help for REXX, use the SEARCH command. You can type SEARCH followed by up to four words on the SDSF command line when using SDSF under ISPF.
- If you are not already familiar with SDSF, you should begin with the SDSF help. To display a brief, interactive tutorial, use the TUTOR command.

Security and REXX

Using SDSF function from a REXX exec is protected just as using SDSF interactively is protected, with the same SAF resources in the JES3 environment. Where special REXX variables correspond to SDSF commands, the authorization for those special variables is the same as for the associated command. In some cases, using a special variable when you are not authorized to the associated command will cause the exec to fail and the invocation of SDSF to end.

SDSF server

The SDSF server is an address space that SDSF uses to:

- Process ISFPARMS statements. ISFPARMS defines global and group options and the format of the panels. The options include things like the name of the JES subsystem to process, what generic and wildcard characters to allow in SDSF commands, and whether to display the action bar on SDSF panels. The format of the panels includes the order and titles of the columns.
- Provide sysplex support. This consists of sysplex-wide data for JES2 devices and for system resources (CK, ENC, INIT, LI, NO, PR, PS, PUN, RDR, RM and SO panels) as well as the most recent SYSLOG data for remote systems (SYSLOG panel).

The SDSF server is not required for sysplex-wide device panels (INIT, LI, NO, PR, PUN, RDR and SO):

- In a JES3 environment. For JES3, all configuration parameters default if there is no server. That is because the assembler ISFPARMS is not supported in JES3 and the server is required to process the ISFPRMxx parmlib member.
- In a JES2 environment when all systems are at the z/OS V1R13 level.

To process ISFPARMS, the server must be active on each system that contains SDSF users. To provide sysplex data, the server must be active on each system that is to be included on SDSF panels. Use the WHO command or pop-up to verify that the server is in use.

Multiple SDSF servers may be run on the same system; however, you must assign them unique names. Only one server with a particular name can be active on the system. The level of the server must match the level of the SDSF application.

You control the server through the MVS operator START, STOP, and MODIFY commands. (The START command names the server; the MODIFY command refreshes the ISFPARMS statements, changes server options, and displays and controls server communications.)

Note: The document *z/OS SDSF Operation and Customization,* SA22-7670 describes the most up-to-date security considerations and customization of SDSF.

Using SDSF with the Java programming language

You can access SDSF function with the Java programming language. This provides a simpler and more powerful alternative to using SDSF in batch, and complements SDSF's support for the REXX programming language.

The principal source of information for using Java with SDSF is the Javadoc supplied with SDSF. To use the Javadoc:

1. Download the `isfjcallDoc.jar` file, in binary, to an empty directory on your workstation. By default, this file is installed into `/usr/include/java_classes/isfjcallDoc.jar.`
2. If you have the Java SDK installed, use this command:
   `jar -xf isfjcallDoc.jar`

   Otherwise, use another utility to unzip the file.
3. Navigate to the `index.html` file and open it with a web browser. Once the `index.html` file is displayed, links allow you to navigate to specific classes or topics, such as:

   Overview Display an overview to using SDSF with Java
   Package Display a list of classes
   Tree Display a hierarchical view of classes
   Index Display an index to the Javadoc

Simplify systems management with SDSF Java

With the z/OS V1R12 SDSF Java API, you can access SDSF panel data and function through a Java program.

Accessing panels and panel data: Each of the panels that you work with when using SDSF interactively (DA, O, PR and so on) has an associated Java interface that describes the returned data and the available methods. Panel data is represented by lists, with each element in a list corresponding to a row on the panel. You access column data within a list element by referencing column values by column name.

Processing system log and issuing commands: You can retrieve records from the system log (SYSLOG) and search for specific messages or events. You can also issue free-form system commands and receive their responses in a manner similar to the SDSF slash (/) command.

Retrieving job output: You can allocate the spool data sets for a job and read them using standard utilities.

Taking action: You use methods to perform functions similar to action characters and overtypable fields, for example, to cancel a job or change the print destination for job output.

Filtering data: For best performance, limit the data that a request returns to the minimum that is required. You do this with request settings, which allow you to specify things such as:

- Filters of various kinds. The same filters that are available when you use SDSF interactively are available with request settings. They include filters by job name, owner and destination, like the PREFIX, OWNER and DEST commands, or any column, like the FILTER command.
- The list of columns to process. Columns are specified by column name.
- Whether to include columns with delayed access. Because gathering the data for "delayed" columns can take significant time, they are not included unless you request them explicitly

View results: You can access messages and return codes that describe the completion of a request through a results object. SDSF messages and system messages, if any, issued in response to commands are contained in lists, with each element corresponding to a message. Return codes from SDSF functions are available both in the results object and as return codes on most methods.

Control access: Standard SDSF authorization checking occurs for all requests and for attempts to modify the row represented by a returned object. SDSF security is described in *z/OS SDSF Operation and Customization*, SA22-7670.

18.4 SDSF tutorial

The SDSF tutorial introduces SDSF and lets you try some of SDSF's most useful functions. The interactive tutorial panel, Figure 18-5, is invoked with the TUTOR command.

```
                    TUTOR - System Display and Search Facility
COMMAND INPUT ===>

        The SDSF tutorial introduces SDSF and lets you
        try some of SDSF's most useful functions.  For detailed
        information such as command syntax, use the help facility.

        The whole tutorial takes about 25 minutes.  Press Enter to
        begin viewing it, or begin with a particular topic by
        typing one of the numbers below:

         1 - Using the tutorial      5 - Purging output
         2 - SDSF panels             6 - Controlling jobs
         3 - Monitoring jobs         7 - Printing data
         4 - Displaying output       8 - Filtering and sorting

                          9 - Quick summary

  F1=Help               F10=Previous           Enter=Forward
  F3=Exit                F7=Up                 TOC=Table of contents
```

Figure 18-5 Primary tutorial panel

Some parts of the tutorial ask you to enter information on simulated SDSF panels. These simulated panels respond to your input. Interacting with them will help you learn how SDSF works. However, if you prefer, the system provides the input on interactive panels if you simply press Enter twice.

Except on the interactive tutorial panels, SDSF commands are not valid on tutorial or help panels.

For detailed information such as command syntax, use the help facility.

18.5 SDSF panels

The SDSF primary option menu lists the panels that you are authorized to use, and the commands that display the panels. (A few panels, shown later in this topic, are accessed with action characters instead of commands, and do not appear on the primary option menu.)

The tabular panels have a fixed field or column, at the left, that does not move as you scroll right and left. You can scroll displayed information up, down, right, and left.

Under ISPF, most SDSF functions can be selected from the action bar at the top of the panel. To display a pull-down menu of action bar choices, place the cursor on an option and press Enter.

Note: The words “field” and “column” are used interchangeably in this document.

Tabular SDSF panel layout in ISPF

Figure 18-6, Display Active Users Panel, is used to describe the SDSF panel layout and features.

```
  Display  Filter  View  Print  Options  Search  Help          Action bar
-------------------------------------------------------------------------------
SDSF DA SC75     SC75     PAG  0  CPU/L/Z   3/  1/  0  LINE 1-     Messages
COMMAND INPUT ===>  SDSF commands                    SCROLL ===> HALF
ACTION=//,=,+,?,A,C,CA,CD,CDA,CP,
ACTION=Q,R,RQ,S,SB,SE,SJ,W,X,XC,X                Action character display
NP   JOBNAME  StepName ProcStep JobID    Owner    C          Column titles
     *MASTER*                   JOB20313 +MASTER+
     PCAUTH   PCAUTH
     RASP     RASP                                          NS  FF  476   0.00
     TRACE    TRACE                                         NS  FF 1155   0.00
     DUMPSRV  DUMPSRV  DUMPSRV                              NS  FF  835   0.00
     XCFAS    XCFAS    IEFPROC                              NS  FF 9779   0.00
     GRS      GRS                                           NS  FF  21T   0.00
     SMSPDSE  SMSPDSE                                       NS  FF 1586   0.00
     SMSPDSE1 SMSPDSE1                                      NS  FF 1862   0.00
     SMSVSAM  SMSVSAM  IEFPROC                              NS  FF 3216   0.00
     CONSOLE  CONSOLE                                       NS  FF 4950   0.00
     WLM      WLM      IEFPROC                              NS  FF 9694   0.00
     ANTMAIN  ANTMAIN  IEFPROC                              NS  FF 1427   0.00
     ANTAS000 ANTAS000 IEFPROC                              NS  FE 1328   0.00
```

Complex wide tabular data, current and customizable

Action character (short commands)

PF10 Scroll PF11

PF7 Scroll PF8

Figure 18-6 Sample SDSF panel layout - DA display

SDSF uses colors on the tabular panels to identify active objects (such as jobs) and overtypable fields, as shown in Table 18-1 on page 859.

Table 18-1 SDSF color indicators

Color	Active/Not Active	Overtypable
Blue	Not Active	No
White	Active	No
Green	Not Active	Yes
Red	Active	Yes

NP action characters for all tabular panels

NP-Description	SDSF action
+	Expand the NP column. (Use RESET to reset.)
//	Block repeat; type // on the first row and another // on the last row to be processed. The action to be repeated should be typed once after the // characters.
=	Repeat previous action character or overtype

Figure 18-7 NP action characters on all tabular panels

SDSF PF keys in ISPF

Under ISPF, you can toggle the display of SDSF's PF keys with the ISPF `PFSHOW` command, and change the setting of the keys with the `KEYS` command. In SDSF running under TSO the PF keys cannot be displayed or changed.

In ISPF each SDSF pop-up panel has PF keys assigned with an ISPF keylist. Although ISPF allows you to change the values of the keys in keylists, and to turn off the use of keylists, you should use the IBM-supplied key definitions and leave keylists on.

The default PF key definitions for SDSF's panels running under ISPF and TSO are shown in Figure 18-8.

KEY	ISPF (Default)	TSO
1 and **13**	Help	Help
2 and **14**	Split the screen	Not used
3 and **15**	End the current panel	End the current panel
4 and **16**	End SDSF	End SDSF
5 and **17**	Repeat the previous FIND	Repeat the previous FIND
6 and **18**	Invoke BookManager	Not used
7 and **19**	Scroll up	Scroll up
8 and **20**	Scroll down	Scroll down
9 and **21**	Swap split screens	Print the screen
10 and **22**	Scroll left	Scroll left
11 and **23**	Scroll right	Scroll right
12 and **24**	Retrieve the previous command. (Some short commands are not retrieved.)	Return the cursor to the command line

Figure 18-8 SDSF panel default key definitions

Action bar

```
 Display  Filter  View
                    ----
  1. Panels...      ----
  2. Logs...
  3. Exit SDSF
I     Input queue
O     Output queue
```

```
                    Panels
Option ===>

                              More:     +
1  Active users (DA)
2  Enclaves (ENC)
3  Health Checker (CK)
4  Held output queue (H)
5  Initiators (INIT)
6  Input queue (I)
7  Job classes (JC)
8  Jobs on any queue (ST)
9  Job O (JO)
10 Lines (LI)
11 Members in the MAS (MAS)
12 Network connections (NC)

F1=Help  F12=Cancel
```

```
                    Panels
Option ===>

                              More:   - +
12 Network connections (NC)
13 Network servers (NS)
14 Nodes (NO)
15 Output queue (O)
16 Printers (PR)
17 Processes (PS)
18 Punches (PUN)
19 Readers (RDR)
20 Resource monitor (RM)
21 Scheduling environments (SE)
22 Spool offload (SO)
23 Spool volumes (SP)

 F1=Help  F12=Cancel
```

Figure 18-9 Action bar Display -> 1. Panels -> Panels list

Under ISPF, SDSF functions can be selected from the action bar, a row of options at the top of the panel shown in Figure 18-9. Select an option from the action bar by placing the cursor on an item and pressing Enter. In the pull-down menu of choices that is displayed, select a

choice by number or by placing the cursor on it and pressing Enter. Choices ending with an ellipsis (...) display a pop-up window.

Most of the SDSF's displays use the same ISPF ISFPCU41 panel. This panel defines the same set of action bar choices for all displays where used.

The action bar choices are, as follows:

- **Display** This action bar choice represents three related choices that appear in the pull-down:
  - **Panels** - This choice displays a menu of the SDSF panels. The SDSF panel menu lists 24 displays. Each display choice is numbered. When a display choice is available, its number is highlighted. The panels list is shown in Figure 18-9.
  - **Logs** - The Logs pop-up allows you to display one of the two SDSF Log panels. The pop-up options are: 1 `System log (LOG)` and 2 `User session log` (U):
    - The `System log` option displays the SDSF log panel, which displays MVS log data.
    - The `User session log` option displays the SDSF User Session Log (ULOG) panel, which shows MVS and JES commands and responses (including commands generated by SDSF) that you issued during your session.
  - **Exit SDSF** - This choice of the Display pull-down ends the SDSF session.
- **Filter** - This action bar choice represents three following choices that appear in the pull-down:
  - **Filter** - The Filter choice of the Filter pull-down displays the Filter pop-up, which allows you to filter the data displayed on SDSF panels that display data in tabular format.
  - **Prefix of jobname** - The Prefix choice of the Filter pull-down displays a pop-up that allows you to filter the DA, I, PS, and ST panels based on job name.
  - **Owner** - The Owner choice of the Filter pull-down displays a pop-up that allows you to filter the DA, I, PS and ST panels based on owning user ID.
  - **Destination** - The Destination choice of the Filter pull-down displays a pop-up that allows you to filter the I, PR, ST, and PUN panels based on destination.
  - **System name** - The System Name choice of the Filter pull-down displays a pop-up that allows you to specify which systems are to be included on the DA panel.
  - **Replies on the Log** - The Replies on the Log choice of the Filter pull-down displays the Replies on the Log pop-up, which lets you filter the WTORs on the Log panel.
- **View** - The View option of the action bar allows you to control the view of the data displayed on SDSF panels. To select the View option, press Enter with the cursor on View:
  - **Sort** - The Sort choice of the View pull-down displays a pop-up that allows you to sort the SDSF panels that display information in tabular format.
  - **Arrange** - The Arrange choice of the View pull-down displays the Arrange pop-up, which allows you to reorder and change the widths of the columns on panels that display data in tabular format.
  - **Set hex to ON** - The Set Hex choice of the View pull-down acts as a toggle to turn hexadecimal formatting of the log or output data sets on or off.
  - **Change field list to ALTERNATE** - The Change Field List choice of the View pull-down acts as a toggle to display the primary or alternate fields on SDSF panels that display information in tabular format.
  - **Who** - The Who choice of the View pull-down displays your user ID, TSO logon procedure name, terminal ID, the index number and name of your group in ISFPARMS, information about software levels, and information about the server. An example is

shown in Figure 18-10. The field for server communications (COMM=) shows information about communications between SDSF servers.

```
USERID=VJUHA,PROC=IKJACCJV,TERMINAL=SC38TC86,GRPINDEX=1,GRPNAME=ISFSPROG,
MVS=z/OS 01.11.00,JES=z 1.11.0,SDSF=HQX7760,ISPF=6.1,RMF/DA=760,SERVER=YES,
SERVERNAME=SDSF,JESNAME=JES3,MEMBER=SC75,JESTYPE=JES3,GLOBAL=SC75,
GLOBALREL=HJS7760,SYSNAME=SC75,SYSPLEX=PLEX75,COMM=NOTAVAIL
```

Figure 18-10 Data in the WHO display and WHO pop-up

- **Print** - The Print option of the action bar allows you to select options for printing data:
  - **Print open sysout** - The Print Open Sysout choice displays a panel that allows you to specify the attributes of a SYSOUT print file.
  - **Print open data set** - The Print Open Data Set choice displays a panel that allows you to specify the attributes of a data set to be used as the print file.
  - **Print open file** - The Print Open File choice displays a panel that allows you to specify the ddname to be used as the print file.
  - **Print** - The Print choice displays a pop-up that allows you to specify the lines to print from a SYSLOG or output data set. If no print file is open, the Print choice opens a default SYSOUT file.
  - **Print close** - The Print Close choice either frees a SYSOUT print file and makes it available for printing, or closes a print data set.
  - **Print screen with ISPF** - The Print Screen with ISPF choice invokes ISPF's PRINT command to print the panel image to an ISPF list file. This choice does not use SDSF's print file.
- **Options** - The Options option of the action bar allows you to set options such as a find limit and colors:
  - **Set action character display** - The Set Action Character Display choice of the Options pull-down displays a pop-up that allows you to control the display of valid action characters on SDSF panels.

    Action characters are typed in the NP column of tabular panels. For example, to purge a job, you type `p` in the NP column next to the job on the Status panel.

    The display of the valid action characters for a panel can also be set with the SET ACTION command.
  - **Find limit** - The Find Limit choice of the Options pull-down displays a pop-up that allows you to limit the number of lines searched when the FIND command is issued on a browse panel.
  - **Change include SYSIN to ON** - The Change Include SYSIN choice of the Options pull-down lets you control whether the Output Data Set panels that you select from the DA, ST, or I panels will include SYSIN data sets.
  - **Set bookshelf** - The Set Bookshelf choice of the Options pull-down displays a pop-up that allows you to set the default bookshelf to be searched by BookManager.
  - **Set display values to ON** - The Set Display Values choice of the Options pull-down acts as a toggle to control the display of values for DEST, OWNER, PREFIX, FILTER, SORT and SYSNAME on the information line. Figure 18-11 is an example of the “Set display values to ON” display.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF DA SC75     (ALL)    PAG  0  CPU/L/Z   4/  1/  0   SET COMMAND COMPLETE
COMMAND INPUT ===>                                          SCROLL ===>
HALF
PREFIX=JES3*  DEST=*  OWNER=*  SYSNAME=*
ACTION=//,=,+,?,A,C,CA,CD,CDA,CP,D,DE,DL,DSD,DSH,DSP,DX,E,H,K,KD,L,LB,LH,LT,
P,
ACTION=Q,R,RQ,S,SB,SE,SJ,W,X,XC,XD,XDC,XF,XFC,XS,XSC,Y,Z
NP   JOBNAME  StepName ProcStep JobID    Owner    C        Pos DP Real
Paging
     JES3     JES3     JES3                                NS  FE  11T
0.00
     JES3AUX  JES3AUX  IEFPROC                             NS  FF  176
0.00
     JES3CI   CI7      JES3CI   JOB20958 JES3              NS  FE 1176
0.00
     JES3DLOG JES3DLOG IEFPROC                             NS  FF  179
0.00
```

Figure 18-11 Snipped of a DA display with Set display values ON set

- **Set screen characteristics** - The Set Screen Characteristics choice of the Options pull-down displays a pop-up that allows you to control the use of color and highlighting on SDSF panels, as well as turn the display of the action bar on or off.
- **Set delay for responses** - The Set Delay choice of the Options pull-down displays a pop-up that allows you to control the default timeout value for awaiting responses to the slash (/) command.
- **Set communications timeout** - The Set communications timeout choice of the Options pull-down displays a pop-up that lets you set the timeout value for awaiting sysplex data.
- **Set console name** - The Set Console Name choice of the Options pull-down displays a pop-up that allows you to set the name of the extended console used by SDSF. The extended console is used by the ULOG panel.
- **Set search characters** - The Set Search Characters choice of the Options pull-down displays a pop-up to let you set the generic and placeholder characters used in pattern matching.
- **Assign PF keys** - The Assign PF Keys choice of the Options pull-down invokes ISPF's KEYS facility to let you change the PF keys for SDSF panels.
- **Change show PF keys** - The Change Show PF Keys choice of the Options pull-down invokes ISPF's PFSHOW command to let you turn the display of PF keys on or off.
- **Set language for help and tutorial** - The Set Language for Help and Tutorial choice of the Options pull-down displays a pop-up that allows you to select English or Japanese for the Help and Tutorial.
- **Set cursor option** - The Set Cursor choice of the Options pull-down acts as a toggle to control how SDSF places the cursor when you work with rows on tabular panels (except OD).

  ON causes the cursor to return to the NP column for the last row you worked with. If the row is not on the panel, either because it would require a scroll, or because your

actions or system activity caused it to be removed from the display, the cursor is returned to the command line.

OFF causes the cursor to return to the command line.

– **Set confirmation** - The Set Confirmation choice of the Options pull-down acts as a toggle to control confirmation of action characters. When confirmation is on, SDSF requests confirmation of cancel, purge, restart and system stop action characters on job-oriented tabular panels (DA, H, I, JDS, O, PS and ST), drain and halt actions on the SP panel, and quiesce on the ENC panel.

– **Operlog limit for filter** - The Operlog Limit for Filter choice of the Options pull-down displays the Operlog Limit for Filter pop-up, which lets you set the amount of Operlog data SDSF will search for records that meet filter criteria.

– **Set date format** - The Set Date Format choice of the Options pull-down displays the Set Date Format pop-up, which lets you select the format SDSF uses for dates. The available date formats are mm/dd/yyyy, dd/mm/yyyy, or yyyy/mm/dd and the separator character, either slash (/), dash (-), or period (.).

– **Set log default** - The Set Log Default choice of the Options pull-down displays a pop-up that allows you to select the default panel for LOG command. The default panel is displayed when you enter LOG with no parameters, or select the Log choice from the Display pull-down.

– **Set default browse action -** The Set Default Browse Action choice of the Options pull-down displays a pop-up that allows you to select the default browse action (S, SB or SE) that is issued when you place the cursor in the NP column and press Enter on the job and output panels. The options are S (SDSF browse), **SB** (ISPF browse), **SE** (ISPF edit), and **None.** The default browse action character is invoked when you select a row on a job or output panel (DA, I, JDS, OD or ST) by placing the cursor in the NP column and pressing Enter. The result is the same as if you had typed the action character in the NP column.

  If you select **None**, then you must type an action character in the NP column to invoke browse.

– **Help** - The Help option of the action bar allows you to select online help for: Extended help, Keys help, Help Index, Tutorial, Book, web sites, REXX help and Columns help.

Cursor-sensitive sort

You can sort a tabular panel by placing the cursor on a column title and pressing Enter. For example, to sort a panel by job name, place the cursor on the JOBNAME column and press Enter. This is a quick alternative to typing the SORT command.

You can disable cursor-sensitive sorting with the **SET CSORT OFF** command and enable with the **SET CSORT ON** command.

18.6 SDSF help panels

Information for users of SDSF, such as commands, action characters, and messages, is mainly provided in the online help for SDSF.

Help panels appear in pop-up windows in response to user requests for assistance during SDSF application sessions. Figure 18-12 shows the table of contents (TOC) for SDSF online help. This panel can be accessed by typing HELP at the Command Input line, by pressing the PF1 key at the SDSF Primary Option Menu, or choosing Option 1. Extended help from the SDSF HELP action bar menu.

```
HELP: SDSF -- Table of Contents
COMMAND INPUT ===>

  Select a topic by number, or press Enter to view topics in sequence.

   1 - What's new                          13 - Action bar
   2 - Job and workload panels             14 - PF keys and keylists
   3 - Device and resource panels          15 - Action characters
   4 - Search and scroll commands          16 - Overtypeable fields
   5 - Filter commands                     17 - JES and MVS commands
   6 - View commands                       18 - Online library
   7 - Print command and actions
   8 - Options commands                    19 - SDSF messages
   9 - Other commands
  10 - Server commands                REXXHelp - SDSF REXX (ISPF only)
  11 - SDSF command
  12 - Help commands

  SEARCH - Search the help (ISPF only)

F1 = Help             F10 = Previous          Enter = Forward
F3 = Exit              F7 = Up                TOC = Menu
```

Figure 18-12 SDSF HELP Table of Contents pop-up

SDSF provides HELP for the HELP too. Figure 18-13 on page 865 and Figure 18-14 on page 866 are the window pop-ups for the HELP command. These panels can be accessed by typing HELP at the command input line from any HELP pop-up window.

```
HELP: HELP Command                  Panel 1 of 2
COMMAND INPUT ===>

    Purpose:  Displays online help for SDSF.

    Where used:  Any SDSF panel, including help and tutorial panels.

    Format: HELP

    Press Enter to see the next help panel in a sequence.
    You can also use:

    SEARCH      - Search the help (ISPF only). Enter outside of help.
    I or F11    - Display an index to help (ISPF only).
    TOC         - Display the table of contents for help.
    TUTOR or F4 - Display an interactive tutorial (ISPF only).

F1 = Help             F10 = Previous          Enter = Forward
F3 = Exit              F7 = Up                TOC = Menu
```

Figure 18-13 HELP: HELP Command (1 of 2) pop-up

```
HELP: HELP Command                    Panel 2 of 2
COMMAND INPUT ===>

    Some PF keys have special uses in help:

    F4 (Tutor)         - Display an interactive tutorial (ISPF only)
    F5 (Extended help) - Display the general help for the topic
                         (ISPF only)
    F7 (Up)            - Display the start of the current topic or,
                         if you're already at the start, display the
                         previous menu
    F10 (Previous)     - Back up one help panel
    F11 (Index)        - Display the help index (ISPF only)

    Under ISPF, you can get more information on a highlighted phrase
    by placing the cursor on the phrase and pressing Enter.

F1 = Help             F10 = Previous          Enter = Forward
F3 = Exit              F7 = Up                TOC = Menu
```

Figure 18-14 HELP: HELP Command (2 of 2) pop-up

Figure 18-15 on page 866 and Figure 18-16 on page 867 are the HELP Index pop-ups. The HELP index can be accessed by entering **I** command in the command input line from any HELP dialog window. To view an index selection, just type that character on the HELP index pop-up `COMMAND INPUT ===>` line.

```
HELP: Index - Special Characters and 'A'        Page 1 of 2
COMMAND INPUT ===>

    Enter the number of a topic, or a letter to view another panel.

    1  / command, issue system commands
    2  A action character, release a job
    3  A action character, release output
    4  abend codes
    5  abend, requesting with ABEND command
    6  ABEND command
    7  action bar
    8  action characters
    9  action characters, confirming
    10 action characters, displaying
    11 action characters, repeating
    12 ACTION command

  F1=Help               F10=Previous           Enter=Forward
  F3=Exit                                      TOC=Table of contents
```

Figure 18-15 HELP index pop-up 1 of 2

```
HELP: Index - Special Characters and 'A'        Page 2 of 2
 COMMAND INPUT ===>

     Enter the number of a topic, or a letter to view another panel.

     12 active jobs, displaying
     13 active users, displaying
     14 address printed on separator pages, changing
     15 alternate form of an SDSF panel
     16 ARRANGE command
     17 arranging columns
     18 attributes of SYSOUT, displaying
     19 authorization group, displaying
     20 automatic update (&)

   F1=Help               F10=Previous           Enter=Forward
   F3=Exit                                      TOC=Table of contents
```

Figure 18-16 HELP Index pop-up 2 of 2

Online HELP example for the DA tabular panel

The following gives an example of the extensive help available for the SDSF tabular panels. The figures in this example introduce the help panels for the active users (DA) display.

Note: The HELP pop-ups for the other SDSF panels have the same basic format and flow.

The *z/OS SDSF Operation and Customization*, SA22-7670 document is intended primarily for system programmers and operators. It contains information about customization, security, operation, maintenance and problem determination and explanations of SDSF messages. The Online HELP offers the *SDSF User's Guide*.

```
                      HELP: Display Active Users Panel
 COMMAND INPUT ===>

         Select a topic by number, or press Enter to view them in sequence.

          1 - Introduction to the DA panel
          2 - Syntax of the DA command
          3 - Action characters: display output, cancel jobs, etc.
          4 - Fields on the DA panel
          5 - Overtyping fields to change their values
          6 - Commands: limit jobs displayed, search, etc.

         These topics are displayed only if selected:

         97 - What's new
         98 - Search and navigate the help
         99 - Messages

 F1 = Help             F10 = Previous          Enter = Forward
 F3 = Exit              F7 = Up                TOC = Menu
```

Figure 18-17 Help panel for Display Active Users Panel

Figure 18-17 shows the general layout of the first pop-up window for an SDSF display panel. Examples of the help text in the DA display pop-up windows follow.

Note: The PF key settings are the same on all the following pop-ups and are shown.

The HELP pop-up for the selection “1- Introduction to the DA panel” is shown in Figure 18-18.

```
                      HELP: Display Active Users Panel         Panel 1 of 1
COMMAND INPUT ===>

        The Display Active Users (DA) panel allows authorized
        users to display information about jobs, users,
        started tasks, and initiators that are active
        in the sysplex.  It also shows system data, such
        as CPU usage and paging information.

        In a JES3 environment, the DA panel requires RMF.
        In a JES2 environment, RMF is required for sysplex-wide
        data and some columns and actions.

        Note: Some of the values on the DA panel, such as CPU% and SIO,
        are approximate.  For detailed and precise performance
        monitoring, use RMF.
```

Figure 18-18 Selection 1 - Introduction to the DA panel

The HELP pop-ups for the selection “2- Syntax of the DA command” are shown in Figure 18-19 on page 869.

Note: The PF key settings are the same for all three pop-ups and are shown only for the 1 of 3 pop-up.

```
                 HELP: Display Active Users Panel -- DA Command Panel 1 of 3
COMMAND INPUT ===>

    Purpose:  Displays the Display Active Users panel.  You
         must be authorized to use this command.

    Where used:  Any SDSF panel.

    Format:  DA (parameters)
        Parameters allow you to limit the display by:
         - Types of address spaces: jobs, TSO users, started tasks,
           or initiators
         - Positions of address spaces: swapped in, swapped out,
           in transition, or ready.

        The parameters are explained on the next panel.

   Example:  DA OIN NOTSU
       Displays only address spaces that are swapped in, not including
       TSO users.
```

```
              HELP: Display Active Users Panel -- DA Command  Panel 2 of 3
COMMAND INPUT ===>

                     Position: Type:   Only:     No:        All:

   Format:        DA (IN   )   (JOB )  (OJOB  )  (NOJOB  )  (ALL )
                     (OUT  )   (TSU )  (OTSU  )  (NOTSU  )  (ALLT)
                     (TRANS)   (STC )  (OSTC  )  (NOSTC  )  (ALLP)
                     (READY)   (INIT)  (OINIT )  (NOINIT )
                                       (OIN   )  (NOIN   )
                                       (OOUT  )  (NOOUT  )
                                       (OTRANS)  (NOTRANS)
                                       (OREADY)  (NOREADY)
```

```
              HELP: Display Active Users Panel -- DA Command  Panel 3 of 3
COMMAND INPUT ===>

       "Position" and "Type" parameters include those address spaces.
       "Only" parameters limit the display to those types or positions.
            Use only one parameter from this column.
       "No" parameters exclude those types or positions.
       "All" parameters show all address spaces, or all types or positions.
            They cannot be used with other parameters.

        The maximum number of parameters is four.

        The information displayed may also be limited by your
        authorization, and by settings for filters such as FILTER,
        PREFIX, and SYSNAME.

        When parameters conflict, the last one is used.
```

Figure 18-19 Selection 2 - Syntax of the DA command pop-ups

The HELP pop-ups for the selection “3 - Action characters ” for the DA panel are shown in Figure 18-20 on page 870 and Figure 18-21 on page 871.

```
       HELP: Display Active Users Panel -- Action Characters  Panel 1 of 5
COMMAND INPUT ===>

      Action characters that can be entered in the NP column
      by authorized users are:

          //    Block repeat; type // on the first row and
                another // on the last row to be processed
          =     Repeat previous action character or overtype
          +     Expand the NP column.  (Use RESET to reset.)
          A     Release a held job.
          C     Cancel a job. For JES3, also print non-held data sets.
          CA    Cancel a job that is defined to Automatic
                Restart Manager (ARM).
          CD    Cancel a job and take a dump.
          CDA   Cancel a job that is defined to ARM, and take a dump.
          CP    Cancel a job and delete held data sets. (JES3 only)
```

```
       HELP: Display Active Users Panel -- Action Characters  Panel 2 of 5
COMMAND INPUT ===>

          D     Display job information in the log. You can add:
                E  - Line, page, record and card counts (JES3 only)
                L  - Long form
                SD - DDNAMES of spool data sets that contain data
                      (JES3 only)
                SH - DDNAMES of spool data sets in spool hold that
                      contain data (JES3 only)
                SP - Spool partition name (JES3 only)
                X  - Extended (JES3 only)
          E     Process a job again. You can add (JES2 only):
                C  - Cancel and hold the job prior to execution
                S  - After the current step completes
                SH - After the current step completes, restart and hold
```

```
       HELP: Display Active Users Panel -- Action Characters  Panel 3 of 5
COMMAND INPUT ===>

          H     Hold a job.
          K     Cancel a started task (system cancel).
          KD    Cancel a started task and take a dump (system cancel).
          L     List output status of a job in the log. For JES3, this
                is job output in the writer queue.  You can add:
                B - SNA/NJE output (JES3 only)
                H - Output on the hold queue (JES3 only)
                L - Long form (JES2 only)
                T - TCP/IP job output (JES3 only)
          P     Cancel a job and purge its output.
          PP    Cancel a protected job and purge its output. (JES2 only)
          R     Reset and resume a job. (RMF)
          RQ    Reset and quiesce a job. (RMF)
          Q     Display output descriptors for all of the data sets
                in an output group.
```

Figure 18-20 Selection 3 - Action characters pop-ups (1 to 3 of 5)

```
        HELP: Display Active Users Panel -- Action Characters  Panel 4 of 5
 COMMAND INPUT ===>

           S     Display the data sets for a job.  You can add:
                 B - Use ISPF Browse
                 E - Use ISPF Edit
                 J - Use ISPF Edit to edit the JCL
           W     Cause job and message logs to spin. (RMF)
           X     Print output data sets.  You can add:
                 C - Close the print file after printing (XC)
                 D - Display the Open Print Data Set panel (XD or XDC)
                 F - Display the Open Print File panel (XF or XFC)
                 S - Display the Open Print panel (XS or XSC)
           Y     Stop a started task (system stop). (RMF)
           Z     Cancel a started task (system force).
           ?     Display a list of data sets for a job.
                 (Access the Job Data Set panel.)
```

```
        HELP: Display Active Users Panel -- Action Characters  Panel 5 of 5
 COMMAND INPUT ===>

           For more information, select a topic by number:

           1 - Using block repeat (//)
           2 - Using repeat (=)
           3 - Displaying valid action characters on the current panel
           4 - Browse (S and SB) in a sysplex
           5 - Using the S action character
           6 - System commands issued for the action characters
           7 - Using the X action character
           8 - Setting the default browse action character
```

Figure 18-21 Selection 3 - Action characters pop-ups (4 and 5 of 5)

The HELP pop-ups for the selection “4 - Fields on the DA pane” are shown in Figure 18-22 and Figure 18-23 on page 873.

```
              HELP: Display Active Users Panel -- Fields   Panel 1 of 9
COMMAND INPUT ===>

        The title line shows the following:

SDSF DA IPO1    IP*     PAG  0   CPU/L/Z/  26/ 26/  0   LINE 1-20 (20)
        |       |        |       |                      |          |
  System ID     |        |       |                 Lines displayed |
  of system     |  Total demand  |                 or first line   |
  you are       |  paging rate   |                 if 100,000      |
  logged        |               Percentage of time                 |
  on to         |               the CPU is busy,                   |
           Systems displayed    MVS, LPAR and zAAP                 |
           (MVS value or        views                  Total lines
           SYSNAME value)                              (**** if more
                                                       than 99,999,999)

  SIO, if shown, is the total system start I/O rate.
  PAG, SIO, and CPU values are for the system you are logged on to.
```

Figure 18-22 Selection 4 - Fields on DA panel (1 of 9)

```
                  HELP: Display Active Users Panel -- Fields   Panel 2 of 9
 COMMAND INPUT ===>
The Display Active Users panel includes some or all of the following
       fields.  (The order and titles may be different, depending upon
       installation and user options.)

         Title       Description
         JOBNAME     Job name of the address space
         StepName    Job step name, or TSO procedure name for TSO users
         ProcStep    Procedure step name, or terminal name for TSO users
         Type        Type of address space: job, started task,
                     TSO user, or initiator
         JNum        JES job number
         Owner       User ID of job creator
         C           JES input class at the time the job was selected for
                     execution
         Pos         Address space position, for example, swapped in,
                     swapped out, nonswappable, in transition
```

```
                  HELP: Display Active Users Panel -- Fields   Panel 9 of 9
 COMMAND INPUT ===>
Title       Description
         SLCPU%      Percentage of time the LPAR is busy for the system,
                     in the most recent interval.  The value for SLCPU%
                     is the same for all rows for a system. (RMF)

         For more information, select a topic by number:
               1 - Address space positions
               2 - Swap out reason codes
               3 - Server values
               4 - Displaying alternate fields
               5 - Changing the colors of fields
               6 - Changing the order of fields
               7 - CPU and SIO fields
               8 - Scaling and abbreviations for values
```

Figure 18-23 Selection 4 - Fields on DA panel (2 and 9 of 9)

```
   HELP: Display Active Users Panel -- Overtypeable Fields  Panel 1 of 1
 COMMAND INPUT ===>

         The following fields can be overtyped by authorized users.

             Field     Description
             SrvClass  Service class name
             Quiesce   Quiesce indicator (QUIESCE or RESUME)

         Overtyping these fields causes an MVS RESET command to be issued.
         SDSF appends an RO command if the MVS command is targeted
         for another system.

         For more information, select this topic by number:
          1 - About overtyping fields
```

Figure 18-24 Selection 5 - Overtypable fields to change their value

The HELP pop-up for the selection “5 - Overtyping fields to change their values” for the DA panel is shown in Figure 18-20 on page 870 and Figure 18-25 shows the HELP pop-up for selection “6 - Commands: limit jobs displayed, search, etc” of DA panel help.

```
                HELP: Display Active Users Panel -- Commands   Panel 1 of 1
COMMAND INPUT ===>
        A few useful commands are shown below.  Some require authorization

        FILTER column oper value - Filter jobs shown on any column
        FIND jobname - Search for a job name
        SET DISPLAY - Show settings for PREFIX, DEST, SORT, etc.
        SORT column-title or place the cursor on the title and press Enter
        SYSNAME system - Filter jobs by system name

        For complete command help, select a topic by number.

        1 - Search and scroll commands (FIND and others)
        2 - Filter commands (FILTER, OWNER, PREFIX, SYSNAME and others)
        3 - View commands (SORT and others)
        4 - Options commands (SET DISPLAY and others)
```

Figure 18-25 Selection 6 - Commands: limit jobs displayed, search, etc

Online HELP for SEARCH command

```
                            HELP: SEARCH Command              Panel 1 of 2
COMMAND INPUT ===>

    Purpose:  Search SDSF's help and tutorial. (ISPF and English only)

    Where used:  Any SDSF panel except help and tutorial panels.
        SEARCH lets you search and display help panels, but
        you enter the command outside of help.
    Format:  SEARCH (phrase)
        Searches the help and tutorial panels.
    Examples:  SEARCH cpu use - Searches for cpu use , cpu and use .
               SEARCH 'cpu use' - Searches for cpu use .

    For more information, select this topic by number:
        99 - Messages

_____________________________________________________________________________
                      HELP: SEARCH Command -- Format          Panel 2 of 2
COMMAND INPUT ===>
SEARCH (phrase)
-----------------------------------------------------------------------
     SEARCH with no search phrase displays the Search pop-up.
     phrase is the word or words (up to 4) you want to search for.
        When phrase is more than one word, SDSF searches
        for the complete set of words first, then for any one
        word. To search only for the whole phrase, enclose it
        in single quotation marks.
        Case is ignored, even when you enclose the phrase in
        quotation marks.
        You cannot search for single quotation marks in the data.
```

Figure 18-26 HELP popup for the SEARCH command

Figure 18-27 on page 875 is an example of the SEARCH command pop-up output.

The pop-up does not display PF settings. PF 8 and PF 7 are for scroll DOWN and UP respectively. PF 1 is HELP and PF 12 CANCEL.

Any character on a search result row will display the related help pop-up.

