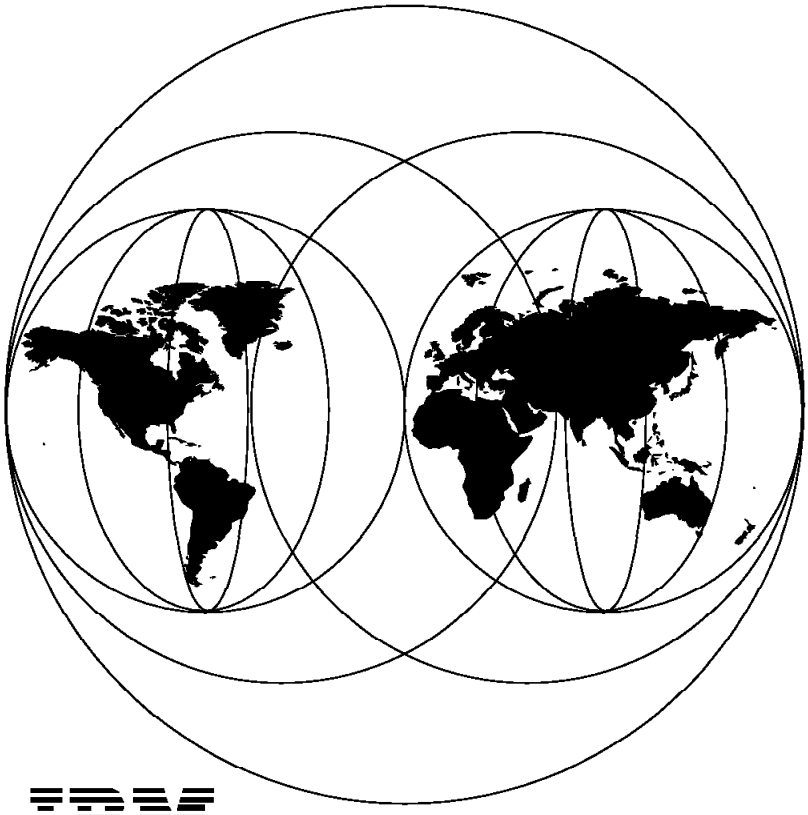
SG24-2067-00

OS/390 Release 3 Implementation

December 1997



IBM

International Technical Support Organization
Poughkeepsie Center

International Technical Support Organization

SG24-2067-00

OS/390 Release 3 Implementation

December 1997

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix C, “Special Notices” on page 221.

First Edition (December 1997)

This edition applies to Release 3 of OS/390, Program Number 5645-001 and to all subsequent releases and modifications until indicated in new editions.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
522 South Road
Poughkeepsie, New York 12601-5400

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

Contents

Figures

Tables

Preface

This redbook describes the new functions available with OS/390 Release 3. These new functional enhancements are for the following components: system logger, OpenEdition, Process SYSOUT interface, workload manager, linklist, APPC/MVS, SDSF, RACF, and the communications server. Additionally, new functions are available as follows: IP PrintWay, generic resource support for TSO/E and APPC/MVS, Recoverable Resource Management Services (RRMS), and a sample functional subsystem.

This redbook will help you to install, tailor and configure the new functions of OS/390 Release 3.

The Team That Wrote This Redbook

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

Bob Haimowitz	ITSO Poughkeepsie
Paul Rogers	ITSO Poughkeepsie
Juha Vainkainen	IBM Finland
Eve Bye	IBM Australia
Benno Aladjem	IBM France
Redelf Janßen	IBM Germany

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 239 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Home Pages at the following URLs:

  For Internet users `http://www.redbooks.ibm.com`
  For IBM Intranet users `http://w3.itso.ibm.com/redbooks`

- Send us a note at the following address:

  `redbook@vnet.ibm.com`

Chapter 1. OS/390 Release 3

With OS/390, IBM integrates a set of MVS base, open, client/server and applications-enabling functions and offers this set as a single product orderable with one program number.

Within OS/390, the levels of all products reflect the level of the OS/390 product itself. Even the word “product” has new meaning; for this reason the products that make up the base of OS/390 are called *base elements* or simply *elements*. Optional products that have an affinity to the elements are called *features*.