```
                              Search Help              Row 1 to 16 of 275
Command ===>

Search for: SEARCH                                          Show titles

_   Select a search result to display the help panel.
_   SEARCH with no search phrase displays the Search pop-up.
_   6 - SEARCH Search the help and tutorial
_   SEARCH - Search the help (ISPF only)
_   SEARCH - Search the help (ISPF only). Enter outside of help.
_   Use SEARCH to search the help (ISPF only). Type this
_   Search option on the action bar, or the SEARCH command.
_   You can add up to four search words to the SEARCH
_   command. SEARCH with no parameters displays the Search
_   SEARCH lets you search and display help panels, but
_   Examples: SEARCH cpu use - Searches for cpu use , cpu and use .
_   SEARCH 'cpu use' - Searches for cpu use .
_   invoke the Search function. With Search,
_   If you supply a search string, BookManager searches
_   The Search choice of the Search pull-down
_   The Search pop-up lets you search SDSF's online help.
_   98 - Search and navigate the help
```

Figure 18-27 SEARCH command output

HELP for SDSF messages

Each SDSF message has a help panel you can display to see an explanation and response to the message. Information about displaying the message helps is included on SDSF help panels. SDSF does not use the ISPF message services.

You can also search in online documents using the BOOK command (see the online help for more information). When the cursor is in the message area, BOOK uses the message text as a search string.

For example, in Figure 18-28, the INVALID COMMAND response is displayed on the right of the panel above the COMMAND INPUT line.

```
 Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 ----
  SDSF INPUT QUEUE DISPLAY ALL CLASSES                  INVALID COMMAND
  COMMAND INPUT ===>                                            SCROLL ===>
 PAGE
  NP   JOBNAME  JobID    Owner    Prty C         Pos  PrtDest           SAff
  AS
  ZZZZ BECKER1  JOB04066 BECKER      1 /              ANYLOCAL
```

Figure 18-28 SDSF message example

You can display help pop-up with the HELP key (default PF 1) or HELP command on the command line. From the pop-up select the `SDSF messages` option. When the `HELP: Messages`

`and Codes` pop-up is displayed, enter the first letter of the message (I) on the `COMMAN LINE` and hit Enter. The `HELP: Messages - 'I'` pop-up is displayed. By pressing Enter, Scroll forward the numbered list of short message texts until you find the subject message and use the number to display the pop-up explaining the meaning of the message.

Alternatively you can use the SEARCH command with the message text as the argument. Select the first line of the `Search Help pop-up` to display the pop-up explaining the meaning of the message.

Action characters

In most cases, action characters cause system commands to be issued for the object on the selected row. Both the ability to issue some action characters, and the command that is generated, depend on your installation options and operating system level.

The action characters are entered in the NP column of tabular panels.

The help for each SDSF panel also includes a list of the action characters that are valid for that panel.

You can display the valid action characters for a panel with the SET ACTION command:

```
SET ACTION (ON|LONG|SHORT|OFF|?)
```

Complete information about using SDSF action characters is provided in the online help for SDSF.

18.7 SDSF server address space

The SDSF server is an address space that SDSF uses to:

- Process ISFPARMS statements.
- Provide sysplex support for JES2. This consists of sysplex-wide data for JES2 devices and for system resources (CK, ENC, INIT, LI, NO, PR, PS, PUN, RDR, RM and SO panels) as well as the most recent SYSLOG data for remote systems (SYSLOG panel). Not required in a JES3 environment and when in a JES2 environment when all systems are at the z/OS V1R13 level.

  CK, ENC, PS and JES2 RM panels:

  An SDSF server is required on each system for sysplex-wide CK, ENC, PS and RM panels, in both JES2 and JES3 environments.

  - When all systems that you want to include are at the z/OS V1R13 level, SDSF uses XCF to communicate between SDSF servers, and does *not* use a server group defined in ISFPARMS.
  - When one or more systems that you want to include is at the z/OS V1R12 or lower level, the server group defined in ISFPARMS is also required, along with WebSphere® MQ.

A server is required in the JES3 environment to process ISFPRMxx member ISFPARMS statements. You start the server using the START command. The command takes the servername as a parameter. Optional START command parameters identify the suffix of PARMLIB member ISFPRMxx.

You can activate new parameters at any time with the MVS MODIFY operator command, which you can enter from the console, or from SDSF by users that are authorized to use the

slash (/) command. Changes take effect the *next* time users access SDSF. A TEST parameter allows you to check the syntax of the statements without activating them.

The server must be active on each system that contains SDSF users to process ISFPARMS statements,. To provide sysplex data, the server must be active on each system that is to be included on SDSF panels.

An SDSF server JCL sample is shown in Figure 18-29.

```
//SDSF     PROC  M=00,         /* SUFFIX FOR ISFPRMXX */
//             P='LC(T)'       /* USE SYSOUT CLASS T FOR SDSFLOG */
//SDSF     EXEC  PGM=ISFHCTL,REGION=32M,TIME=1440,PARM='M(&M),&P'
```

Figure 18-29 SDSF server JCL sample

18.8 SDSF security and ISFPARMS overview

SDSF global options and the format of the panels are defined with the ISFPARMS statements. The options include things such as the names of SDSF data sets, what generic and wildcard characters to allow in SDSF commands, and whether to display the action bar on SDSF panels. The format of the panels includes the order and titles of the columns.

SDSF for JES3 implementation requires the SDSF server address space to be started to process ISFPARMS. The server uses dynamic ISPFPARMS, which are defined with statements rather than assembler macros that can be used in JES2 SDSF. Statements are easier to code and are more dynamic than the assembler macros: they can be updated without reassembling or link-editing.

An overview of SDSF SAF security

To accomplish security through SAF, you permit or deny users access to the SDSF resources by use of their classes and resource names. In addition, you can supplement SAF security with the SAF exit points and installation exit routines.

The IBM-supplied class descriptor table provides a resource group class (GSDSF) and a resource member class (SDSF) for SDSF. For a resource group class, each user or group of users permitted access to that resource group is permitted access to all members of the resource group. For each GSDSF class created, a second class representing the members must also be created.

As with generic profiles, resource group class profiles enable you to protect multiple resources with one profile. However, the resources need not have similar names.

A resource group profile is a general resource profile with the following special characteristics:

- Its name does not match the resources it protects.
- The ADDMEM operand (not the profile name itself) specifies the resources it protects.
- Its class is a resource group class or grouping class (for example GSDSF).
- The related member class (not the resource group class itself) must be RACLISTed. For example, the SDSF class must be RACLISTed, not the GSDSF class. Depending on the class, RACLISTing is accomplished using the SETROPTS command or RACROUTE REQUEST=LIST macro service.

To accomplish SDSF security through SAF with RACF, you:

1. Activate generic processing before defining profiles, using the SETROPTS command.
2. Define profiles to protect the resources in the appropriate classes, using the RDEFINE command. (Classes are already defined for RACF. You must define them for other security products.)

   Begin with generic profiles for broad access to resources and then define generic or discrete profiles that are more restrictive.
3. Permit users to access appropriate profiles in each class with the necessary access levels, using the PERMIT command.
4. Activate the classes, using the SETROPTS command.

The sample ISFPARMS definitions in the ISF.SISFJCL(ISFPRM00) data set are used in the following discussion. This sample is included in Appendix A, “SDSF ISFPARMS default definitions” on page 991 for your reference.

The sample ISFPRM00 defines security for three SDSF groups of users that are common to most installations:

- Group 1 - System programmers. Users have JCL, OPER and ACCT TSO authority.
- Group 2 - Operators. Users have JCL and OPER TSP authority.
- Group 3 - General users. Users have JCL TSO authority. `ISF024I USER userid NOT AUTHORIZED TO SDSF, NO GROUP ASSIGNMENT` is issued if a user with none of the above TSO authorities attempts to invoke SDSF.

You have to choose SAF to protect SDSF functions in the JES3 environment. Even when SAF is used for all of SDSF security, you need ISFPARMS to control the following:

- Global values (OPTIONS statement)
- Any values for groups that are not related to security (GROUP statement)
- Code page - ITRTAB statement

The control of user membership into a group is accomplished with SAF profiles, as shown in Table 18-2.

Table 18-2 SAF profiles for controlling user membership in a group

User group	SAF profile	Access
System Programmers	GROUP.ISFSPROG.*	READ
Operators	GROUP.ISFOPER.*	READ
General users	GROUP.ISFUSER.*	READ

The access authorities to other profiles in classes SDSF, OPERCMDS, CONSOLE, WRITER, XFACILIT, and LOGSTRM control the actions allowed for members in a group. Refer to *z/OS SDSF Operation and Customization,* SA22-7670 for a complete description of the SAF profiles.

SDSF users do not need access authority to work with their own jobs and output. In general, all TSO users can access the JESSPOOL resources they own. When you provide SAF authority to the SDSF resources by group, go from broad access (for example, RACF generic profiles) to limited access (RACF discrete profiles).

System programmers need access to all profiles for each group in order to attain access to all resources. Likewise, the operators, in addition to having access to their own profiles, also need access to all profiles defined for users.

ISFPARMS statements in the JES3 environment

The statements for the JES3 environment that make up ISFPARMS are as follows:

OPTIONS Specifies global SDSF initialization parameters.

CONNECT Defines a server as a default server.

GROUP Defines a group of users and the SDSF functions that are available to a member of the group. It also includes initialization parameters. In the JES3 environment you must use SAF along with your group definitions to control membership and authorization.

FLD and **FLDENT** Customizes the fields shown on an SDSF primary or alternate tabular panels for members of the associated group.

NTBL and **NTBLENT** Specifies such things as user IDs, job names, and destination names to further qualify group membership and authority. Associated with a GROUP statement.

PROPLIST and **PROPERTY**
Specifies a property to customize. Provides an alternative to a user exit routine. Associated with a GROUP statement.

FLD/FLDENT statements and tabular displays

An FLD statement along with FLDENT statements, defines the fields that are displayed on a tabular SDSF panel. It is associated with the field list for a particular panel by a GROUP statement.

You can define a *primary* and *alternate* variable field list for each SDSF panel. The primary field list contains those fields that are shown upon entry into a panel. The alternate field list contains fields that can be displayed with the ? command.

It is also important to locate overtypable fields on the panel so that the entire field is visible on one panel. An overtypable field can be overtyped only when the entire field is visible.

The fields that are available on the display depend on your JES level and installation options. The ARRANGE command allows users to change the order and widths of the fields in each field list.

The columns on SDSF panels that display data in a tabular format are customized with FLD statements. The NAME on an FLD statement is referenced by a group. The TYPE on FLD statements name the SDSF panel for which the list of following FLDENT statements defines columns that are included on a tabular panel, as well as their order, titles, and widths.

. TITLE is the title that appears on a panel for the column defined by column. WIDTH is the width of the column on the panel.

REXX execs reference columns by their names rather than by their titles.

The syntax of FLD and FLDENT statements is shown in Figure 18-30.

```
FLD NAME(FLD-statement-name),TYPE(panel-ID)
       FLDENT COLUMN(column),TITLE(title),WIDTH(width)
```

Figure 18-30 FLD and FLDENT syntax

The source of the data for each column is extracted from either of the following:

- From in-storage control blocks. These columns are in the primary field list. SDSF performance is best when the columns with data from in-storage control blocks are at the beginning of the field list.
- From the JES spool data set, requiring an I/O operation. These columns are in the alternate field list. I/O operations are only done when the columns are visible on the panel or being sorted. SDSF performance is best when the columns with data from the spool data set are at the end of the field list.

```
Display Active Users panel (DA)       Enclaves panel (ENC)
Health Check History panel (CKH)      Health Checker panel (CK)
Held Output panel (H)                 Initiator panel (INIT)
Input Queue panel (I)                 JESPLEX panel (JP)
Job Class panel (JC)                  Lines panel (LI)
Network Connections (NC)              Network Servers (NS)
Nodes panel (NO)                      Output Descriptors panel (OD)
Output Queue panel (O)                Printer panel (PR)
Processes panel (PS)                  Punch panel (PUN)
Reader panel (RDR)                    Resource panel (RES)
Resource Monitor (RM) panel           Environment panel (SE)
Spool Offload panel (SO)              Spool Volumes panel (SP)
Status panel (ST)                     System Requests panel (SR)
```

Figure 18-31 Panel columns described in the z/OS SDSF Operation and Customization document

The *z/OS SDSF Operation and Customization,* SA22-7670 document has tables for panel column and column source descriptions. The panels are listed in Figure 18-31.

COLSHELP command

The COLSHELP command displays a table of column information for SDSF panels. In Figure 18-32 on page 881 the table display shows some of the columns and the column titles on the SDSF DA (Display Active Users) panel. The COLSHELP command requires ISPF. The ISPF PFSHOW command displays PF key settings for Columns on SDSF Panels. A slash (/) in the *ALL panels* and *Include descriptions* input fields indicates that the options have been selected.

```
                          Columns on SDSF Panels      Row 76 to 81 of 1,244
 Command ===>

 Sort with F5 (panel), F6 (column), F10 (title). Use Filter to filter rows.

 /  All panels            /  Include descriptions

 Panel     Column      Title                 Delayed?  Overtype?
    Description
 DA        JNAME       JOBNAME
    Job name, fixed field
 DA        STEPN       StepName
    Job step name (TSO logon proc for TSO users)
 DA        PROCS       ProcStep
    Procedure step name (terminal ID for TSO users)
 DA        JTYPE       Type
    Type of address space
F1=Help        F5=SortPnl     F6=SortFld     F7=Backward    F8=Forward
 F10=SortTtle   F11=SortDesc   F12=Cancel
```

Figure 18-32 COLSHELP command output panel

An X in the *Delayed?* column indicates that access for the column is delayed. Inclusion of these columns on a panel may impact SDSF performance.

By default, the table shows the columns for the SDSF panel on which it is issued. When you use the COLSHELP command from the SDSF primary option menu, the table includes columns for all panels. You may scroll to the first row for a specific panel with the LOCATE panel command. LOCATE can be abbreviated as L, for example, L DA.

Filter the columns based on panel name or title with the FILTER command. For more information, tab to the link and press the Help PF (F1).

If you cannot find a field on a tabular panel that you are interested in, use the COLSHELP command to display all fields of the panel you are working with. If the field you are interested in is listed in the Columns on SDSF Panels (COLSHELP) display, but is marked with an X in the *Delayed?* column, you can switch the panel field list to ALTERNATE to see the contents of the delayed field. To switch to the panel field list, select the action bar `View action` and then option *4. Change field list to PRIMARY/ALTERNATE* from the View pop-up.

Overtyping fields

In Figure 18-33 on page 882 is a sample display of a few rows on the SDSF INPUT QUEUE panel and a display of the COLSHELP command output for the overtypable job priority and job class columns. You can change the data on these columns by typing over it. For example, you can change the class of a job by typing a new value for the class. (JES3 command **`*F,J=jobno,C=new_class`** is issued for the overtype value.) Overtype columns are highlighted (red or green, by default) on panel displays.

```
  Display  Filter  View  Print  Options  Search  Help
--------------------------------------------------------------------------------
SDSF INPUT QUEUE DISPLAY ALL CLASSES                  LINE 1-26 (41)
COMMAND INPUT ===>                                          SCROLL ===> HALF
NP   JOBNAME  JobID    Owner    Prty C         Pos  PrtDest            SAff  AS
     VAINISCT JOB07219 VAINI       1 A              ANYLOCAL
:::::
_______________________________________________________________________________
                           Columns on SDSF Panels          Row 211 from 1244
  Command ===>

  Sort with F5 (panel), F6 (column), F10 (title). Use Filter to filter rows.

     All panels          Descriptions

  Panel     Column      Title                 Delayed?  Overtype?
  I         JPRIO       Prty                            X
  I         JCLASS      C                               X
```

Figure 18-33 Sample COLSHELP for JES3 I queue JPRIO and JCLASS columns

To be able to overtype a column you must be authorized and the entire column must be visible.You can use the tab key to move from one overtypable column to another. Blanking out a value with the space bar does not delete the value. Some fields, where the associated system command allows it, support deleting the value by typing a comma by itself in the field.

The help for each SDSF panel includes guidance on valid values for overtypable fields. In most cases, overtyping a field causes a system command to be issued.

The overtype extension function also lets you delete values when the field supports a set of related values. You can display the overtype extension pop-up by typing a + by itself in any overtypable field. Figure 18-34 on page 883 is an example of the overtype extension pop-up.

Overtype extension pop-up

When using the overtype extension pop-up, type a value on the pop-up. The input field on the pop-up may be longer than the maximum valid value for the column; use the maximum length displayed on the pop-up as a guide. To remove a value from a set of related fields, blank them out. To repeat an overtype, type = in the NP column to repeat the last action character or overtyped fields. The action character or overtype does not have to be on the immediately preceding row. You cannot enter another action character or overtype a field in the row where the **=** action character is.

```
  Display  Filter  View  Print  Options  Search  Help
--------------------------------------------------------------------------------
SDSF INPUT QUEUE DISPLAY ALL CLASSES                  LINE 1-7 (45)
COMMAND INPUT ===>                                          SCROLL ===> HALF
PREFIX=VAIN*  DEST=*  O
ACTION=//-Block,=-Repea                                   ,CA-CancelARM,
ACTION=CD-CancelDump,CD                                   splay,
ACTION=DE-DisplayEstima                                   DMA-DisplayMDSAlloc,
ACTION=DME-DisplayMDSEr                                   layMDSSysSel,
ACTION=DMSV-DisplayMDSS                                   playDDnames,
ACTION=DSH-DisplaySpool                                   isplayExtended,
ACTION=E-Restart,H-Hold                                   H-ListHold,
ACTION=LT-ListTCP,P-Pur                                   ,SE-ISPFEdit,
ACTION=SJ-JCLEdit,W-Spi                                   XDC-PrintDSClose,
ACTION=XF-PrintFile,XFC                                   PrintSysoutClose
NP   JOBNAME  JobID                                       t            SAff  AS
     VAINISCT JOB07219 VAINI       1 +              ANYLOCAL
     VAINISCT JOB07278 VAINI       1 A              ANYLOCAL
     VAINX11  JOB20554 VAINI       1 A              ANYLOCAL
     VAINX22  JOB20555 VAINI       1 A              ANYLOCAL
     VAINX33  JOB20556 VAINI       1 A              ANYLOCAL
```

```
Overtype Extension

Column C
Maximum length 8

Type a value.
===> A

F1=Help  F12=Cancel
```

Figure 18-34 Overtype extension pop-up

18.9 Working with JES3

SDSF offers many panels and features for interacting with JES3. This chapter covers the following:

- User authorization to SDSF panels and commands
- Display data filtering
- Viewing alternate forms (delayed columns) of a tabular SDSF panel
- The Input Queue (I) panel, which displays information about jobs on the JES input queue or that are executing.
- The Output Queue (O) panel, which displays information about output for jobs, started tasks, and TSO users on any nonheld output queue.
- The Held Output Queue (H) panel, which displays information about SYSOUT data sets for jobs, started tasks, and TSO users on any held JES output queue. There is one row for each held sysout class for each job.
- The Status (ST) panel, which displays information about jobs, started tasks, and TSO users on the JES queues.
- The Job zero (J0) panel, which displays information about JES3 job zero (aka JES3 or J=0). With this panel, you can work with data sets that were created by JES3.
- Viewing output job output using the S action character, or the SB, SE, or SJ action characters to invoke ISPF Browse or Edit.

  The Job Data Set (JDS) panel displays information about the SYSOUT data sets for a job, started task, or TSO user. The JDS panel is accessed with the **?** action character.

  To view output formatted for a page printer, you can use the V action character, which requires GDDM® (Version 2 Release 2 or a later release).

- The Initiator (INIT) panel, which displays information about JES-managed and WLM-managed initiators.
- Printers, punches and readers (PR, PUN and RDR) panels that display JES3 printer, punch and reader support unit information and jobs on the units.
- The Nodes (NO) panel, which displays information about nodes node in an NJE network.
- The Spool Volumes (SP) panel, which allows users to display information about JES spool volumes and spool partitions.
- The Network Server (NS) panel, which displays information about server-type networking devices on the node:
  - NETSERV devices used to communicate between JES and TCP/IP
  - Bulk Data Transfer (BDT) instances used to communicate between JES3 and VTAM
- The Network Connection (NC) panel, which displays information about networking connections to an adjacent node:
  - SOCKET devices that represent a TCP/IP networking connection
  - Active Binary Synchronous Communication (BSC) NJE lines
  - Associated NJE transmitters and receivers
- The JESPlex (JP) panel, which displays and controls the main processors of a JES3 complex.
- The Job Class (JC) panel, which allows users to display and control the job classes in a JES3 complex. It shows both the JES and the WLM managed classes.
- The hardcopy log (LOG) panel, which displays a merged, sysplex-wide system message log, which contains console messages, operator commands, and operator responses for the MVS systems.

  The term "hardcopy log" refers to the system log (SYSLOG), the operations log (OPERLOG) and the JES3 DLOG. The DLOG centrally records command and message traffic for systems in a JES3 complex in JES3 format. The JES3 DLOG is written to SYSLOG on the global processor.
  - Access OPERLOG with the LOG O command. Messages are displayed in their original color.
  - Access MVS SYSLOG or JES3 DLOG with the LOG S command.
- The user session log (ULOG) panel, which displays the MVS and JES commands and responses issued during the user's session, including commands generated by SDSF and SAF. SDSF deletes the user session log when an SDSF session is ended or when the ULOG CLOSE command is issued.

  SDSF uses MVS console services to acquire an extended console that is used to issue commands and receive responses.

Users' authorization

The target objects (for example, the job, output group, initiator, or printer) of the SDSF action characters are controlled as resources in the SAF SDSF class and in the JESSPOOL class. JES uses the JESSPOOL class to protect SYSIN/SYSOUT data sets. Printers and punches are controlled in the WRITER class. Most SDSF action characters generate MVS or JES commands that are protected in the OPERCMDS class.

SDSF uses a console when issuing MVS or JES commands that were entered with a / command. The console used varies:

- If the user session log (for display on the ULOG panel) is active, SDSF uses an extended console.
- If the user log is not active, SDSF uses a console ID of 0.

CONSOLE class controls access to MCS consoles. It may also be used for restricting access (conditional access) to other resources in WRITER and OPERCMDS for commands originating from an MCS console.

SAF security provides a dynamic means of authorizing SDSF users to issue commands and process job output. Once a user starts an SDSF session, SDSF checks user authorization for virtually every interaction with SDSF resources. SAF authorization dynamically affects the next user interaction. You must end an SDSF session and restart it when changes are made to SAF authorization for destination names and for operator authority by destination.

If you are using RACF as a security product, RACF logs access attempts to protected SDSF resources according to the audit setting in the RACF profile for the resource. Logging is performed for all access attempts except for the following resource names in the SDSF class:

- ISFOPER.DEST.**
- ISFOPER.ANYDEST.jesx
- All resource names beginning with ISFATTR.

Logging is not performed for these access attempts because the user is not specifically trying to gain access to those resources.

SAF protection

Protection for each type of resource can be defined separately, so that, for example, a user may be authorized to issue action characters for a job, but not be authorized to browse that job's data sets. Users can always access the JESSPOOL resources they own; they do not need additional authority to work with their own jobs and output.

SDSF checking authority requirements to JESSPOOL class resources:

- Action characters `//`, `=`, `+`, `?`, or `Q` on the Active users (DA), Input queue (I), Output queue (O), Held output queue (H), Job data set (JDS), Output descriptor (OD) and Status of jobs (ST) panels:

  No SDSF security checking is done.
- Action characters S (browse), X (print), or V (view) on the DA, I, O, H, JDS, OD, and ST panels:

  READ access to the *nodeid.userid.jobname.jobid.Ddsid.dsname* resource
- Action character SJ (JCLEdit) on the DA, I, O, H, JDS, OD, and ST panels:

  READ access to the *nodeid.userid.jobname.jobid.JCL* resource
- Action character SB (ISPFBrowse), SE (ISPFEdit) on the DA, I, O, H, JDS, OD, and ST panels:

  READ access to the *nodeid.userid.jobname.jobid.JESMSGLG* resource

  READ access to the *nodeid.userid.jobname.jobid.JESYSMSG* resource
- Action characters D (display) and L (list) on the DA, I, and ST panels:

  READ access to the *nodeid.userid.jobname.jobid* resource

- All other action characters on the DA, I, and ST panels:

  ALTER access to the *nodeid.userid.jobname.jobid* resource
- All other action characters on the JDS and OD panels:

  ALTER access to the *nodeid.userid.jobname.jobid.Ddsid.dsname* resource

Where:

nodeid Is the NJE node ID of the target JES subsystem.

userid Is the local user ID of the job owner.

jobname Is the name of the job.

- jobid Is the JES job ID of the: Job (for jobs on DA, I, and ST)
- Job with which the data set is associated (for SYSIN or SYSOUT data sets).

Ddsid Is the data set ID number that identifies the job data set prefixed by the required letter D.

dsname Is the user-specified or system-assigned data set name.

???Typically, when you define SAF authority for JESSPOOL resources, you also need to define other authorities for action characters and overtypable fields. For most action characters, a user must be authorized for jobs or job output. However, the S, V, and X action characters require authorization only for SYSIN/SYSOUT data sets. No security checking is done for the object when the ? or Q action characters are used.

Some other profiles for commands generated by SDSF action characters are also required, such as:

- In the OPERCMDS class: JESx.** profiles for JES3 commands
- In the OPERCMDS class: MVS.** profiles for MVS commands

To protect resources individually in the OPERCMDS class with restrictive profiles, you would use the specific resource name for the command generated by the action character.

Authorized SDSF commands are protected by defining resource names in the SAF SDSF class profiles ISFCMD.** with READ access. These commands include:

ABEND, ACTION, CK, DA, DEST, ENC, FINDLIM, H, I, INIT, INPUT, JC, the JESNAME parameter on the SDSF command, the JES3NAME parameter om on the SDSF command, JP, J0, LI, LOG, NC, NO, NS, O, OWNER, PR, PREFIX, PS, PUN, RDR, RES, RSYS, SE, the SERVER parameter on the SDSF command, the SDSF command, SO, SP, SR, ST, SYSID, SYSNAME, TRACE, and ULOG

Notes:

- In a JES3 environment, when an SAF class for a resource is inactive, or the profile to protect the resource is not defined, requests will be failed.
- The ISPFPARM's GROUP function AUTH parameter (authorized-command-list) applies to JES2 only.

For a complete description of protecting SDSF, refer to Chapter 7 in *z/OS SDSF Operation and Customization,* SA22-7670.

Resource group profiles

The IBM-supplied class descriptor table provides a resource group class (GSDSF) and a resource member class (SDSF). For a resource group class, each user or group of users

permitted access to that resource group is permitted access to all members of the resource group. For each GSDSF class created, a second class representing the members must also be created.

A resource group profile is a general resource profile with the following special characteristics:

- Its name does not match the resource it protects.
- The ADDMEM operand of the RDEFINE command specifies the resources it protects (not the profile name itself).
- The related member class (not the resource class itself) must be RACLISTed. For example, the SDSF class must be RACLISTed, not the GSDSF class. Use the SETROPTS command with the RACLIST operand for this task.

18.9.1 Filtering display data

Filtering limits data on SDSF panels. SDSF data filtering can be set either using the Filter pull-down of the action bar or filter commands.

Filter option of the action bar

The Filter pull-down of the action bar allows you to filter the data displayed on SDSF panels. SDSF displays a list of filtering choices in the pull-down:

```
1. Filter...
2. Prefix of jobname...
3. Owner...
4. Destination...
5. System name...
*. Change APPC to OFF    (JES2 environment only)
7. Replies on the Log...
```

To select the choice, type the number of the choice or place the cursor on the choice and press Enter. Under ISPF, the values you specify are saved across sessions.

1. Filter Displays Filter pop-up (Figure 18-35 on page 888).

```
 Displa                                                                  
--------                          Filter         Row 1 to 9 of 25        ---------
SDSF STA  Command ===>                                                   
COMMAND                                                                  ==> HALF
PREFIX=V  Type filter criteria.  Type a / in the Column or Oper          
ACTION=/  fields for valid values. Press F11/23 to clear all             M,
ACTION=C  filter criteria.                                               
ACTION=D                                                                 DSAlloc,
ACTION=D  Filtering is  OFF                                              ,
ACTION=D                                                                 
ACTION=D  AND/OR between columns   AND  (AND/OR)                         ed,
ACTION=E  AND/OR within a column   OR   (AND/OR)                         
ACTION=L                                                                 ,
ACTION=S  Column             Oper  Value (may include * and %)           lose,
ACTION=X                                                                 lose
NP   JOB                                                                 Sys     S
     VAI                                                                         H
     VAI                                                                         H
     VAI                                                                         H
```

Figure 18-35 Filter pop-up on top of a STATUS DISPLAY

You can either type the column names directly or select them from a prompt pop-up.

You can abbreviate the column name to the shortest string of characters that uniquely identifies that column. The value field data may include * and % pattern matching characters.

Prefix of jobname Type a prefix to limit jobs on the DA, H, I, O, PS and ST panels. The prefix string may include * and % pattern matching characters.

Owner Type an owner to limit jobs on the DA, H, I, O, PS and ST panels. The owner string may include * and % pattern matching characters.

Destination Type up to four destinations to limit jobs on the H, I, J0, O, ST, PR and PUN panels. Only those jobs whose names match the destination are displayed.

To delete a destination, simply blank it out. Blank out all destinations on the pop-up to display jobs for all destinations, or for the destinations named filter criteria in the IDEST parameter of ISFPARMS if one is coded.

System name Type a system name or leave blank for the system you are logged on to. The system name string may include * and % pattern matching characters.

Replies on the Log Type a system name to limit WTORs on the Log panels. Leave blank for the system you are logged on to. The system name string may include * and % pattern matching characters.

Filter commands

Filter commands limit data on the SDSF panels. Under ISPF, filters are saved (one set for each JES type).

Figure 18-36 on page 889 lists the SDSF filter commands.

```
                        HELP: Commands to Filter
 COMMAND INPUT ===>

   Select a command by number, or press Enter to view them in sequence.

             1 - ACTION     Control the display of WTORS on the Log
             2 - APPC       Control the display of APPC transactions *
             3 - DEST       Limit jobs displayed by destination
             4 - FILTER     Limit data on SDSF panels
             5 - OWNER      Limit job displays by owner ID
             6 - PREFIX     Limit jobs displayed by job name or netmail ID
             7 - RSYS       Limit WTORS on the Log panels by system
             8 - SELECT     Display selected rows
             9 - SYSID      Specify the SYSLOG data set for the LOG panel *
            10 - SYSNAME    Limit data by system

           * JES2 only

F1 = Help             F10 = Previous          Enter = Forward
 F3 = Exit              F7 = Up                 TOC = Menu
```

Figure 18-36 SDSF commands to limit data on panels (Filter)

Filter command

The `FILTER` command format is shown in Figure 18-37.

```
FILTER ON | OFF | OR | FIL | (+|-)column (operator) value | ?

OFF turns filtering off but retains filter criteria.
ON turns filtering on.
OR and AND specify the relationship between filters both within a column and between
columns.
(+|-)column names a column for filtering and turns filtering on. Column name can be
abbreviated to the shortest unique name.
 + adds the filter to any previous filters. There is a limit of 25 filters under ISPF,
but no limit with isffilter under REXX. When you use this with isffilter, you must
specify an operator.
 - discards all filters for the column. (ISPF only)
operator is one of the following:
EQ or =    Equal (the default)     NE or ¬=   Not equal
LT or <    Less than               GT or >    Greater than
LE or <=   Less than or equal      GE or >=   Greater than or equal
Operators with less than or greater than are valid only when the value does not contain
pattern matching characters (* and % by default).
value is the value to be used for comparison.  value can contain pattern matching
characters.  If it includes embedded blanks, enclose it in quotation marks.
? displays filters and their current state. Under ISPF, it
displays the Filter pop-up.
```

Figure 18-37 FILTER command format

Use pattern matching characters (* and % by default) for an inexact or partial match. For example:

```
FILTER JOBNAME EQ %A* matches jobs with a name that has A in the second
position.
```

You can change the pattern matching characters with the SET SCHARS command.

ACTION command

The ACTION command controls the display of Write-To-Operator-with-Reply (WTOR) messages on the log by specifying which WTOR messages are displayed at the bottom of the Log panel. You must be authorized to use this command. ACTION may be used on any SDSF panel. The ACTION command format is shown in Figure 18-38.

```
ACTION routing-code-list | ?
routing-code-list is up to four routing codes separated by blanks (1-28)
MVS is all routing codes reserved for MVS (1-12).
USER is all routing codes reserved for customer use (13-28).
ALL requests the display of WTORs for all routing codes.
OFF requests the display of no WTORs.  This is the default.
? displays the current setting for ACTION on the message line.
Use up to 4 parameters. The routing-code-list, MVS, and USER parameters may be combined.
ACTION commands are cumulative.
```

Figure 18-38 ACTION command format

DEST command

The DEST command limits jobs to be selected for display by destination. You must be authorized for the command and for the destination. The DEST command may be used on any panel. It affects only the ST panel. The DEST command format is shown in Figure 18-39.

```
DEST (+ or -) (destination-names) | ?
+ add-destination-names
- delete-destination-names
destination-names are destination names of up to 18 characters.  Enter up to 4
destination names.
? displays the current setting on the command line or pop-up.
```

Figure 18-39 DEST command format

OWNER command

The OWNER command limits jobs selected for display by owner ID. You must be authorized to use this command. OWNER may be used on any SDSF panel but affects only the DA, I, PS, and ST panels. The OWNER command format is shown in Figure 18-40.

```
OWNER ownerid|?
ownerid is the owning user ID of the job, or the netmail ID.  It can be up to 8
characters including * (any string of characters) or % (any single character).
?  displays the current setting on the command line or pop-up.
OWNER with no parameters displays all jobs.
```

Figure 18-40 OWNER command format

PREFIX command

The PREFIX command limits jobs selected for display by job name or netmail ID. This command may be used on any SDSF panel, but affects only the DA, I, PS, and ST panels. The PREFIX command format is shown in Figure 18-41 on page 891.

```
PREFIX string | ?
string is the name of the job, up to 8 characters, including * (any string of
characters) or % (any single character).
? displays the current setting on the command line or pop-up.
PREFIX with no parameters displays all jobs, except on the Held Output Queue panel,
where it displays all jobs with names that begin with your user ID.
```

Figure 18-41 PREFIX command format

RSYS command

The RSYS command limits WTORs displayed at the bottom of the Log panels. You must be authorized for this command. This command may be used on any SDSF panel, but affects only the syslog and operlog panels. The RSYS command format is shown in Figure 18-42.

```
RSYS system-name|?
 system-name is the MVS system name, up to 8 characters, including * (any string of
 characters) or % (any single character).
 ? displays the current setting on the command line or pop-up.
 RSYS with no parameters displays only WTORS from the system you are logged on to.
```

Figure 18-42 RSYS command format

SYSNAME command

The SYSNAME command specifies the systems in the sysplex that are included on the CK, DA, ENC, and PS panels. You must be authorized to use this command. The SYSNAME command may be used on any SDSF panel. This command format is shown in Figure 18-43.

```
SYSNAME system-name|?
   system-name is the MVS system name, up to 8 characters, including * (any string of
   characters) or % (any single character).
   ? displays the current setting on the command line or pop-up.
   SYSNAME with no parameters displays only data for the system you are logged on to.
```

Figure 18-43 SYSNAME command format

SELECT command

The SELECT command temporarily limits the jobs (rows) displayed on tabular panels. This command only lasts until you exit the panel or issue another SELECT with no parameters. The SELECT command may be used on any tabular panel. Its format is shown in Figure 18-44 on page 892.

```
SELECT | S (selection-criteria)
   SELECT with no parameters removes any filtering done with SELECT.
   selection-criteria specifies the rows to be selected. The selection criteria varies
   depending on the current panel.
Queue panels (DA and ST):
   jobname {jobid}. The jobid is the job number. You do not need leading zeros.
   job number. You do not need to type leading zeros.
On these panels, SELECT overrides other filters (parameters on panel commands, FILTER,
and, if you are authorized, PREFIX, OWNER and DEST. For DEST, you must also be
authorized to the destination).

JDS panel:
   ddname {stepname}
   dsid
CK panel:
   checkname {checkowner}
You may use special characters (* and %) except with jobid.
```

Figure 18-44 SELECT command format

SET TIMEOUT command

The SET TIMEOUT command sets the timeout value for awaiting sysplex data on browse panels. Its format is shown in Figure 18-45.

```
SET TIMEOUT timeout-value| ?
   timeout-value specifies the default timeout value (in seconds). The timeout
   value must be in the range of 0 to 9999, where 0 indicates that SDSF does not wait,
   that is, the sysplex support for device panels is disabled. When the sysplex support
   is disabled, the device panels show only the devices for the system you are logged on
   to.
   ? displays the current setting on the command line or pop-up.

   SET TIMEOUT with no parameters results in the timeout value specified in ISFPARMS.
```

Figure 18-45 SET TIMEOUT command format

Note: The sysplex-wide DA panel requires RMF in the JES3 environment. Some of the values on the DA panel, such as CPU% and SIO, are approximate. For detailed and precise performance monitoring, use RMF.

18.9.2 View alternate form of a tabular SDSF panel fields

The ? command (not NP field action) displays the alternate form of a tabular panel. You may need to scroll right to see the alternate fields.

The action bar `View` pull-down choice *4. Change field list to ALTERNATE* also toggles the display of the primary or alternate fields on SDSF tabular panels.

Locate command

The LOCATE command can be used to scroll a panel to a specific line or column. Its syntax is shown in Figure 18-46 on page 893.

```
{LOCATE|LOC|L} [line-number | time | time-date | column]

  line-number is up to 8 digits (Log and Output Data Set panels) time is a
  time of day in the form hh:mm:ss or hh.mm.ss (Log panels)
  time-date is the time and date in the current format (Log panels)
  column-heading is the heading of the column to be located (tabular
  panels). The panel is scrolled horizontally so that the column is the
  first column after the fixed field.
```

Figure 18-46 LOCATE command syntax

Note: If LOCATE column returns with message `COLUMN NOT FOUND`, switch the panel view (? command) and retry the LOCATE command. If the `COLUMN NOT FOUND` message is repeated, use the COLSHELP command to check the spelling of the column title you are trying to locate.

18.10 Input queue (I) panel

The SDSF input queue panel displays information about jobs that are on the JES input queue or that are executing. The input display is accessed with the I command from any SDSF panel. Figure 18-47 shows the syntax of the I command.

```
I[class] [H|NH]

I with no parameters displays all jobs in all classes (but not TSO users or started
tasks). The jobs displayed may be limited by your authorization and by filter settings
such as PREFIX or FILTER.

class - displays information for a specific input class. Enter a single class, up to 7
characters. You can also use special characters for class:
   @ - jobs waiting to be transmitted to another node.
   $ - TSO users
   # - started tasks
   ! - hardcopy queue
The hardcopy queue contains all jobs that have any type of output in the system.
Accessing the hardcopy queue by using the I command allows you to find output for a job,
whether it is on a held or nonheld JES output queue.
   H - displays only held jobs.
   NH - displays only jobs that are not held.
Examples:
   IA H        Displays jobs in classes A that are held.
   IA NH        Displays jobs in class A that are not held.
   I$           Displays the input queue for all TSO users.
```

Figure 18-47 I command syntax

Input queue panel fields

The input queue panel in the JES3 environment may include the following columns, shown in Figure 18-48 on page 894. (The order and titles may be different, depending on installation and user options.)

Title	Description
JOBNAME	Job name of the address space
JobID	JES job ID, or work ID
Type	Type of address space: job, started task, TSO user, or initiator
JNum	JES job number
Owner	User ID of job creator
Prty	JES input queue priority
C	JES input class
Pos	Position in the JES input queue
PrtDest	JES print destination name or default print routing
SAff	JES execution system affinity (if any)
ASys	JES execution system ID
Status	Status of job
SecLabel	Security label of job
OrigNode	Origin node name
ExecNode	Execution node name
Device	Device or JES processor name
PhaseName	Name of the job phase
Phase	Number of the job phase
SrvClass	Service class
Dly	Indicator that job processing is being delayed. Use the I action character for details.
Mode	Subsystem managing the job (WLM or JES)

Figure 18-48 Input queue panel columns

Figure 18-49 lists the delayed access fields on the Input queue panel (except Spin).

WPos	Position in the WLM queue
Scheduling-Env	Scheduling environment for the job
RNum	Room number on job card
Programmer-Name	Programmer name
Acct	Account number
Notify	TSO user ID from the NOTIFY parameter
ISys	JES input system ID
Rd-Time	Time the job was read in
Rd-Date	Date the job was read in
ESys	JES execution system ID
St-Time	Time execution began
St-Date	Date execution began
Cards	Number of cards read for the job
MC	MSGCLASS of the job
Tot-Lines	Total number of spool records for job
Spin	Indicator that jobs in the class can be spun
SubGroup	Submittor group
SubUser	Submittor user
JobAcct1-5	Account fields from the job card

Figure 18-49 Input queue columns with delayed access

Figure 18-50 on page 895 shows a snippet of an SDSF input queue PRIMARY field list display.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF INPUT QUEUE DISPLAY ALL CLASSES                    LINE 1-13 (50)
COMMAND INPUT ===>                                         SCROLL ===> HALF
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=SC75
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,A-Release,C-Cancel,CA-CancelARM,
ACTION=CD-CancelDump,CDA-CancelARMDump,CP-CancelPrint,D-Display,
ACTION=DE-DisplayEstimates,DL-DisplayLong,DM-DisplayMains,DMA-DisplayMDSAlloc,
ACTION=DME-DisplayMDSError,DMR-DisplayMDSRestart,DMSS-DisplayMDSSysSel,
ACTION=DMSV-DisplayMDSSysVer,DMU-DisplayUnavailVol,DSD-DisplayDDnames,
ACTION=DSH-DisplaySpoolHold,DSP-DisplaySpoolPartition,DX-DisplayExtended,
ACTION=E-Restart,H-Hold,I-Info,J-Start,L-List,LB-ListBDT,LH-ListHold,
ACTION=LT-ListTCP,P-Purge,Q-OutDesc,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,
ACTION=SJ-JCLEdit,W-Spin,X-Print,XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,
ACTION=XF-PrintFile,XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose
NP   JOBNAME  JobID    Owner    Status            Prty C         Pos  PrtDest
     BECKER1  JOB04066 BECKER                       1 A              ANYLOCAL
     MDSWAIT  JOB14437 VAINI                        1 A              ANYLOCAL
     MDSWAIT0 JOB14440 VAINI                        1 A              ANYLOCAL
     MDSWAIT2 JOB14442 VAINI                        1 A              ANYLOCAL
     NOVOLS   JOB14503 VAINI                        1 A              ANYLOCAL
```

Figure 18-50 Input queue panel with SET ACTION LONG data

Tip: You can define a primary and alternate variable field list for each SDSF panel. The primary field list contains those fields that are shown upon entry into a panel. The alternate/primary field list contains fields that can be displayed with of the ? command. In alternate field display the ? command toggles to the primary fields display.

Action bar View option choice 4. `Change field list to PRIMARY/ALTERNATE` allows you to select between the primary and alternate variable field list display.

Figure 18-51 displays some fields on the SDSF input queue ALTERNATE field list for the Input queue panel.

```
 Display  Filter  View  Print  Options  Search  Help
 -------------------------------------------------------------------------------
 SDSF INPUT QUEUE DISPLAY ALL CLASSES                    LINE 1-13 (50)
 COMMAND INPUT ===>                                         SCROLL ===> HALF
 PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=SC75
NP   JOBNAME   ISys     Rd-Time  Rd-Date  ESys     St-Time  St-Date  Cards MC
      BECKER1   SC75     11:29:11 2006.130          0:00:00 0000.000     8 H
      MDSWAIT   SC75     11:22:00 2009.117          0:00:00 0000.000     3 A
      MDSWAIT0  SC75     11:52:08 2009.117          0:00:00 0000.000     3 A
      MDSWAIT1  SC75     11:52:14 2009.117          0:00:00 0000.000     3 A
      MDSWAIT2  SC75     11:52:22 2009.117          0:00:00 0000.000     3 A
      NOVOLS    SC75     13:06:42 2009.120          0:00:00 0000.000     4 T
```

Figure 18-51 Input queue panel with some ALTERNATE fields and with SET ACTION OFF

Tip: The SET SCREEN command displays a panel that allows you to set the colors, highlighting, and intensities used on SDSF panels, and control display of the action bar. It is valid only if SDSF was accessed through ISPF. The values are saved across SDSF sessions.

The ARRANGE (parameters) command allows you to reorder and change the widths of columns on the current panel:

```
ARRANGE from-column A|B to-column
ARR     from-column FIRST|LAST|width
        DEFAULT
        ?
```

from-column, to-column each name a column on an SDSF panel.

The column can be abbreviated to the shortest name that is unique for that panel.

A moves from-column after to-column.

B moves from-column before to-column.

FIRST or **F** makes from-column the first column after the fixed field (the first column). The fixed field cannot be moved.

LAST or **L** makes from-column the last column (furthest to the right).

width sets the width of from-column; it is a number (1-127).

DEFAULT resets the column arrangement to the default

? under ISPF, displays the Arrange pop-up.

Under ISPF the ARRANGE criteria are saved (one set for each JES type).

When a value is too large to fit in a column, SDSF scales the value using these abbreviations:

K Kilo (hexadecimal scaling)
T Thousands (decimal scaling) or Tera (hexadecimal scaling)
M Millions (decimal scaling) or Mega (hexadecimal scaling)
B Billions (decimal scaling)
G Giga (hexadecimal scaling)
P Peta (hexadecimal scaling)
KB Kilobytes
MB Megabytes
GB Gigabytes
TB Terabytes
PB Petabytes

The SORT command sorts the rows on the current tabular panel, including its alternate form (displayed with the ? command). It is available on any panel that displays tabular data.

```
SORT (major-column) (A or D) (minor-column) (A or D)
     (OFF) or (?)
```

with no parameters sorts a panel in ascending order using the fixed output field for that panel.

Input queue panel NP field actions

Figure 18-52 on page 897 lists actions available on the SDSF input queue display and the JES3 command that it uses to get the response information.

The list is in ascending sort order by NP field. The JES3 commands and command responses resulting from actions are recorded in the hardcopy log.

NP-Description	JES3 command / SDSF action
?-JDS	SDSF job data set display
A-Release	*F J=jobno,R - Release a held job
C-Cancel	*F J=jobno,CO - Cancel a job and process output data sets
CA-CancelARM	*F J=jobno,C,ARMR - Cancel a job that is defined to Automatic Restart Manager (ARM)
CD-CancelDump	*F J=jobno,C,D - Cancel a job and take a dump
CDA-CancelARMDump	*F J=jobno,C,D,ARMR - Cancel a ARM defined job and take a dump
CP-CancelPrint	*F J=jobno,CP - Cancel a job and print data sets
D-Display	*I J=jobno - Display job information in the log
DE-DisplayEstimates	*I J=jobno,E - Display line, page, record, and card counts
DM-DisplayMains	*I J=jobno,M - Display eligible mains
DMA-DisplayMDSAlloc	*I S,A,J=jobno - Display MDS allocate queue information
DME-DisplayMDSError	*I S,E,J=jobno - Display MDS error queue information
DMR-DisplayMDSRestart	*I S,R,J=jobno - Display MDS restart queue information
DMSS-DisplayMDSSysSel	*I S,SS,J=jobno - Display MDS system select allocate queue
DMSV-DisplayMDSSysVer	*I S,SV,J=jobno - Display MDS verify queue information
DMU-DisplayUnavailVol	*I S,U,J=jobno - Display MDS unavailable volume queue
DSD-DisplayDDnames	*I J=jobno,SD - Display spool DD-names for a job
DSH-DisplaySpoolHold	*I J=jobno,SH - Display held spool DD-names for a job
DSP-DisplaySpoolPartition	*I J=jobno,SP - Display spool partitions for a job
DX-DisplayExtended	*I J=jobno,X - Display extended information for a job
E-Restart	*R main,jobno - Restart a job
H-Hold	*F J=jobno,H - Hold a job
I-Info	SDSF job information pop-up (See Figure 18-54 on page 898)
J-Start	*F J=jobno,RUN - Start a WLM-managed job immediately
L-List	*I U,Q=WTR,J=jobno - List Q=WTR output status of a job
LB-ListBDT	*I U,Q=BDT,J=jobno - List Q=BDT output status of a job
LH-ListHold	*I U,Q=HOLD,J=jobno - List Q=HOLD output status of a job
LT-ListTCP	*I U,Q=TCP,J=jobno - List Q=TCP output status of a job
P-Purge	*F J=jobno,C - Cancel a job and purge its output
Q-OutDesc	SDSF output descriptor display
S-Browse	SDSF browse of a job's spool data set data
SB-ISPFBrowse	ISPF browse of a job's spool data set data
SE-ISPFEdit	ISPF edit of a job's spool data set data
SJ-JCLEdit	ISPF edit of a job's JCL
W-Spin	*F J=jobno,SPIN - Cause job and message logs to spin
X-Print	Print a job's spool data
XC-PrintClose	Print a selected spool data and close SDSF print
XD-PrintDS	SDSF open SDSF print and print selected spool data
XDC-PrintDSClose	Open SDSF print, print spool data and close SDSF print
XF-PrintFile	Print the selected spool data using DD-name
XFC-PrintFileClose	Print the selected spool data using DD-name and close
XS-PrintSysout	SDSF open SYSOUT data set and print selected data
XSC-PrintSysoutClose	Open SDSF SYSOUT, print selected data and close SYSOUT

Figure 18-52 Input queue NP field actions

Input queue overtypable fields - JES3 commands

You can change the columns listed in Figure 18-53 on page 898 on the SDSF JES3 input queue panel by typing over them.

```
Title                JES3 command - Action
C                    *F J=,C= - Change job's JES3 class
Prty                 *F J=,P= - Change job's JES3 priority
```

Figure 18-53 Input queue panel overtypable fields

Job information pop-up for NP action I

Figure 18-54 displays the Job Information panel when action character I (Info) is entered for a row on the panel.

```
   Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF INPUT QUEUE DISPLAY ALL CLASSES                  LINE 49-57 (57)
COMMAND INPUT ===>                                          SCROLL ===> HALF
AC
AC                              Job Information
I

     Job name         VAINIAR    Job class limit exceeded? NO
     Job ID           JOB20830   Duplicate job name wait?  YES
     Job schedulable? NO         Time in queue             N/A
     Job class mode   JES        Average time in queue     N/A
     Job class held?  NO         Position in queue         N/A    of N/A
                                 Active jobs in queue      N/A

     Scheduling environment JES3             available on these systems:

     F1=Help  F12=Cancel
```

Figure 18-54 Job Information pop-up

The Job Information pop-up displays:

- Job name and ID.
- Job class mode. Manager of the job class: WLM or JES.
- Job schedulable. If this is NO, look to the fields that follow for the reason. Other reasons are possible, including:
  - The job is held (shown in the STATUS column on the panel)
  - The job has affinity for a system that is in independent mode, or is stopped, or for which job scheduling is stopped.

  If this is YES and the job is not running, there may be no available initiators. For WLM-managed job classes, the systems must be in goal mode before jobs can be initiated.
- Job class held. YES if the job is in a held job class.
- Job class limit exceeded. YES if the limit for the number of jobs executing in the job class has been reached.
- Duplicate job name. YES if another job executing in the MAS has the same name as this one and the system cannot accept duplicate names.
- Time in queue. The amount of time the job has been in the job queue, waiting to be initiated. WLM-managed job classes only.

- Average time in queue. The average time that a job in the service class waits to be initiated. WLM-managed job classes only.
- Position in queue. The position of the job compared to the total number of jobs in the service class waiting to be initiated. WLM-managed job classes only.
- Active in queue. The number of jobs in the service class that are executing. WLM-managed job classes only.
- Scheduling environment. The scheduling environment required for the job, and the systems in the MAS, if any, on which the scheduling environment is available.

Input queue NP action D-display example

SDSF Input queue panel displays status HOLD for jobs VAINZLL to VAINZMM. The response for action **D-Display** for job VAINZLL is shown in Figure 18-55. The response indicates that the job is in a JES3 DJC net. Job VAINZ11 is in operator hold and can be released with an **A-Release** NP action character.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF INPUT QUEUE DISPLAY ALL CLASSES                    COMMAND ISSUED
COMMAND INPUT ===>                                        SCROLL ===> HALF
RESPONSE=SC75      IAT8674 JOB VAINZLL  (JOB20600) P=01 CL=A        NET=NT2
RESPONSE=    HOLD=(N) CI(RESCHEDULED)
RESPONSE=SC75      IAT8699 INQUIRY ON JOB STATUS COMPLETE,       1 JOB
RESPONSE=DISPLAYED
     VAINZLL        ANYLOCAL                          HOLD
     VAINZ11        ANYLOCAL                          HOLD
```

Figure 18-55 Action D-Display response

Note: SDSF may not display responses for the NP actions if the SDSF Extended MCS (EMCS) console actvation fails. However, the action commands will be executed successfully. The ULOG displays:

```
ISF032I CONSOLE nnnnnnnn ACTIVATE FAILED, RETURN CODE 0004, REASON CODE 0000
```

The hardcopy log shows the result of all action commands.

18.11 Output queue (O) panel

The Output Queue panel allows the user to display information about SYSOUT data sets for jobs, started tasks, and TSO users on the JES3 output service writer queue.

There is one row on the panel for SYSOUT data sets of a job on the JES3 output service writer queue that share the same characteristics, such as class and destination.

The output display is accessed with the O command from any SDSF panel. Figure 18-56 on page 900 shows the syntax of the O command.

```
O(classes) (form-number)

O with no parameters displays information for all output data sets. The information
displayed may be limited by your authorization and by settings for filters such as
FILTER, PREFIX, and so on.

O can be issued on any SDSF panel.
class - displays information about job output in specific output class. Enter up to 7
classes, without blanks, including:
   @ - output waiting to be transmitted to another node. If other classes are
   specified, the output must be in one of those classes.
form-number - displays only data sets with this form number. The form number can be up
to 8 characters long, including * (any string of characters) or % (any single
character).

Examples:
   OJAB      Displays output in classes J, A, and B.
   OBK STD    Displays output in classes B and K, with a form number of STD.
```

Figure 18-56 O command syntax

Output queue panel fields

The output queue panel in the JES3 environment may include the following columns, shown in Figure 18-57. (The order and titles may be different, depending on installation and user options.)

Title	Description
JOBNAME	Job name. This is the fixed field.
JNum	JES job number (not included in the default field list)
JobID	JES job ID or work ID X
Owner	User ID of SYSIN/SYSOUT owner, or default values of ++++++++ or ???????? if user ID not defined to RACF
Prty	JES output group priority
C	JES output class
Forms	Output form number
Dest	JES print destination name
Tot-Rec	Output total record count (lines). Blank for page-mode data.
Tot-Page	Output page count. Blank if not for page-mode data.
Device	Output device name (only if it is printing)
Status	JES job status
SecLabel	Security label of output group
JP	JES job priority
FCB	Output FCB ID
UCS	Output UCS ID (print train required)
Wtr	Output external writer name
Flash	Output flash ID
Burst	3800 burst indicator
PrMode	Printer process mode
OHR	Output hold reason code
Output-Hold-Text	Output hold reason text
Max-RC	Return code information for the job
Type	Type of address space

Figure 18-57 Output queue panel columns

Figure 18-58 on page 901 lists the delayed access fields on the output queue panel.

```
Title                 Description
Programmer-Name       JES programmer name field
Acct                  JES account number
Notify                TSO user ID from NOTIFY parameter on job card
ISys                  JES input system ID
Rd-Time               Time that the job was read in
Rd-Date               Date that the job was read in
ESys JES              execution system ID
St-Time               Time that execution began
St-Date               Date that execution began
End-Time              Time that execution ended
End-Date              Date that execution ended
Cards                 Number of cards read for job
JC                    JES input job class
MC                    Message class of job
SubGroup              Submittor RACF group
JobAcct1              Job accounting field 1 (not included in the default field list)
JobAcct2              Job accounting field 2 (not included in the default field list)
JobAcct3              Job accounting field 3 (not included in the default field list)
JobAcct4              Job accounting field 4 (not included in the default field list)
JobAcct5              Job accounting field 5 (not included in the default field list)
```

Figure 18-58 Output queue panel columns with delayed access

REXX execs and Java programs reference columns by name rather than by title. The COLSHELP command displays both column (field) names and titles.

Figure 18-59 is an SDSF output queue display on a 24x80 TSO panel.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF OUTPUT ALL CLASSES ALL FORMS    LINES 10,073     LINE 1-21 (21)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
PREFIX=%%%N*  DEST=ANYLOCAL  OWNER=*  SYSNAME=SC75
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,Q-OutDesc,S-Browse,SB-ISPFBrowse,
ACTION=SE-ISPFEdit,SJ-JCLEdit,X-Print,XC-PrintClose,XD-PrintDS,
ACTION=XDC-PrintDSClose,XF-PrintFile,XFC-PrintFileClose,XS-PrintSysout,
ACTION=XSC-PrintSysoutClose
NP   JOBNAME  JobID    Owner    Prty C Forms    Dest                Tot-Rec
T
     PCPNTJV  JOB20847 VAINI       1 A BLUE     ANYLOCAL                 63
     PCPNTJV  JOB20847 VAINI       1 B RED      SECRET                   14
     PCPNTJV  JOB20847 VAINI       1 C YELLOW   TELLIT                   32
     PCPNTJV  JOB20847 VAINI       1 E PINK     ANYLOCAL                  4
     PCPNTJV  JOB20847 VAINI       1 E WHITE    REMOTE                    8
     PCPNTJV  JOB20847 VAINI       1 H PINK     ANYLOCAL                 10
     PRINTWAY JOB20357 IBMUSER    15 A STD      ANYLOCAL                 72
     VAINIAR  JOB20828 VAINI       1 A STD      ANYLOCAL                  51
```

Figure 18-59 Output queue panel

In Figure 18-59 there are several rows for job PCPNTJV 20847. A row is created for a group of output data sets (one or more) that share the same characteristics, such as class and destination.

The title line of the Output Queue panel is described in Figure 18-60 on page 902. The characteristics of the print data represented by each row differ.

```
SDSF OUTPUT ALL CLASSES   ALL FORMS  LINES    94372    LINE  1-20 (20)
            |             |                   |              |      |
   JES output classes     |                   |              |      |
   on displayed     SYSOUT forms              |              | Totallines
                    being displayed           |              | (**** if
                                              |              | more than
                Total number of print lines for              | 99,999,999)
                output being displayed. Scaled      Lines displayed
                if needed, for example, 1G          or first line
                rather than 1,000,000,000.          if 100,000 or more
```

Figure 18-60 Output queue panel title line

Output queue panel NP field actions

Figure 18-61 describes the NP column action characters for the JES3 Output queue panel. Only actions for a job's data sets are available. DS NP-action lists the spool data sets for a job.

NP Description	JES3 command / SDSF action
?-JDS	Display a list of the data sets for an output group.
Q-OutDesc	Display output descriptors for all of the data sets with similar characteristics.
S-Browse	Display the data sets for an output group. **SB** - Use ISPF Browse. **SE** - Use ISPF Edit. **SJ** - Use ISPF Edit to edit the JCL.
X-Print	Print output data sets. You can add: **C** - Close the print file after printing (XC) **D** - Display the Open Print Data Set panel (XD or XDC) **F** - Display the Open Print File panel (XF or XFC) **S** - Display the Open Print panel (XS or XSC)

Figure 18-61 Output queue panel NP field actions

Output queue panel overtypable fields

The output queue panel does not have any overtypable fields.

18.12 Held output queue (H) panel

The Held output queue panel allows the user to display information about SYSOUT data sets for jobs, started tasks, and TSO users on the JES3 output service hold queue, both external writer and TSO held queues.

There is one row on the panel for SYSOUT data sets of a job on the JES3 output service held queue that share the same characteristics, such as class and destination.

Operator hold JES3 writer queue SYSOUT is not displayed on the SDSF held output queue panel.

The held output panel is accessed with the H command from any SDSF panel. Figure 18-62 on page 903 shows the syntax of the H command

```
H(classes) (string|ALL)

H with no parameters displays information for all output data sets. The information
displayed may be limited by your authorization and by settings for filters such as FILTER,
PREFIX, and so on.

class - is a list of up to 7 output classes. Do not use blanks between H and the classes
or between classes.

string - is a character string that limits the panel to jobs with names that match the
character string.
   string may be up to 8 characters, including * (any string of characters) and (any
   single character).

ALL displays all jobs.

Examples:
   HDE ALL - Displays information for all jobs in output classes D and E.
   H ABC -   Displays information for jobs with the name abc.
   H ABC* -  Displays information for jobs with names that begin with abc.
```

Figure 18-62 H command syntax

The Held output queue panel in the JES3 environment may include the following columns, shown in Figure 18-63. (The order and titles may be different, depending on installation and user options.)

Held output queue panel fields

Title	Description
JOBNAME	Job name. This is the fixed field.
JNum	JES job number. Not included in the default field list.
JobID	JES job ID X
Owner	User ID of SYSIN/SYSOUT owner, or default values of ++++++++ or ????????, if user ID not defined to RACF
Prty	JES output group priority
C	JES output class
Dest	JES print destination name
Tot-Rec	Output total record count (lines). Blank for page-mode data.
Tot-Page	Output page count (lines). Blank if not for page-mode data.
Forms	Output form number
FCB	Output FCB ID
Status	JES job status
UCS	Output UCS ID (print train required)
Wtr	Output external writer name
Flash	Output flash ID
Burst	3800 burst indicator
PrMode	Printer process mode
SecLabel	Security label of data sets
JP	Job priority
OHR	Output hold reason code
Output-Hold-Text	Output device name
Max-RC	Return code information for the job
Type	Type of address space

Figure 18-63 Held output queue panel columns

Figure 18-64 lists the delayed access fields on the held output queue panel.

Title	Description
RNum	JES job room number X
Programmer-Name	JES programmer name X
Acct	JES account number X
Notify	TSO user ID from NOTIFY parameter on job card X
ISys	JES input system ID X
Rd-Time	Time that the job was read in X
Rd-Date	Date that the job was read in X
ESys	JES execution system ID X
St-Time	Time that execution began X
St-Date	Date that execution began X
End-Time	Time that execution ended X
End-Date	Date that execution ended X
Cards	Number of cards read for job X
JC	JES input job class
MC	Message class of job X
SubGroup	Submittor group X
JobAcct1	Job accounting field 1. Not included in the default field list.
JobAcct2	Job accounting field 2. Not included in the default field list.
JobAcct3	Job accounting field 3. Not included in the default field list.
JobAcct4	Job accounting field 4. Not included in the default field list.
JobAcct5	Job accounting field 5. Not included in the default field list.

Figure 18-64 Held output queue panel columns with delayed access

REXX execs and Java programs reference columns by name rather than by title. The COLSHELP command displays both column (field) names and titles.

Figure 18-65 on page 905 is an SDSF held output queue display on a 24x80 TSO panel.

In Figure 18-65 there are several rows for job PCPNHJV 20850. A row is created for a group of output data sets (one or more) that share the same characteristics, such as class and destination.

Output queue panel overtypable fields

The SDSF output queue panel for the JES3 environment does not have any overtypable fields.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF HELD OUTPUT DISPLAY ALL CLASSES LINES 2,864      LINE 1-14 (16)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
PREFIX=%%%N*  DEST=*  OWNER=*  SYSNAME=SC75
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,Q-OutDesc,S-Browse,SB-ISPFBrowse,
ACTION=SE-ISPFEdit,SJ-JCLEdit,X-Print,XC-PrintClose,XD-PrintDS,
ACTION=XDC-PrintDSClose,XF-PrintFile,XFC-PrintFileClose,XS-PrintSysout,
ACTION=XSC-PrintSysoutClose
NP   JOBNAME  JobID    Owner    Prty C Dest                 Tot-Rec
Tot-Page F
     PCPNHJV  JOB20850 VAINI       1 U ANYLOCAL                  9
P
     PCPNHJV  JOB20850 VAINI       1 W ANYLOCAL                  5
P
     PCPNHJV  JOB20850 VAINI       1 V REMOTE                    8
W
     PCPNHJV  JOB20850 VAINI       1 G TELLIT                   32
Y
     PCPNHJV  JOB20850 VAINI       1 T ANYLOCAL                 63
B
     PCPNHJV  JOB20850 VAINI       1 X SECRET                   14
R
```

Figure 18-65 Held output panel display

Held output panel NP column actions

Figure 18-66 describes the NP column action characters for the JES3 Output queue panel. Only actions for a job's data sets are available.**-JDS** NP-action lists the spool data sets for a job.

NP Description	JES3 command / SDSF action
?-JDS	Display a list of the data sets for an output group.
Q-OutDesc	Display output descriptors for all of the data sets with similar characteristics.
S-Browse	Display the data sets for an output group. **SB** - Use ISPF Browse. **SE** - Use ISPF Edit. **SJ** - Use ISPF Edit to edit the JCL.
X-Print	Print output data sets. You can add: **C** - Close the print file after printing (XC) **D** - Display the Open Print Data Set panel (XD or XDC) **F** - Display the Open Print File panel (XF or XFC) **S** - Display the Open Print panel (XS or XSC)

Figure 18-66 Held output queue panel NP field actions

18.13 Status (ST) panel

The status panel displays information about jobs, started tasks, and TSO users on all the JES3 queues.

The syntax of the ST command is described in Figure 18-67 on page 906.

```
ST[classes] [string]

classes displays information for a specific class. Enter a single class, up
to 6 characters. To filter the panel using more than one class, use the
FILTER command.

You can use these special characters for class
   * - converter queue              $ - TSO users in execution
   # - started tasks in execution   ! - hard-copy queue
   + - output queue                 - - input queue
   ? - purge queue                  ) - receiver queue
   = - spin queue                   / - setup queue
   @ - jobs waiting to be transmitted to another queue

string is a character string that limits the panel to jobs whose names match
the character string. The string can be up to 8 characters, including:
   *  - to represent any character or string of characters
   %  - to represent any single character.
   Note: SET SCHARS command may be used to change the characters for pattern
   matching.

ST with no parameters displays all jobs. The information displayed may be
limited by your authorization and by settings for SDSF filters such as
FILTER and PREFIX.
```

Figure 18-67 ST command syntax

“Status panel fields” on page 907 shows an example of some columns on an SDSF status display for a JES3 job.

. A select command with no parameters returns the display to the original display.

```
   Display  Filter  View  Print  Options  Search  Help

------------------------------------------------------------------------------
---
 SDSF STATUS DISPLAY ALL CLASSES                      LINE 1-1 (1)
 COMMAND INPUT ===>                                            SCROLL ===>
HALF
 PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=SC75
 ACTION=//-Block,=-Repeat,+-Extend,?-JDS,A-Release,C-Cancel,CA-CancelARM,
 ACTION=CD-CancelDump,CDA-CancelARMDump,CP-CancelPrint,D-Display,

ACTION=DE-DisplayEstimates,DL-DisplayLong,DM-DisplayMains,DMA-DisplayMDSAllo
c,
 ACTION=DME-DisplayMDSError,DMR-DisplayMDSRestart,DMSS-DisplayMDSSysSel,
 ACTION=DMSV-DisplayMDSSysVer,DMU-DisplayUnavailVol,DSD-DisplayDDnames,
 ACTION=DSH-DisplaySpoolHold,DSP-DisplaySpoolPartition,DX-DisplayExtended,
 ACTION=E-Restart,H-Hold,I-Info,J-Start,L-List,LB-ListBDT,LH-ListHold,
 ACTION=LT-ListTCP,P-Purge,Q-OutDesc,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,
 ACTION=SJ-JCLEdit,W-Spin,X-Print,XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,
 ACTION=XF-PrintFile,XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose
 NP   JOBNAME  JobID    Owner    Prty Queue      C         Pos  SAff  ASys
S
      VAINISCT JOB07219 VAINI       1 EXECUTION  A           49
H
```

Figure 18-68 Status panel (ST) display

Status panel fields

The status panel in the JES3 environment may include the following columns, shown in Figure 18-69 on page 908. (The order and titles may be different, depending on installation and user options.)

Title	Description
JOBNAME	Job name. This is the fixed field.
Type	Type of address space
JNum	JES job number. Not included in the default field list.
JobID	JES job ID
Owner	User ID of job owner, or default values of ++++++++ or ????????, if user not defined to RACF
Prty	JES job queue priority
Queue	JES queue name for job
C	(JES2) 8 (JES3) JES input class
Pos	Position in JES queue
SAff	JES execution system affinity (if any)
ASys	JES active system ID (if job active)
Status	Status of job
PrtDest	JES print destination name
SecLabel	Security label of job
OrigNode	Origin node name
ExecNode	Execution node name
Device	JES device name
Max-RC	Return code information for the job
SrvClass	Service class
WPos	Position on the WLM queue
Scheduling-Env	Scheduling environment for the job
Dly	Indicator that job processing is delayed
Mode	Subsystem managing the job (JES or WLM)

Figure 18-69 Status panel columns

Figure 18-70 lists the delayed access fields on the status panel.

Title	Description
RNum	JES job room number
Programmer-Name	JES programmer name
Acct	JES account number
Notify	TSO user ID from NOTIFY parameter on job card
ISys	JES input system ID
Rd-Time	Time that the job was read in
Rd-Date	Date that the job was read in
ESys	JES execution system ID
St-Time	Time that execution began
St-Date	Date that execution began
End-Time	Time that execution ended
End-Date	Date that execution ended
Cards	Number of cards read for job
MC	MSGCLASS of job
Tot-Lines	Total number of spool records for job
Spin	Indicator of whether the job is eligible to be spun
SubGroup	Submittor group
PhaseName	Name of the phase the job is in
Phase	Number of the phase the job is in
JobAcct1	Job accounting field 1. Not included in the default field list.
JobAcct2	Job accounting field 2. Not included in the default field list.
JobAcct3	Job accounting field 3. Not included in the default field list.
JobAcct4	Job accounting field 4. Not included in the default field list.
JobAcct5	Job accounting field 5. Not included in the default field list.
SubUser	Submitting user ID

Figure 18-70 Status panel columns with delayed access

REXX execs and Java programs reference columns by name rather than by title. COLSHELP command displays both column names and titles.

Status panel NP field actions

The status panel actions are the same as the input queue actions. See “Input queue panel NP field actions” on page 896 and Figure 18-52 on page 897.

18.14 Job zero (J0) panel

The Job zero panel displays information about JES3 job zero spool data sets. With this panel, you can work with JES3’s spin-off data under JES3 job zero. Figure 18-71 shows the syntax.

```
J0
J0 can be issued on any SDSF panel.
```

Figure 18-71 J0 command syntax

J0 panel fields

Figure 18-72 lists the fields on the J0 panel.

Title	Description
DSPNAME	Name or job number of the DSP that created the data set
DSID	Data set ID
Owner	User ID that created the data set
C	Output class
CC	Data set copy count
PrMode	Process mode
Burst	Burst indicator
Forms	Creation form number
FCB	FCB name
UCS	UCS name
Wtr	External writer name
Flash	Flash name
FlashC	Flash count
SegID	Segment number
Chars	Character arrangement tables
CpyMod	Copy-modification module for the 3800 printer
Queue	Queue the SYSOUT is on
Dest	Print destination name
SecLabel	Security label for the data set
CrDate-CrTime	Creation date and time
Spin	Spin data set indicator
Sel	Selectable indicator
Rec-Cnt	SYSOUT record count
Page-Cnt	SYSOUT page count (for page-mode data)
Byte-Cnt	SYSOUT byte count
RecFM	Record format
DDName	DD name of the data set
DSName	Full data set name
StepName	Name of the job step that created the data set
ProcStep	Name of the procedure step that created the

Figure 18-72 J0 panel columns with delayed access

Figure 18-73 is an SDSF J0 panel display. The rows on the panel are for a spin-off spool data set created by JES3 DSPs. The rows with the DSP name JOBnnnnn are created by the *CALL DSIPDJC command.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF JOB O DISPLAY                  LINES 187,048      DATA SET DISPLAYED
COMMAND INPUT ===>                                            SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,C-Cancel,D-Display,H-Hold,O-Release,
ACTION=P-Purge,Q-OutDesc,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,X-Print,
ACTION=XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,XF-PrintFile,
ACTION=XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose
NP   DSPNAME  DSID Owner    C CC PrMode   Burst Forms    FCB  UCS  Wtr
Fla
     DC         18 JES3     A  1 LINE     C     STD      STD3 ANY
NON
     JOB17005   20 JES3     A  1 LINE     C     STD      STD3 ANY
NON
     DISPLAY    21 JES3     A  1 LINE     C     STD      STD3 ANY
NON
     DC         23 JES3     A  1 LINE     C     STD      STD3 ANY
NON
     JOB20658   81 JES3     A  1 LINE     C     STD      STD3 ANY
NON
     DISPLAY   157 JES3     A  1 LINE     C     STD      STD3 ANY
NON
  *SDSF    -OL
```

Figure 18-73 J0 panel display

Tip: The bottom line in Figure 18-73 is the ISPF list of logical sessions activated by entering the SWAPBAR command. The panel name SDSF is set by SDSF. It may be changed with the ISPF command SCRNAME name. The **'*'** in front of a panel name is set for an active panel. The **'-'** is for the inactive split panel.

J0 panel NP field actions

Figure 18-74 on page 911 describes the NP column action characters for the J0 panel. Only actions for data sets are available. The J0 panel displays only spool data sets.

```
NP-Description     JES3 command / SDSF action
C-Cancel           *F U,J=0,Q=WTR,DSN= - Purge a data set.
D-Display          *I U,J=0,Q=WTR,DSN= - Display information
H-Hold             *F U,J=0,Q=WTR,DSN= - Hold a data set.
O-Release          *F U,J=0,Q=WTR,DSN= - Release a data set.
P-Purge            *F U,J=0,Q=WTR,DSN= - Purge a data set.
Q-OutDesc          Display output descriptors for the data set.
S-Browse           Display a data set. Other forms:
                       SB (ISPF Browse)
                       SE (ISPF Edit)
X-Print            Print a data set. You can add the following:
                       C - Close the print file after printing (XC)
                       D - Display the Open Print Data Set panel (XD or XDC)
                       F - Display the Open Print File panel (XF or XFC)
                       S - Display the Open Print panel (XS or XSC)
```

Figure 18-74 J0 panel NP-column actions

J0 panel overtypable fields

You can change the columns listed in Figure 18-75 on page 911 on the SDSF JES3 J0 panel by typing over them.

Field	Description
Burst	Data set burst indicator
C	JES output class: A-Z, 0-9
CC	Data set copy count
Chars	Character arrangement table names
CpyMod	Copy modification table name
Dest	Print destination name
FCB	Output FCB name
Flash	Output flash ID
Forms	Data set creation form number
PrMode	Process Mode
UCS	UCS ID
Wtr	Output special writer ID or data set ID

The JES3 **`*F U J=0,Q=WTR,DSN=`** command is used to set the overtype value for a field

Figure 18-75 J0 panel overtypable fields

18.15 Viewing jobs' spool data

On `I`, `O`, `H`, `ST`, and `DA` panels the NP field `S-Browse` action invokes SDSF browse for all spool data sets of a job. The SDSF output display panel is used to view data for action code `S`. Instead SDSF browse, ISPF Browse or Edit can be invoked with the `SB` or `SE` action characters. `SJ` action displays just the JCL for a job. The displayed JCL can be changed and resubmitted; changes are not saved.

If action bar *Options selection 3. Change include SYSIN to ON/OFF* or the INPUT ON|OFF command is ON, the SYSIN data sets for the job are included in the display entered from DA, I and ST panels. If the data to be viewed contains nondisplay characters, the SET HEX ON|OFF command may be used to display the data in hexadecimal format.

The NP action ? displays a list of data sets for a job on the Job Data Set (JDS) panel. The NP actions S, SB, SE and SJ can be used to work with the individual spool data sets. Note that

the SJ action always invokes a job's JCL edit independent of the job information panel from where it is used.

NP field action S-Browse

When used to view a job's spool data, the displayed data includes the JES job log, JCL for the job, any job-related messages and SYSIN data sets if requested in addition to the job's SYSOUT data available so far.

The INPUT {ON|OFF} command specifies whether jobs' input data sets are to be included into the display when you view jobs from the DA, ST, or I panels. You must be authorized to use this command. The action bar *Options pull-down choice 3* can be used to toggle between input ON or OFF.

The SET HEX {ON|OFF} command controls the display in hexadecimal for this session. The action bar *View pull-down choice 3 "Set hex to ON|OFF"* may be used instead of the SET HEX command to control the display in hexadecimal.

The SDSF browse does not support the ISPF type labels for data lines and the ISPF picture string find commands.

NP field action V-View GDDM browse

To view output formatted for a page printer on the job data set panel, you can use the V-View action character, which requires GDDM. If GDDM is not available or if the data to be viewed is not formatted for a page printer, the spool data set display is formatted for a line-mode printer.

On the Job Data Set panel page-mode output is indicated by the PrMode column value PAGE, a value other than blanks in the Page-Cnt columns and RecFm value VM.

NP field action SB-ISPFBrowse

Spool data viewing for a job using the ISPF browse is invoked with the SB action character. ISPF browse ignores the SDSF SET HEX setting. To display data in hexadecimal, use the ISPF HEX primary command. The ISPF label assignment and picture string find commands are available.

When the ISPF browse is active, SDSF commands are not available. To use SDSF commands (such as / or PRINT) you must access SDSF's browse with the S action character.

NP field actions SE-ISPFEdit

To display the job's output with ISPF Edit, use SE. To display just the JCL for the job, use SJ. You can change and resubmit the JCL from the display; changes you make to the data are not saved.

The SET BROWSE command controls the default browse action character (S, SB, or SE) that is issued when you place the cursor in the NP column and press Enter. Figure 18-76 shows an example of the SDSF job output browse display on the SDSF output display panel.

```
   Display  Filter  View  Print  Options  Search  Help
 -------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY PELEGDSF JOB17189  DSID     2 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===>                                            SCROLL ===> HALF
********************************* TOP OF DATA *********************************
 IAT6140 JOB ORIGIN FROM GROUP=ANYLOCAL, DSP=IR , DEVICE=INTRDR  , 0000
 15:18:17 ---- IAT6853 THE CURRENT DATE IS MONDAY,    10 MAY 2010 ----
IRR010I  USERID PELEG    IS ASSIGNED TO THIS JOB.
 15:18:17  IAT2000 JOB PELEGDSF (JOB17189) SELECTED SC75     GRP=A
 15:18:17  ICH70001I PELEG    LAST ACCESS AT 15:02:03 ON MONDAY, MAY 10, 2010
 15:18:17  ICK091I   DD65 NED=002107.900.IBM.75.0000000BALB1
 15:18:17 *020 ICK003D REPLY U TO ALTER VOLUME DD65 CONTENTS, ELSE T
 15:18:26  IEE600I REPLY TO 020 IS;U
 15:18:26  -                                        ---------TIMINGS (MINS.)---
 15:18:26  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB   VECT   VAFF
 15:18:26  -PELEGDSF REFORMAT             12    100    .00    .00    .00    .00
 15:18:26  -PELEGDSF ENDED.  NAME-                     TOTAL CPU TIME=   .00  TO
//PELEGDSF JOB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//REFORMAT EXEC PGM=ICKDSF
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
/*
        1 //PELEGDSF JOB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
```

Figure 18-76 Output display for an S-Browse action

NP field actions SJ-JCLEdit for JCL

To display just the JCL for a job, use SJ NP-column action. You can change and resubmit the JCL from the display; changes you make to the data are not saved. The job must have executed on your node or not yet executed to be eligible for SJ action.