The OS/390 base elements and optional features include a set of services that provide an integrated set of MVS, UNIX, LAN, distributed computing, and application enablement services, as shown in Figure 1.

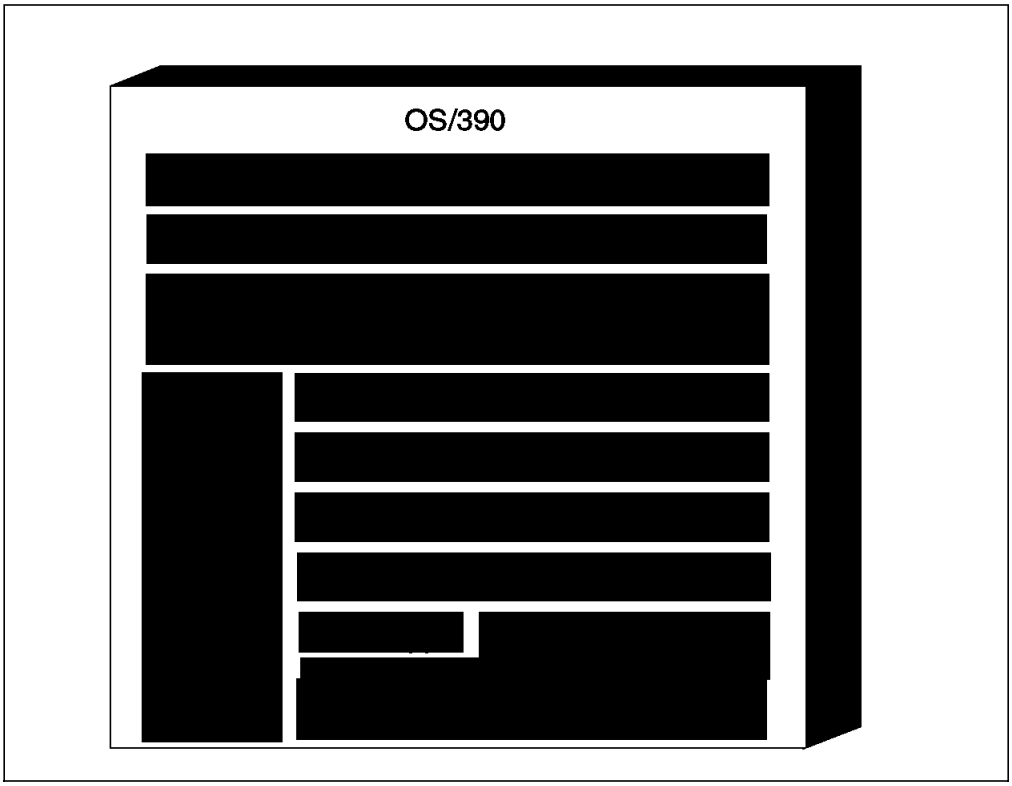


Figure 1. Overview of OS/390 Release Structure

In addition to the base elements, OS/390 has optional features that have an affinity to the base.

There are two types of features:

- One type is shipped with OS/390 whether you order the feature or not. These features support dynamic enablement. If you order the feature they are shipped enabled, otherwise they are shipped disabled. If you later want to use a disabled feature, let IBM know and enable it dynamically through parmlib.
- The second type of feature does not support dynamic enablement and is not shipped with OS/390 unless you specifically order it in addition to the base.

Most of the OS/390 elements are based on products that have been available for some time and you may have been running some of them. To distinguish the element from its prior or preceding product, the term *root product* can be used. The OS/390 level of an element can be any of the following:

- A repackaging of the root product
- The root product with some additional function
- The root product unchanged

This chapter provides an overview of the new function included with OS/390 Release 3.

1.1 Internet Connection Secure Server

With OS/390 Release 3, the IBM Internet Connection Secure Server Version 2 Release 1 is integrated into the base.

Note: Version 2 Release 2 of this product is shipped on a separate tape in the box with the OS/390 order.

The benefits of the IBM Internet Connection Secure Server for OS/390 are that you have S/390 security, the utilization of large storage capacity, use of centralized skills, a single point of entry and control, consolidation of multiple Web sites, and secure Internet transactions.

Also shipped with an OS/390 Release 3 order is the Internet BonusPak II, which includes samples HTML pages and CGI programs to get you started with the Internet Connection Secure Server.

1.2 Security Server

The OS/390 Security Server is an integrated optional feature which includes RACF and the OpenEdition DCE Security Server at the OSF DCE 1.1 level. Now, non-DCE applications can use DCE security through the Generic Security Services API. This provides the following capabilities:

- Allows a single sign-on to the RACF and DCE domain
- Gives enterprise-wide security
- Enables porting of applications that use DCE security services to OS/390
- Allows DCE application services to associate a DCE user with a corresponding RACF-defined user

The Security Server includes the following enhancements in OS/390 Release 3:

- Support for OS/390 UNIX OE permanent kernel
- Support for VM SFS profile administration from OS/390 when sharing a RACF database with VM
- New SOM grouping class (GSOMOBJ) to allow grouping profiles
- RACF remote sharing facility (RRSF) enhancements
- New exit point for RACF TSO commands
- Creator user ID not automatically added to access list

- Enhanced search algorithm for CICS 4.1 users with large numbers of RACLIST profiles

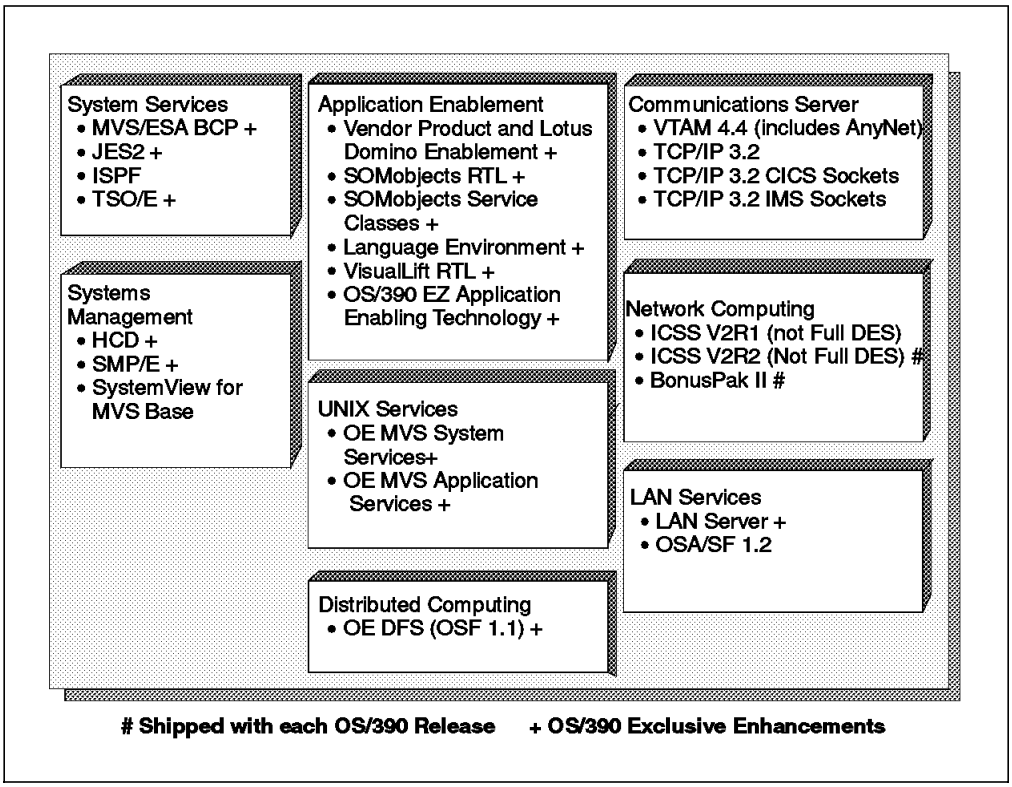


Figure 2. Overview of OS/390 Release 3 Enhancements

1.3 System Services

The OS/390 Release 3 System Services enhancements are the primary focus of this book. These enhancements are the new functional changes to the MVS components of OS/390. The enhancements, as shown in Figure 2, change the following components:

- MVS system logger
- OpenEdition services
- SYSOUT application programming interface (SAPI) support for obtaining SYSOUT from JES2 and JES3
- A sample functional subsystem (SFSS) shipped in SYS1.SAMPLIB
- TSO/E generic resource support (TSO/GR) in a Parallel Sysplex
- New workload manager (WLM) services
- Dynamic linklist
- XCF enhancements for signalling services
- APPC/MVS API trace support for debugging applications
- JES2 64Kb jobs number support
- IOS component tracing of dynamic configuration changes
- ISPF enhancements for improved workstation support and security

1.3.1 OS/390 System Logger Enhancements

The OS/390 system logger was introduced with MVS/ESA 5.2.0 and is enhanced in OS/390 Release 3 to include the following functions:

- The system logger can automatically delete log data whose user-defined retention period has expired.
- The logger provides archival support such that log data can remain in the log stream for the length of time required by the customer, thus eliminating the costly functions of moving the log data. This admits a common mechanism for handling log data archival across various transaction managers that use the OS/390 system logger. In addition, the 168 log data set limit per log stream is removed.
- The OS/390 system logger will dynamically manage a structure′s entry-to-element ratio by periodically checking to determine if the actual usage is significantly different from the structure allocation and if so, it attempts to alter the structure′s entry-to-element ratio to match the current usage.
- The system logger provides the ability to allocate staging data sets during rebuild failures, to dump the dataspace log data relating to the log stream. This ensures that a non-volatile copy of the log data is maintained and protected between the failure and the correction.
- Support for remote site recovery allows log stream data at a local (or primary) site to be duplicated at a remote or secondary site. The remote site becomes the primary site when a disaster occurs at the primary site. The OS/390 system logger allows transmitted log data to be written into a log stream at the remote site with a block ID and GMT time stamp identical to the block ID and GMT time stamp assigned at the local site.

IBM plans to provide in the OS/390 Release 3 time frame, as an integral part of the system, Syncpoint Manager services (transactional coordination of two-phase commit protocols) together with Recoverable Resource Managers′ services. These services, called Recoverable Resource Management Services (RRMS), can provide transactional processing capabilities to all of OS/390′s application environments and support both local and distributed commit scopes. RRMS also coordinates recovery and resynchronization of distributed work with the participating distributed communication managers and the participating client/server platforms.

The OS/390 system logger is used by RRMS for transaction logging and to record resource manager information necessary to recover incomplete transactions on eligible systems in the Parallel Sysplex.

1.3.2 OpenEdition Services Enhancements

OpenEdition services now provide the following enhancements in OS/390 Release 3:

- OpenEdition System Services (FIMD HOMxxxx) has been merged with the Base Control Program (BCP) and is now part of the BCP (FMID HBB6603).
- The OMVS address space is started automatically at IPL by means of the OMVS= statement in the IEASYSxx parmlib member.
- BPXOINIT is the started procedure that runs the initialization process. BPXOINIT is also the jobname of the initialization process.

- The START OMVS and STOP OMVS commands are no longer supported. You are able to make some configuration modifications active by using SETOMVS or SET OMVS. If a full restart of OpenEdition services is required, the system must be re-IPLed.
- Activation of OpenEdition services is available in two levels *minimum mode* or *full-function mode*. Mimimum mode is intended for installations that do not intend to use OpenEdition services.
- A temporary file system (TFS) is now available. A TFS is an in-memory physical file system that supports in-storage mountable file systems. Because it is an in-storage file system, the TFS delivers high speed I/O.

This redbook provides a brief overview of installation and customization tasks for OpenEdition Services. The redbook *OS/390 Release 3 OpenEdition Installation and Customization*, SG24-2087, provides step-by-step descriptions on how to install, customize, and use the OpenEdition product set.

1.3.3 Process Sysout Interface Enhancements

The process sysout interface (PSO) is enhanced through a new SSI interface call to select and process sysout (print data) data sets from JES. This general use programming interface (GUPI), called SYSOUT Application Programming Interface (SAPI), provides a platform for application development for on-demand SYSOUT viewing, printing, and modification that compliments the traditional role of JES printing.