Figure 18-77 shows an example of a re-edit request for a job's JCL. The SJ action can be entered on any line on the job data set display panel or on a row of any queue display panel.

```
 SDSF EDIT    VAINISCT (JOB07219) JCLEDIT                   Columns 00001 00072
 Command ===>                                                 Scroll ===> HALF
 ****** ***************************** Top of Data ******************************
 000001 //VAINISCT JOB 1,EXPERT,MSGLEVEL=1,MSGCLASS=T,NOTIFY=&SYSUID,CLASS=A
 000003 //*MAIN SYSTEM=SC75
 000004 //TMP    EXEC PGM=IKJEFT01,DYNAMNBR=99
 000005 //SYSPROC  DD DSN=VAINI.U.CLIST,DISP=SHR
 000006 //STEPLIB  DD DSN=VAINI.U.LOAD,DISP=SHR
 000007 //SYSTSPRT DD SYSOUT=*
 000010 //SYSTSIN  DD *
 000011   BPXCAN
 ****** **************************** Bottom of Data ****************************
```

Figure 18-77 Output display for a SJ-JCLEdit action

NP field action ?-JDS on I, O, H, ST, and DA panels - Job Data Set panel

The Job Data Set panel allows users to list and display information about the SYSOUT data sets for a job, started task, or TSO user. The Job Data Set panel is accessed with the NP column **?** action character.

Figure 18-78 shows an example of the SDSF job data set panel. INPUT ON is set. SDSF displays the type of the spool data sets.

```
  Display  Filter  View  Print  Options  Search  Help
--------------------------------------------------------------------------------
SDSF JOB DATA SET DISPLAY - JOB VAINISCT (JOB07219)    LINE 1-9 (9)
COMMAND INPUT ===>                                          SCROLL ===> HALF
PREFIX=*  DEST=(ALL)  OWNER=*  SYSNAME=SC75
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,H-Hold,O-Release,P-Purge,
ACTION=Q-OutDesc,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,SJ-JCLEdit,V-View,X-Print,
ACTION=XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,XF-PrintFile,
ACTION=XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose
NP   DDNAME   StepName ProcStep DSID Owner    C Dest               Rec-Cnt Page
     JESJCLIN                     1 VAINI                                0
     JESMSGLG                     2 VAINI    T                           0
     JESJCL                       3 VAINI    T                          29
     JESYSMSG                     4 VAINI    T                           2
     J3SCINFO                     5 VAINI                                0
     J3JBINFO                     6 VAINI                                1
     JCBLOCK                      7 VAINI                                0
     JOURNAL                      8 VAINI                                0
```

Figure 18-78 Job data set (JDS) panel

Job data set (JDS) panel fields

The job data set (JDS) panel allows users to list and display information about the SYSOUT data sets for a job, started task, or TSO user.

JDS panel fields in the JES3 environment are shown in Figure 18-79 and Figure 18-80 on page 915 (default titles and order).

```
     SDSF JOB DATA SET DISPLAY - JOB  VAINISCT (JOB07219)  LINE 1-9  (9)
                                      |         |                |
     |
                                      Job name  |                |    Total
     lines
                                                Job or work ID   Lines
     displayed
```

Figure 18-79 Job data set panel title

Note: When JDS is accessed from H or O panels, access for fields marked with * is immediate. Otherwise, access for all fields is delayed.

Title	Description
DDNAME	Ddname of the data set
StepName	Output creation step name
ProcStep	Output creation procedure step name
DSID	Data set ID
Owner*	User ID of SYSIN/SYSOUT owner
C *	Original or released output class
Dest*	Print destination name
Rec-Cnt	Output record count
Page-Cnt	Output page count (for page-mode data only)
Byte-Cnt	Byte count for the data set
CC	Data set copy count
SecLabel	Security label for data set
PrMode	Data set process mode
Burst	Data set burst indicator
CrDate-CrTime	Data set creation date and time
Forms	Data set creation form number
FCB	Output FCB name
UCS	Output UCS name
Wtr	Output special writer ID or data set ID
Flash	Output flash name
FlashC	Output flash count
SegID	Data set segment number
DSName	Output data set name
Chars	Character arrangement tables (1-4)
CpyMod	Copy-modification module for the 3800 printer
PageDef	Library member used by PSF to specify print characteristics
FormDef	Library member used by PSF to specify print characteristics
Title	Output title
Name	Output name
Building	Output building
Department	Output department
Room	Output room
Address-Line1 - Address-Line4	Output address lines 1 to 4
OutBn	Output bin
ComSetup	Printer setup options
FormLen	Form length
ColorMap	AFP resource for the data set containing color translation
ITy	Paper source (in-tray)
OverlayB	Overlay for the back of each sheet
OverlayF	Overlay for the front of each sheet
OffsetXB	Offset in x direction from the page origin for the back of each page
OffsetXF	Offset in x direction from the page origin for the front of each page
OffsetYB	Offset in y direction from the page origin for the back of each page
OffsetYF	Offset in y direction from the page origin for the front of each page
Port	Number of the TCP/IP port where the FSS connects to the printer
Notify	Print complete notification message
UserLib	Libraries containing AFP resources
UserData1	User data
AFPParms	Data set containing parameters used by the AFP Print Distributor
Queue	JES3 queue the dataset is on

Figure 18-80 JDS panel fields (1 of 2)

Title	Description
Spin	Indicates if it is a spin data set
Sel	Indicates if it is selectable
TP	Indicates if it was created by a transaction program
TPJName	Job name of the transaction program that created it
TPJobID	Job ID of the transaction program that created it
TRd-Time	Start time and date for entry of the transaction
TRd-Date	program
TSt-Time	Start time and date for execution of the
TSt-Date	transaction program
TPAcct	Account number of the transaction program
RecFm	Record format

Figure 18-81 JDS panel fields (2 of 2)

NP field action characters on JDS panels

JDS panel action characters that can be entered in the NP column by users are shown in Figure 18-82.

NP-Description	JES3 command / SDSF action
C-Cancel	*F U,J=,DD=,C - Purge an output data set.
H-Hold	*F U,Q=WTR,DD=,NQ=HOLD - Hold an output data set.
O-Release	*F U,Q=HOLD,DD=,NQ=WTR - Release an output data set.
P-Purge	*F U,Q=,J=,DD=,C - Purge an output data set.
Q-OutDesc	Display output descriptors for the data set.
S-Browse	Display data set or data sets. Other forms: SB (ISPF Browse), SE (ISPF Edit), SJ (Edit JCL)
V-View	View a job's page-mode data sets using GDDM.
X-Print	Print output data sets. You can add the following: **C** - Close the print file after printing (XC) **D** - Display the Open Print Data Set panel (XD or XDC) **F** - Display the Open Print File panel (XF or XFC) **S** - Display the Open Print panel (XS or XSC)

Figure 18-82 JDS panel action characters

Job data set (JDS) panel overtypable fields

JDS panel overtypable fields are shown in Figure 18-83 and Figure 18-84 on page 917.

Title	Description
Address	Output address (lines 1-4). Type + alone to modify ADDRESS2-4
AFPParms	Data set containing parameters to be used by the AFPPrint Distributor
Building	Output building
Burst	Data set burst indicator
C	JES output class: A-Z, 0-9
CC	Data set copy count
Chars	Character arrangement table names
ColorMap	AFP resource for the data set containing color translation

Figure 18-83 JDS panel overtypable fields 1 of 2

```
Title          Description
Address        Output address (lines 1-4). Type + alone to modify ADDRESS2-4
ComSetup       Output microfiche printer options
CpyMod         Copy modification table name (JES3 only)
Department     Output department
Dest           Print destination name
FCB            Output FCB name (JES3 only)
Flash          Output flash ID (JES3 only)
FormDef        Output form definition
FormLen        Form length
Forms          Data set creation form number
ITy            Paper source (1-255)
Name           Output name
Notify         Output print complete notification. Type + alone to modify NOTIFY2-4.
OffsetXB       Offset in x direction from the page origin for the back of each page
OffsetXF       Offset in x direction from the page origin for the front of each page
OffsetYB       Offset in y direction from the page origin for the back of each page
OffsetYF       Offset in y direction from the page origin for the front of each page
OverlayB       Overlay for the back of each sheet
OverlayF       Overlay for the front of each sheet
OutBn          Output bin
PageDef        Output page definition
Port           Number of the TCP/IP port at which the FSS connects to the printer
PrMode         Process Mode
Room           Output room
Title          Output title
UCS            UCS ID
UserLib        Output libraries with AFP resources to be used by PSF when processing
               SYSOUT. Type + alone to modify USERLIB2-4.
UserData1      Output user data. Type + alone to modify UserData2-16.
Wtr            Output special writer ID or data set ID

The JES3 *F U J=O,Q=WTR,DSN= command is used to set the overtype value for
fields managed by JES3. If the overtype value is associated with an output
descriptor, then the output characteristics are modified with the scheduler
facilities call.
```

Figure 18-84 JDS panel overtypable fields 2 of 2

JES3 spool security and SYSIN/SYSOUT data access

Note: JES uses the JESSPOOL class to protect SYSIN/SYSOUT data sets. SDSF extends the use of the JESSPOOL class to protect SDSF job and output group resources as well. SDSF checks a user's SAF authorization to:

- Job resources on the DA, I, and ST panels.
- Output groups on the H, JDS, O, and OD panels.
- SYSIN/SYSOUT data sets on the JDS, J0 and any other panel used for browsing with the S or V action characters and printing with the X action character.

NP field action Q-OutDesc on I, O, H, JDS, ST, and DA panels

The output descriptors panel (OD) displays JES output descriptors. Output descriptors provide information about a SYSOUT data set, for example, an address or a building. The output descriptors display, shown in Figure 18-85 on page 918, is accessed with the Q-OutDesc action character.

```
 Display  Filter  View  Print  Options  Search  Help
 -------------------------------------------------------------------------------
 SDSF OUTPUT DESCRIPTORS - JOB VAINIJV  (JOB20671)      LINE 1-21 (107)
 COMMAND INPUT ===>                                          SCROLL ===> HALF
 PREFIX=V*  DEST=(ALL)  OWNER=*  SYSNAME=SC75
 ACTION=//,=,+,?,E,S,SB,SE,SJ,V,X,XC,XD,XDC,XF,XFC,XS,XSC
NP  DDNAME   Output Descriptors
     SYSUT2   PageDef

     SYSUT2   FormDef

     SYSUT2   Title
              SDSF OD TEST
     SYSUT2   Name
              JUHA VAINIKAINEN
     SYSUT2   Building
              BLDG8
     SYSUT2   Department
              ITSO
```

Figure 18-85 Output Descriptors panel

The output descriptor fields, shown in Figure 18-86, can be overtyped if the output descriptors panel was accessed from the DA, I, O, H or ST panels. The data set must be closed.

The Output Descriptors panel in Figure 18-86 and Figure 18-87 on page 919 have the following fields.

Title	Description
DDNAME	Ddname of the data set
PageDef	Library member used by PSF to specify print characteristics
FormDef	Library member used by PSF to specify print characteristics
Title	Title of output
Name	Output name
Building	Output building
Department	Output department
Room	Output room
Address	Output address lines 1 through 4
OutBin	Output bin
ComSetup	Printer setup options
FormLen	Form length
ColorMap	AFP resource for the data set containing color translation information
InTray	Paper source
OverlayB	Overlay for the back of each sheet
OverlayF	Overlay for the front of each sheet
OffsetXB	Offset in x direction from the page origin for the back of each page
OffsetXF	Offset in x direction from the page origin for the front of each page
OffsetYB	Offset in y direction from the page origin for the back of each page
OffsetYF	Offset in y direction from the page origin for the front of each page

Figure 18-86 Output descriptor panel fields 1 of 2

Output descriptors (OD) panel fields and overtypable fields

Title	Description
PortNo	Number of the TCP/IP port where the FSS connects to the printer
Notify	Print completion notification for 1 to 4 IDs
UserLib	User resource (AFP) libraries to be used by PSF
RetainS	Retain time for successful transmissions (hh:mm:ss)
RetainF	Retain time for unsuccessful attempts (hh:mm:ss)
RetryL	Maximum number of retries
RetryT	Time between retries (hh:mm:ss)
PrtOptns	Entry in the PrintWay options data set
PrtQueue	Print queue name
IP Destination	IP address or TCP/IP name (for example, node.IP:1.2.333.444.5)
UserData	User data
AFPParms	Data set containing parameters used by the AFP Print Distributor

Figure 18-87 Output descriptor panel fields 2 of 2

Output descriptors panel NP field action characters

The output descriptors panel action characters that can be entered in the NP column by users are listed in Figure 18-88. The action characters are entered on lines with DDNAME.

All output descriptor fields can be overtyped when the output descriptors panel is accessed from the DA, I or ST panels.

NP-Description	SDSF action
E-Erase	Erase an output descriptor. E is valid only when the Output Descriptors panel was accessed from the: - Output Queue panel - Held Output Queue panel - Job Data Set panel if it was accessed from the Output Queue panel or the Held Output Queue panel.
S-Browse	Display data sets - access the output data set panel. Also SB - Use ISPF Browse and SE - Use ISPF Edit action are available.
V-View	View page-mode data sets using GDDM.
X-Print	Print output data sets. You can add the following: **C** - Close the print file after printing (XC) **D** - Display the Open Print Data Set panel (XD or XDC) **F** - Display the Open Print File panel (XF or XFC) **S** - Display the Open Print panel (XS or XSC)
?-JDS	Display a list of data sets - access the JDS panel).

Figure 18-88 Output descriptors panel action characters

18.16 JESPlex (JP) panel

The JESPlex (JP) panel displays and controls the main processors in a JES3 complex.

Figure 18-89 on page 920 shows a JESPLEX panel for a three main processor JES3 complex on an 80-byte line length panel. The Status and ConnStat columns have been transposed --

ARR Status A ConnStat; ARR ConnStat A NAME. The Version and SLevel columns are moved to be the last columns of the JESPLEX panel -- ARR Version L; ARR SLevel L. The ARRANGE command reorders and changes the widths of columns on the current panel.

The C (Command character) field has been shortened (**ARR C 1**) to one byte to fit more visible fields on the display lines.

```
  Display  Filter  View  Print  Options  Search  Help
-----------------------------------------------------------------------------
--
SDSF JP DISPLAY  SC75                1% SPOOL        LINE 1-3 (3)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Connect,D-Display,DL-DisplayLong,F-Flush
,
ACTION=JS-MonitorStatus,P-Stop,S-Start,SM-MonStart,V-StartScheduling,
ACTION=VF-StopScheduling,ZM-MonStop
NP   NAME     ConnStat      SysName  Version  C JESN SLevel Global
Start-Type
     SC74     FLUSHED       SC74             *           0 NO     LOCAL
     SC70     CONNECTED     SC70     z 1.13.0 *           0 NO     LOCAL
     SC75     CONNECTED     SC75     z 1.13.0 * JES3      3 YES    HOT
```

Figure 18-89 JESPlex panel

Attention: The C column shows the JES3 SYN= specification. The JES3 SYN= parameter on CONSTD initialization statement specifies a set of prefixes (or synonyms) to be used as SYSTEM scope command prefixes. A command entered with a SYSTEM scope prefix will execute on the system on which the command is entered.

JESPlex panel fields

Figure 18-90 and Figure 18-91 on page 921 list columns in the JES3 environment on the JESPLEX panel. (The order and titles may be different, depending upon installation and user options.)

NAME	Member name. The names of undefined systems have a leading *.
Status	Status of the member
SysName	System name of the MVS image on which this JES system is active
Version	Version of JES the member is running
C	Command character
JESN	JES subsystem name
SLevel	JES3 service level
Global	JES3 global indicator
Start-Type	Last start type for the member
Start-Date-Time	Date and time the member was started
LastGCon-Date-Time	Last time global was contacted
PrimTG	Primary track group allocation
SecTG	Secondary track group allocation

Figure 18-90 JESPlex panel fields 1 of 2

```
Title              Description
WTOLim             WTO message limit
WTOInt             WTO message interval
PBufCSA            PBUF CSA limit
PBufAux            PBUF JES3AUX
PBufFixed          Fixed PBUFs
UserPages          User pages per open SYSOUT dataset
SelectModeName     Selection mode name
PartName           Spool partition name
MsgPrefix          Message prefix
MsgDest            Message Destination
ConnStat           Connect status
AttStat            Attach status
```

Figure 18-91 JESPlex panel fields 2 of 2

JESPlex panel NP field actions

Figure 18-92 displays the action characters for the JESPLEX panel NP-column.

```
Title              JES3 command / SDSF action
C-Connect          *S main,CONNECT - Connect the local to the global
D-Display          *I MAIN=main - Display main processor
DL-DisplayLong     *I MAIN=main,X - Display main processor extended
F-Flush            *S main,FLUSH - Flush jobs from a local after the local fails
JS-MonitorStatus   *S MONITOR,DI - Display the monitoring DSP parameters
P-Stop             RO main,*RETURN - Stop a main processors
S-Start            *S JSS - Start job scheduling on the global
SM-MonStart        *CALL MONITOR - Invoke the monitor DSP
V-StartScheduling  *MODIFY VARY,main,ON - Make a main available for JES3 scheduling
VF-StopScheduling  *MODIFY VARY,SC70,OFF - Make a main unavailable for JES3 scheduling
ZM-MonStop         *C MONITOR - Cancel the monitor DSP
```

Figure 18-92 Action characters on JESPlex panel NP-column

Note: ULOG should be active when the JP panel's NP actions are used or fields are overtyped. The SDSF User Session Log (ULOG) allows users to view the MVS and JES generated by SDSF and SAF. SDSF deletes the user session log when an SDSF session is ended or when the ULOG CLOSE command is issued.

JESPlex panel overtypable fields

Figure 18-93 shows the JESPLEX panel overtypable fields and JES3 commands issued for overtyped new values.

```
Field            JES3 command - Action
PartName         *F G,main,SP,partname - Modify spool partition name
SelectModeName   *F G,main,SELECT,MODE,modename - Change selection mode
```

Figure 18-93 JESPlex panel overtypable fields

18.17 Job class (JC) panel

The Job Class (JC) panel displays and controls the job classes in the JES3 JESplex. Both JES-managed and WLM-managed classes are shown.

Figure 18-94 shows the syntax of the JC command.

```
JC[one_class]

   one_class selects the job class information to be displayed. There is no
   blank between JC and the one_class. JC with no parameters displays all
   job classes.

   JC command can be issued on any SDSF panel.

Example: JCa displays job classes A.
```

Figure 18-94 JC command syntax

Figure 18-95 is an example of a job class panel for JES3-managed class A.

```
   Display  Filter  View  Print  Options  Search  Help

------------------------------------------------------------------------
---
 SDSF JOB CLASS DISPLAY CLASS A                        LINE 1-3 (3)
 COMMAND INPUT ===>                                          SCROLL ===>
HALF
 ACTION=//-Block,=-Repeat,+-Extend,D-Display,DC-DisplayClass,DG-DisplayGroup
,
 ACTION=ST-Status
 NP   CLASS    Status   Member   Group    Mode Xeq-Cnt  TDepth Log   Jrnl
Rst Je
      A        ACTIVE   SC70     A        JES        1    NONE STD   NO   NO
NO
      A        ACTIVE   SC74     A        JES        1    NONE STD   NO   NO
NO
      A        ACTIVE   SC75     A        JES        1    NONE STD   NO   NO
NO
```

Figure 18-95 Job class (JC) panel

Job class panel fields

Figure 18-96 on page 923 shows the columns on the job class panel in the JES3 environment. (The order and titles may be different, depending on installation and user options.)

```
Title       Description
CLASS       Job class
Status      Status of the job class
Member      Member name
Group       Group name
Mode        Manager of the class (JES or WLM) is released by an operator command
Xeq-Cnt     Number of active jobs
TDepth      Job class limit for the system
Jrnl        Save job-related information in a job journal
Log         Print the job log
Rst         Restart the job from the first step. The job will be restarted on the
            processor on which it was active.
JESLog      Spin options for JES joblogs
SDepth      Setup depth
PartName    Spool partition name
PriTrk      Primary track group allocation
SecTrk      Secondary track group allocation
Prio        Priority
```

Figure 18-96 Job class panel columns

Job class panel NP field actions

Action characters that can be entered in the NP column are described in Figure 18-97.

```
NP-Description       JES3 command / SDSF action
D-Display            *I C=class - Display information about a job class
DC-DisplayClass      *I G,main,C,class - Display status for the class
DG -DisplayGroup     *I G,main,G,group_for_class - Display status for the group
ST-Status            Display the ST panel for all jobs in the class.
```

Figure 18-97 Job class panel NP field actions

Job class panel overtypable fields

Figure 18-98 lists job class panel overtypable fields and JES3 commands issued for overtyped values.

```
Title     JES3 command - Action
TDepth    *F C=class,TDEPTH= - Job class limit for the system
Log       *F C=class,JESMSG= - Print the job log (Yes or No)
JESLog    *F C=class,SPIN= - Spin options for JES joblogs, for example
          (SPIN,line-count) or (NOSPIN)
SDepth    *F C=class,SDEPTH= - Setup depth
```

Figure 18-98 Job class panel overtypable fields

18.18 Initiator (INIT) panel

The Initiator panel displays information about JES-managed and WLM-managed initiators. The INIT command can be issued on any SDSF panel. Figure 18-99 on page 924 presents the syntax of the INIT command.

```
INIT [JES | WLM | ALL]

   JES displays JES-managed initiators
   WLM displays WLM-managed initiators
   ALL displays all initiators. This is the default.
```

Figure 18-99 INIT command syntax

Figure 18-100 is a sample display of the SDSF initiator panel display.

There are three types of rows on the initiator display for each system in the JESPlex: ResType GROUP, CLASS, and INIT. The GROUP rows are displayed on each system for the defined JES3 the GROUP initialization statement in the JES3 initialization deck. CLASS rows, for classes defined to the JES3 job class group, follow each GROUP row. Highlighted INIT rows, with an active job, follow the CLASS rows.

Tip: The order of the columns on the default initiator panel starts with NP, ID, Status, JobName, Stepname, JobID, C, ASID, ASIDX, and Owner on an 80-byte panel row.

In Figure 18-100 on page 924 the order of columns has been arranged to NP, ID, ResType, Status SysName and the rest as they were originally. The filter for the panel requests rows with ID S or T for sysname SC74 or SC75 to be displayed.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
--
SDSF INITIATOR DISPLAY  (ALL)                          LINE 1-7 (7)
COMMAND INPUT ===>                                            SCROLL ===>
HALF
PREFIX=VAIN*  DEST=*  OWNER=*  FILTERS=3  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,D-Display,DL-DisplayLong,P-Stop,S-Start
NP   ID       ResType Status       SysName  JobName  Stepname JobID    C
     S        GROUP   ON           SC74
     S        CLASS   ON           SC74                                S
     T        CLASS   ON           SC74                                 T

     S        GROUP   ON           SC75
     S        CLASS   ON           SC75                                S
     S        INIT    ACTIVE       SC75     CLASSS2  XYZZY    JOB21135 S
     S        INIT    ACTIVE       SC75     CLASSS   XYZZY    JOB20889 S
     S        INIT    ACTIVE       SC75     CLASSS3  XYZZY    JOB21136 S

     T        CLASS   ON           SC74                                 T
```

Figure 18-100 Initiator panel

Initiator panel fields

Figure 18-101 on page 925 presents the initiator panel columns in the JES3 environment. None of the columns are delayed access fields. Some of the fields are displayed only on the ALTERNATE view field list.

Title	Discription
ID	Initiator class name
Status	Initiator status
JobName	Job name
StepName	Job step name
Type	Type of user (job)
JobID	Job ID
C	Input class at the time the job was selected for execution
ASID	Address space identifier
ASIDX	Address space identifier in hexadecimal
Owner	Owner of the active job
SysName	System name
SysID	JES member name
JESN	JES subsystem name
JESLevel	JES version and release

Figure 18-101 Initiator panel fields 1 of 2

Title	Description
Seclabel	Security label of the job
SrvClass	Service class of the active job (JES) or the initiator is running (WLM)
Mode	Initiator mode (WLM or JES)
Barrier	Group barrier
Default	Default group indicator
DefCount	Defined initiator count (Group only)
AllocCount	Allocated initiator count
UseCount	Count of initiators in use
Alloc	Allocation indicator (manual/dynamic) (Group only)
Unalloc	Unallocation indicator (manual/dynamic) (Group only)
Group	Group name (Class only)
ResType	Resource type (GROUP, CLASS or INIT)

Figure 18-102 Initiator panel fields 2 of 2

Initiator panel NP field actions

Action characters that can be entered in the NP column on the initiator panel are shown in Figure 18-103.

NP-Description	JES3 Command / SDSF action
D-Display	*I G,main,C,class - Display information about an initiator.
DL-DisplayLong	*I G,ALL,C,class - Display the long form of information about an initiator.
P-Stop	*F G,main,C,class,OFF - Stop an initiator when the current job completes.
S-Start	*F G,main,C,class,ON - Start an initiator.

Figure 18-103 Initiator panel NP field actions

Initiator panel overtypable fields

Figure 18-104 on page 926 shows initiator panel overtypable fields and JES3 commands issued for overtyped values on ResType GROUP rows.

Note: The SDSF panel display issues message `COMMAND ISSUED` even if an invalid overtype value is entered. Enter the ULOG command to view the JES3 action for an overtyped column.

```
Field       JES3 command / SDSF description
Alloc       *F G,main,G,group,ALLOC,DYNAMIC|MANUAL - Allocation indicator
Barrier     *F G,main,G,group,BAR,barrier - Group barrier
DefCount    *F G,main,G,group,INIT,count - Defined intiator count
Mode        *F G,ALL,G,group,MODE,JES|WLM - Initiator mode: JES or WLM
Group       *F C=class,GROUP=group - Group name
Unalloc     *F G,main,G,group,UNALLOC,DYNAMIC|MANUAL- Unallocation indicator
```

Figure 18-104 Initiator panel overtypable fields

18.19 Printers (PR) panel

The Printer panel allows users to display information about JES printers and jobs being printed. The Printers panel is accessed with the PR command. The PR command syntax is shown in Figure 18-105.

```
PR [printer-list]

   printer-list is one or both of the following:
      LCL - all local printers
      RMT - all remote printers.
   PR with no parameters displays information about all printers.
```

Figure 18-105 PR command syntax

Figure 18-106 is a sample display of the SDSF printers panel display.

```
 Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
--
SDSF PRINTER DISPLAY  (ALL)                            LINE 1-14 (14)
COMMAND INPUT ===>                                            SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,BC-BackCkpt,BCnP-BackNumCkpt,BD-BackTop,
ACTION=BN-BackCkptN,BNnP-BackNumCkptN,C-Cancel,CG-CancelGroup,CJ-CancelJob,
ACTION=CP-CancelPosition,CT-CancelStop,D-Display,DL-DisplayLong,
ACTION=E+ADHJLMRTX-RestartOptions,E-Restart,EH-RestartHold,EJ-RestartJob,
ACTION=ER-RestartRescan,Fn-ForwardNum,FC-ForwardCkpt,FCnP-ForwardNumCkpt,
ACTION=FN-ForwardCkptN,FNnP-ForwardNumCkptN,K-ForceFSS,L-Fail,LD-FailDump,
ACTION=S+ADMTX-StartOptions,S-Start,V-VaryOn,VF-VaryOff,X+DRTX-CallWtrOption
s,
ACTION=X-CallWtr,XR-CallWtrResched
NP     PRINTER    Status   Group    SForms   SClass          JobName  JobID
       IAZFSS     AC       LOCAL    STD      CD              VAINI
JOB20994
       IPDPOK     AC       IPDS     STD      I
       IPDWAY     OFF      IPDS     STD      K
       NS         OFF      TCPIP    VTAM     J
       PRTWAY     OFF      TCPIP    STD      J
       PRTWA2     OFF      TCPIP    STD      J
       RM001PR1   AV       RM001    STD
```

Figure 18-106 SDSF printers panel

Printers panel fields

Figure 18-107 and Figure 18-108 on page 929 list the printers panel columns in the JES3 environment. None of the columns are delayed access fields. Some of the fields are displayed only on the ALTERNATE view field list.

Title	Description
PRINTER	Name of the printer
Status	Printer status
Group	Device group
SForms	Printer selection form number
SClass	Printer output selection classes
JobName	Job name
JobID	JES job ID, or work ID
Owner	Owner ID of the active job
Rec-Cnt	Number of line-mode records
Rec-Prt	Number of line-mode records printed
Page-Cnt	Number of output pages
Page-Prt	Number of output pages printed
JP	JES job priority
DP	Output data set priority
C	JES output class
SecLabel	Security label of output group
Forms	Output form number
FCB	Output FCB ID
UCS	Output UCS ID (print train required)
Flash	Output flash ID
Burst	3800 burst indicator
SepDS	Separator page between data sets
PrMode	Printer process mode
SFCB	Printer selection FCB ID
SUCS	Printer selection USC ID
SFlh	3800 or FSS printer selection flash ID
Work-Selection	Printer work selection criteria
SBurst	3800 output selection burst mode
SPrmode1-4	Output selection process mode 1-4
M	3800 or FSS mark forms control
NPro	FSS nonprocess run-out time, in seconds
Mode	FSS control mode of printer
CkptRec	Number of logical records per checkpoint
CkptPage	Number of logical pages per checkpoint
CkptSec	3800 or FSS default checkpoint interval
CkptMode	Checkpoint interval used by FSS
CpyMod	3800 or FSS copy modification module ID
Unit	Printer unit name
DFCB	Device default forms control buffer (FCB)
Setup	Setup mode
CopyMark	Copymark mode
Pau	Pause mode (pause between data sets)
Tr	Printer tracing
FSSName	FSS defined for the printer
FSSProc	Proc used to start the FSS
SysName	System name
JESN	JES subsystem name
JESLevel	JES version and release

Figure 18-107 Printers panel fields 1 of 2

Title	Description
Type	Type of address space: job, started task, or TSO user
Trans	Data translation
Char1-4	Character arrangement table 1-4
FSASysNm	FSA system name
DSPName	Dynamic support program name
DevType	Device type name
Line-Lim-Lo	Printer line limit, minimum
Line-Lim-Hi	Printer line limit, maximum
Page-Lim-Lo	Printer page limit, minimum
Page-Lim-Hi	Printer page limit, maximum
DGrpY	Device cannot process data sets that are destined for any local device
Dyn	Device can be started dynamically
OpLog	Operator command actions are logged in the output of the FSS device
CGS	Character generation storage
B	Burst
PDefault	P default
Copies	Copy count
CB	Clear printer processing indicator
TRC	Table reference character
HFCB	Use designated FCB until status is changed
HChars	Use designated CHARS until status is changed
HUCS	Use designated UCS until status is changed
HCpyMod	Use designated Copy Mod until status is change
HFlash	Use designated Flash until status is changed
HBurst	Use designated Burst until status is changed
HForms	Use designated Forms until status is changed
CCtl	Data carriage control stream
Cmpct	Compaction for SNA remote punches
Comp	Compression
Compact	Compaction table name for SNA remote punches
FCBL	JES will load FCB
LRecL	Logical record length
Select	Device type and subaddress for output

Figure 18-108 Printers panel fields 2 of 2

Printers panel NP field actions

Action characters for the NP column on the printers panel are listed in Figure 18-109 and Figure 18-110 on page 930.

NP Description	JES3 command / SDSF description
B-Back	*RESTART,devname - Backspace a printer:
	C - Most recent checkpoint
	Cnumber - Before the most recent checkpoint lines
	CnumberP - Pages before the most recent checkpoint
	D - Top of the current data set
	N - Last internally-noted checkpoint
	Nnumber - Lines before the last internally-noted checkpoint
	NnumberP - Pages before the last internally-noted checkpoint

Figure 18-109 Printers panel NP column actions 1 of 2

NP Description	JES3 command / SDSF description
C-Cancel	*CANCEL,devname - Canceling output for current job on a printer:
	G - Cancel only the output destined for this device for
	J - Cancel all output of the type PRT or PUN
	P - Stop printer and determine the position of data being processed
	T - Stop the printer once the current activity is canceled
D-Display	*INQUIRY,D,D=devname - Display printer information
DL-DisplayLong	*INQUIRY,D,D=devname - Display the long form of the information
E-Restart	*RESTART,devname - Restart a printer. Use parameters:
	A - Automatic mode. Mutually exclusive with M.
	D - Turn on diagnostic mode. Mutually exclusive with X.
	H - Suspend activity on the current data set and place it in hold
	J - Requeue all data sets for the current job
	L - Reload FCB and UCS/CHARS buffer
	M - Manual mode. Mutually exclusive with A.
	R - Request that it perform a scheduling pass
	T - End it automatically once the current job is rescheduled
	X - Turn off diagnostic mode. Mutually exclusive with D.
F-Forward	*RESTART,devname - Forward space a printer. Required parameters:
	number - Number of lines
	C - Mmost recent checkpoint
	Cnumber - Lines from the most recent checkpoint
	CnumberP - Pages from the most recent checkpoint
	N - last internally-noted checkpoint
	Nnumber - Lines from the last internally-noted checkpoint
	NnumberP - Pages from the last internally-noted checkpoint
K-ForceFSS	FORCE jobname - Force termination of the FSS
L-Fail	*FAIL,devname - Fail device
LD-FailDump	*FAIL,devname,DUMP - Fail the device with a dump
S-Start	*START,devname - Start a printer. Use one or more of parameters:
	A - Automatic mode. Mutually exclusive with M.
	D - Turn on diagnostic mode. Mutually exclusive with X.
	M - Manual mode. Mutually exclusive with A.
	T - End it when this request completes
	X - Turn off diagnostic mode. Mutually exclusive with D.
V-VaryOn	*VARY,devname,ON - Vary the printer online
VF-VaryOFF	*VARY,devname,OFF - Vary the printer offline
X-CallWtr	*X,WTR,OUT=devname - Invoke a writer. Use one or more of parameters:
	D - Turn on diagnostic mode. Mutually exclusive with X.
	R - Suspend writer output until the device is available
	T - End it after the output is printed
	X - Turn off diagnostic mode. Mutually exclusive with D.

Figure 18-110 Printers panel NP column actions 1 of 2

Printers panel overtypable fields

Figure 18-111 on page 931 is a list of overtypable fields on the SDSF printers panel.

Important: For some fields, you must also type an action character when overtyping. See the description of each field.

Field	Description
B	Burst. Action: S
CB	Clear printer processing indicator. Actions: S, X
Char1	Character arrangement table 1. JES3: Type + alone to modify Char2-4. JES3 actions: Bx, Fx, E, S, X
CkptPage	Number of logical pages per checkpoint: 1-32767. JES3 actions: Bx, Fx, E, S, X
CkptSec	3800 or FSS default checkpoint interval: 0-32767 JES3 actions: Bx, Fx, E, S, X
Copies	Copy count. JES3 actions: Bx, Fx, E, S
Copymark	Copymark mode: DATASET, JOB, CONSTANT, DEFAULT, NONE. JES3 actions: Bx, Fx, E, S, X
CpyMod	3800 or FSS CPYMOD ID. JES3 actions: S
DGrpY	Device cannot process data sets destined for a local device
Dyn	Device can be started dynamically
Line-Lim-Hi	Selection output size, maximum number of lines. JES3 actions: Bx, Fx, E, S, X
Line-Lim-Lo	Selection output size, minimum number of lines. JES3 actions: Bx, Fx, E, S, X
Mode	Mode of printer: FSS or JES
NPro	Nonprocess run-out time, in seconds. JES3 actions: Bx, Fx, E, S, X
Oplog	Operator command actions will be logged
Page-Lim-Hi	Selection output size, maximum number of pages. JES3 actions: Bx, Fx, E, S, X
Page-Lim-Lo	Selection output size, minimum number of pages. JES3 actions: Bx, Fx, E, S, X
PDefault	Print default
SBurst	3800 output selection burst mode: Yes or No. JES3 actions: S or X
SClass	Printer output selection classes: classes with no delimiters. JES3 actions: Bx, Fx, E, S, X
SepDs	Separator page between data sets: Yes or No. JES3 actions: Bx, Fx, E
Setup	Printer setup mode
SFCB	Printer selection FCB ID. JES3 actions: Bx, Fx, E, S, X
SFlh	3800 or FSS printer selection flash ID. JES3 actions: Bx, Fx, E, S
SForms	Printer selection form number. JES3 actions: Bx, Fx, E, S, X
SPrMode1	Selection process mode 1. JES3 actions: Bx, Fx, E, S, X
SUCS	Printer selection USC ID. JES3 actions: Bx, Fx, E, S, X
Trans	Data translation: Yes or No
Work-Selection	Printer work selection criteria. JES3 actions: Bx, Fx, E The list of criteria must be enclosed in parentheses. Criteria must be separated by a comma. /value specifies that the characteristic prefixed with a slash (/) is not to be used as work-selection criterion.

Figure 18-111 Printers panel overtypable fields

18.20 Punches (PUN) panel

The punches panel displays information about JES punches and jobs being punched. Punches panel is accessed with the PUN command. Figure 18-112 on page 932 displays the PUN command syntax.

```
PUN [punch-list]

punch-list is one or both of the following:
   LCL - all local punches
   RMT - all remote punches.
   PUN with no parameters displays information about all punches.
```

Figure 18-112 PUN command syntax

Figure 18-113 is an example of the SDSF punches panel display.

```
 Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
---
SDSF PUNCH DISPLAY  SC75                              LINE 1-7 (7)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,BC-BackCkpt,BCn-BackNumCkpt,BD-BackTop,
ACTION=BN-BackCkptN,BNn-BackNumCkptN,C-Cancel,CG-CancelGroup,CJ-CancelJob,
ACTION=CP-CancelPosition,CT-CancelStop,D-Display,DL-DisplayLong,
ACTION=E+ADHJMRTX-RestartOptions,E-Restart,EH-RestartHold,EJ-RestartJob,
ACTION=ER-RestartRescan,FC-ForwardCkpt,FCn-ForwardNumCkpt,FN-ForwardCkptN,
ACTION=FNn-ForwardNumCkptN,L-Fail,LD-FailDump,S+ADMTX-StartOptions,S-Start,
ACTION=V-VaryOn,VF-VaryOff,X+DRTX-StartOptions,X-CallWtr,XR-CallWtrResched
NP   PUNCH      Status   Group    SForms   JobName  JobID    Owner
Rec-Cnt R
     RM002PU1   AV       RM002    STANDARD
     RM003PU1   AV       RM003    STANDARD
     RM004PU1   AV       RM004    STANDARD
     RM005PU1   AV       RM005    STANDARD
```

Figure 18-113 Punches panel

Punches panel fields

Figure 18-114 and Figure 18-115 on page 933 show column titles on the SDSF punches panel in the JES3 environment.

Title	Description
PUNCH	Name of the punch
Status	Punch status
Group	Device group name
SForms	Punch selection form
JobName	Job name
JobID	JES job ID, or work ID
JNum	JES job number
Owner	User ID of job creator
SClass	Punch output selection classes
Rec-Cnt	Number of line-mode records in the job
Rec-Prt	Number of line-mode records punched
Page-Cnt	Output page count
Page-Prt	Output pages punched

Figure 18-114 Punches panel fields 1 of 2

```
Title           Description
SepDS           Separator page between data sets
CCtl            Data carriage control stream
Cmpct           Compaction for SNA remote punches
Comp            Compression for remote SNA punches
Compact         Compaction table name for SNA remote punches
Work-Selection  Punch work selection criteria
SPrMode1-4      Output selection process modes 1 through 4
Select          Device type and device subaddress (remote punches only)
CkptRec         Number of records per checkpoint
Unit            Punch unit name
LRecL           Logical record length of transmitted data (SNA only)
Pau             Pause mode
Tr              Punch tracing
SysName         System name
JESN            JES subsystem name
JESLevel        JES version and release
SecLabel        Security label of the job on the device
Line-Lim-Lo     Punch line limit, minimum
Line-Lim-Hi     Punch line limit, maximum
Forms           Output forms
PrMode          Output PRMODE
DP              Output priority
JP              Job priority
C               Output class
DevType         Device type
DSPName         Dynamic support program name
HForms          Use designated Forms until status is changed
Copies          Copy count
Dyn             Start device dynamically
DGrpY           Device cannot process data sets that are destined for any local device
B               Burst
```

Figure 18-115 Punches panel fields 2 of 2

Punches panel NP field actions

Action characters for the NP column of the punches panel are listed in Figure 18-116 and Figure 18-117 on page 934

```
NP Description  JES3 command / SDSF description
B-Back          *RESTART,devname - Backspace a punch:
                C - Most recent checkpoint
                Cnumber - Before the most recent checkpoint lines
                CnumberP - Pages before the most recent checkpoint
                D - Top of the current data set
                N - Last internally-noted checkpoint
                Nnumber - Lines before the last internally-noted checkpoint
                NnumberP - Pages before the last internally-noted checkpoint
C-Cancel        *CANCEL,devname - Canceling output for current job on a punch:
                G - Cancel only the output destined for this device for
                J - Cancel all output of the type PRT or PUN
                P - Stop punch and determine the position of data being processed
                T - Stop the punch once the current activity is canceled
```

Figure 18-116 Punches panel NP field actions 1 of 2

NP Description	JES3 command / SDSF description
D-Display	*INQUIRY,D,D=devname - Display punch information
DL-DisplayLong	*INQUIRY,D,D=devname - Display the long form of the information
E-Restart	*RESTART,devname - Restart a punch. Use parameters:
	A - Automatic mode. Mutually exclusive with M.
	D - Turn on diagnostic mode. Mutually exclusive with X.
	H - Suspend activity on the current data set and place it in hold
	J - Requeue all data sets for the current job
	M - Manual mode. Mutually exclusive with A.
	R - Request that it perform a scheduling pass
	T - End it automatically once the current job is rescheduled
	X - Turn off diagnostic mode. Mutually exclusive with D.
F-Forward	*RESTART,devname - Forward space a punch. Required parameters:
	C - Mmost recent checkpoint
	Cnumber - Lines from the most recent checkpoint
	CnumberP - Pages from the most recent checkpoint
	N - last internally-noted checkpoint
	Nnumber - Lines from the last internally-noted checkpoint
	NnumberP - Pages from the last internally-noted checkpoint
K-ForceFSS	FORCE jobname - Force termination of the FSS
L-Fail	*FAIL,devname - Fail device
LD-FailDump	*FAIL,devname,DUMP - Fail the device with a dump
S-Start	*START,devname - Start a punchrinter. Use one or more of parameters:
	A - Automatic mode. Mutually exclusive with M. **D** - Turn on diagnostic mode. Mutually exclusive with X. **M** - Manual mode. Mutually exclusive with A. **T** - End it when this request completes **X** - Turn off diagnostic mode. Mutually exclusive with D.
V-VaryOn	*VARY,devname,ON - Vary the punch online
VF-VaryOFF	*VARY,devname,OFF - Vary the punch offline
X-CallWtr	*X,WTR,OUT=devname - Invoke a writer. Use one or more of parameters:
	D - Turn on diagnostic mode. Mutually exclusive with X.
	R - Suspend punch writer output until the device is available
	T - End it after the output is punched
	X - Turn off diagnostic mode. Mutually exclusive with D.

Figure 18-117 Punches panel NP field actions 2of 2

Punches panel overtypable fields

The fields listed in Figure 18-118 on page 935 can be overtyped on the punches panel.

Important: For some fields, you must also type an action character when overtyping. See the description of each field.