1.3.4 Sample Functional Subsystem

A sample functional subsystem (FSS) is provided in SYS1.SAMPLIB. The sample FSS illustrates the use of the functional subsystem interface (FSI). It is a working example of how one might implement a functional subsystem. This example can be used as a starting point for application programmers to develop their own FSS applications, which can include functions such as driving output devices, plotters, microfiche writers, or anything else the application wishes to do with the data.

1.3.5 TSO/E Generic Resource Support

In previous releases, a TSO/E instance was identified by a unique application name. To start a session with TSO/E, the TSO/E user specified this unique name together with the TSO/E user ID and other logon parameters. This, however, requires that this particular TSO/E system be up and running. If the system is down, a session cannot be started with it until the system comes up. In addition, there is no mechanism to evenly distribute sessions across a group of TSO/Es.

In a Parallel Sysplex running OS/390 Release 3, TSO/E on one OS/390 system can be known by the same generic name as TSO/E on any other OS/390 system in the sysplex. Any of the TSO/Es sharing the same generic name can be concurrently active. This provides both increased TSO/E availability and workload balancing.

The VTAM generic resource support to TSO/E, which is called TSO/E generic resource (TSO/GR), increases TSO/E availability because all TSO/Es can be accessed by the same generic name. If a particular TSO/E system is down, a session request using the generic name can still be satisfied by another TSO/E system with the same generic name. The MVS workload manager makes efficient use of system resources by selecting the TSO/E system, based on

balanced load distribution, from among all TSO/E systems with the same generic name.

TSO/GR support also includes reconnection to the original system using the generic TSO/E name if the TSO/E user loses connection to the system and then logs on again before a system-specific reconnect interval has expired.

1.3.6 Workload Manager (WLM) Support

The following items are enhancements to workload manager for OS/390 Release 3:

In support of managing resources:

- Sysplex I/O priority management
- Business unit-of-work management
- Server address space management

In support of work environments that now exploit workload management:

- DB2 stored SQL procedures support
- DSOM sysplex workload management
- DCE sysplex workload management
- TSO sysplex workload management
- APPC/MVS Support for generic resources

New workload manager delays are available as RMF expands WLM-related reports of Monitor III and the postprocessor to display these values. In addition, DASD response time values on service-class level are shown.

1.3.7 Dynamic Linklist Enhancements

Enhancements to LNKLST processing are provided to allow the LNKLST to be changed without requiring an IPL. Having the ability to dynamically modify the LNKLST can reduce the number of planned outages, allowing installations to make new software available to their customers without disrupting their computing services.

1.3.8 XCF Enhancements

In OS/390 Release 3, the cross-system coupling facility (XCF) system initialization processing is enhanced in many cases to eliminate the need for operator intervention when a system in the sysplex has been re-IPLed without the prior instance of that system first having been partitioned from the sysplex. This enhancement simplifies the operational procedures associated with re-IPLing a system in a sysplex.

The XCF signalling service provides a basic high performance message passing mechanism between members of an XCF group. The XCF signalling services are extended to provide several higher level signalling functions:

- Ordered message delivery
- Broadcast to members of a group
- Consolidation of response message(s)
- Queue messages for busy conditions
- Message timeout support

These enhancements reduce the cost of using the XCF signalling services, increasing the attractiveness of the Parallel Sysplex platform for software developers.

Also, whenever a system joins or leaves a sysplex, XCF provides a descriptive ENF signal on all other active systems in the sysplex. This enhancement improves the capabilities available to monitoring and automation programs which operate in a sysplex environment.

1.3.9 Advanced Program-to-Program Communications/MVS (APPC/MVS)

APPC/MVS provides distributed transaction processing capabilities for both inbound and outbound protected conversations. In the OS/390 Release 3, APPC/MVS supports the SNA syncpoint architecture for LU 6.2 conversations enabling LUs on OS/390 to participate in distributed two-phase commit through the use of the OS/390 Recoverable Resource Management Services. This allows for distributed transactional capabilities with a remote system that supports that LU 6.2 syncpoint protocals, or between LUs that are on the same system, or between LUs that are on different systems in a Parallel Sysplex. Since APPC/MVS supports all MVS application environments, new applications in all these environments are able to participate in distributed as well as local commit scopes.

APPC/MVS supports balancing of work among eligible systems in the sysplex using VTAM′s generic resources function and WLM recommendations for session allocation. On system failures, APPC/MVS and the OS/390 transaction manager support recovery and resynchronization of incomplete work over protected conversations on other eligible systems in the sysplex.

APPC/MVS supports network-qualified LU names. Application programmers can request APPC/MVS API traces to help in debugging their APPC/MVS applications.

1.4 System Management Services

The following enhancements are available in the system management services area:

- OS/390 Release 3 SDSF
- Hardware configuration definition (HCD)
- Resource measurement facility (RMF)
- SMP/E
- Recoverable resource management services (RRMS)

1.4.1 OS/390 Release 3 SDSF

SDSF delivers improved usability through new options to allow the user to extend the width of over-typeable fields and columns. This provides the ability to more efficiently use space on tabular panels. Additionally, a new tabular display is provided to display and control JES2 readers and punches. SDSF enhances its National Language (NLS) support with the ability to recognize an increased number of date formats. SDSF enhances security services with the delivery of functions to convert dynamic ISFPARMS to SAF or RACF. There are also

enhancements to SDSF's print services display which support the designation of Internet protocol (IP) addresses.

1.4.2 Hardware Configuration Definition (HCD) Enhancements

HCD includes support for new hardware. HCD also provides enhancements which make it easier to configure the OS/390 system and to create accurate configuration documentation as follows:

- IODF reports support more granularity for IODF compare reports, including a limitation on LPAR level and improved readability. A user gets a better overview of configuration changes, which is useful for configuration planning tasks. In addition, all IODF reports now show device number ranges.
- The number of messages during a build production IODF is reduced. The message collection at build production IODF time results in a significantly shorter message list, providing a better overview of changes.
- A preview of dynamic activation changes provides information about devices, control units, and channel paths that are to be deleted, added, or changed during dynamic activation.
- A new report type shows IOGEN information per device type, based on unit information module (UIM) information. Together with the other reports already provided by HCD, this information is useful for configuration planning purposes.

1.4.3 Resource Measurement Facility (RMF)

The traditional host-based performance monitoring of the Resource Measurement Facility is expanded significantly by the function called Performance Monitoring of OS/390, which allows users to monitor and analyze performance data from several OS/390 systems and the network on one workstation running OS/2. It is integrated into the performance monitoring common functions running on OS/2, and through this integration, it enables access to performance data from other applications (for example, performance monitoring of SNA).

1.4.4 SMP/E Enhancements

The OS/390 Release 3 SMP/E provides an enhancement to the unconditional CHANGE statement of the ZONEEDIT command by providing support for pathnames within DDDEF entries. Support of pathnames on the ZONEEDIT CHANGE statement provides the user with a simplified method of changing pathnames. This capability is especially useful to OpenEdition customers during the service process.

In OS/390 Release 3, SMP/E also provides enhanced support for cross-zone conditional requisite checking. In the past, the only SMP/E support in this area was the REPORT CROSSZONE command. The user needed to run the REPORT CROSSZONE command to identify cross-zone conditional requisites and then run any needed APPLY, ACCEPT, or RESTORE commands. With OS/390 Release 3, SMP/E can enforce conditional cross-zone requisites and provides immediate feedback from the APPLY, ACCEPT, and RESTORE commands regarding these requisites. This immediate feedback is reinforced by the fact that a user must bypass errors in order to get SYSMODs installed in the set-to zone.

1.4.5 Recoverable Resource Management Services (RRMS)

The OS/390 Recoverable Resource Management Services coordinates the resource manager′s recovery of protected resources across application, resource manager, and OS/390 system failures.