```
Title            Description
B                *R, *S or *X - Burst. Action: S
Copies           *R or *S - Copy count (JES3 only). Actions: E or S
DGrpY            Process local data sets
Dyn              Start device dynamically
Line-Lim-Hi      Selection output size, maximum number of cards. Actions: E, S or X
Line-Lim-Lo      Selection output size, minimum number of cards. JES3 actions: E, S or
                 X
SClass           Punch output selection classes: classes with no delimiters, up to 36.
                 JES3 actions: Actions: E, S or X
SepDs            Separator card between data sets: Yes or No. JES3 actions: E, S or X
SPrMode1         Selection process mode 1.  Type + alone to work with multiple values.
                 JES3 actions: E, S or X
Work-Selection   Punch work selection criteria. JES3 actions: E, S or X
                 The list of criteria must be enclosed in parentheses.  Criteria must
                 be separated by a comma.  /value specifies that the characteristic
                 prefixed with a slash (/) is not to be used as work-selection
                 criterion.
```

Figure 18-118 Punches panel overtypable fields

18.21 Readers (RDR) panel

The readers panel, Figure 18-119, displays information about JES readers and jobs being processed by readers.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF READER DISPLAY  SC75                                LINE 1-8 (8)
COMMAND INPUT ===>                                          SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,CH-CancelHold,CHN-CancelNoHold,
ACTION=CK-CancelAlloc,CKN-CancelPurge,D-Display,DL-DisplayLong,L-Fail,
ACTION=LD-FailDump,S-Start,SH-StartHold,SHN-StartNoHold,SK-StartAlloc,
ACTION=SKN-StartPurge,V-VaryOn,VF-VaryOff,X-Call,XC-CallCardImage,XH-CallHold,
ACTION=XHN-CallNoHold,XK-CallAlloc,XKN-CallPurge
NP   READER     Status   Group    JobName  JobID    Owner    Rec-Cnt    Rec-Pro
     RM001RD1   AV       RM001
     RM002RD1   AC       RM002     (NONE)
     RM003RD1   AV       RM003
     RM004RD1   AV       RM004
     RM005RD1   AV       RM005
     RM006RD1   AV       RM006
     RM007RD1   AV       RM007
```

Figure 18-119 Readers panel

The readers panel is accessed with the RDR command. Figure 18-120 on page 936 shows the RDR command syntax.

```
RDR [reader-list]

   reader-list is up one or both of the following, in any combination:
         LCL - all local readers
         RMT - all remote readers.
   RDR with no parameters displays information about all readers.
```

Figure 18-120 RDR command syntax

Readers panel fields

Figure 18-121 is the list of the readers panel columns in the JES3 environment.

Title	Description
READER	Device name. This is the fixed field.
Status	Reader status
Group	Device group name
JobName	Job name
JobID	Active job ID
Type	Type of active address space. Not in the default field list
JNum	Active job number. Not in the default field list
Owner	User ID of owner
Rec-Cnt	Number of records in the job
Rec-Proc	Number of records processed
C	Default execution class
MC	Message class
Unit	Reader unit name
SysName	System name
JESN	JES subsystem name
JESLevel	z/OS JES level
SecLabel	Security label of the job on the reader
DevType	Device type name
DSPName	DSP name
AReq	Account number required on job card
PReq	Programmer name required on job card
SWA	SWA ABOVE or BELOW
BLP	BLP label setting is respected
DP	Default job priority
ML	Default job message level
AL	Default allocation message level
Time	Default time limit
Region	Default region size

Figure 18-121 Readers pane fields

Readers panel NP field actions

Figure 18-122 on page 937 lists action characters that can be entered in the NP column on the readers panel.

```
NP Description   JES3 command / SDSF description
C-Cancel         *C devname. You can add one or more of these parameters:
                 H - Hold the control-card processor
                 HN - Process jobs that are completely entered.
                 K - Leave hot readers allocated.
                 KN - Do not leave hot readers allocated.

                 You cannot combine H and HN or K and KN.
D-Display        *I D,D=devname,S - Display information
DL-DisplayLong   *I D,D=devname - Display the long form of information
L-Fail           *FAIL devname - Fail the reader DSP (JES3 only)
LD-FailDump      *FAIL devname,DUMP - Fail the reader DSP and take a dump
S-Start          *S devname = Start. You can add one or more of parameters:
                 H - Hold the control-card processor.
                 HN - Process jobs after the batch is created.
                 K - Keep active once end-of-file is reached.
                 KN - Purge when end-of-file is reached.

                 You cannot combine H and HN or K and KN.
V-VaryOn         *MODIFY VARY,devname,ON - Vary online
VF-VaryOff       *MODIFY VARY,devname,OFF - Vary offline
X-Call           *X CR,IN=devname - Invoke card reader support. You can add one or more
                 parameters:
                 C - Enable card image support
                 H - Place the control-card processor in hold
                 HN - Allow jobs to be processed
                 K - Remain active after end-of-file is reached
                 KN - Purge after end-of-file is reached

                 You cannot combine H and HN or K and KN.
```

Figure 18-122 Readers panel NP field actions

18.22 Lines (LI) panel

The Lines (LI) panel displays information about JES NJE and RJP lines. The lines panel is accessed with the LI command. Figure 18-123 shows a lines panel with no line data.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF LINE DISPLAY  SC75                                LINE 0-0 (0)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,DE-DisplayErrors,
ACTION=DL-DisplayLong,DS-DisplayStatus,E-Restart,I-Interrupt,L-Fail,
ACTION=LD-FailDump,S-Start,SL-StartLog,SNL-StartNoLog,SNR-StartNoRcv,
ACTION=SR-StartRcv,SRJP-StartRJP,V-VaryOn,VF-VaryOff
NP   DEVICE       Status   Unit  Node     JobName  JobID    Owner
Proc-Lines
```

Figure 18-123 An empty SDSF lines panel

Lines panel fields

Figure 18-124 on page 938 is a list of the lines panel columns in the JES3 environment.

Title	Description
DEVICE	Device name. This is the fixed field.
Status	Line status
Unit	Line address or type
Node	Node that the line is connected to
JobName	Job name
JobID	JES job ID
Owner	User ID of owner
Proc-Lines	Number of lines processed for the job.
Tot-Lines	Number of lines in the job.
Type	Type of line (RJP or NJE)
ADisc	Line disconnect option
Code	BSC adaptor code
Comp	BSC data compression option
Duplex	BSC line mode
Intf	BSC adapter interface
Speed	Speed of the line
Tr	Trace I/O option
Transp	BSC transparency feature
Password	Password
Discon	Disconnect status: NO, INTERRUPT, or QUIESCE (only for active lines).
SysName	System Name
JESN	JES subsystem name
JESLevel	JES version and release

Figure 18-124 Lines panel fields

Lines panel NP column actions

Figure 18-125 is a list of the lines panel NP column actions.

NP action	JES3 command / SDSF description
C-Cancel	*C - Cancel a or line
D-Display	*I - Display the line, transmitter or receiver in the log Optional parameters: **L** - long form, for the line **S** - status of the names of the BSC line **E** - cumulative error statistics for the line
E-Restart	*R - Restart the line.
I-Interrupt	*C - Interrupt the line.
L-Fail	*FAIL - Fail the line DSP
LD-FailDump	*FAIL - Fail the line DSP with a dump
S-Start	*S - Start a line
SL-StartLog	*S - Start the line with tracing
SNL-StartNoLog	*S - Start the line without tracing
SR-StartRcv	*S - Start and allow network jobs to be received
SNR-StartNoRcv	*S - Start but prevent network jobs from being received
SRJP-StartRJP	*S - Start RJP on the line
V-VaryOn	*F VARY - Vary online
VF-VaryOff	*F VARY - Vary offline

Figure 18-125 Lines panel NP field action

18.23 Nodes (NO) panel

The nodes panel displays information about JES NJE nodes. The nodes panel is accessed with the NODE command shown in Figure 18-126.

```
{NODES |NODE |NO}

NODES with no parameters displays all nodes.
```

Figure 18-126 NODE command syntax

Figure 18-127 is an example of the SDSF nodes panel display. The command ARR LineName L was issued for the panel. -- JES3 NJE TCPIP and SNA connections do not use lines.

```
 Display  Filter  View  Print  Options  Search  Help
---------------------------------------------------------------------------
---
SDSF NODE DISPLAY  SC75     WTSC75J3                  LINE 1-10 (10)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,A-Release,D-Display,DL-DisplayLines,
ACTION=EL-ResetLines,H-Hold,SN-Start
NP   NODENAME Status                Path     PType Hold VerifyP  SendP
SysNa
     WTSCMXA  CONNECTED             WTSCNET        NONE NOTSET   NOTSET
SC75
     WTSCNET  CONNECTED             WTSCNET  TCPIP NONE NOTSET   NOTSET
SC75
     WTSCPLX1 UNCONNECTED           WTSCPLX1 TCPIP NONE NOTSET   NOTSET
SC75
     WTSCPLX2 UNCONNECTED           WTSCPLX2 SNA   NONE NOTSET   NOTSET
SC75
     WTSCPLX3 UNCONNECTED           WTSCPLX3 SNA   NONE NOTSET   NOTSET
SC75
     WTSCPLX4 CONNECTED             WTSCPLX4 TCPIP NONE NOTSET   NOTSET
SC75
     WTSCPLX7 CONNECTED             WTSCPLX7 TCPIP NONE NOTSET   NOTSET
SC75
     WTSCPLX9 CONNECTED             WTSCPLX9 TCPIP NONE NOTSET   NOTSET
SC75
     WTSCPOK  UNCONNECTED           WTSCPLX1       NONE NOTSET   NOTSET
SC75
     WTSC75J3 OWNNODE               WTSC75J3       NONE NOTSET   OWNNODE
SC75
```

Figure 18-127 Nodes panel

Nodes panel fields

Figure 18-128 and Figure 18-129 on page 940 are the list of columns on the nodes panel in the JES3 environment.

Title	Description
NodeName	Node name. This is the fixed field
Status	Node status for the first path.
Hold	Job hold indicator for the local node
LineName	Line dedicated to NJE for this node
Tr	Trace option
VerifyP	Password received from the node
SendP	Password sent to the node
SysName	System Name
JESN	JES subsystem name
JESLevel	JES version and release

Figure 18-128 Nodes panel fields 1 of 2

Title	Description
MaxRetries	Number of retries to attempt before ending the BSC NJE line
Path	Name of the adjacent node in the path
PType	Protocol type
BDTName	Bulk Data Transfer (BDT) ID
PartName	Spool partition name to which all incoming NJE streams are written
MaxLines	Maximum number of lines for the node.
Direct	Specifies whether the node can be directly attached only
SSignon	Specifies whether secure signon protocol is to be used
JTNum	Number of job transmitters associated with the TCP/IP node
JRNum	Number of job receivers associated with the TCP/IP node
STNum	Number of SYSOUT transmitters associated with the TCP/IP node
SRNum	Number of SYSOUT receivers associated with the TCP/IP node
Secure	Use secure (TLS) socket
PwCntl	Password encryption control
XNameReq	Specifies whether inbound SYSOUT can be held for an external writer if no external writer name is supplied
Connect	Automatically reconnect
Conn-int	Connection interval (minutes)
BufSz	Buffer size
Strm	Number of concurrent streams
PrtDef	Print class default for networking output received at the home node
PrtTSO	TSO data set default class for networking output received at the home node
PrtXwtr	External writer data set default class for networking output received at the home node
PunDef	Punch class default for networking output received at the home node
NetPr	Number of logical network printers on the home node
NetPu	Number of logical network punches on the home node
CTC	Channel to channel node

Figure 18-129 Nodes panel fields 2 of 2

Nodes panel NP field actions

Figure 18-130 is a list of the nodes panel NP column actions.

```
NP action        JES3 command / SDSF description
D-Display        *I NJE,N=node - Display information about a node in the log.
DL-DisplayLines  *I NJE,N=node,LINE - Display lines defined to a BSC connected node
EL-ResetLines    *F NJE,N=node,FORCE - Reset BSC lines to the node
H-Hold           *F NJE,N=node,HOLD - Hold jobs destined for this directly-attached
                 node
SN-Hold          *S,TCP,SOCKET=node - Start node communication on a line.
```

Figure 18-130 Nodes panel NP column actions

Nodes panel overtype fields

Figure 18-131 on page 941 is a list of the nodes panel overtypable columns and the JES3 command to implement the overtyped value.

```
Title      JES3 command / SDSF description
Connect    *F,NJE - Automatically reconnect
Conn-int   *F,NJE - Connection interval in minutes
JRNum      *F,NJE - Number of job receivers for the TCP/IP node
JTNum      *F,NJE - Number of job transmitters for the TCP/IP node
MaxRetries *F,NJE - Number of retries to attempt before ending the BSC NJE line
NHold      *F,NJE - Process inbound SYSOUT in NETDATA format (Yes or No)
PartName   *F,NJE - Spool partition name for all incoming streams
Path       *F,NJE - Adjacent node in the path
PrtDef     *F,NJE - Print class default for networking output received at the home node
PrtTSO     *F,NJE - TSO data set default class for networking output received
PrtXwtr    *F,NJE - External writer data set default for networking output received
PType      *F,NJE - Protocol type: TCP, SNA, BSC
PunDef     *F,NJE - Punch class default for networking output received at the home node
PWCntl     *F,NJE - Password encryption control
Secure     *F,NJE - Use secure (TLS) socket
SRNum      *F,NJE - SYSOUT receivers for the TCP/IP node
SSignon    *F,NJE - Use secure signon protocol
STNum      *F,NJE - SYSOUT transmitters for the TCP/IP node
```

Figure 18-131 Nodes panel overtypable fields

18.24 Network servers (NS) panel

The Network Server (NS) panel allows users to display information about server-type networking devices on the node:

- NETSERV devices used to communicate between JES and TCP/IP
- BDT instances used to communicate between JES3 and VTAM

The Network Server panel is accessed with the NS command, which does not have any arguments.

Figure 18-131 is an example of the SDSF network servers panel display.

```
 Display  Filter  View  Print  Options  Search  Help
------------------------------------------------------------------------------
---
SDSF NS DISPLAY  SC75                                LINE 1-2 (2)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,E-Restart,K-SysCancel,
ACTION=KD-SysCancelDump,L-Fail,LD-FailDump,S-Start,X-CallTCP,Z-SysForce
NP   DEVICE       Status   DSPName  Stack    CTr VTr JTr ASID  SrvJobNm
IPName
     JES3NS       ACTIVE   JOB20820 TCPIP    NO  NO  NO  003B  JOB20941
WTSC75.
     JES3NS9      INACTIVE          TCPIP    NO  NO  NO
WTSC75.
```

Figure 18-132 Network servers panel

Network servers panel fields

Figure 18-133 on page 942 lists the fields in the JES3 environment on the network servers panel.

Title	Description
DEVICE	Name of the network server. This is the fixed field.
Status	Device status
DSPName	Dynamic support program name (JES3 only)
Stack	Name of the TCP/IP stack
Restart	Restart the device automatically
Rest-Int	Restart interval (minutes)
CTr	Common tracing
VTr	Verbose tracing
JTr	JES tracing
ASID	ASID of the network server
SrvJobNm	Job name of the network server address space
IPName	Local TCP/IP host name
Port	Local TCP/IP port number
Secure	Secure (TLS) socket
SysName	System Name
JESN	JES subsystem name
JESLevel	z/OS JES level

Figure 18-133 Network servers panel fields

Network servers panel NP field actions

Figure 18-134 is a list of the network servers panel NP column actions.

```
NP-Description     JES3 command / SDSF action
C-Cancel           *C - Cancel a network server (JES3 only)
D-Display          *I - Display the network server in the log.
E-Restart          *R - Restart the network server
K-SysCancel        CANCEL - Cancel the network server address space
KD-SysCancelDump   C jobname,DUMP - Cancel the network server address space with a dump
L-Fail             *FAIL - Fail the device DSP (JES3 only)
LD-FailDump        *FAIL,devname,DUMP - Fail the device DSP with a dump (JES3 only)
X-CallTCP          *X - Invoke the network server DSP (JES3 only)
Z-SysForce         FORCE - Force the network server address space
```

Figure 18-134 Network servers panel NP field actions

Network servers panel overtypable columns

Figure 18-135 is a list of the network servers panel overtypable columns and the JES3 command to carry out the overtyping.

```
Field      JES3 command / SDSF description
CTr        *F NETSERV= - Common tracing
IPName     *F NETSERV=- Local TCP/IP host name
JTr        *F NETSERV=- JES tracing
Port       *F NETSERV=- Local TCP/IP port number
VTr        *F NETSERV=- Verbose tracing
```

Figure 18-135 Network server panel overtypable fields

18.25 Network Connection (NC) panel

The Network Connection (NC) panel allows users to display information about networking connections to an adjacent node:

- SOCKET devices that represent a TCP/IP networking connection
- Active BSC NJE lines
- Associated NJE transmitters and receivers

The Network Connection panel is accessed with the NC command. The NC syntax is shown in Figure 18-136.

```
NC [SHORT]

with no parameters displays network connections, transmitters and receivers.

SHORT or S displays information about network connections only. Transmitters and
receivers are not displayed.
```

Figure 18-136 NC command syntax

Figure 18-137 is an example of the SDSF network connections panel display, invoked with the NC command, in the JES3 environment.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF NC DISPLAY  SC75                                  INVALID COMMAND
COMMAND INPUT ===>                                             SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,SN-StartNetComm
NP   DEVICE       Status   Type ANode    JobName  JobID    JType    Owner
Pr
     @0000002     ACTIVE   TCP  WTSCPLX9
     @0000002.JR1 INACTIVE
     @0000002.JT1 INACTIVE
     @0000002.OR1 INACTIVE
     @0000002.OT1 INACTIVE
     WTSCNET      ACTIVE   TCP  WTSCNET
     WTSCNET.JR1  INACTIVE
     WTSCNET.JT1  INACTIVE
     WTSCNET.OR1  INACTIVE
     WTSCNET.OT1  INACTIVE
     WTSCPLX1     INACTIVE TCP  WTSCPLX1
     WTSCPLX4     ACTIVE   TCP  WTSCPLX4
     WTSCPLX4.JR1 INACTIVE
     WTSCPLX4.JT1 INACTIVE
     WTSCPLX4.OR1 INACTIVE
     WTSCPLX4.OT1 ACTIVE                 VAINI    JOB20994 JOB
     WTSCPLX7     INACTIVE TCP  WTSCPLX7
```

Figure 18-137 Network connection panel

In Figure 18-137 the TCPIP NJE connection to the local node (WTSC75J3) was started from the foreign node WTSCPLX9. JES3 on the local socket side dynamically created a SOCKET definition with a unique name of @0000002 for the connection.

Note: The concept of foreign and local sockets exists in TCP/IP. A JES3 socket defines JES3's usage of a foreign socket only. The local socket is implicitly defined by the NETSERV statement.

A socket definition, defining a foreign socket that is used to connect to TCP/IP: Each socket runs as a subtask under a Netserv address space. The socket definition consists of the following information:

- A unique name representing the view of the socket by JES3 global and used in inquiry and modify commands as well as internal JES3 processing of outbound and inbound TCP/IP data.
- A host name.
- A port number, handled the same way as the Netserv port number.
- The Netserv under which the socket task runs.

If communication is started on a foreign socket, TCP/IP creates an ephemeral socket on the foreign socket side. JES3 on the local socket side, in turn, dynamically creates a SOCKET definition with a unique name of *@nnnnnnn*, where *nnnnnnn* starts at 0000001 and is assigned to the first available number. JES3 calls this socket definition a server socket, because when a TCP/IP connection is established, the node on which the connection is initiated is known to TCP/IP as a client and the other node is known to TCP/IP as a server. The server node always creates a server socket, even if an inactive socket definition exists on the server side.

Network servers panel fields

Title	Description
DEVICE	Name of the connection, transmitter or receiver. This is the fixed field.
Status	Device status
Type	Connection type (SNA, BSC, TCP)
ANode	Adjacent node
Jobname	Job name
JobID	JES job ID
JType	Type of address space
Owner	User ID of job creator
Proc-Lines	Number of lines processed for the job
Tot-Lines	Number of lines in the job
Unit	Unit associated with line
JRNum	Job receiver count
JTNum	Job transmitter count
SRNum	SYSOUT receiver count
STNum	SYSOUT transmitter count
CTr	Common tracting
JTr	JES tracing
VTr	Verbose tracing
IPName	IP host name
Port	TCP/IP port number
Secure	Secure (TLS) connection
RelConn	Related connection name
SrvName	Name of the associated server device
SysName	System Name
JESN	JES subsystem name
JESLevel	z/OS JES version and release

Figure 18-138 Network connections panel fields

Figure 18-138 on page 945 lists network connection panel fields for the JES3 environment.

Network connection panel NP field actions

Figure 18-139 is a list of the network connections panel NP column actions.

```
NP-Description         JES3 command / SDSF description
C-Cancel               *C - Cancel the connection (JES3 only)
D-Display              *I - Display the network connection in the log.
SN-StartNetComm        *S - Start network communication
```

Figure 18-139 Network connections panel NP column actions

Network connection panel overtypable columns

Figure 18-140 is a list of the network connection panel overtypable columns and the JES3 command to carry out the overtyping.

```
Field          JES3 command - SDSF description
ANode          *F - Adjacent node
CTr            *F - Common tracing
IPName         *F - IP host name
JTr            *F - JES tracing
Port           *F - TCP/IP port number
SrvName        *F - Name of the associated server device
VTr            *F - Verbose tracing
```

Figure 18-140 Network connections panel overtypable columns

18.26 Spool volumes (SP) panel

The spool volumes panel displays information about JES spool volumes.The spool volumes panel is accessed with the SP command, which does not have any arguments. Figure 18-141 is an example of data on the spool volumes panel.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF SPOOL DISPLAY SC75      1% ACT  1772K FRE  1759K LINE 1-5 (5)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,A-Release,D-Display,DL-DisplayLong,H-Hold,
ACTION=HC-HoldCancel,HP-HoldStop,J-Jobqueue,P-Purge,U-Use
NP   NAME     Status       TGPct TGNum TGUse Ext LoCyl    LoTrk
LoHe
     DRAINED  INACTIVE       100     0     0 00  00000000 0000000000000000
0000
     UNAVAIL  INACTIVE       100     0     0 00  00000000 0000000000000000
0000
     JES3PART ACTIVE           1 1772K 13728 00  00000000 0000000000000000
0000
     SPOOL1   ACTIVE          44 15000  6703 02  00000014 000000000000012C
0000
     SPOOL2   ACTIVE           1 1758K  7025 03  0000000F 00000000000000E1
0000
```

Figure 18-141 Spool volumes panel

Spool volumes panel fields

The spool volumes panel title line shows the following in Figure 18-142 on page 947.

```
   Display  Filter  View  Print  Options  Search  Help

-------------------------------------------------------------------------------
---
 SDSF SPOOL DISPLAY SC75       1% ACT  1772K FRE  1759K LINE 1-5 (5)
                      |         |      |          |          |     |
                      |         |    Active    Free        Lines   |
                      |         |    tracks    tracks     displayed|
                      |    Spool utilization                     Total
                  System name the user is logged on to            lines
```

Figure 18-142 Spool volumes panel title line

The Spool Volumes panel of the JES3 environment includes some or all of the following fields listed in Figure 18-143. (The order and titles may be different, depending upon installation and user options.)

Title	Description
NAME	DDNAME, DRAINED or UNAVAIL. This is the fixed field.
Status	Spool or partition status (active, starting, halting, draining, inactive)
TGPct	Spool utilization
TGNum	Total track groups
TGUse	Track groups in use
Ext	Extent number, in hexadecimal
LoCyl	Low cylinder
LoTrk	Absolute low track number, in hexadecimal
LoHead	Low head
HiCyl	High cylinder
HiTrk	Absolute high track number, in hexadecimal
HiHead	High head
TrkPerCyl	Tracks per cylinder
RecPerTrk	Records per track
TrkPerTG	Tracks per track group
Type	Spool type (PARTITION or EXTENT)
PartName	Partition name
OverFNam	Overflow partition name
OverAllow	Indicates if overflow from this partition to another partition is allowed
OverOccur	Indicates if overflow from this partition to another partition occurred
OverInto	Indicates if overflow into this partition is allowed
PTracks	Total tracks in the partition
PTrackU	Tracks in use in the partition
DTracks	Total tracks in the data set
DTrackU	Tracks in use in the data set
Default	Default partition indicator
STT	Single track table indicator
MargPct	Marginal SLIM threshold percentage shown only on the row for the partition
MargExc	Marginal threshold exceeded
MinPct	Minimal SLIM threshold percentage
MinExc	Marginal threshold exceeded
DataSetName	Data set name

Figure 18-143 Spool volumes panel fields

Spool volumes panel NP field actions

Figure 18-144 on page 948 is a list of spool volumes panel NP field actions.

NP-Description	JES3 command / SDSF description
A-Release	*F Q,DD=(NAME),RELEASE - Release the spool data set and all jobs that have data on spool for scheduling
D-Display	*I Q,DD=|SP=(NAME) - Display the status of a spool volume or spool partition
DL-DisplayLong	*I Q,SP=(NAME),DD - Display the long form of spool partition status
H-Hold	*F Q,DD=(NAME),HOLD - Hold the spool data set and further scheduling for jobs with data on the data set
HC-HoldCancel	*F Q,DD=(NAME),CANCEL - Hold the spool data set and cancel all jobs using it
HP-HoldStop	*F Q,DD=(NAME),STOP - Hold the spool data set and hold further scheduling of jobs with data on it. Cancel jobs active on the main and using the data set.
J-Jobqueue	*I Q,SP=(NAME),U - Display all jobs using the spool partition
P-Purge	*F Q,DD=(NAME),DRAIN - Drain a spool volume.
U-Use	*F Q,DD=(NAME),USE - Resume allocating space on the spool data set

Figure 18-144 Spool volumes panel NP field actions

Spool volumes panel overtypable fields

Figure 18-145 is a list of the spool volumes panel overtypable columns and the JES3 command to carry out the overtyping.

```
Field          JES3 command / SDSF description
MinPct         *F Q,SP=spart,MIN=nn - Minimal SLIM threshold percentage
OverFNam       *F Q,SP=spart,O={spart2 | YES | NO] - Overflow partition name
PartName       *F Q,DD=ddname,SP=spart - Partition name
```

Figure 18-145 Spool volumes panel overtypable fields

18.27 User Session Log (ULOG) panel

The User Session Log (ULOG) panel displays the MVS and JES commands and responses issued during the user's session, including commands generated by SDSF and SAF. SDSF deletes the user session log when an SDSF session is ended or when the ULOG CLOSE command is issued.

SDSF uses MVS console services to acquire an extended console that is used to issue commands and receive responses.

Responses can be returned to ULOG only if:

- The command processor issues the message using the console ID of the extended console.
- The command processor supports use of the CART (command and response token). To get a command response on the same panel as the / command was entered, the command processor must specify both console ID and CART. To get the response in the ULOG, only the console ID is required.
- The message response is not being suppressed through MPF (the message processing facility).
- The D R,L command filters the response based on the issuing console ID. To see all outstanding replies, issue D R,L,CN=(ALL).

The ULOG panel is accessed with the ULOG command, shown in Figure 18-146.

```
{ULOG|U} [CLOSE]

   CLOSE deletes all entries in the user session log and deactivates the
   extended console.
   ULOG with no parameters displays the ULOG panel. An extended console is
   activated if one is not already active.
```

Figure 18-146 ULOG command syntax

The SDSF ULOG extended console activation may fail with message “`ISF032I CONSOLE emcs_name ACTIVATE FAILED, RETURN CODE 0004, REASON CODE 0000` “. Use the SET CONSOLE command or action bar Options choice 9 to set a new name for the extended console.

Figure 18-147 shows an example of the JES3 commands entered on the spool volumes panel and command responses on the ULOG panel. The D EMCS command was issued as a

slash (/) command on a COMMAND INPUT line. The ULOG panel is scrolled to the right past the message text prefix.

All commands and command responses are always logged into the system hardcopy log.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF ULOG  CONSOLE NEWCONS                        LINE 0       COLUMNS 42-
121
COMMAND INPUT ===>                                            SCROLL ===>
HALF
******************************** TOP OF DATA
**********************************
  ISF032I CONSOLE METOO ACTIVATE FAILED, RETURN CODE 0004, REASON CODE 0000
  ISF031I CONSOLE NEWCONS ACTIVATED
 -*I Q,SP=(DRAINED)
  IAT8980 DRAINED  HAS NO SPOOL DATA SETS
  IAT8607 INQUIRY ON SPOOL PARTITION STATUS COMPLETE
 -*I Q,SP=(JES3PART),DD
  IAT8513 SPOOL1   JES3PART  15,000 GRPS,  8,281 LEFT ( 55%), STT
  IAT8513 SPOOL2   JES3PART  1,800K GRPS, 1,793K LEFT (100%), STT
  IAT8607 INQUIRY ON SPOOL PARTITION STATUS COMPLETE
-D EMCS
  IEE129I  11.16.19 DISPLAY EMCS 651
  DISPLAY EMCS
  NUMBER OF CONSOLES MATCHING CRITERIA: 28
  SC74     *ROUTE74 *AXR0374 *SYSLG75 *OPLOG75 *AXR0375 SYSJ3N01
  XYZ      U2M2     *SYSLG74 *OPLOG74 *AXR0174 *DICNS75 *AXR0175
  *AXR0275 SYSJ3D01 VAINJ    *DICNS74 *AXR0474 *AXR0274 SC75
  *ROUTE75 *AXR0475 SYSJ3R01 VAINI    METOO    VJUHA    NEWCONS
```

Figure 18-147 Example of commands and responses on the ULOG panel

The message text on the ULOG message lines is prefixed with the system name, julian date, time stamp and ID of job that issued the message:

```
SC75      2011095  11:10:32.12  JES3       IAT8607 INQUIRY ON SPOOL PARTITION S
```

When a JES or MVS operator command is issued on a active SDSF panel, the responses of the commands are displayed on the active panel, as shown in Figure 18-148 on page 951.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF ULOG  CONSOLE NEWCONS                        LINE  COMMAND ISSUED
COMMAND INPUT ===>                                          SCROLL ===>
HALF
RESPONSE=SC75
 CNZ4100I 11.51.40 CONSOLE DISPLAY 680
 CONSOLES MATCHING COMMAND: D C,CN=METOO
 MSG:CURR=0    LIM=1500 RPLY:CURR=0    LIM=999   SYS=SC75     PFK=00
 METOO     TYPE=EMCS     STATUS=ACT-SC75
           DEFINED=(*ALL)
           MATCHED=(*ALL)
    ATTRIBUTES ON *DEFINED
       AUTH=(INFO)      CMDSYS=SC75         NBUF=N/A
       KEY=NONE
       MFORM=(M)
       LEVEL=(ALL)
       MONITOR=(NONE)                      INTIDS=N  UNKNIDS=N
       ROUT=(NONE)
       MSCOPE=(*ALL)
   SC74     *ROUTE74 *AXRO374 *SYSLG75 *OPLOG75 *AXRO375 SYSJ3N01
   XYZ      U2M2     *SYSLG74 *OPLOG74 *AXRO174 *DICNS75 *AXRO175
   *AXRO275 SYSJ3D01 VAINJ    *DICNS74 *AXRO474 *AXRO274 SC75
    *ROUTE75 *AXRO475 SYSJ3R01 VAINI    METOO    VJUHA    NEWCONS
```

Figure 18-148 MVS command response on an active SDSF panel

Users can request that SDSF use a console ID of 0 with the **i** parameter on the **/** command (i/ command). The i/ command responses are not displayed on the ULOG panel, but are logged into the system hardcopy log.

18.28 Hardcopy log panels

The MVS hardcopy processing allows your installation to have a permanent record of system activity and helps you audit the use of operator commands. You can record system messages and, optionally, commands, by using either the system log (SYSLOG) or the operations log (OPERLOG). Hardcopy processing is required in a sysplex.

The SDSF hardcopy log includes:

- The SDSF OPERLOG (LOG O) panel displays a merged, sysplex-wide system message log, which contains console messages, operator commands, and operator responses for the MVS systems. The SDSF OPERLOG messages are retrieved from the MVS operations log (OPERLOG).

  The MVS operations log (OPERLOG) is a log stream that uses the system logger to record and merge communications from each system in a sysplex. Only the systems in a sysplex that have specified and activated the operations log will have their records sent to the MVS OPERLOG.

- The SDSF SYSLOG (LOG S) panel displays the MVS system log messages.

  The system log (SYSLOG) is a data set residing in the primary job entry subsystem's spool space. In MVS, the system log data set includes all entries made by the WTL

(write-to-log) macro as well as the hardcopy log (console messages, operator commands, and operator responses for a z/OS system). SYSLOG is maintained by JES in the JES SPOOL space.

- JES3 DLOG centrally records command and message traffic for systems in a JES3 complex in JES3 format. The JES3 DLOG is written to SYSLOG on the global processor.

  SYSLOG on the global processor must be active when DLOG is active.

IBM recommends use of OPERLOG on all systems in the sysplex as the only normally active hardcopy medium. The OPERLOG MDB records contain considerably more information than either the JES3 DLOG or SYSLOG formats. In addition, with OPERLOG each system writes its own command and message traffic to the common log, rather than all log activity taking place on the JES3 global processor, as with DLOG.

You control which messages are included in the hardcopy message set with the VARY,HARDCPY command. The HARDCPY operand on the VARY command assigns SYSLOG or OPERLOG as the hardcopy medium. You can assign both SYSLOG and OPERLOG as the hardcopy medium. To display information about the hardcopy medium, enter:

```
DISPLAY CONSOLES,HARDCOPY or D C,HC
```

Unless you specify otherwise, the system includes all operator and system commands, responses, and status displays in the hardcopy message set. To request that some commands and command responses not be included in the hardcopy message set, the system gives you the following choices on the VARY,HARDCPY command:

- NOCMDS - The system does not include operator commands or their responses in the hardcopy message set.
- INCMDS - The system includes all operator commands and their responses, excluding any status displays, in the hardcopy message set.
- STCMDS or CMDS - The system includes all operator and system commands, their responses, and status displays in the hardcopy message set. As of z/OS V1R8, STCMDS and CMDS are equivalent.

Use the JES3 *MODIFY,O command to activate or deactivate the JES3 DLOG:

```
*MODIFY,O,DLOG={ON|OFF}
```

The JES3 *INQUIRY,O,DLOG command displays the status of the DLOG.

Defaults for the SDSF hardcopy log panels

The default hardcopy log panel can be set with the SET LOG command on any SDSF panel. The command specifies the panel that is displayed when you enter the LOG command with no parameters. Instead of the SET LOG command, you can use SDSF panel action bar *Options* choice 18 Set Log Default for setting the log panel default. The format of the SET LOG command is shown in Figure 18-149.

```
SET LOG {OPERACT|OPERLOG|SYSLOG|?}

   SET LOG with no parameters is the same as SET LOG OPERACT.
   OPERACT or A specifies that the OPERLOG panel is displayed if the Operlog
   component is active on the system you are logged on to; otherwise, the
   SYSLOG panel is displayed.
   OPERLOG or O specifies that the OPERLOG panel is displayed.
   SYSLOG or S specifies that the SYSLOG panel is displayed.
   ? displays the current setting for SET LOG command.
```

Figure 18-149 SET LOG command syntax

Commands to search data on panels

Find and scroll to specified characters:

- FIND (string) (parameters) command - Searches all data on the Log, ULOG, and Output Data Set panels, and the fixed (first) field on the tabular panels.
- FINDLIM **c**ommand - Resets the maximum number of lines searched by the FIND command on the Log, ULOG, and Output Data Set panels. You must be authorized to use this command.
- {LOCATE | LOC | L} [line-number | time | time-date | column] command - Scrolls a panel to a specific line or column.

The FIND command syntax is shown in Figure 18-150.

```
FIND (string)    (start-col) (end-col) (PREV)  (CHARS)
F    (*)                               (NEXT)  (WORD)
     (X'string')                       (FIRST) (PREFIX)
                                       (LAST)  (SUFFIX)
                                       (ALL)
   string is the string of characters to be searched for.
   * uses the string entered with the previous FIND command.
   X'string' specifies a string of hexadecimal characters.
   start-col starts the search in the specified column.  If used without
   end-col, the string must begin there.
   end-col ends the search in the specified column.
   PREV searches backward.
   NEXT searches forward.
   FIRST starts at the beginning of the data.
   LAST starts at the end of the data.
   ALL starts at the beginning, scrolls to the first occurrence, and indicates
   the number of occurrences.
   CHARS (or CHAR) indicates a character string.  It is the default.
   WORD indicates the string is preceded and followed by a non-alphanumeric
   character.
   PREFIX (or PRE) indicates the string is preceded by a nonalphanumeric
   character and followed by an alphanumeric character.
   SUFFIX (or SUF) indicates the string is preceded by an alphanumeric
   character and followed by a nonalphanumeric character.

   Note: X'string', WORD, PREFIX, and SUFFIX are valid only on the Log, and
   Output Data Set panels. FIRST, LAST, and ALL are not limited by FINDLIM.
```

Figure 18-150 The FIND command syntax

The syntax of the FINDLIM command is shown in Figure 18-151.

```
   FINDLIM {number|?}

   number is any number between 1000 and 9999999.
   ? displays the current value on the command line or pop-up.