RRMS exploits the OS/390 system logger for logging and for recording resource manager information necessary to recover and restore critical resources to a consistent state. Resources of this nature are termed *protected* resources (or *recoverable* resources, the two terms are often used interchangeably); a protected resource is defined as a local or distributed resource that is to be updated in a synchronized and controlled manner. RRMS can also participate in coordination, recovery, and resynchronization of distributed work with distributed communication managers and client/server platforms.

In addition, Automatic Restart Manager (ARM) services may be used to facilitate fast restarts of the failing resource managers as a group on a system within the Parallel Sysplex.

The OS/390 Recoverable Resource Management Services are shipped disabled in the initial OS/390 product and can be made available through an enabling PTF.

1.5 Distributed Computing Services

An enhancement to the distributed computing services is IP PrintWay and NetSpool.

1.5.1 IP PrintWay and NetSpool

IP PrintWay, a feature of OS/390 Release 3, provides an automated facility to transmit output data sets from the MVS Job Entry Subsystem (JES) to printers or servers in your TCP/IP network. PrintWay, acting as a line printer requester (LPR), can translate output data sets from EBCDIC to ASCII and transmit the data sets to any line printer daemon (LPD) in a TCP/IP network. The LPD must adhere to the TCP/IP protocol between LPRs and LPDs as defined in Request for Comments (RFC) 1179 and amendments. PrintWay can transmit data to LPDs running in a printer or to LPDs running on a host system, including LPDs running on MVS, AIX, OS/2, OS/400, or UNIX systems.

NetSpool provides the capability to route VTAM application output (such as CICS and IMS) directly to the JES Spool without changing your application for processing by PSF/MVS or IP PrintWay. You can define NstSpool logical printers in the same routing and options data sets used by IP PrintWay.

The IP PrintWay and NetSpool solution has better performance, usability, capacity, and functionality characteristics than the Network Print Facility (NPF) and is the strategic replacement for NPF.

1.6 Communications Server

The OS/390 base includes a communications server. The communications server supports the programming interfaces most common in the current information technology industry: APPC, SOCKETS/RPC,and SNA(3270). It also supports wide-area networking protocols such as SNA, TCP/IP, and ATM. Because it integrates the VTAM AnyNet feature, which implements the multiprotocol transport networking (MPTN) architecture, the OS/390 Communications Server gives you a choice of which application to use over which network.

The communications server opens OS/390 for networking applications. The communications server in OS/390 Release 3 provides improved performance, high availability, and usability for S/390. This further enhances S/390′s superior strength as a server for the top tiers of an enterprise client/server solution. The following overview offers more detail on these significant enhancements for OS/390 networking. See Chapter 14, “Communications Server for OS/390” on page 189.

1.6.1 Performance and RAS Improvements for TCP/IP Function

Communications server enhancements include improved performance, reliability, availability, and serviceability for TCP/IP.

The TCP/IP CICS Sockets feature and IMS Sockets feature are now included as part of the OS/390 base. In OS/390 Releases 1 and 2, these were optional, priced features.

1.6.2 Multi-Node Persistent Sessions (MNPS)

A very important aspect of high availability for mission-critical applications in the network is addressed in the communications server through the capability known as MNPS. MNPS extends current persistent sessions capability across multiple hosts connected through the S/390 coupling facility. MNPS provides for the recovery of VTAM, MVS, hardware or application failures by restarting the application on another host in the sysplex without requiring users to re-logon. In conjunction with the automatic restart manager (ARM) function of OS/390, the application restart can be automatically executed, thus making an outage transparent to end users.

Chapter 2. OS/390 System Logger

The OS/390 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.

An OS/390 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 Figure 3.