   Under ISPF, this command remains in effect across SDSF sessions.
```

Figure 18-151 The FINDLIM command syntax

Note: The SDSF panel action bar Options pull-down choice 16 “Operlog limit for filter...“ can be used to limit the amount of OPERLOG data SDSF will search for records that meet filter criteria.

Figure 18-152 shows the LOCATE command syntax.

```
LOCATE | LOC | L line-number | time | time-date | column

   line-number is up to 8 digits (Log and Output Data Set panels) time is a
   time of day in the form hh:mm:ss or hh.mm.ss (Log panels)
   time-date is the time and date in the current format (Log panels)
   column is the title of the column to be located (tabular panels). The panel
   is scrolled horizontally so that the column is the first column after the
   fixed field.
```

Figure 18-152 The LOCATE command syntax

OPERLOG (LOG O) panel

The SDSF OPERLOG panel displays the MVS hardcopy message set of the MVS operations log. The OPERLOG panel is accessed with the LOG O command.

The first time you access the OPERLOG panel in a session, SDSF positions the data to show the most recent OPERLOG entries. If you exit the panel and then re-access it, you must scroll to the bottom to see the most recent entries.

```
   Display  Filter  View  Print  Options  Help

------------------------------------------------------------------------
---
 SDSF OPERLOG  DATE 04/05/2011     1 WTOR    1 FILTER          COLUMNS 02-
81
 COMMAND INPUT ===>                                            SCROLL ===>
HALF
---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+--
--8-
D                                          062 00000090  IRRH204E The
RACF_SENS
E                                          062 00000090  more potential
errors
N 0000000 SC75     2011095 15:11:00.02 JES3     00000090  IAT6395 00002
REQUEST(
M 0080000 SC74     2011095 15:14:56.16          00000090  IRR812I PROFILE **
(G)
E                                          063 00000090          TO START
BPXAS
N 0200000 SC74     2011095 15:14:56.18 STC04095 00000090  $HASP100 BPXAS
ON S
N 4000000 SC74     2011095 15:14:56.25 STC04095 00000090  $HASP373 BPXAS
STAR
N 0000000 SC74     2011095 15:14:56.26 STC04095 00000290  BPXP024I BPXAS
INITIAT
S                                                       0041
N 0000000 SC75     2011095 15:21:00.02 JES3     00000090  IAT6395 00002
REQUEST(
NC0000000 SC75    2011095 15:24:18.87 NEWNAME  00000290  *I O DLOG
NR0000000 SC75     2011095 15:24:18.88 JES3     00000090  IAT8617 DLOG
STATUS -
******************************** BOTTOM OF DATA
********************************
```

Figure 18-153 OPERLOG (LOG O) panel

Figure 18-153 shows an OPERLOG panel display. The COLS command has been used to display the formatted line for identifying display columns. The RESET command resets the results of a previous COLS command. The ACTION OFF command is also in effect; no WTOR messages are displayed at the bottom of the panel.

The format of the lines on the OPERLOG panel is shown in Figure 18-154 on page 957. The MVS mapping macro for hardcopy log format is IHAHCLOG in the SYS1.MODGEN library.

```
NR0000000 SYS1   2001363 09:46:34.34 HZSPROC  00000090  HZS0002E CHECK(...
 | |       |        |     |           |        |         |
 | First   |        |     Time        |        User exit |
 | 28      |        Julian            |        flags     |
 | routing |        date              |                  |
 | codes   |                          Console name,      Message text,    |
 Record    System name                jobname, or        beginning with
 and Request type                     multi-line ID      message ID
```

Figure 18-154 OPERLOG panel format

Messages are displayed on the OPERLOG panel with the color and highlighting that were assigned to them when they were issued. You can customize the colors with the SET SCREEN command.

Useful commands for the OPERLOG panel

A few useful commands are shown. Some require authorization.

- (W) | (I)/[command] - Issue a system command
- ACTION routing-code-list - Specify WTORs displayed
- FILTER column oper value - Filter log records
- FIND string - Search for a character string
- LOCATE hh:mm:ss mm/dd/yyyy - Locate a time and date
- NEXT / PREV number H | D | M | S - Scroll by hours, days, minutes, seconds
- PRINT begin-time begin-date end-time end-date - Print data
- RSYS system - Limit WTORs by system
- SET SCREEN - Set colors

The / command is used to issue a system command. The syntax is shown in Figure 18-155.

```
(W)|(I)/[command] Issue a system command
   / with no command displays the system command extension pop-up, primed with
   the text of the previous slash command.
   W waits the full delay interval before displaying messages. The delay
   interval is specified with the SET DELAY command.
   I uses console ID 0 (INTERNAL) to issue the command. command is any MVS or
   JES command.
   + displays the system command extension pop-up. The system command extension
   pop-up lets you enter longer commands and select commands from a list of
   recently issued commands.
```

Figure 18-155 {W} | {I} / command syntax

The ACTION command is used to specify which WTORs to display. The syntax is shown in Figure 18-156.

```
ACTION routing-code-list Specify WTORs displayed
   ACTION {routing-code-list|MVS|USER|ALL|OFF|?}
   Specifies which Write-To-Operator-with-Reply (WTOR) messages are displayed
   at the bottom of the Log panel. You must be authorized to use this command.
```

Figure 18-156 ACTION command syntax

The FILTER command is used to filter log records. The format is shown in Figure 18-157 on page 958.

```
FILTER column oper value Filter log records
   FILTER ON | OFF | OR | AND
   FIL    (+|-)column (operator) value
   Filters data on the current SDSF panel. Under ISPF, filters are saved (one
   set for each JES type).
   OFF turns filtering off but retains filter criteria.
   ON turns filtering on.
   OR and AND specify the relationship between filters both within a column and
   between columns.
   (+|-)column names a column for filtering and turns filtering on. column can
   be abbreviated to the shortest unique name.
   + adds the filter to any previous filters. There is a limit of 25 filters
   under ISPF
   - discards all filters for the column.
```

Figure 18-157 FILTER command syntax

The FIND command is used to search for a character string. The syntax is shown in Figure 18-150 on page 954.

The LOCATE command is used to locate a time and date in the log. The syntax is shown in Figure 18-152 on page 955.

The NEXT and PREV commands control scrolling options. The syntax is shown in Figure 18-158.

```
NEXT|N (number) (D|H|M|S)
   number is the number of days, hours, minutes, or seconds (1-99) to scroll
   forward.  The default is 1 H.
   D|H|M|S indicate the unit for number is days, hours, minutes, or seconds
   respectively. Hours is the default.
```

Figure 18-158 NEXT / PREV command syntax

The RSYS command is used to limit WTORs by the system. The format is shown in Figure 18-159.

```
RSYS (system-name | ?)
RSYS with no parameters displays only WTORS from the system you are logged on to.
system-name is the MVS system name, up to 8 characters, including * (any string of
characters) or % (any single character).
? displays the current setting on the command line or pop-up.
```

Figure 18-159 RSYS command syntax

The PRINT command is used to print data. The format is shown in Figure 18-160 on page 959.

```
RINT  (first-line last-line)
PRT   (begin-time begin-date end-time end-date)
PT    (SCREEN)
   PRINT with no parameters opens a default SYSOUT data set if the print data
   set is not already open. On the output data set panel, it also prints the
   entire data set.
   first-line is the first line to be printed, in the current data set. To
   specify a quantity rather than a range of lines, use * for the first line.
   You must use * with the operlog.
   last-line is the last line to be printed. If * is used for first-line this
   is a quantity of lines.
   begin-time is the beginning time for a range of lines on the log, in the
   form hh:mm:ss or hh.mm.ss.
   begin-date is the beginning date for a range of lines on the log, in the
   current date format. The default is the date of the current top line.
   end-time is the ending time for the range of lines.
   end-date is the ending date for the range of lines. The default is the date
   of the current top line.
   SCREEN prints the screen image. (Under ISPF, use the ISPF PRINTL command.)
```

Figure 18-160 PRINT command syntax

The SET SCREEN command displays a panel that allows you to set the colors, highlighting and intensities used on SDSF panels, and control display of the action bar. It is valid only if SDSF was accessed through ISPF. The values are saved across SDSF sessions.

Columns for filtering on the OPERLOG panel

The OPERLOG panel columns to use with the FILTER command are shown in Figure 18-161.

```
   SYSNAME    MVS system name.
   DATE       Character format date of the message. Enter the date as it is displayed.
   TIME       Character format time of the message. Enter the time as it is displayed.
   DATETIME   Date and time the message was logged, in date/time format. This column
              accepts the date format set with SET DATE. Use operators with > or <.
   JOBNAME    Originating job name.
   JOBID      Job ID. OPERLOG in JES3 system does not show job ID.
   CONSOLE    Console name.
   MSGID      Message ID (first 8-character token of the message text).
   MSGTEXT    Message text (includes message ID). Note that because this column includes
              the message ID, you may want to include a leading generic pattern matching
              character in your filter command, for example, FIL MSGTEXT EQ *STARTED*.
```

Figure 18-161 OPERLOG panel filter fields

You can use the FILTER function to define up to 25 filters with boolean operators. The filter criteria are column, operator and value, and can include pattern matching.

When entering multiple filters, you can specify AND or OR to define the relationship between filters.

When using SDSF interactively under ISPF, type `FILTER ?` to display the Filter pop-up, then type values on the pop-up or select from lists of valid values.

The usage of the FILTER command is explained in Figure 18-157 on page 958. An SDSF panel's action bar Filter pop-up choice '1. Filter' guides you to set up a complex filter.

SYSLOG as hardcopy

The MVS hardcopy log records command and message traffic on the systems. The three forms of the hardcopy log are:

- OPERLOG centrally records command and message traffic for systems in a sysplex in Message Data Block (MDB) format. OPERLOG is controlled by the MVS VARY OPERLOG,HARDCPY command.
- SYSLOG individually records command and message traffic for each system in MVS format. SYSLOG is controlled by the MVS VARY SYSLOG,HARDCPY command.
- JES3 DLOG centrally records command and message traffic for systems in a JES3 complex in the JES3 format. The JES3 DLOG is written to SYSLOG on the global processor. The JES3 DLOG is controlled using the *MODIFY O,DLOG=ON|OFF command. SYSLOG on the global processor must be active when DLOG is active.

At least one hardcopy log must be active on each MVS system.

Users can access the SYSLOGs of all individual systems in the JES complex as a single entity. Instead of searching multiple SYSLOG data sets, viewing and searching is performed on a single, logical SYSLOG data set.

> **Note:** JES3 DLOG activates an extended MCS console to receive messages from the sysplex systems that are defined to belong to the JES3 complex. The DLOG processing, on the JES3 global, extracts the messages from the data space, formats them in JES3 DLOG format, and writes them to SYSLOG using a WTL macro service.
>
> The SYSLOG on the global may contain messages from JESPLEX systems that are IPLed, but do not have an active JES3 primary subsystem.

SYSLOG (LOG S) panel

The SYSLOG panel displays the system log, which is a collection of JES3 data sets that contain console messages, operator commands, and operator responses for a z/OS system. It is accessed with the LOG S command.

The first time you access the SYSLOG panel in a session, SDSF positions the data to show the most recent SYSLOG entries. If you exit the panel and then re-access it, you must scroll to the bottom to see the most recent entries.

> **Note:** SYSLOG panel data filtering is not available. FIND and LOCATE (with some limitations) for scrolling are available.

Figure 18-162 on page 961 is an example of the MVS SYSLOG and JES3 DLOG data display formats of the messages issued for the MVS S DEALLOC command.

```
  Display  Filter  View  Print  Options  Search  Help
 ------------------------------------------------------------------------------
 SDSF SYSLOG  20313.073 SC75      03/12/2011 0W         897     COLUMNS 02- 133
 COMMAND INPUT ===>                                            SCROLL ===> HALF
---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8-
NC0000000 SC75     2011072 11:32:20.70 VAINI    00000290  S DEALLOC
M 0080000 SC75     2011072 11:32:20.72          00000090  IRR812I PROFILE ** (G)
E                                          317 00000090          TO START DEALL
N 0000000 SC75     2011072 11:32:20.73 JES3     00000090  IAT6100 ( DEMSEL ) JOB
N 0020000 SC75     2011072 11:32:20.75 DEALLOC  00000290  IEF695I START DEALLOC
S                                                         IBMUSER , GROUP SYS1
N 0000000 SC75     2011072 11:32:20.76          00000290  IEA989I SLIP TRAP ID=X
NC0000000 SC75     2011072 11:33:06.24 VAINI    00000290  *F O DLOG=ON
::::::::: ::::
JES         2011072 1133063  IAT7114 DLOG INITIALIZATION SUCCESSFUL
    VAINI   2011072 1133063  IAT8020 DLOG FACILITY ENABLED
    VAINI   2011072 1133225 -S DEALLOC
SEC         2011072 1133225  SC75= DEALLOC  IRR812I PROFILE ** (G) IN THE STARTE
SEC         2011072 1133225  SC75= DEALLOC          TO START DEALLOC WITH JOBNAM
LOG         2011072 1133225  IAT6100 ( DEMSEL ) JOB DEALLOC  (JOB20676), PRTY=15
MLG         2011072 1133225  SC75= DEALLOC  IEF695I START DEALLOC  WITH JOBNAME
MLG         2011072 1133225  SC75= DEALLOC   GROUP SYS1
MLG         2011072 1133226  SC75=          IEA989I SLIP TRAP ID=X33E MATCHED.
```

Figure 18-162 SYSLOG (LOG S) panel (MVS SYSLOG and JES3 DLOG data on display)

The MVS format of data description of the SYSLOG panel is shown in Figure 18-163.

```
  ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+-
  ---8-
   M 0080000 SC75     2011072 11:31:57.60          00000090  IRR812I PROFILE
  ** (G)
   | |       |        |       |           |        |         |
   | First 28|        |       Time        |        User exit |
   | routing |        Julian date         |        flags     |
   | codes   |                            Console name,      Message text,
   Record    System name                  jobname, or        beginning
  with
   and Request type                       multi-line ID      message ID
```

Figure 18-163 Description of the MVS format of a SYSLOG message

The JES3 DLOG format of data on the SYSLOG panel is shown in Figure 18-164.

```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+-
---8-
 SEC         2011072 1133225  SC75= DEALLOC  IRR812I PROFILE ** (G) IN THE
STARTE
  |    |     |       |        |     |        |       |
  |    |     Julian  Time     |     Job name |       Message
text
  |    |     date             System         Message ID
  |    Console name           name                        |
 JES3 Destination Class
```

Figure 18-164 Description of the JES3 DLOG format of a SYSLOG message

The MVS format of data on the SYSLOG panel is the same as on the OPERLOG panel.

Useful commands for the SYSLOG panel

A few useful commands are shown here:

- (W)|(I)/[command] - Issue a system command - Figure 18-155 on page 957
- ACTION routing-code-list - Specify WTORs displayed - Figure 18-156 on page 957
- FIND string - Search for a character string - Figure 18-150 on page 954
- LOCATE hh:mm:ss mm/dd/yyyy - Locate a time and date - Figure 18-152 on page 955
- PRINT begin-time begin-date end-time end-date - Print data - Figure 18-160 on page 959
- RSYS system - Limit WTORs by system - Figure 18-159 on page 958
- SYSID system-id - Specify the system ID - Figure 18-165

```
SYSID (system-id)
      (*)
      (?)
   with no parameters indicates the SYSLOG panel displays the SYSLOG for the
   all systems in the JESPlex.
   system-id is a system name 1-8 characters.
   * specifies that the JES3 global system is to be used.
   ? displays the current SYSID setting on the command line, as well as a list
   of the systems defined in the JESPLEX beginning on the message line. The
   member the user is logged on to is shown in parentheses.
```

Figure 18-165 SYSID command syntax

18.29 Working with MVS

In this section the following topics are discussed:

- The Display Active Users (DA) panel, which allows users to view and control information about active jobs, TSO users, started tasks, and initiators. It also shows system data, such as processor usage and paging information.

  In a JES3 environment, the DA panel requires Resource Measurement Facility™ (RMF) to be active.

- The System Requests (SR) panel, which displays information about reply and action messages.
- The Scheduling Environment (SE) panel, which displays the Scheduling Environments in the sysplex.
- The Resource (RES) panel, which displays Workload Management (WLM) resources.
- The Enclaves (ENC) panel, which displays information about WLM enclaves.
- The Processes (PS) panel, which displays information about z/OS UNIX System Services processes.
- The Health Checker (CK) panel, which displays information from IBM Health Checker for z/OS. The panel shows the active checks. Checks that are currently running are highlighted.

Display active users (DA) panel

The Display Active Users (DA) panel displays information about active jobs, TSO users, started tasks, and initiators in the sysplex. It also shows some system data, such as processor usage and paging information.

The DA panel in a JES3 environment requires RMF Monitor I to be started. By default, Monitor I is started when RMF is started.

The DA panel is invoked with the DA command, shown in Figure 18-166.

```
 Position:   Type:   Only:     No:        All:
DA (IN   )   (JOB )  (OJOB  )  (NOJOB  )  (ALL )
   (OUT  )   (TSU )  (OTSU  )  (NOTSU  )  (ALLT)
   (TRANS)   (STC )  (OSTC  )  (NOSTC  )  (ALLP)
   (READY)   (INIT)  (OINIT )  (NOINIT )
                     (OIN   )  (NOIN   )
                     (OOUT  )  (NOOUT  )
                     (OTRANS)  (NOTRANS)
                     (OREADY)  (NOREADY)
"Position" and "Type" parameters include address spaces.
   Only parameters limit the display to those types or positions. Use only one
   parameter from this column.
   No parameters exclude those types or positions.
   All parameters show all address spaces, or all types or positions. They
   cannot be used with other parameters.
The maximum number of parameters is four.
The information displayed may also be limited by your authorization, and by settings for
filters such as FILTER, PREFIX, and SYSNAME.
When parameters conflict, the last one is used.
```

Figure 18-166 DA command syntax

Figure 18-167 is a sample active panel display invoked with a DA JOB NOSTC command.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF DA SC75     (ALL)    PAG  0  CPU/L/Z   5/  1/  0  LINE 1-8 (13)
COMMAND INPUT ===>                                            SCROLL ===>
HALF
PREFIX=%*  DEST=*  OWNER=*  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,?-JDS,A-Release,C-Cancel,CA-CancelARM,
ACTION=CD-CancelDump,CDA-CancelARMDump,CP-CancelPrint,D-Display,
ACTION=DE-DisplayEstimates,DL-DisplayLong,DSD-DisplayDDDnames,
ACTION=DSH-DisplaySpoolHold,DSP-DisplaySpoolPartition,DX-DisplayExtended,
ACTION=E-Restart,H-Hold,K-SysCancel,KD-SysCancelDump,L-List,LB-ListBDT,
ACTION=LH-ListHold,LT-ListTCP,P-Purge,Q-OutDesc,R-Reset,RQ-ResetQuiesce,
ACTION=S-Browse,SB-ISPFBrowse,SE-ISPFEdit,SJ-JCLEdit,W-Spin,X-Print,
ACTION=XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,XF-PrintFile,
ACTION=XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose,Y-SysStop,
ACTION=Z-SysForce
NP   JOBNAME     SIO   CPU% ASID ASIDX  EXCP-Cnt   CPU-Time SR Status
SysName
     VAINI      0.00   0.00   76 004C       2947       2.51 TI        SC74
     INITS      0.00   0.00   31 001F          2       0.00 DW
     EATING     0.00   0.00   61 003D          2       0.00 DW        SC75
     JUST       0.00   0.00   62 003E          2       0.00 DW        SC75
     SOME       0.00   0.00   63 003F          2       0.00 DW        SC75
     CLASSA     0.00   0.00   64 0040          2       0.00 DW        SC75
     JOBS       0.00   0.00   66 0042          2       0.00 DW        SC75
```

Figure 18-167 Display active users panel

Note: The DA panel shows information about jobs, TSO users, started tasks, and initiators that are active in the JESPLEX even if some of the systems are not running JES3 as the primary job entry subsystem.

The title line description of the display active users panel is in Figure 18-168.

```
SDSF DA SC75    (ALL)    PAG  0  CPU/L/Z   1/  1/  0  LINE 1-25 (132)
        |       |        |       |                     |          |
  System ID     |        |       |                 Lines displayed |
  of system     |  Total demand  |                 or first line   |
  you are       |  paging rate   |                 if 100,000      |
  logged        |               Percentage of time                 |
  on to         |               the CPU is busy,                   |
           Systems displayed    MVS, LPAR and zAAP                 |
           (MVS value or        views                  Total lines
           SYSNAME value)
SIO, if shown, is the total system start I/O rate. PAG, SIO, and CPU values
are for the system you are logged on to.
```

Figure 18-168 Display active users panel title line

Display active users panel fields

The display active users panel includes some or all of the fields in Figure 18-169 on page 965 and. (The order and titles may be different, depending upon installation and user options.)

Title	Description
JOBNAME	Job name of the address space
StepName	Job step name, or TSO procedure name for TSO users
ProcStep	Procedure step name, or terminal name for TSO users
Type	Type of address space: job, started task, TSO user, or initiator
JNum	JES job number. Not included in the default field list.
Owner	User ID of job creator
C	JES input class at the time the job was selected for execution
Pos	Address space position; swapped in, swapped out, nonswappable, in transition
DP	Address space dispatching priority in hexadecimal
Real	Current utilization of real storage in frames
Paging	Demand paging rate (only present if the address space was swapped in for the entire interval)
SIO	Address space's EXCP rate in EXCPs per second
CPU%	Percent of CPU time used on behalf of this address space during the most recent interval measured
ASID	Address space identifier
ASIDX	Address space identifier in hexadecimal
EXCP-Cnt	Address space's EXCP count for the current job step. Uses hexadecimal scaling.
CPU-Time	Accumulated CPU time (TCB plus SRB) consumed on behalf of the address space, for the current job step, in seconds
SR	Swap out reason code
JobID	JES job ID or work ID
Status	PROT (job is protected)
Workload	Workload name
SrvClass	Service class name
SP	Service class period
ResGroup	Resource group name
Server	Server indicator, indicates if resource goals are being honored
Quiesce	Quiesce indicator (address space is quiesced)
SysName	System on which the address space is running
SPag	Demand paging rate for the system (see note)
SCPU%	System CPU utilization for the system that is processing the job (see note)
ECPU-Time	Accumulated CPU time consumed within the address space, for the current job step, in seconds
ECPU%	CPU usage consumed within the address space
CPUCrit	Current address space CPU protection
StorCrit	Current address space storage protection
RptClass	Report class
MemLimit	Memory limit
Tran-Act	Elapsed time the transaction has been active
Tran-Res	Elapsed time the transaction was swapped in
Spin	Indicator of whether jobs in the job class can be spun (
Seclabel	Security label
GCP-Time	Accumulated general processor service time, in seconds
zAAP-Time	Accumulated zAAP service time, in seconds
zACP-Time	Accumulated general processor service time that was eligible for a zAAP, in seconds
GCP-Use%	Percent of the total general processor time used by the address space in the most recent interval (not normalized

Figure 18-169 Display active users panel fields 1 of 2

Title	Description
zAAP-Use%	Percent of the total zAAP time used by the address space in the most recent interval (not normalized)
SzAAP%	zAAP view of CPU use for the system, in the most recent interval. The same for all rows for a system.
SzIIP%	zIIP view of CPU use for the system, in the most recent interval. The same for all rows for a system.
Promoted	Promoted due to a chronic resource contention
zAAP-NTime	Normalized zAAP service time, in seconds
zIIP-Time	Accumulated zIIP service time, in seconds
zICP-Time	Accumulated general processor service time that was eligible for a zIIP, in seconds
zIIP-NTime	Normalized zIIP service time, in seconds
zIIP-Use%	Percent of the total zIIP time used by the address space in the most recent interval (not normalized)
SLCPU%	Percentage of time the LPAR is busy for the system, in the most recent interval. The value for SLCPU% is the same for all rows for a system.

Figure 18-170 Display active users fields 2 of 2

Display active users panel NP field actions

Figure 18-171 and Figure 18-172 on page 967 show the action characters that can be entered in the NP column on the display active users panel.

NP action	JES3 command / SDSF description
A-Release	*F J=jobno,R - Release a held job.
C-Cancel	*F J=jobno,CO - Cancel a job, also print non-held data sets.
CA-CancelARM	*F J=jobno,C,ARMR - Cancel a job that is defined to Automatic Restart Manager (ARM).
CD-CancelDump	*F J=jobno,C,D - Cancel a job and take a dump.
CDA-CancelARMDump	*F J=jobno,C,D,ARMR - Cancel a job that is defined to ARM, and take a dump.
CP-CancelPrint	*F J=jobno,CP - Cancel a job and delete held data sets.
D-Display	*I J=jobno - Display job information in the log.
DE-DisplayEstimates	*I J=jobno,E - Display line, page, record and card counts
DL-DisplayLong	*I A,J=jobno - Display long form
DSD-DisplayDDDnames	*I J=jobno,SD - Display DDNAMES of spool data sets that contain data
DSH-DisplaySpoolHold	*I J=jobno,SH - Display DDNAMES of spool data sets in spool hold that contain data
DSP-DisplaySpoolPartition	*I J=jobno,SP - Display Spool partition name
DX-DisplayExtended	*I J=jobno,X - Display extended
E-Restart	*R main,jobno - Process a job again.
H-Hold	*F J=jobno,H - Hold a job.
K-SysCancel	C jobname.identifier,A=asidx - Cancel a started task (MVS cancel).
KD-SysCancelDump	C jobname.identifier,DUMP,A=asidx - Cancel a started task and take a dump (MVS cancel)

Figure 18-171 Display active users panel NP field actions 1 of 2

```
NP action                JES3 command / SDSF description
L-List                   *I U,Q=WTR,J=jobno - List WTR queue output status of a job
LB-ListBDT               *I U,Q=BDT,J=jobno - ListSNA/NJE outputstatus of a job
LH-ListHold              *I U,Q=HOLD,J=jobno - List HOLD queue output status of a job
LT-ListTCP               *I U,Q=TCP,J=jobno - List TCP/IP queue output status of a
                         job
P-Purge                  *F J=jobno,C - Cancel a job and purge its output.
R-Reset                  RESET jobname,A=asidx - Reset and resume a job.
RQ-ResetQuiesce          RESET jobname,Q,A=asidx - Reset and quiesce a job.
Q-OutDesc                Display output descriptors for all of the data sets.
S-Browse                 Display the data sets for a job. You can add:
SB-ISPFBrowse            Use ISPF Browse
SE-ISPFEdit               Use ISPF Edit
SJ-JCLEdit                Use ISPF Edit to edit the JCL
W-Spin                    *F J=jobno,SPIN - Cause job and message logs to spin.
X-Print                  Print output data sets. You can add:
XC                       C - Close the print file after printing (XC)
XD or XDC                D - Display the Open Print Data Set panel (XD or XDC)
XF or XFC                F - Display the Open Print File panel (XF or XFC)
XS or XSC                S - Display the Open Print panel (XS or XSC)
Y-SysStop                P jobname.identifier,A=asidx - Stop a started task (MVS
                         stop).
Z-SysForce               FORCE jobname.identifier,A=asidx - Cancel a started task
                         (MVS force).
?-JDS                    Display a list of data sets for a job. (Access the Job Data
                         Set panel.)
```

Figure 18-172 Display active users panel NP field actions 2 of 2

Display active users panel overtypable fields

Display active users panel overtypable fields are listed in Figure 18-173.

```
Field       MVS command / SDSF description
SrvClass    E jobname[,A=asid],SRVCLASS=classname - Service class name
Quiesce     E jobname[,A=asid],Q - Quiesce indicator (QUIESCE or RESUME)
```

Figure 18-173 Display active users panel overtypable fields

18.30 System Requests (SR) panel

The system requests panel displays information about operator reply and action messages. The system requests panel is invoked with the SR command. Figure 18-174 on page 968 shows the SR command syntax.

```
SR ALL | {ACTIONS|A} | CEM | EM | IM | {MOUNTS|M} | {REPLIES|R|RM})
   ALL displays all reply and action messages. This is the default.
   ACTIONS or A displays action messages.
   CEM displays critical eventual action messages.
   EM displays eventual action messages.
   IM displays immediate action messages.
   MOUNTS or M displays DASD and tape mount messages. SDSF considers a message
   to be a mount if it has tape or DASD pool routing codes.
   REPLIES or R or RM displays reply messages.
```

Figure 18-174 SR command syntax

If the MVS action message retention facility (AMRF) is not active, the SR panel shows only reply messages. The AMRF parameter in the CONSOLxx PARMLIB member INIT statement specifies whether AMRF is to be active.

You change the status of AMRF with the CONTROL M,AMRF={Y|N} command. To learn the status of AMRF, issue the CONTROL M,REF command.

Figure 18-175 is an example of the SDSF system requests panel.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF SYSTEM REQUESTS  RM 1    IM 11   CEM 6    EM 11   LINE 14-29 (29)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,AI-AutoReplyIgnore,C-Remove,D-Display,R-Re
ply
NP      REPLYID    SysName  JobName  Message-Text
        33019      SC74     HZSPROC   HZS0002E
CHECK(IBMCS,CSTCP_CINET_PORTRNG_
        34019               DB8YMSTR  DSNY020I DSNYASCP -DB8Y MEASURED USAGE
DA
        80018      SC75     SMF      *IEE986E SMF HAS USED   100% OF
AVAILABLE
        64019      SC74     SMF      *IEE986E SMF HAS USED   100% OF
AVAILABLE
        81018      SC75     SMF      *IEE979W SMF DATA LOST - NO BUFFER
SPACE A
        65019      SC74     SMF      *IEE979W SMF DATA LOST - NO BUFFER
SPACE A
        179        SC75     VAINIAR  @179 REPLY SOMETHING
        624018     SC75     HZSPROC   HZS0002E
CHECK(IBMRACF,RACF_TEMPDSN_ACTIV
        626018     SC75     HZSPROC   HZS0002E CHECK(IBMDAE,DAE_SHAREDSN):
        622018     SC75     HZSPROC   HZS0002E
CHECK(IBMRACF,RACF_TAPEVOL_ACTIV
        623018     SC75     HZSPROC   HZS0002E
CHECK(IBMRACF,RACF_IBMUSER_REVOK
        494019     SC74     HZSPROC   HZS0002E
CHECK(IBMRACF,RACF_TAPEVOL_ACTIV
        727018     SC75     HZSPROC  *HZS0003E
CHECK(IBMRACF,RACF_SENSITIVE_RES
```

Figure 18-175 System requests panel

System requests panel fields

The title line description of the system requests panel is in Figure 18-176.

```
SDSF SYSTEM REQUESTS  RM 4    IM 3    CEM 8    EM 24   LINE 1-10 (10)
                      |                                  |     |
 Counts of reply, immediate action, critical             Lines   Total
 eventual action and eventual action messages          displayed lines
```

Figure 18-176 System requests panel title line

The system requests panel includes some or all of the following fields listed in Figure 18-177 on page 970.

Title	Description
REPLYID	Reply ID of the message
SysName	Originating system name
JobName	Name of the issuing job
Message-Text	Message ID and text
JobID	JES job ID of the issuing job (JES2) or Proc name or job name (JES3)
Date	Date when the message was logged
Time	Time when the message was logged
Console	Target console
RouteCd	First 28 routing codes, in hexadecimal
Desc	Descriptor codes, in hexadecimal
Type	Message type
Queue	Queue the message is on (CEM - critical eventual action, EM - eventual action, IM - immediate, RM - reply)
AutoReply	Automatic reply indicator
AutoRDelay	Message delay time until the automatic reply is done, in seconds
AutoReplyTime	Date and time when auto reply will be done
AutoReplyText	Automatic reply text

Figure 18-177 System request panel fields

System requests panel NP field actions

Figure 18-178 shows the action characters that can be entered in the NP column on the system request panel by authorized users.

NP action	MVS command / SDSF description
AI-AutoReplyIgnore	SETAUTOR IGNORE - Ignore auto reply for the message.
C-Remove	K C,A,id - Remove an action message.
D-Display	D R,CN=(ALL),MSG= - Display a message in the logs or ULOG.
R[command]	R id,reply - Reply to the message. R by itself displays a pop-up on which you can complete the command.

Figure 18-178 System requests panel NP field actions

18.31 Scheduling environment (SE) panel

The SDSF scheduling environment panel displays the scheduling environments in the sysplex.

A scheduling environment is a list of abstract resource names along with their required states. If an MVS image satisfies all of the requirements in the scheduling environment associated with a given unit of work, then that unit of work can be assigned to that MVS image. If any of the requirements are not satisfied, then that unit of work cannot be assigned to that MVS image.

For every resource name that is referenced by a scheduling environment, a corresponding resource state must be set on each system in the sysplex. The resource state can be:

- ON, which will satisfy a resource state requirement of ON.
- OFF, which will satisfy a resource state requirement of OFF.
- RESET, which will not satisfy any resource state requirement.

  Resources are put into the RESET state when:

  – A system is IPLed

– A policy is activated that defines a resource name that did not exist in the previously active policy

A scheduling environment is dynamic. It identifies the dependency that a job has to run on particular systems without specifically naming the systems. Since a scheduling environment can change state, the systems where a job is eligible to run can change without modification to its JCL.

The JES3 //*MAIN JECL SYSTEM parameter is specific and static, since it lists system names. You can use scheduling environments and the SYSTEM parameter together.

Figure 18-179 displays a scheduling environment panel. The SE panel is invoked with the SE primary command (Figure 18-180). You must be authorized to use the command.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF SCHEDULING ENVIRONMENT DISPLAY ALL SYSTEMS       LINE 1-4 (4)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
ACTION=//-Block,=-Repeat,+-Extend,D-Display,R-Resource,ST-Status
NP   SCHEDULING-ENV   Description                     Systems
     JES2             MAS                             SC74
     JES3             JESPLEX                         SC75
     NAV              Not available
     PLEX75            SYSPLEX                         SC74,SC75
```

Figure 18-179 Scheduling environment panel

The SE panel displays the same data that is returned by the D WLM,SCHENV=*,SYSTEMS command IWM036I response message.

```
SE {MAS|ALL}
   ALL displays scheduling environments for all systems in the sysplex. This is
   the default for JES3. MAS under JES3 is treated as ALL.
```

Figure 18-180 SE command syntax

Scheduling environment panel fields

The SE panel includes the fields listed in Figure 18-181.

Title	Description
SCHEDULING-ENV	Scheduling environment name
Description	Description of the scheduling environment
Systems	Systems with the scheduling environment available

Figure 18-181 Scheduling environment panel fields

Scheduling environment panel NP field actions

Figure 18-182 on page 972 shows the action characters that can be entered in the NP column on the scheduling environment panel by authorized users.

```
NP action    MVS command / SDSF description
DD-Display   D WLM,SCHENV=JES2,SYSTEMS - Display scheduling environments in the log.
             This issues the MVS D command.
R-Resource   Display resources for a scheduling environment. F WLM,RESOURCE= command
             used to change resource state.
ST-Status    Display the ST panel for all jobs requiring the scheduling environment.
```

Figure 18-182 Scheduling environment panel NP field actions

18.32 Resources (RES) panel

The resources panel displays WLM resources used in that scheduling environment. To display sysplex resources with SDSF, access the resource panel with the RES command. To display resources for a scheduling environment, access the panel with the R action character from the SE panel.

Resource, when used as part of a scheduling environment, is an abstract element that can represent an actual physical entity (such as a peripheral device), or an intangible quality (such as a certain time of day). A resource is listed in a scheduling environment along with a required state of ON or OFF. If the corresponding resource state on a given system matches the required state, then the requirement is satisfied for that resource.

Figure 18-183 shows an SDSF WLM resources panel display. The resources panel is invoked with the RES command. You must be authorized to use the command.

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
SDSF RESOURCE DISPLAY ALL SYSTEMS                      LINE 1-4 (4)
COMMAND INPUT ===>                                        SCROLL ===> HALF
ACTION=//-Block,=-Repeat,+-Extend,D-Display
NP   RESOURCE         SC74     SC75
     PLEX75           ON       ON
     SC70             ON       OFF
     SC74             ON       OFF
     SC75             OFF      ON
```

Figure 18-183 Resources panel

Figure 18-184 shows the RES command syntax.

```
RES {MAS|ALL}
   ALL displays WLM resources for all systems in the sysplex. This is the
   default for JES3. MAS under JES3 is treated as ALL.
```

Figure 18-184 RES command syntax

Resources panel fields

The resources panel includes the fields explained in Figure 18-185 on page 973.

```
Title       Description
RESOURCE    WLM resource name
ReqState    Required state of the resource as defined in WLM (displayed only when the
            panel is accessed with the R action character from the SE panel)
System      System in the Sysplex, showing the state of the resource on the system
```

Figure 18-185 Resources panel fields

Resources panel NP field action

Figure 18-186 shows the action characters that can be entered in the NP column on the resources panel by authorized users.

```
NP action   MVS command / SDSF description
D -Display  D WLM,RESOURCE=resource,SYSTEMS - Display resources in the Log.
```

Figure 18-186 Resources panel NP field action

Resource panel overtypable columns

Each column showing the state of the resource for that system, highlighted in Figure 18-183 on page 972, can be overtyped to change the state of the resource for that system. This issues an MVS F WLM,RESOURCE= command. SDSF appends an RO command if the MVS command is targeted for another system.

18.33 Enclaves (ENC) panel

The enclaves panel displays information about WLM enclaves. It is accessed with the ENC command.

An enclave is a transaction that can span multiple dispatchable units (SRBs and tasks) in one or more address spaces and is reported on and managed as a unit. A multisystem enclave can run in multiple address spaces spanning multiple systems within a sysplex. With all units of work of a job running in the same enclave, WLM can manage all of the work to a single performance goal.

SDSF displays multisystem enclaves on multiple rows. When you act against any of these rows, SDSF issues the WLM service against the original enclave.

Figure 18-188 on page 974 is an SDSF enclave display. It is invoked with the ENC command (Figure 18-187). You must be authorized to use this command.

```
ENC {ACTIVE|ALL}
   ACTIVE displays only active enclaves
   ALL displays all enclaves. This is the default.
```

Figure 18-187 ENC command syntax

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF ENCLAVE DISPLAY  (ALL)    ALL                    LINE 1-8 (8)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
PREFIX=%*  DEST=*  OWNER=*  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,I-Info,M-Match,R-Reset,RQ-ResetQuiesce
NP   NAME             SSType Status   SrvClass Per PGN RptClass ResGroup
CPU-
     2400000002       STC    INACTIVE SYSSTC     1
     2000000001       STC    INACTIVE SYSTEM     1
     2800000003       STC    INACTIVE SYSSTC     1
     2C00000004       TCP    INACTIVE SYSOTHER   1
     2400000002       STC    INACTIVE SYSSTC     1
     2000000001       STC    INACTIVE SYSTEM     1
     2800000003       STC    INACTIVE SYSSTC     1
     2C00000004       TCP    INACTIVE SYSOTHER   1
```

Figure 18-188 Enclave panel

The Enclave panel includes some or all of the following fields explained in Figure 18-189. (The order and titles may be different, depending upon installation and user options.)

Enclave panel fields

Title	Description
NAME	Enclave token
SSType	Subsystem type (for example, DB2, MQ)
Status	Status of the enclave
SrvClass	Service class
Per	Period number
PGN	Performance group
RptClass	Report class
ResGroup	Resource group
CPU-Time	Total CPU time
OwnerSys	Enclave owner system
OwnerJob	Enclave owner jobname
OwnerAS	Enclave owner ASID
OwnerASX	Enclave owner ASID in hexadecimal
Scope	Scope of the enclave, either LOCAL (single-system) or MULTISYS (the enclave has an export token and so is multisystem-capable)
Type	Enclave type: IND (independent) or DEP (dependent)
Original	For an enclave that has been exported, YES if this is the original enclave
Quiesce	Indicates if the enclave is in a quiesce delay, which occurs if the address space has been reset with the MVS RESET,QUIESCE command
Workload	Workload name
SysName	System that reported the data
SysLevel	Version and release
Subsys	Subsystem name
zAAP-Time	Accumulated zAAP time, in seconds
zACP-Time	Accumulated zAAP on CP time, in seconds
zIIP-Time	Accumulated zIIP time, in seconds
zICP-Time	Accumulated zIIP on CP time, in seconds
Promoted	Promoted due to a chronic resource contention
zAAP-NTime	Normalized zAAP time, in seconds
zIIP-NTime	Normalized zIIP time, in seconds

Figure 18-189 Enclave panel fields

Enclave panel NP field actions

Figure 18-190 shows the action characters that can be entered in the NP column on the enclaves panel by authorized users.

NP action	SDSF description
I-Info	Display additional information about the enclave.
M-Match	Match the enclave by export token, to display only the instances of a multisystem enclave. Valid only for multisystem enclaves, as indicated in the Scope column. To see all enclaves again, re-access the panel.
R-Reset	Reset and resume an enclave.
RQ-ResetQuiesce	Reset and quiesce an enclave.

Note: If you reset a dependent enclave, the owner address space is reset.

Figure 18-190 Enclaves panel NP field actions

Pop-up display for enclave I-Info action

Figure 18-191 shows the I action additional information pop-up display.

```
Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
SDSF ENCLAVE DISPLAY  (ALL)    ALL                    LINE 1-8 (8)
COMMAND INPUT ===>                                       SCROLL ===> HALF
PR
AC                 Enclave 2C00000004 on System SC75
NP
     Subsystem type    TCP          Plan name
     Subsystem name    TCPIP        Package name
     Priority                       Connection type
     Userid                         Collection name
     Transaction name  TCPENC01     Correlation
     Transaction class              Procedure name
     Netid                          Function name
I   TCPENC01
     Logical unit name              Performance group
     Subsys collection              Scheduling env
     Process name
     Reset             NO

     F1=Help  F12=Cancel
```

Figure 18-191 Enclave I action pop-up

I-Info action data on the additional information pop-up about an enclave is shown in Figure 18-192 on page 976. WLM uses this information to classify the enclave.

Subsystem type	Type of the subsystem
Subsystem name	Name of the subsystem
Priority	Priority associated with the subsystem
Userid	User ID associated with the request
Transaction name	Transaction program name for the request
Transaction class	Class name within the subsystem
Netid	Network identifier associated with the requester
Logical unit name	Local LU name associated with the requester
Subsys collection	Subsystem collection name
Process name	Process name associated with the request
Plan name	Access plan name for the set of associated SQL statements
Package name	Package name for the set of associated SQL statements
Connection type	Name associated with the environment that is creating the request
Collection name	Customer-defined name for the group of associated packages
Correlation	Name associated with the user/program creating the request, which may reside anywhere in the network
Procedure name	DB2-stored SQL procedure name associated with the request
Function name	Function name
Performance group	Performance group number (PGN) associated with the request
Scheduling env	Scheduling environment

Figure 18-192 I-Info pop-up enclave data

Enclaves panel overtypable field

The overtypable field for the enclaves panel is shown as follows:

Field	Description
SrvClass	Service class name

18.34 Processes (PS) panel

The Processes (PS) panel displays information about z/OS UNIX System Services processes.

A UNIX process is defined as being an instance of a program running on a system and the resources that it uses. A process can have one or more threads; a thread is a single flow of control within a process. Application programmers create multiple threads to structure an application in independent sections that can run in parallel for more efficient use of system resources.

Figure 18-193 shows the PS command syntax.

```
PS {ALL | ACTIVE}
  ALL displays all z/OS UNIX System Services processes. This is the default.
  ACTIVE displays only active processes.
```

Figure 18-193 PS command syntax

Figure 18-194 on page 978 shows the processes panel invoked by the PS command.

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF PROCESS DISPLAY  SC75     ALL                     LINE 1-17 (28)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
PREFIX=%*  DEST=*  OWNER=*  SYSNAME=SC75
ACTION=//-Block,=-Repeat,+-Extend,C-Cancel,D-Display,K-Kill,T-Terminate
NP   JOBNAME  JobID    Status                           Owner    State
CPU-Time
     BPXOINIT          RUNNING                          OMVSKERN MR
7.66
     RESOLVER          RUNNING                          IBMUSER  1R
0.63
     RESOLVER          RUNNING                          IBMUSER  1R
0.63
     JES3NS            RUNNING                          JES3     1R
15.65
     HZSPROC  JOB20946 RUNNING                          IBMUSER  1R
63.07
     TCPIP    JOB20969 FILE SYS KERNEL WAIT             TCPIP    1F
417.69
     TN3270   JOB20974 RUNNING                          IBMUSER  1R
189.31
     TN3270   JOB20974 RUNNING                          IBMUSER  1R
189.31
     FTPOE1   JOB20966 SWAPPED,FILE SYS KERNEL WAIT     TCPIP    1FI
0.01
     CEA               FILE SYS KERNEL WAIT             CEA      1F
0.10
     CBDQDISP JOB20972 SWAPPED,FILE SYS KERNEL WAIT     TCPIP    1FI
0.01
     PFA      JOB20978 SWAPPED,RUNNING                  PFA      1RI
25.00
     DB9YDIST JOB20982 FILE SYS KERNEL WAIT             IBMUSER  MF
0.79
```

Figure 18-194 Processes panel

In the UNIX operating environment, the innermost level of UNIX is the kernel. This is the actual UNIX operating system, a program that always resides in memory. Sections of the code in this program are executed on behalf of users to do needed tasks, like access files or terminals. The OMVS address space is not considered a process.

BPXOINIT is the started procedure that runs the z/OS UNIX initialization process. The BPXOINIT address space has two categories of functions:

1. It behaves as PID(1) of a typical UNIX system. It is the parent of /etc/rc, and it inherits orphaned children so that their processes get cleaned up using normal code in the kernel. BPXOINIT is also the parent of MVS address spaces that are dubbed and not created by fork or spawn. Therefore, TSO/E commands and batch jobs have a parent PID of 1.
2. Certain functions that the kernel performs need to be able to make normal kernel calls. The BPXOINIT address space is used for these activities.

Processes panel fields

The Processes panel includes some or all of the fields shown in Figure 18-195. Figure 18-196 on page 979 shows the values for the state column.

Value	Description
1	State is for a single thread process
A	Message queue receive wait
B	Message queue send wait
C	Communication system kernel wait
D	Semaphore operation wait
E	Quiesce frozen
F	File system kernel wait
G	MVS pause wait
H	Process state is for multiple threads and pthread was used to create one of the threads. Process state is obtained from the initial pthread created task (IPT).

Figure 18-195 Processes panel state column values

Title	Description
JOBNAME	Job name
JobID	JES job ID
Status	Status of the process
Owner	Userid of the owner
State	State of the process or the most recently created thread
CPU-Time	Compute time in hundredths of seconds
PID	Process ID
PPID	Parent process ID
ASID	Address space ID
ASIDX	Address space ID in hexadecimal
LatchWaitPID	PID on which this process is waiting
Command	Command that created the process
ServerName	Server name
Type	Server type
ActFiles	Number of active files
MaxFiles	Maximum number of files
St-Time	Time the process was started
St-Date	Date the process was started
SysLevel	Level of z/OS on the system
SysName	System name where the process is executing
SecLabel	Security label

Figure 18-196 Processes panel fields

Processes panel NP field actions

Figure 18-197 shows the action characters that can be entered in the NP column on the processes panel.

NP action	MVS command / SDSF description
C-Cancel	C jobname,[A=asidx|U=userid] - Cancel the address space that owns the process
D -Display	D OMVS,PID=processid - Display information about processes
K-Kill	F BPXOINIT,FORCE=processid - Kill the process (SIGKILL)

Figure 18-197 Process panel NP field actions

18.35 Health Checker (CK) panel

The SDSF Health Checker (CK) panel displays information from IBM Health Checker for z/OS. The panel shows the active checks. Checks that are currently running are highlighted.

IBM Health Checker for z/OS is a z/OS component that is used 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.

Figure 18-199 on page 981 is an example of the Health Checker panel displayed. The CK command invokes the Health Checker panel (Figure 18-198 on page 980). You must be authorized to use this command.

```
CK (category|E|EH|EM|EL|EN|D|ALL)
   with no parameters displays active checks.
   category shows only checks for that category. The value can include * (any
   string of characters) or % (any single character).
   E displays all exception checks, with these variations:
      EH - exception-high
      EM - exception-medium
      EL - exception-low
      EN - exception-none
   D displays deleted checks.
   ALL displays deleted as well as active checks.
```

Figure 18-198 CK command syntax

```
  Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
---
SDSF HEALTH CHECKER DISPLAY  (ALL)                    LINE 1-11 (333)
COMMAND INPUT ===>                                          SCROLL ===>
HALF
PREFIX=%*  DEST=*  OWNER=*  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,A-Activate,D-Display,DD-DisplayDiag,
ACTION=DL-DisplayLong,DP-DisplayPolicies,DPO-DisplayOutdatedPolicies,
ACTION=DS-DisplayStatus,E-Refresh,H-Deactivate,L-ListHistory,P-Delete,
ACTION=PF-DeleteForce,R-Run,S-Browse,SB-ISPFBrowse,SBI-ISPFBrowseIn,
ACTION=SBO-ISPFBrowseOut,SE-ISPFEdit,SEI-ISPFEditIn,SEO-ISPFEditOut,
ACTION=U-RemoveCat,X-Print,XC-PrintClose,XD-PrintDS,XDC-PrintDSClose,
ACTION=XF-PrintFile,XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose
NP   NAME                            CheckOwner       State
Status
     ALLOC_ALLC_OFFLN_POLICY         IBMALLOC         ACTIVE(ENABLED)
SUCCES
     ALLOC_SPEC_WAIT_POLICY          IBMALLOC         ACTIVE(ENABLED)
SUCCES
     ALLOC_TIOT_SIZE                 IBMALLOC         ACTIVE(ENABLED)
SUCCES
     ASM_LOCAL_SLOT_USAGE            IBMASM           ACTIVE(ENABLED)
SUCCES
     ASM_NUMBER_LOCAL_DATASETS       IBMASM           ACTIVE(ENABLED)
SUCCES
     ASM_PAGE_ADD                    IBMASM           ACTIVE(ENABLED)
SUCCES
     ASM_PLPA_COMMON_SIZE            IBMASM           ACTIVE(ENABLED)
SUCCES
     ASM_PLPA_COMMON_USAGE           IBMASM           ACTIVE(ENABLED)
SUCCES
     CATALOG_IMBED_REPLICATE         IBMCATALOG       ACTIVE(ENABLED)
EXCEPT
     CEE_USING_LE_PARMLIB            IBMCEE           ACTIVE(ENABLED)
EXCEPT
```

Figure 18-199 Health Checker panel

Health Checker panel fields

The Health Checker panel includes some or all of the fields listed in Figure 18-200 and Figure 18-201 on page 982.

Title	Description
NAME	Check name
CheckOwner	Check owner
State	Check state
Status	Check status
Result	Result from the last invocation of the check
Diag1	Diagnostic data from the check (first word)
Diag2	Diagnostic data from the check (second word)
DiagFrom	Source for the diagnostic data: ABEND, HCHECKER or CHECKRTN

Figure 18-200 Health Checker panel fields 1 of 2

Title	Description
Global	Indicator if this is a global check
GlobalSys	System on which the global check is running
ExcCount	Number of exceptions detected in the last iteration of the check
RunCount	Number of times the check has been invoked
Fail	Number of times the check failed
Severity	Severity level of the check
SevCode	Numeric severity level of the check
WTOType	WTO type or descriptor code
ModifiedBy	How the check was modified
PolicyStatus	Policy error status
WTONum	Number of WTOs issued by the check
NumCat	Number of categories in which the check is defined
Category	Category name
Category2-16	Category names two through sixteen
ExitName	Exit module name that added the check
ModName	Check module name at which the check runs
MsgName	Message load module name
UserDate	Current date of the check (YYYYMMDD)
DefDate	Default date of the check (YYYYMMDD)
Debug	Debug mode indicator
Start-Date-Time	Date and time the check last started
Interval	Interval at which the check runs
NextSch-Date-Time	Date and time the check is next scheduled to run (YYYY.DDDD HH:MM:SS)
NextSch-Int	Time remaining until the check runs, in hhhhh:mm:ss
Log-Date-Time	Date and time of the last successful write to System Logger
Deleted-Date-Time	Date and time the check was deleted
ProcName TaskID	Procedure name and started task ID for IBM Health Checker for z/OS
Reason	Description of the reason for the check
TaskID	Health Checker started task ID
UpdateReason	Description of updates to the check
ParmLen	Length of the check parameters
Parameters	Check parameters. Unprintable characters are translated to periods (.).
SysLevel	Level of the operating system
SysName	System name
EInterval	Interval at which the check will run when it has raised an exception
ExecName	Name of the exec to run
Locale	Where the check is running
Origin	Origin of the check
Verbose	Verbose mode for the check
RexxIn	Rexx input data set name
RexxOut	Rexx output data set name
LogStream	Name of the logstream used to record this check

Figure 18-201 Health Checker panel fields 2 of 2

Health Checker panel NP field actions

Action characters that can be entered in the NP column by authorized users are displayed in Figure 18-202 on page 983.

```
NP action                    MVS command / SDSF description
A-Activate                   F stcid,ACTIVATE - Activate
D-Display                    F stcid,DISPLAY - Display
DD-DisplayDiag               F stcid,DISPLAY,CHECKS,DETAIL,DIAG,CHECK= - Display
                             information, diagnostic form
DL-DisplayLong               F stcid,DISPLAY - Display long
DP-DisplayPolicies           F stcid,DISPLAY,POLICY,DETAIL - Display policies
DPO-DisplayOutdatedPolicies  F stcid,DISPLAY,POLICY,OUTDATED - Display policies that
                             are outdated and not applied
DS-DisplayStatus             F stcid,DISPLAY,STATUS - Display status
E-Refresh                    F stcid,REFRESH - Refresh
H-Deactivate                 F stcid,DEACTIVATE - Deactivate
L-ListHistory                List history (display the CKH panel). The check must have
                             a history (see the Log-Date-Time column and Figure 18-204
                             on page 984).
P-Delete                     F stcid,DELETE - Delete
PF-DeleteForce               F stcid,DELETE,FORCE=YES - Delete force: delete even if
                             it is running
R-Run                        F stcid,RUN - Run
S-Browse                     Browse (access SDSF's Output Dataset Panel)
SB-ISPFBrowse                Browse using ISPF Browse
SBI-ISPFBrowseIn             Browse REXX input data set using ISPF browse
SBO-ISPFBrowseOut            Browse REXX output data set using ISPF browse
SE-ISPFEdit                  Browse using ISPF Edit
SEI-ISPFEditIn               Edit REXX input data set using ISPF Edit
SEO-ISPFEditOut              Edit REXX output data set using ISPF Edit
U-RemoveCat                  Remove all categories for the check
X-Print                      Print the check output. You can add the following:
   C - Close the print file after printing (XC)
   D - Display the Open Print Data Set panel (XD or XDC)
   F - Display the Open Print File panel (XF or XFC)
   S - Display the Open Print panel (XS or XSC)
Note: When Log-Date-Time value is set to ****** N/A ******, the L action
issues SDSF message 'NOT VALID FOR TYPE' -- Explanation: The action
character is not a valid action against that object type.
```

Figure 18-202 Health Checker panel NP field actions

Health Checker panel overtypable fields

The overtypable fields for the Health Checker panel are in Figure 18-203.

```
Field        MVS command / SDSF description
Category     F stcid,UPDATE,CHECK - Category of the check. Type + alone to work with
             the full set of categories.
Debug        F stcid,UPDATE,CHECK - Debug mode indicator (OFF, ON)
EInterval    F stcid,UPDATE,CHECK - Interval at which the check runs when it finds an
             exception (SYSTEM, HALF, hhh:mm)
Interval     F stcid,UPDATE,CHECK - Interval at which the check runs (hhh:mm)
Parameters   F stcid,UPDATE,CHECK - Parameters for the check.
Severity     F stcid,UPDATE,CHECK - Severity level of the check (HIGH, MEDIUM, LOW,
             NONE)
UserDate     F stcid,UPDATE,CHECK - Date of the check
Verbose      F stcid,UPDATE,CHECK - Verbose mode for the check
WTOType      F stcid,UPDATE,CHECK - WTO type or descriptor code, in decimal
```

Figure 18-203 Health Checker panel overtypable fields

Health Check history (CHK) panel for CK panel L NP action

The CKH panel, Figure 18-204, shows information about instances of a check selected from the Health Checker panel with the L-ListHistory NP action.

```
SDSF CK HISTORY  DAE_SHAREDSN                          LINE     1 COLS   1
113
COMMAND INPUT ===>
PREFIX=*  DEST=**  OWNER=*  SYSNAME=*
ACTION=//-Block,=-Repeat,+-Extend,S-Browse,SB-ISPFBrowse,SE-ISPFEdit,X-Print
,XC-
ACTION=XF-PrintFile,XFC-PrintFileClose,XS-PrintSysout,XSC-PrintSysoutClose
NP     COUNT      CheckOwner       Status            Result Diag1    Diag2
S
       30         IBMDAE           SUCCESSFUL              0 00000000
00000000 0
       29         IBMDAE           SUCCESSFUL              0 00000000
00000000 0
       28         IBMDAE           SUCCESSFUL              0 00000000
00000000 0
       27         IBMDAE           SUCCESSFUL              0 00000000
00000000 0
       26         IBMDAE           SUCCESSFUL              0 00000000
00000000 0
       25         IBMDAE           SUCCESSFUL              0 00000000
00000000 0
```

Figure 18-204 Health Check history panel

The Health Checker history panel includes fields listed in Figure 18-205. Browse and print NP field action characters are the only ones available on the Health Checker history panel.

Title	Description
Count	Count of this instance of the check
CheckOwner	Check owner
Status	Check status
Result	Result code from the check
Diag1	Diagnostic data from check, word 1
Diag2	Diagnostic data from check, word 2
Start-Date-Time	Date and time the check started (YYYY.DDD HH:MM:SS)
End-Date-Time	Date and time the check ended (YYYY.DDD HH:MM:SS)
Sysplex	Sysplex name for the sysplex on which the check ran
SysName	System name for the system on which the check ran
Name	Check name

Figure 18-205 Health Check History panel fields

18.36 SDSF REXX and SDSF in batch

This section briefly discusses:

- How to access SDSF data and functions with the REXX programming language. Using REXX with SDSF provides a simpler and more powerful alternative to using SDSF in batch.
- How to use batch processing to issue often-repeated SDSF commands by creating a list of the commands as control statements. In the list, you specify the SDSF panel you wish to use and the operation you wish to perform on it.

The Restructured Extended Executor (REXX) language is a procedural language that allows you to write programs and algorithms in a clear and structural way. It is an interpreted and compiled language, and you do not have to compile a REXX command list before executing it.

SDSF with REXX merges the power of SDSF with the simplicity of REXX. REXX with SDSF integrates with your REXX executable by executing commands and returning the results in REXX variables. To understand the REXX with SDSF API you need to understand the commands and what they do and you need to know which variables are set and what these variables include. For a full description of REXX with SDSF, refer to the chapter "Using SDSF with the REXX programming language" *in z/OS SDSF Operation and Customization,* SA22-7670. You can also refer to the interactive tutorial panels that display when you press PF1when using SDSF or use the REXXHELP command from any SDSF panel.

18.37 SDSF REXX

SDSF REXX provides access to SDSF functions through the REXX programming language. The REXX support is a simpler and more powerful alternative to SDSF batch.

To access SDSF functions with REXX, you use:

- The ISFCALLS command, to add and delete the SDSF host command environment
- The ISFEXEC command, for SDSF commands, such as the commands that access SDSF panels
- The ISFACT command, for action characters and overtyping columns
- Special REXX variables, to provide function equivalent to other SDSF commands, and for messages and table data

You must be authorized to use SDSF from REXX, and to the SDSF functions that you invoke from REXX. Depending on how your SDSF security is implemented, you may be placed in a different SDSF user group when you use SDSF from REXX than when you use SDSF interactively. In some cases, invoking an SDSF function from REXX when you are not authorized to the function will cause the exec to fail and the invocation of SDSF to end.

REXXHELP command

To display the online help for using REXX with SDSF, type REXXHELP on any command line in SDSF when using SDSF under ISPF. Figure 18-206 and Figure 18-207 on page 986 display the REXXHELP pop-ups.

```
ISFG90                    Using REXX with SDSF
                                                              More:     +
Tab to a topic and press Enter, or press Enter to view the topics in
order.

o   Introduction                                - Search - Index -
o   Programming practices
o   Quick start
o   Add the SDSF host command environment
o   Issue SDSF commands
    - Commands for tabular panels (ISFEXEC)
    - Log panels (ISFLOG and ISFULOG)
    - Slash (/) commands (ISFSLASH)
    - Other commands (ISFEXEC)
    - Filter commands (special variables)
    - Options commands (special variables)
o   Take actions and modify columns on SDSF panels
o   Browse output and Print output
o   Examples
o   Diagnose errors in a REXX exec
 F1=Help   F7=Up   F8=Down   F12=Cancel
```

Figure 18-206 REXXHELP pop-up index 1 of 2

```
ISFG90                    Using REXX with SDSF
                                                              More:   -
order.

o   Introduction                                - Search - Index -
o   Programming practices
o   Quick start
o   Add the SDSF host command environment
o   Issue SDSF commands
    - Commands for tabular panels (ISFEXEC)
    - Log panels (ISFLOG and ISFULOG)
    - Slash (/) commands (ISFSLASH)
    - Other commands (ISFEXEC)
    - Filter commands (special variables)
    - Options commands (special variables)
o   Take actions and modify columns on SDSF panels
o   Browse output and Print output
o   Examples
o   Diagnose errors in a REXX exec
o   Special variables
 F1=Help   F7=Up   F8=Down   F12=Cancel
```

Figure 18-207 REXXHELP pop-up index 2 of 2

Support for SDSF REXX commands:

- Using SDSF with REXX requires that you add a host command environment prior to any other SDSF host environment commands. The host command environment is what allows you to use Address SDSF on the ISFEXEC and ISFACT commands.You add the host command environment with the ISFCALLS() function.

- The ISFEXEC host command, which allows you to issue commands that access tabular panels (DA, ST, and so on), the slash (/) command, and a few others.

  You issue SDSF commands using the ISFEXEC host environment command, as:

  ```
  Address SDSF "ISFEXEC sdsf-command ( options )"
  ```

  The ISFEXEC command returns data as follows:

  Column data. The column data is returned in stem variables in this format:

  ```
  column-name.row-number
  ```

  The value for stem variable number 0 is a count of the number of variables returned.

- You invoke SDSF action characters and modify column values using the ISFACT host environment command, as follows:

  ```
  Address SDSF " ISFACT command Token PARM ( parms) ( options ) "
  ```

  `command` is the command for the panel. It must be the same SDSF command, including any parameters, that was previously entered with the ISFEXEC command.

  `Token` identifies the rows to be acted upon.

  `parms` is the list of parameters that specifies the action characters and modifications.

  `options` is an optional list of options. For example, requests the alternate field list for a panel or specifies that SDSF should wait the full delay interval before retrieving responses to a command.

- You access both the SYSLOG and the OPERLOG using the ISFLOG host environment command. When used with the SYSLOG, the ISFLOG command processes the JES logical log.

  The syntax of the ISFLOG command is as follows:

  ```
  Address SDSF " ISFLOG ALLOC TYPE ( SYSLOG ) [(VERBOSE)] “
  ```

  The allocation is done with the FREE=CLOSE option so that the file is automatically de-allocated when closed. The stem variable ISFDDNAME contains the ddname that is returned and ISFDSNAME contains the data set name.

  ```
  Address SDSF " ISFLOG READ TYPE ({OPERLOG | SYSLOG}) [(VEROSE)] “
  ```

  The records read are returned in the ISFLINE stem variable. ISFLINE.0 contains the number of variables. By default, SDSF retrieves the current day's records. You can customize the results with the special variables:

  - ISFLINELIM sets a limit on the number of variables created.
  - ISFLOGSTARTTIME, ISFLOGSTARTDATE, ISFLOGSTOPTIME and ISFLOGSTOPDATE define the date and time range for the records. Use them to ensure that your date and time range is reasonable, so that an excessive number of variables is not created.

- You issue system commands using the ISFSLASH host environment command. The syntax of the ISFSLASH command is:

  ```
  Address SDSF " ISFSLASH ({stem-name | list}) (options) “
  ```

  Options for slash (/) commands:

  - INTERNAL specifies that console ID 0 (INTERNAL) should be used to issue the command
  - WAIT specifies that SDSF should wait the full delay interval before retrieving responses. This option is strongly recommended to ensure that the responses are accessible in the ISFULOG special variable. The delay interval is specified with the ISFDELAY variable.

Special variables for slash (/) commands set options such as the delay limit and the console name.

ISFCMDLIM limits the number of commands that may be issued through ISFSLASH.

ISFCONS specifies the console name for the user session log (ULOG).

- You can request all of the column values for a specific row using the ISFGET host environment command, as follows:

  `Address SDSF " ISFGET command TOKEN ('" token "') ( options ) “`

  `command` is the command for the panel. It must be the same SDSF command, including any parameters, that was previously entered with the ISFEXEC command.

  `token` identifies the row to be acted upon. The token was previously set by ISFEXEC or ISFACT for the panel accessed with command.

- To browse the output of jobs and browse check output on the CK or CKH panel, you use a combination of action characters, with ISFACT, and special REXXvariables.

  To browse job output from the DA, H, I, JDS, O and ST panels, allocate the output data sets with special REXX-only action characters, then browse the data sets using EXECIO or a similar utility. The action characters are:

  – SA - Allocate all data sets associated with the item. On the DA, I or ST panels, this will be all data sets in the job. On the O and H panels, it will be all data setsin the output group. On the JDS panel, it will be a single data set.

  – SJA - Allocate the JCL data set.

  To browse check output from the CK or CKH panel, use the S action character with the special variable ISFLINE.

- SDSF defines several REXX variables to supplement host environment commands and provide feedback for requests. These special variables all begin with the prefix ISF. Any variable starting with that prefix is considered reserved for use by SDSF. Do not name variables in your REXX execs with the prefix ISF.

  The names of special variables are not affected by the PREFIX option used with the ISFEXEC or ISFACT commands.

  Some special variables correspond to SDSF commands and result in a command being issued. The authorization for those special variables is the same as for the associated command. In some cases, using a special variable when you are not authorized to the associated command will cause the exec to fail and the invocation of SDSF to end.

  Special REXX variables provide function equivalent to many of the SDSF commands. The special variables use the format:

  `variable-name='parameters'`

  The parameters for the variable are the same as for the associated command, with the exception that the ? parameter is not supported. The values of special variables are not saved across sessions (or invocations) in the REXX environment.

  The variables are grouped by command type:

  – SDSF command
  – Filter commands
  – Options commands
  – Trace commands

  To drop SDSF special variables (that is, unassign the variables and restore them to their original undefined state) use the ISFRESET() function. The option to use with ISFRESET corresponds to the variable type (Input, InOut, or Output).

For a full description of the SDSF REXX special variables refer to the chapter *"*Using SDSF with the REXX programming language" *in z/OS SDSF Operation and Customization,* SA22-7670*.* REXXHELP also provides details on the special variables for panels, entering system commands, filtering, options, other commands, browsing, printing and messages.

SDSF REXX examples

See Appendix B, "SDSF REXX and SDSF in batch examples" on page 999 for SDSF REXX examples:

- "SDSF REXX - Show jobs in the JES3 MDS queue" on page 1000
- "SDSF REXX - Access SYSLOG sample" on page 1000
- "SDSF REXX - Send a JES3 command to the global" on page 1001
- "SDSF REXX - Display JES3 DJC nets and the DJC net status" on page 1002

A

SDSF ISFPARMS default definitions

This appendix provides listings of the sample ISFPARMS definitions in the ISF.SISFJCL data set ISFPRM00 member.

```
/*********************************************************************/
/*                                                                   */
/*              Sample SDSF Initialization Statements                */
/*                                                                   */
/*                                                                   */
/*  Proprietary Statement =                                          */
/*                                                                   */
/*    Licensed Materials - Property of IBM                           */
/*    5694-A01                                                       */
/*    Copyright IBM Corp. 1981, 2011.                                */
/*                                                                   */
/*    Status = HQX7780                                               */
/*                                                                   */
/*  EXTERNAL CLASSIFICATION = OTHER                                  */
/*  END OF EXTERNAL CLASSIFICATION:                                  */
/*                                                                   */
/*                                                                   */
/*  This is a sample SDSF parameter definition.  It is equivalent    */
/*  to the macros supplied in ISFPARMS.                              */
/*                                                                   */
/*  To use this member, copy it to SYS1.PARMLIB or a dataset         */
/*  concatenated to it and edit the member as appropriate.           */
/*  Alternatively, you can modify the SDSF server JCL to point       */
/*  to a data set that contains the member.                          */
/*                                                                   */
/*  Note that, even with conditional processing, if you want         */
/*  to use a common member with different levels of SDSF, you        */
/*  must ensure that the member does not include support (such       */
/*  as new keywords or values) that was introduced in a              */
/*  higher level of SDSF.                                            */
/*                                                                   */
/*  The SDSF server must be started for the member to be used.       */
```

```
/*  If the SDSF server is not active, the macros in ISFPARMS        */
/*  are used instead.                                               */
/*                                                                  */
/*  The following are general syntax rules for coding the SDSF      */
/*  initialization statements.  Refer to the SDSF Operation and     */
/*  Customization manual for more details.                          */
/*                                                                  */
/*    - Statements are free form, and can appear in any column 1-72. */
/*      An optional sequence number may be coded in columns 73-80,  */
/*      but it is not used by SDSF.                                 */
/*                                                                  */
/*    - A statement can span any number of lines.  Use a trailing   */
/*      comma to indicate that a statement is continued.            */
/*                                                                  */
/*    - Comments can be coded at any point a blank is allowed using */
/*      the slash-asterisk notation.  Blank lines can be inserted   */
/*      at any point to improve readability.                        */
/*                                                                  */
/*    - All values are translated to upper case.  Enclose the value */
/*      in quotes if it contains special characters or contains     */
/*      mixed case.                                                 */
/*                                                                  */
/*    - Statements may appear in any order, except that the FLDENT  */
/*      must follow an FLD, and the NTBLENT must follow an NTBL.    */
/*      SERVER statements must follow a SERVERGROUP.                */
/*                                                                  */
/*    - A keyword value of blanks may be specified by coding one    */
/*      or more blanks enclosed in quotes for the value.            */
/*                                                                  */
/********************************************************************/

    /**************************************************/
    /* WHEN Statement - Provide Conditional Processing */
    /**************************************************/

 WHEN                        /* Reset any prior WHEN conditions      */


/*******************************************************************/
/* Note: The following statements are commented out to show the    */
/*       syntax.  The statements are only needed when the sysplex  */
/*       support is to be used.                                    */
/*                                                                 */
/*       Refer to the Operation and Customization Guide for the    */
/*       complete set of options that may be specified.            */
/*******************************************************************/

    /*********************************************************/
    /* SERVERGROUP, SERVER, and COMM - Define Communications */
    /*********************************************************/

/* SERVERGROUP */                /* Defines a group of SDSF servers */

/*******************************************************************/
/*   Each SERVER statement defines an SDSF server in the sysplex.  */
```

```
/*   The server in turn relates to a specific JES2 member for which */
/*   data is to be gathered.  Repeat the SERVER and COMM statements */
/*   as many times as necessary to define all the JES2 members for  */
/*   which data is to be shown.                                      */
/*                                                                   */
/*   Note: All servers must be in the same sysplex and all JES2     */
/*   members must be in the same MAS.                                */
/*******************************************************************/


/*   SERVER NAME(sdsf-servername),    /* Names the SDSF server      */
/*          SYSNAME(system-name),     /* System name for server     */
/*          JESNAME(jes2-subsystem-name),  /* JES2 procedure name   */
/*          MEMBER(jes2-member-name), /* JES2 member name           */
/*          COMM(comm-statement-name) /* Related COMM statement     */

/*   COMM NAME(statement-name),    /* Defines communications parms  */
/*        QMGR(qmgr-name)             /* QMgr name for connections  */
/*        CLUSTER(clustername),       /* Cluster name for queues    */
/*        QREPLACE(YES),              /* Replace prior queue defs   */
/*        QDELETE(NO),                /* Do not delete queues       */
/*        QDEFINE(YES)                /* Define required queues     */


   /********************************/
   /* CONNECT - Connection Options */
   /********************************/
CONNECT DEFAULT(COND),/* Default server if not already assigned    */
                      /* DEFAULT(NO) to not assign server as default */
                      /* DEFAULT(YES) to unconditionally assign    */
                      /*   server as default                       */
        XCFSRVNM(SAME)  /* Use server name as XCF appl name        */

   /*******************************************/
   /* OPTIONS Statement - Global SDSF Options */
   /*******************************************/

OPTIONS ATHOPEN(YES),       /* Use authorized open for datasets    */
  DCHAR('?'),               /* Command query character             */
  DSI(NO),                  /* Bypass ENQ for dynamic allocation   */
  FINDLIM(5000),            /* Maximum lines to search for FIND    */
  IDBLKS(4096),             /* HASPINDX blocksize                  */
  INDEX(ISF.HASPINDX),      /* HASPINDX dataset name               */
  LINECNT(55),              /* Print lines per page                */
  LOGLIM(0),                /* OPERLOG search limit in hours       */
  MENUS(ISF.SISFPLIB),      /* Panels dataset name for TSO         */
  NIDBUF(5),                /* Number of haspindx buffers          */
  SCHARS('*%'),             /* Generic and placeholder characters  */
  SCRSIZE(1920),            /* Maximum screen size                 */
  SYSOUT(A),                /* Default print sysout class          */
TIMEOUT(5),               /* Communications timeout in seconds   */
  TRACE(C000),              /* Default trace mask                  */
  TRCLASS(A),               /* Default trace sysout class          */
  UNALLOC(NO)               /* Do not free dynalloc data sets      */
```

```
   /***************************************/
   /* GROUP ISFSPROG - System Programmers */
   /***************************************/

GROUP NAME(ISFSPROG),       /* Group name                          */
  TSOAUTH(JCL,OPER,ACCT),   /* User must have JCL, OPER, ACCT      */
  ACTION(ALL),              /* All route codes displayed           */
  ACTIONBAR(YES),           /* Display the action bar on panels    */
  APPC(ON),                 /* Include APPC sysout                 */
  AUPDT(2),                 /* Minimum auto update interval        */
  AUTH(ALL),                /* All authorized functions            */
  BROWSE(NONE),             /* Browse default action character     */
  CMDAUTH(ALL),             /* Commands allowed for all jobs       */
  CMDLEV(7),                /* Authorized command level            */
  CONFIRM(ON),              /* Enable cancel confirmation          */
  CPUFMT(LONG),             /* Long format CPU utilization on DA   */
  CTITLE(ASIS),             /* Allow mixed case column titles      */
  CURSOR(ON),               /* Leave cursor on last row processed  */
/*CUSTOM(SPRGPROP),*/       /* Uncomment for custom properties     */
  DADFLT(IN,OUT,TRANS,STC,TSU,JOB), /* Default rows shown on DA    */
  DATE(MMDDYYYY),           /* Default date format                 */
  DATESEP('/'),             /* Default datesep format              */
  DFIELD2(DAFLD2),          /* Sample alternate field list for DA  */
  DISPLAY(OFF),             /* Do not display current values       */
  DSPAUTH(ALL),             /* Browse allowed for all jobs         */
  EMCSAUTH(MASTER),         /* Activate EMCS cons with master auth */
  EMCSREQ(NO),              /* EMCS console not required           */
  GPLEN(2),                 /* Group prefix length                 */
  ILOGCOL(1),               /* Initial display column in log       */
  INPUT(OFF),               /* Initial value for INPUT command     */
  ISYS(LOCAL),              /* Initial system default              */
  LANG(ENGLISH),            /* Default language                    */
  LOG(OPERACT),             /* Default log option                  */
  OWNER(NONE),              /* Default owner                       */
  RSYS(NONE),               /* Initial system default for wtors    */
  UPCTAB(TRTAB2),           /* Upper case translate table name     */
  VALTAB(TRTAB),            /* Valid character translate table     */
  VIO(SYSALLDA)             /* Unit name for page mode output      */

   /****************************/
   /* GROUP ISFOPER - Operators */
   /****************************/
GROUP NAME(ISFOPER),        /* Group name                          */
  TSOAUTH(JCL,OPER),        /* User must have JCL and OPER         */
  ACTION(ALL),              /* All route codes displayed           */
  ACTIONBAR(YES),           /* Display action bar on panels        */
  APPC(ON),                 /* Include APPC sysout                 */
  AUPDT(2),                 /* Minimum auto update interval        */
  AUTH(ALLOPER),            /* All operator authorized functions   */
  BROWSE(NONE),             /* Browse default action character     */
  CMDAUTH(ALL),             /* Commands allowed for all jobs       */
  CMDLEV(7),                /* Authorized command level            */
  CONFIRM(ON),              /* Enable cancel confirmation          */
```

```
  CPUFMT(LONG),             /* Long format CPU utilization on DA   */
  CTITLE(ASIS),             /* Allow mixed case column titles      */
  CURSOR(ON),               /* Leave cursor on last row processed  */
/*CUSTOM(OPERPROP),*/       /* Uncomment for custom properties     */
  DADFLT(IN,OUT,TRANS,STC,TSU,JOB), /* Default rows shown on DA    */
  DATE(MMDDYYYY),           /* Default date format                 */
  DATESEP('/'),             /* Default datesep format              */
  DISPLAY(OFF),             /* Do not display current values       */
  DSPAUTH(USERID,NOTIFY,AMSG), /* Browse authority                 */
  EMCSAUTH(MASTER),         /* Activate EMCS cons with master auth */
  EMCSREQ(NO),              /* EMCS console not required           */
  GPLEN(2),                 /* Group prefix length                 */
  ILOGCOL(1),               /* Initial display column in log       */
  ISYS(LOCAL),              /* Initial system default              */
  LANG(ENGLISH),            /* Default language                    */
  LOG(OPERACT),             /* Default log option                  */
  OWNER(NONE),              /* Default owner                       */
  RSYS(NONE),               /* Initial system default for wtors    */
  UPCTAB(TRTAB2),           /* Upper case translate table name     */
  VALTAB(TRTAB),            /* Valid character translate table     */
  VIO(SYSALLDA)             /* Unit name for page mode output      */


   /*********************************/
   /* GROUP ISFUSER - General Users */
   /*********************************/
GROUP NAME(ISFUSER),        /* Group name                          */
  TSOAUTH(JCL),             /* User must have JCL                  */
  ACTION(11,12,USER),       /* Default route codes in log          */
  ACTIONBAR(YES),           /* Display action bar on panels        */
  APPC(ON),                 /* Include APPC sysout                 */
AUPDT(10),                /* Default auto update interval        */
  AUTH(ALLUSER),            /* All user authorized functions       */
  BROWSE(NONE),             /* Browse default action character     */
  CMDAUTH(USERID,NOTIFY),   /* Command authority                   */
  CMDLEV(2),                /* Command level                       */
  CONFIRM(ON),              /* Enable cancel confirmation          */
  CPUFMT(LONG),             /* Long format CPU utilization on DA   */
  CTITLE(ASIS),             /* Allow mixed case column titles      */
/*CUSTOM(USERPROP),*/       /* Uncomment for custom properties     */
  CURSOR(ON),               /* Leave cursor on last row processed  */
  DADFLT(IN,OUT,TRANS,STC,TSU,JOB), /* Default rows on DA          */
  DATE(MMDDYYYY),           /* Default date format                 */
  DATESEP('/'),             /* Default datesep format              */
  DISPLAY(OFF),             /* Do not display current values       */
  DSPAUTH(USERID,NOTIFY),   /* Browse authority                    */
  EMCSAUTH(MASTER),         /* Activate EMCS cons with master auth */
  EMCSREQ(NO),              /* EMCS console not required           */
  ILOGCOL(1),               /* Initial display column in log       */
  LANG(ENGLISH),            /* Default language                    */
  LOG(OPERACT),             /* Default log option                  */
  OWNER(USERID),            /* Default owner                       */
  PREFIX(USERID),           /* Default prefix                      */
  UPCTAB(TRTAB2),           /* Upper case translate table name     */
  VALTAB(TRTAB),            /* Valid character translate table     */
```

```
  VIO(SYSALLDA)             /* Unit name for page mode output      */


   /********************/
   /* Sample NTBL list */
   /********************/
NTBL NAME(SLIST)
  NTBLENT STRING($S),OFFSET(1)
  NTBLENT STRING(P),OFFSET(7)
  NTBLENT STRING(PAY),OFFSET(3)


   /********************************/
   /* Define default SDSF Codepage */
   /********************************/
TRTAB CODPAG(SDSF) VALTAB(TRTAB) UPCTAB(TRTAB2)


   /**********************************************/
   /* Sample alternate field list for DA display */
   /**********************************************/
FLD NAME(DAFLD2) TYPE(DA)  /* Name is referenced by GROUP statement */

  FLDENT COLUMN(STEPN),TITLE('StepName'),WIDTH(D)
  FLDENT COLUMN(PROCS),TITLE('ProcStep'),WIDTH(D)
  FLDENT COLUMN(JOBID),TITLE('JobID'),WIDTH(D)
  FLDENT COLUMN(OWNERID),TITLE('Owner'),WIDTH(D)
  FLDENT COLUMN(JCLASS),TITLE('C'),WIDTH(D)
  FLDENT COLUMN(ASID),TITLE('ASID'),WIDTH(D)
  FLDENT COLUMN(ASIDX),TITLE('ASIDX'),WIDTH(D)
  FLDENT COLUMN(EXCP),TITLE(' EXCP-Cnt'),WIDTH(D)
  FLDENT COLUMN(CPU),TITLE('  CPU-Time'),WIDTH(D)
  FLDENT COLUMN(REAL),TITLE('Real'),WIDTH(D)
  FLDENT COLUMN(PAGING),TITLE('Paging'),WIDTH(D)
  FLDENT COLUMN(EXCPRT),TITLE('   SIO'),WIDTH(D)
  FLDENT COLUMN(CPUPR),TITLE('  CPU%'),WIDTH(D)
  FLDENT COLUMN(DP),TITLE('DP'),WIDTH(D)
  FLDENT COLUMN(POS),TITLE('Pos'),WIDTH(D)
  FLDENT COLUMN(SWAPR),TITLE('SR'),WIDTH(D)
  FLDENT COLUMN(PGN),TITLE('PGN'),WIDTH(D)
  FLDENT COLUMN(DOMAIN),TITLE('DmN'),WIDTH(D)
  FLDENT COLUMN(STATUS),TITLE('Status'),WIDTH(D)
  FLDENT COLUMN(WORKLOAD),TITLE('Workload'),WIDTH(D)
  FLDENT COLUMN(SRVCLASS),TITLE('SrvClass'),WIDTH(D)
  FLDENT COLUMN(PERIOD),TITLE('SP'),WIDTH(D)
  FLDENT COLUMN(RESGROUP),TITLE('ResGroup'),WIDTH(D)
  FLDENT COLUMN(SERVER),TITLE('Server'),WIDTH(D)
  FLDENT COLUMN(QUIESCE),TITLE('Quiesce'),WIDTH(D)
  FLDENT COLUMN(SYSNAME),TITLE('SysName'),WIDTH(D)
  FLDENT COLUMN(SPAGING),TITLE('SPag'),WIDTH(D)
  FLDENT COLUMN(SCPU),TITLE('SCPU%'),WIDTH(D)
  FLDENT COLUMN(ECPU),TITLE(' ECPU-Time'),WIDTH(D)
  FLDENT COLUMN(ECPUPR),TITLE(' ECPU%'),WIDTH(D)
  FLDENT COLUMN(CPUCRIT),TITLE('CPUCrit'),WIDTH(D)
  FLDENT COLUMN(STORCRIT),TITLE('StorCrit'),WIDTH(D)
```

```
  FLDENT COLUMN(RPTCLASS),TITLE('RptClass'),WIDTH(D)
  FLDENT COLUMN(MEMLIMIT),TITLE('MemLimit'),WIDTH(D)
  FLDENT COLUMN(TRANACT),TITLE('Tran-Act'),WIDTH(D)
  FLDENT COLUMN(TRANRES),TITLE('Tran-Res'),WIDTH(D)
  FLDENT COLUMN(SPIN),TITLE('Spin'),WIDTH(D)
  FLDENT COLUMN(SECLABEL),TITLE('SecLabel'),WIDTH(D)
  FLDENT COLUMN(GCPTIME),TITLE('GCP-Time') WIDTH(D)
  FLDENT COLUMN(ZAAPTIME),TITLE('zAAP-Time') WIDTH(D)
  FLDENT COLUMN(ZAAPCPTM),TITLE('zACP-Time') WIDTH(D)
  FLDENT COLUMN(GCPUSE),TITLE('GCP-Use%') WIDTH(D)
  FLDENT COLUMN(ZAAPUSE),TITLE('zAAP-Use%') WIDTH(D)
  FLDENT COLUMN(SZAAP),TITLE('SzAAP%') WIDTH(D)
  FLDENT COLUMN(SZIIP),TITLE('SzIIP%') WIDTH(D)
  FLDENT COLUMN(PROMOTED),TITLE('Promoted') WIDTH(D)
  FLDENT COLUMN(ZIIPTIME),TITLE('zIIP-Time') WIDTH(D)
  FLDENT COLUMN(ZIIPCPTM),TITLE('zICP-Time') WIDTH(D)
  FLDENT COLUMN(ZIIPNTIM),TITLE('zIIP-Ntime') WIDTH(D)
  FLDENT COLUMN(ZIIPUSE),TITLE('zIIP-Use%') WIDTH(D)
  FLDENT COLUMN(SLCPU),TITLE('SLCPU%') WIDTH(D)

    /*********************/
    /* Custom Properties */
    /*********************/

/* The custom properties are defined using a PROPLIST statement */
/* which is referenced by the CUSTOM keyword on the GROUP.  For */
/* each PROPLIST, define the PROPERTY statements for the custom */
/* properties that are required.  See the SDSF Operation and    */
/* Customization manual for the complete list of properties     */
/* that may be specified.                                       */

/* PROPLIST NAME(SPRGPROP)          Group ISFSPROG properties */
  /* PROPERTY NAME(property-name),VALUE(TRUE or FALSE) */

/* PROPLIST NAME(OPERPROP)          Group ISFOPER properties  */
  /* PROPERTY NAME(property-name),VALUE(TRUE or FALSE) */

/* PROPLIST NAME(USERPROP)          Group ISFUSER properties  */
  /* PROPERTY NAME(property-name),VALUE(TRUE or FALSE) */
```

B

SDSF REXX and SDSF in batch examples

This appendix contains sample SDSF REXX EXECs for JES3 SDSF. These EXECs can be downloaded from the IBM Redbooks website.

The following SDSF REXX code are examples in the appendix:

- Show jobs in the JES3 MDS queue
- Access SYSLOG
- Send a JES3 command to the global
- Display JES3 DJC nets and the DJC net status
  - A table is provided to display a panel for DJC nets
- JCL for a SDSF in batch example using the TSO terminal monitor program
  - A BATSD REXX EXEC to display the status for all JES3 jobs
  - Find jobs in the MDS queues using stem variables

SDSF REXX - Show jobs in the JES3 MDS queue

Example B-1 SDSF REXX - Show jobs in the JES3 MDS queue

```
/* Show some SDSF REXX basic features - List JES3 MDS alloc queue */
/* for the user who invokes this REXX exec                        */
mdsq = "MDS Q empty"      /* Nothing in MDS queue                 */
zrc=isfcalls("ON")        /* Add the SDSF host command environment*/
isfprefix="*"             /* Set filtering PREFIX                 */
owner=SYSVAR("sysuid")    /* Get userid for filtering OWNER       */
isfowner=owner            /* Set userid for filtering OWNER       */
zcn="SD"owner             /* Get ULOG console name                */
isfcons=strip(substr(zcn,1,8),"T"," ")  /* Set ULOG console name  */
Address SDSF "ISFEXEC ST" /* Access ST panel with ISFEXEC command */
do i=1 to JNAME.0         /* Loop for all rows returned           */
  if PhaseName.i="AWAIT RES ALLOC" then /* Job in JES3 MDS queue? */
  do                      /* Yes - Query why? - Ask JES3          */
    isfdelay="5"          /* Wait for JES3 response               */
    Address SDSF "ISFACT ST TOKEN('"TOKEN.i"') PARM(NP DMA)"
                          /* Issue DMA action with ISFACT cmd     */
    do j=2 to isfulog.0   /* JES3 response in ULOG for DMA action */
      Say substr(isfulog.j,41)          /* Copy what JES3 says    */
      mdsq = ""
    end
  end
end
if mdsq <> "" then say mdsq
zrc=isfcalls("OFF")       /* Delete SDSF command environment      */
```

The highlights in the REXX exec are examples of SDSF REXX panel access commands, action character commands, and some special REXX variables that provide function equivalent to other SDSF commands.

SDSF REXX - Access SYSLOG sample

Example B-2 SDSF REXX - Access SYSLOG

```
/* SDSF REXX - Access SYSLOG                                      */
rc=isfcalls( ON )         /* Add the SDSF host command environment */
/* Read the SYSLOG using the ISFLOG command                       */
z = TIME("E")
Address SDSF "ISFLOG READ TYPE(SYSLOG)"
z = msgrtn()              /* Tell about the ISFLOG copletion      */
do i=max(1,isfline.0-10) to isfline.0
  Say i "-" isfline.i     /* Write a line to the output stream    */
end
Say TIME(R)
Drop isfline.
/* ALLOCate the SYSLOG using the ISFLOG command                   */
Address SDSF "ISFLOG ALLOC TYPE(SYSLOG) (VERBOSE)"
z = msgrtn()              /* Tell about the ISFLOG VERBOSE        */
say "SYSLOG DD("ISFDDNAME.1") DSN('"ISFDSNAME.1"') ISFLOG ALLOC rc" rc
/* EXECIO READ a few SYSLOG lines                                 */
Address TSO "EXECIO * DISKR" ISFDDNAME.1 "( STEM SLINE. FINIS"
do i=max(1,sline.0-10) to sline.0
  Say i "-" sline.i       /* Write a line to the output stream    */
```

```
end
Say TIME(R)
if rc <> 0 then           /* EXECIO problem - try to free SYSLOG ds*/
  Address TSO "FREE DD("ISFDDNAME.1")"
/* FREE the SYSLOG data set                                        */
say "SYSLOG FREE rc" rc
rc=isfcalls( OFF)         /* Undo the SDSF host command environment*/
Exit 0
/*
Messages issued in response to a command or special variable are in:
  ISFMSG -  contains the SDSF short message
  ISFMSG2 - is a stem variable that contains SDSF numbered messages.
*/
msgrtn: procedure expose isfmsg isfmsg2.
/* The isfmsg variable contains a short message                    */
Say " "
if isfmsg<>"" then
  Say "isfmsg is:" isfmsg
/* The isfmsg2 stem contains additional descriptive error messages */
do ix=1 to isfmsg2.0
  Say "isfmsg2."ix "is:" isfmsg2.ix
end
Say " "
return 0
```

SDSF REXX - Send a JES3 command to the global

Example B-3 SDSF REXX - Send a JES3 command to the global

```
/* SDSF REXX - Send a JES3 command to the global                  */
trace 0
zrc=isfcalls("ON")        /* Add the SDSF host command environment*/
/* Check for JES3 environment                                     */
jt = ""                                  /* JES type             */
jg = ""                                  /* JES3 global system   */
js = ""                                  /* JES3 CONSTD SYN      */
Address SDSF "ISFEXEC WHO"               /* Ask SDSF for WHO info*/
do i=1 to  ISFRESP.0                     /* Loop through ISPRESP.*/
  parse var isfresp.i keyw "=" kval      /* Parse ISPRESP.i      */
  if keyw = "JESTYPE" then jt = kval     /* If type OK - save it */
  if keyw = "JESNAME" then jn = kval     /* Save JES name        */
  if keyw = "GLOBAL"  then jg = kval     /* Save global system   */
end
if jt<>"JES3" then do; z=sayit(jt "not JES3 - Exiting..."); exit 16; end
Address "ISPEXEC" "VGET (ZSCREEN)"       /* Possible ISPS118L    */
if rc<>0 then ZSCREEN = "Z"              /* ISPF not available   */
/* If JES3 is unable to define (*) as a SYSPLEX prefix, the (*) will
be used by JES3 as a SYSTEM scoped prefix. WHO does not return JES3
PLEXSYN=. However the MVS ROUTE and * command prefix will work.   */
cmd = "I,MAIN=JGLOBAL X"                 /* A JES3 command       */
jgp = "RO "jg",*"                        /* Cmd prefix: 'RO sys,'*/
z = entcmd(jgp||cmd)                     /* Issue JES3 command   */
/* Use JESPlex panel to find JES3 system scopre command prefix    */
isffilter = "Global EQ YES"              /* Filter JP global row */
Address SDSF "ISFEXEC JP"                /* 'Display' JP panel   */
  if rc<>0 then do
    call msgrtn                          /* List error messages  */
    return 16
```

```
  end
/* REXX execs reference columns by name rather than by title.     */
if name.0 = 1 then
  jpg = "RO " name.1","comchar.1
else exit 16
cmd = "I O DLOG"
z = entcmd(jgp||cmd)                        /* Issue JES3 command   */
zrc=isfcalls("OFF")       /* Delete SDSF command environment      */
exit 0
/* Subroutine to SDSF ISFSLASH a JES3 command end echo the respose*/
entcmd:
  isfcons =    userid() || zscreen        /* Console name         */
  parse arg ca
  Address SDSF "ISFSLASH '"ca"' (WAIT)"   /* Issue a command      */
  /* Extract * command output from ULOG                           */
  if rc<>0 then do
    call msgrtn                           /* List error messages  */
    return 16
  end
  do ix=1 to isfulog.0
    Say isfulog.ix
  end
return 0
/* Subroutine to issue error messages                             */
msgrtn: procedure expose isfmsg isfmsg2.
/* The isfmsg variable contains a short message                   */
  Say " "
  if isfmsg<>"" then
    Say "isfmsg is:" isfmsg
/* The isfmsg2 stem contains additional descriptive error messages*/
  do ix=1 to isfmsg2.0
    Say ix ":" isfmsg2.ix
  end
  Say " "
return 0
```

SDSF REXX - Display JES3 DJC nets and the DJC net status

Example B-4 SDSF REXX - Display JES3 DJC nets and the DJC net status

```
/* Rexx exec for JES3 DJC net displays using SDSF services           */
/* e&oe - minimal error checking - SDSF REXX sample only             */

/* Function:                                                         */
/*     Checks for JES3 environment - If none, EXIT                   */
/*     Re-invoke this REXX with ISPF APPLID "USRT" (as required)     */
/*     Check z/OS release level => "01.13.00" required - If not, EXIT */
/*     Set up an ISPF table for JES3 *I N command response           */
/*     Issue *I N command using SDSF ISFSLASH                        */
/*     - Read *I N command response (IAT8578 messages) from SDSF ULOG */
/*     - Add IAT8578 message data into ISPF table                    */
/*     - ISPF display the IAT8578 message table (until END requested) */
/*     For valid table selections (B, D, and S show DJC data)        */
/*     - Issue *X DISPDJC command using SDSF ISFSLASH                */
/*       (*X DISPDJC command spins the DJC net data to JES3 job zero) */
/*     - Read, ISFLOG, the OPERLOG, locate IAT6306 message for the   */
/*       DISPDJC job and parse out the DISPDJC's job number          */
/*     - Access using ISFEXEC JO spool data set list and find the    */
```

```
/*       entry for the DISPDJC job output                              */
/*     - Spool allocate using ISFACT JO NP SA the DISPDJC data set    */
/*     - Allocate a temporary data set and copy the DISPDJC into it   */
/*       (ISPF browse and view do nor work with spool data set)       */
/*     - ISPF VIEW the temporary data set - *X DIDPDJC,NET= output    */
/*     - Purge using the ISFACT JO NP P DISPDJC spool output          */
/*     - Re-invoke the ISPF IAT8578 message table                     */
/*                                                                    */
/* Rexx variables plib and vio, below, are installation dependent!    */

Parse upper arg arg
parse source . . src sdd sds .

plib = "VAINI.U.PANELS"  /*<===UPDATE as needed - Application panels  */
vio  = "VIO"             /*<===UPDATE as needed - UNIT name for VIO   */

/* Checks for JES3 environment                                        */
rc=isfcalls( ON )
jt = ""
jg = ""
Address SDSF "ISFEXEC WHO"
  do i=1 to  ISFRESP.0                          /* Check WHO response */
    parse var isfresp.i keyw "=" kval
    if keyw = "JESTYPE" then jt = kval
    if keyw = "JESNAME" then jn = kval
    if keyw = "GLOBAL"  then jg = kval
  end
/* If JES3 is not our JES, EXIT ....                                  */
if jt<>"JES3" then do; z=sayit(jt "not JES3 - Exiting..."); exit 16; end
/* If ISPF is not our environment - EXIT...                           */
if sysvar(SYSISPF)<>"ACTIVE" then do; say "ISPF not active - Exiting..."
  exit 16; end
rc=isfcalls( OFF )
/* Invoke this ISPF rexx with APPLID "USRT" - PF setting will change  */
address "ISPEXEC" "VGET (ZAPPLID) SHARED"
if zapplid <> "USRT" then do
  address "TSO" ,
  "ALTLIB  ACT APPL(CLIST) DD("sdd")"           /* APPL level CLISTs  */
  address "ISPEXEC" "SELECT CMD("src  arg") NEWAPPL(USRT) PASSLIB"
  address "TSO" ,
  "ALTLIB  DEACT  APPL(CLIST)"
  return
end
/* Now running under APPLID "USRT" - Start tracing if requested       */
z = wordpos("DB",arg)
if z ¬= 0 then do                               /*Rexx trace requested*/
  x = MSG("ON")
  Trace "I"                                     /* Trace...           */
  arg = delword(arg,z)
end
else Trace "O"                                  /* No tracing         */
/* Change some PF key settings - Scroll - PF 10 Left; PF 11 Right     */
ZPF10 = "ULEFT"; ZPF11 = "URIGHT"; ZEDLMSG = "" ZEDSMSG = ""
address "ISPEXEC" "VPUT (ZPF10 ZPF11) PROFILE "
/* Check z/OS release level => "01.13.00" required                    */
z = MVSVAR('SYSOPSYS')
parse var z . rel .
if rel < "01.13.00" then do
   ZEDLMSG = src "SDSF rexx is not supported in" z"!"
   Address "ISPEXEC" "SETMSG MSG(ISRZ001)"
```

```
   exit 16
end
/* Local ISPF table display panel USRHOQW title data                 */
Owner = " "; JOBNAME = " "; JobID = " "; dsd = " "; Q = "DJC"
/* Make local ISPF panels available                                  */
address "ISPEXEC"
"CONTROL ERRORS RETURN "
"LIBDEF ISPPLIB DATASET ID('"plib"') "         /* Activate APPL panels*/
ng = 0                                         /* Set GO switch to go */
/* Invoke JES3 DJC processing                                        */
z = donets()                                   /* Do JES3 DJC nets    */
address "TSO" ,
"ALTLIB  DEACT  APPL(CLIST)"
address "ISPEXEC"
"LIBDEF ISPPLIB"
"LIBDEF ISPLLIB"
return 0
/* Propcess JES3 DJC nets                                            */
donets:
Address "ISPEXEC" "VGET (ZSCREEN)"
/* Invoke subroutine to set prefix for JES3 commands & SDSF console   */
z = jescm()
if z <> 0 then return z                        /* Failure - EXIT...   */
/* Ask ISPF to tolerate goofs...                                     */
Address "ISPEXEC" "CONTROL ERRORS RETURN"
Address "TSO"                                  /*Default REXX addr env*/
/* Set ISPF initial table display values                             */
TNW = "TNW" || ZSCREEN ; SEL = 0 ; np = "" ; ps = ""
cmd = gpfx'I N'                                /*JES3 *I N command    */
again = 1
hd   = "NP   IAT8578 messages for the JES3 DJC job nets  "
cvar = "_NP  ~JOBNET                                     "
tnam = "NP JOBNET "
/* Invoke ISPF table create for the JES3 DJC nets                    */
z= tbc()
/* Show JES3 DJC NETs loop                                           */
Do while again
  ng = 0; errorm. = ""; shortm  = ""           /* Init some variables */
  z = entcmd()                                 /* Issue *I N command  */
  again = 0                                    /* Set tbl disp switch */
  sds. = ""; sds.0 = 0
  z = tbt()                                    /* Set tbl disp to top */
  z = tbd()                                    /* Display DJC table   */
end
return rc                                      /* All done here...    */
/* Subroutine to issue an ISPF message                               */
sayit:
  parse arg xyz
  if xyz ¬= "" then do
    zedlmsg = xyz
    If sysvar(SYSISPF) = "ACTIVE" then do
      zedsmsg = ""
      address "ISPEXEC" "SETMSG MSG(ISRZ000)"
    end
    else Say zedlmsg
  end
  return 0
/* Subroutine to create an ISPF table                                */
tbc:
Address "ISPEXEC"
```

```
  "TBCREATE" TNW "NAMES("tnam") NOWRITE REPLACE"
Address "TSO"
return rc
/* Subroutine to add a row into the ISPF table                        */
tba:
Address "ISPEXEC"
  NP = ""
  "TBADD" TNW
Address "TSO"
return rc
/* Subroutine to set the ISPF table display to TOP                    */
tbt:
Address "ISPEXEC"
  "TBTOP" TNW
 Address "TSO"
return rc
/* Subroutine to display the ISPF table and process row actions       */
tbd:
TRC = 0; TOP = 0; CRW = 1; SEL = 0
Address "ISPEXEC"
Do while TRC < 8                                /*Loop until END hit   */
  If again = 1 then  Leave
  /* ISPF display DJC NET table                                       */
  "TBTOP" TNW
  "TBSKIP" TNW "NUMBER("TOP")"
  src = RC
  "TBDISPL" TNW "PANEL(USRHOQW) POSITION(TBL) CSRROW("CRW") AUTOSEL(NO)" ,
                "ROWID(RID)"
  TRC = RC                                      /* Save tbl disp rc    */
  zedlmsg = ""
    /* Process first selected table row, if any non-blank action set  */
    If ZTDSELS <> 0 Then do                     /* Rows to process     */
      CRW = TBL                                 /* Save table position */
      if np <> "" then z = selpro()             /* Process action      */
      NP = ""                                   /* Clear old action    */
    End
    /* Process remainin pending table row selections                  */
    Do while ZTDSELS > 1
      "TBDISPL" TNW "POSITION(TBL) CSRROW("RID") ROWID(RID)"
      MRC = RC
      CRW = TBL
      if np <> "" then z = selpro()             /* Process action      */
      np = ""
    End
  if zcmd <> " " then z = selpro()
  If TRC = 8 then Leave
End
/* All done - Exiting... */
  "TBCLOSE" TNW
Address "TSO"
return rc
/* Subroutine to process one table row section                        */
selpro:
  zcmdmsg=""
  select
    when (np = "S" | np = "B"| np = "D") & ng = 0  then do
      "CONTROL DISPLAY SAVE"
      nid = word(jobnet,1)                      /*Invoke..             */
      z = vjOnet(nid)                           /* *X DISPDJC process  */
      "CONTROL DISPLAY RESTORE"
```

```
    end
    when (np = "H" | np = "R" | np = "C") & ng = 0 then do
      zedlmsg = zedlmsg "NP="np "TBD"          /* Set NP TBD message  */
    end
    when zcmd <> "" then do
      zcmdmsg = "C=>"zcmd "TBD"                /* Save zmcd message   */
      zcmd = ""
    end
    otherwise do
      if np <> "" then zedlmsg = zedlmsg "NP="np" ????"
      "TBDISPL" TNW "POSITION(TBL) MSG(ISRZ001) CSRROW("RID") ROWID(RID)"
      XRC = RC
      Z = ztdsels
      zedlmsg = ""
      if np <> "" then z = selpro()
    end
  End
  /* Issue message for command line data - TBD - to_be_defined        */
  if (zcmdmsg<>"" & ZTDSELS<=1) then do;zedlmsg=zcmdmsg;zcmdmsg="";end
  if zedlmsg <> "" then do
    "SETMSG MSG(ISRZ001)"
  end
return rc
/* Subroutine to use SDSF ISFSLASH for a JES3 command                */
entcmd:
  rc=isfcalls( ON )
  isfcons =    userid() || zscreen
  Address SDSF "ISFSLASH '"cmd"' (WAIT)"
  lrc=rc
  /* Find and extract *I N command output from the SDSF ULOG          */
  if lrc<>0 then do
    call msgrtn                                /* List error messages */
    ng = 1                                     /* Set failure         */
    return  16
  end
  /* Process ULOG for ISFSLASH response                               */
  jobnet = "No DCJ jobs or nets"
  do ix=1 to isfulog.0
    if pos("IAT8578",isfulog.ix)<>0 then do
      parse var isfulog.ix . "NET-ID=" jobnet
      np = " "
      z = tba() /* Insert IAT8578 message text into the ISPF table */
    end
  end
  if jobnet = "No DCJ jobs or nets" then do; z = tba(); ng = 1; end
  rc=isfcalls( OFF )
  return lrc
  /* Subroutine to list SDSF error messages                           */
  msgrtn:
    /* The isfmsg variable contains a short message                   */
    if isfmsg <>"" then jobnet = isfmsg
    z = tba()                                  /* TBADD message       */
    /* The isfmsg2 stem contains additional descriptive error messages*/
    do ix=1 to isfmsg2.0
      jobnet = isfmsg2.i
      z = tba()                                /* TBADD message       */
    end
  return 4
  /* Subroutine to *X DISPDJC NET= for argument DJC NET               */
  /* *X DISPDJC NET= command output is spooled to JES3 JO             */
```

```
/* The *X DISPDJC NET= command response is read from OPERLOG        */
vj0net:
  rc=isfcalls( ON )
  lst=time("N")
  /* SDSF ISFSLASH *X DISPDJC NET=net_id command                    */
  cmd=gpfx"X DISPDJC NET="nid
  isfcons =    userid() || zscreen
  Address SDSF "ISFSLASH '"cmd"' (WAIT)"     /* Issue JES3 command  */
  isfsysid=MVSVAR('SYSNAME')                 /* Sysname filter      */
  isflogstarttime=lst                        /* Log read start at   */
  isflinelim=1000                            /* Read record limit   */
  Address SDSF "ISFLOG READ TYPE(OPERLOG)"
  /* Find JES3 job number for *X DISPDJC job                        */
  do ix=isfline.0 to 1 by -1                 /* Scan log records    */
    if pos("IAT7450",isfline.ix)<>0 then do
      parse var isfline.ix "(" jid ")" .
      leave
    end
  end
  /* Find JES3 job 0 data sets using SDSF ISFEXEC J0 data           */
  Address SDSF "ISFEXEC J0 (ALTERNATE DELAYED)"
  nr = isfrows
  /* Locate *X DISPDJC output data set index in ISFEXEC J0 output   */
  /* using the job number found in the IAT7450 response message.    */
  do i=1  to nr
    if name.i = jid then do; leave ; end
  end
  /* Find *X DISPDJC output data set using the ISFACT J0 NP SA      */
  /* SA (browse allocate) returns variables for DD- and DSNAME.     */
  Address SDSF "ISFACT J0 TOKEN('"TOKEN.i"') PARM(NP SA)"
  sarc = rc
  ndd = isfddname.0
  /* Allocate a temporary data set for the copy of J0 data          */
  /* ISPF browse does not work with spool data sets...              */
  do j=1 to ndd
    address ISPEXEC "VGET (ZSCREEN)"
    bdd = "NETJ0"zscreen
    address TSO ,
    "alloc dd("bdd") spa(1 5) cyl new recf(v a) lrecl(255) uni("vio") reu"
    /* Copy J0 spool data into the temporary data set               */
    address TSO ,
    "EXECIO * DISKR" isfddname.j "(STEM line. FINIS"
    y = line.0+1
    z = value("line."y,"")
    line.0 = y
    address TSO ,
    "EXECIO * DISKW" bdd "(STEM line. FINIS"
    /* ISPF VIEW the temporary data set                             */
    address ISPEXEC "LMINIT DATAID(DID) DDNAME("bdd")"
    lin = rc
    if  lin = 0 then do
      address  ISPEXEC "VIEW   DATAID("DID")"
      address  ISPEXEC "LMFREE DATAID("DID")"
    end
    else say "LMINIT" bdd "RC =" lin zerrlm
    "free dd("bdd")"
  end
  /* PURGE the J0 *X DISPDJC output spool data sets...              */
  Address SDSF "ISFACT J0 TOKEN('"TOKEN.i"') PARM(NP P)"
  rc=isfcalls( OFF )
```

```
  return rc
/* Subroutine to set prefix for JES3 global commands & SDSF console   */
  jescm:
    isfcons = userid() || zscreen               /* Set console name   */
    gpfx = "*"
    if jg =  "" then z = sayit("JES3 sysplex scope prefix set to '*'")
    else gpfx = "RO" jg",*"                     /* Prefix = "RO gbl,*"*/
  return 0
```

Table display panel for DJC nets

Example B-5 USRHOQW - table display panel for DJC nets

```
)ATTR
 _ TYPE(INPUT) INTENS(HIGH)  PADC(NULLS) JUST(LEFT) CAPS(ON)
 % TYPE(OUTPUT) INTENS(HIGH) PAD(' ')  JUST(ASIS) CAPS(ON)
 ~ TYPE(OUTPUT) INTENS(HIGH) PAD(' ')  JUST(ASIS) CAPS(ON) COLOR(YELLOW)
 ? TYPE(OUTPUT) INTENS(HIGH) PAD('-')  JUST(ASIS) CAPS(ON)
 ¢ TYPE(OUTPUT) INTENS(HIGH) PAD(' ')  JUST(LEFT) CAPS(ON)
 @ TYPE(OUTPUT) INTENS(HIGH) PAD(' ')  JUST(ASIS) CAPS(OFF)
 + TYPE(OUTPUT) INTENS(LOW)  PAD(' ')  JUST(ASIS) CAPS(OFF)
 \ TYPE(OUTPUT) INTENS(LOW)  PAD(' ')  JUST(LEFT) CAPS(ON)
 | TYPE(TEXT) INTENS(HIGH)   PAD(' ')  JUST(ASIS) CAPS(ON)
 ! TYPE(TEXT) INTENS(LOW)    PAD(' ')  JUST(ASIS) CAPS(ON)
)BODY  ASIS WIDTH(&ZSCREENW) EXPAND(//)
|-----------------------  JES3|&Q  |-------------------/-/----------------------
|C =>_ZCMD                           / /                          |S =>_SCR |
@HD                                                                            |
?ZX                                 / /                                        |
)MODEL
&CVAR
)INIT
  &ZSCRNAME = &Q
  VPUT (ZSCRNAME) SHARED
  &ZX = '----'
)REINIT
)PROC
)END
/*--+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---*/
/* USRHOQW - JES3 DJC Net display sample */
```

SDSF in batch example

The TSO terminal monitor program, in batch, is used to invoke a REXX exec (BATSD), which invokes SDSF to display the status (ST) panel for all JES3 jobs. The output of the SDSF program is directed to a DASD data set. The ISFOUT data is read into stem variables, which are filtered to find jobs in the JES3 MDS queues.

Tip: The terminal monitor program (TMP) provides an interface between the user, command processors, and the TSO/E control program. It obtains commands, gives control to command processors, and monitors their execution.

Example B-6 SDSF in batch execution JCL

```
//RXBATSD  JOB (999,POK),EXPERT,MSGLEVEL=1,MSGCLASS=A,NOTIFY=&SYSUID
//TMP    PROC
//TMP    EXEC PGM=IKJEFT01,DYNAMNBR=99
//SYSPROC  DD DSN=VAINI.U.CLIST,DISP=SHR
//SYSTSPRT DD SYSOUT=*
```

```
//       PEND
//       EXEC TMP
//SYSTSIN  DD *
  BATSD  /* REXX to execute SDSF in batch TSO TMP*/
  BATSDR /* SDSF REXX to execute under batch TSO TMP*/
```

REXX to execute SDSF in TSO batch

Example B-7 SDSF in batch - BATSD REXX

```
/* Rexx    Sample JCL from the z/OS SDSF Operation and Customization document
            //       EXEC PGM=SDSF,PARM='32,500'
            //ISFOUT DD SYSOUT=*
            //ISFIN  DD *
*/
"alloc dd(isfout) uni(sysallda) spa(1 1) cyl new reu"
"alloc dd(isfin)  uni(vio) spa(1 1) tra new reu" ,
  "recf(f) lrec(80) dsor(PS)"
isfi. = ""               /* Build ISFIN data              */
isfi.1 = "PREFIX *"      /* Set PREFIX for ST             */
isfi.2 = "OWNER *"       /* Set OWNER for ST              */
isfi.3 = "ST"            /* ST command                    */
isfi.4 = "? "            /* Request alternate fields      */
isfi.5 = "++ALL"         /* Set everything action         */
isfi.0 = 5
"EXECIO * DISKW ISFIN (FINIS STEM ISFI." /* Update ISFIN  */
"call  *(SDSF)  '++32,500'"              /* Invoke SFSF   */
"EXECIO * DISKR ISFOUT (FINIS STEM SDVAR." /*Read ISFOUT  */
sayit = 0                /* Contol switch                 */
z = substr(date(s),1,2) || date(j)
say "Present time:" substr(z,1,4) || "." || substr(z,5)  date(j) time()
do i = 1 to sdvar.0      /* Loop through SDSF output      */
   if pos('SDSF STATUS DISPLAY',sdvar.i) <> 0 & sayit = 0 then do
     j = i + 2
     tp = pos("Rd-Time",sdvar.j) /* Find an alternate fld */
    if tp > 0 then do   /* Process alternate field data   */
       sayit = 1        /* Set switch = start processing  */
       jp = pos("JOBNAME",sdvar.j) /* Find field offsets  */
       qp = pos("Queue",sdvar.j)
       pp = pos("PhaseName",sdvar.j)
       dp = pos("Rd-Date",sdvar.j)
     end
   end
   if sayit then do     /* Find JES3 jobs in MDS queues   */
     if pos('SDSF STATUS DISPLAY',sdvar.i) <> 0 then iterate
     if pos('COMMAND INPUT',sdvar.i) <> 0 then iterate
     if substr(sdvar.i,jp,15) = " " then iterate
     qinf = substr(sdvar.i,qp,10)
     pinf = substr(sdvar.i,pp,21)
     if qinf = "SETUP" | wordpos("MDS",pinf) <> 0 then do
       jinf = substr(sdvar.i,jp,25)
       pinf = substr(sdvar.i,pp,21)
       tinf = substr(sdvar.i,tp,8)
       dinf = substr(sdvar.i,dp,8)
       say left(jinf,25) left(qinf,10) left(pinf,21) dinf right(tinf,8)
     end
   end
end
```

Sample output from the BATSD REXX execution is in Example B-9 on page 1010. The REXX program in Example B-7 on page 1009 and the SDSF REXX program in Example B-8 create the same output.

SDSF REXX to execute under batch TSO TMP

Example B-8 SDSF REXX in batch

```
/* Show some SDSF REXX basic features - List JES3 MDS alloc queue */
/* for the user who invokes this REXX exec                        */
trace O
zrc=isfcalls("ON")        /* Add the SDSF host command environment*/
isfdest   = "*"
isfowner  = "*"
isfprefix = "*"
isfsysname = "*"
isffiltermode = "or"
isffilter = "PHASENAME EQ 'AWAIT RES ALLOC' PHASENAME EQ 'MDS ERROR'"
isffilter = isffilter "PHASENAME  EQ UNA*"
/*  isffilter assignments that do not work
isffilter = "PHASENAME EQ 'AWAIT RES ALLOC' +PHASENAME EQ 'MDS ERROR'"

isffilter = "PHASENAME EQ 'AWAIT RES ALLOC'"
isffilter = "+PHASENAME EQ 'MDS ERROR'"
*/
Address SDSF "ISFEXEC ST (ALTERNATE DELAYED)"
do i=1 to JNAME.0               /* Loop for all rows returned     */
  say left(jname.i,8) left(jobid.i,8) left(ownerid.i,7) left(queue.i,10) ,
    left(phasename.i,21) right(dater.i,8) right(timer.i,8)
end
zrc=isfcalls("OFF")       /* Delete SDSF command environment      */
exit O
```

Example B-9 SDSF in batch - BATSD REXX and BADSDR SDSF REXX sample output

```
MDSWAIT3 JOB14444 VAINI   EXECUTION  MDS ERROR            2009.117 12:08:49
NOUNITS  JOB14506 VAINI   EXECUTION  MDS ERROR            2009.120 13:23:19
BECKER1  JOBO4066 BECKER  SETUP      AWAIT RES ALLOC      2006.130 11:29:11
MDSWAIT  JOB14437 VAINI   SETUP      AWAIT RES ALLOC      2009.117 11:22:00
MDSWAIT0 JOB14440 VAINI   SETUP      AWAIT RES ALLOC      2009.117 11:52:08
MDSWAIT1 JOB14441 VAINI   SETUP      AWAIT RES ALLOC      2009.117 11:52:14
MDSWAIT2 JOB14442 VAINI   SETUP      AWAIT RES ALLOC      2009.117 11:52:22
NOVOLS   JOB14503 VAINI   SETUP      UNAVAIL VOL          2009.120 13:06:42
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 1012. Note that some of the documents referenced here may be available in softcopy only.

- *ABCs of z/OS System Programming Volume 9*, SG24-6989-05
- *ABCs of z/OS System Programming Volume 10*, SG24-6990-03
- *ABCs of z/OS System Programming Volume 6*, SG24-6986-00
- *ABCs of z/OS System Programming Volume 5*, SG24-6985-02
- *ABCs of z/OS System Programming Volume 3*, SG24-6983-03
- *ABCs of z/OS System Programming Volume 2*, SG24-6982-02
- *ABCs of z/OS System Programming Volume 1*, SG24-6981-02
- *ABCs of z/OS System Programming Volume 11*, SG24-6327-01
- *ABCs of z/OS System Programming Volume 8*, SG24-6988-00
- *ABCs of z/OS System Programming Volume 7*, SG24-6987-01
- *ABCs of z/OS System Programming Volume 12*, SG24-7621-00
- *ABCs of z/OS System Programming Volume 4*, SG24-6984-00

Other publications

These publications are also relevant as further information sources:

- *z/OS MVS Planning: Operations*, SA22-7601
- *z/OS MVS System Codes*, SA22-7626
- *z/OS MVS JCL Reference*, SA22-7597
- *Network Job Entry Formats and Protocols*, SA22-7539
- *z/OS SDSF Operation and Customization*, SA22-7670
- *z/OS JES3 Initialization and Tuning Reference*, SA22-7550
- *JES3 Initialization and Tuning Guide*, SA22-7549
- *z/OS JES3 Customization*, SA22-7542
- *z/OS JES3 Diagnosis*, GA22-7547
- *z/OS JES3 Diagnosis Reference*, GA22-7548

Online resources

These Web sites are also relevant as further information sources:

- SDSF home page: usage tips, presentations, as well as a wizard to help you enable the sysplex support can be found at:

  http://www.ibm.com/servers/eserver/zseries/zos/sdsf

- The latest edition of *z/OS SDSF Operation and Customization*, SA22-7670 is available at:

  http://publibz.boulder.ibm.com/epubs/pdf/isf4cs90.pdf

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

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Redbooks

ABCs of z/OS System Programming Volume 13

(1.5” spine)
1.5”<-> 1.998”
789 <->1051 pages

ABCs of z/OS System Programming Volume 13

JES3 internals and externals

JES3 initialization statements, JES3 SDSF

JES3 operator commands

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

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

JES3 considers job priorities, device and processor alternatives, and installation-specified preferences in preparing jobs for processing job output. This IBM Redbooks publication describes a JES3 environment that includes the following:

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

This book will help you install, tailor and configure a JES3 system.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-7717-01 ISBN0738436259