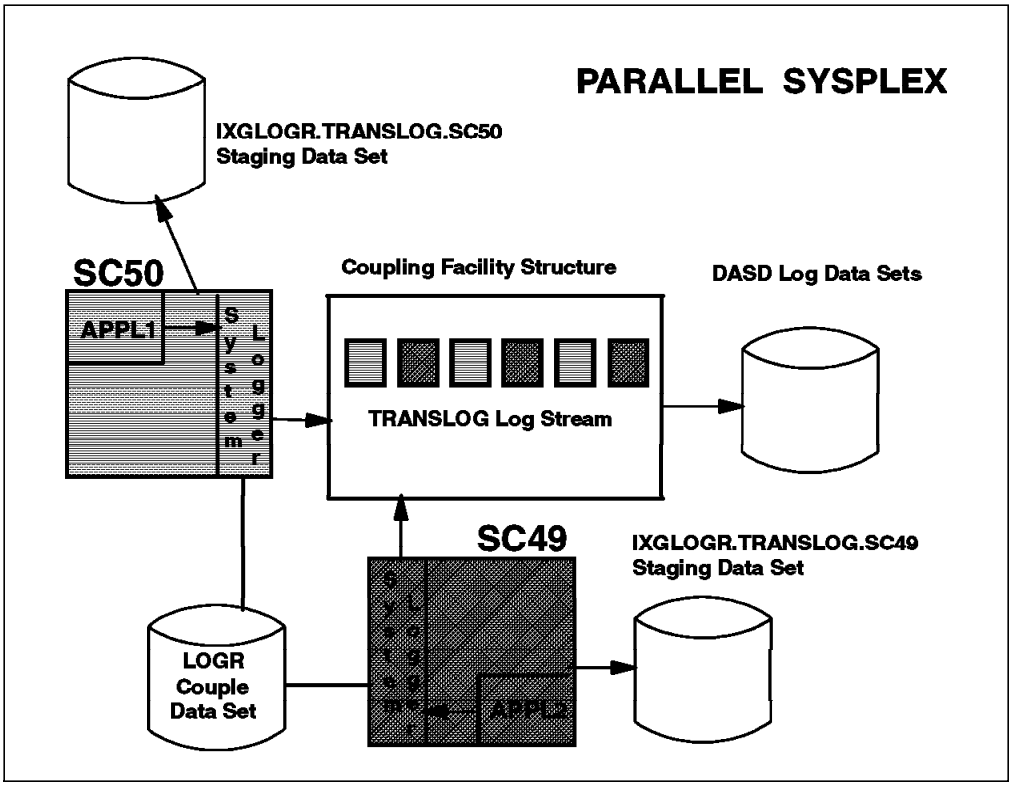


Figure 3. Overview of OS/390 System Logger Processing

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.

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.

You can use system logger services to merge data from different systems in a sysplex, as shown in Figure 3, 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)

OS/390 Release 3 adds additional services. See 2.1, “OS/390 System Logger Enhancements in OS/390 Release 3.”

The following documentation describes all aspects of the OS/390 system logger:

- *Setting Up a Sysplex*
- *MVS/ESA SP V5 Planning: Operations*
- *MVS/ESA SP V5 Assembler Services Guide*
- *MVS/ESA SP V5 Assembler Services Reference*
- *Sysplex Migration Guide*

2.1 OS/390 System Logger Enhancements in OS/390 Release 3

Many of the system logger functions are enhanced in OS/390 Release 3. The enhancements include:

- The 168 simultaneous DASD log data set limit is removed and replaced by a virtually unlimited number of data sets that can be defined.

  You can, if necessary, increase the number of log data sets available for log streams using the DSEXTENT parameter in a OS/390 Release 3 level LOGR couple data set. If you have log streams that exceed 168 DASD log data sets, perhaps because you retain data in your log stream for a long period of time, increase the number of log data set directory extents available for the sysplex on the DSEXTENT parameter in a OS/390 Release 3 or higher LOGR couple data set using the format utility.

  Each additional extent specified goes into a common pool available to any log stream in the sysplex. System logger allocates these extents as needed. Each extent allows a log stream to extend by 168 log data sets. Whenever all the log data sets in a data set directory extent have been physically deleted, system logger returns the data set directory extent record to the available pool of extents for the sysplex.

  Each log stream has one data set directory extent which is part of the log stream record, not part of the data set directory extent pool. This permanent extent is always used first for a log stream, before retrieving an extent from the common data set directory pool. This extent does not revert to the common pool when unused and is not available for use by any other log stream; it is held for the owning log stream.

The size of the LOGR couple data set may have to be increased to accommodate the increase in the number of DASD log data sets per log stream.

- DASD log data sets may be deleted automatically using user-defined retention periods defined in the LOGR policy. System logger provides support to assist installations in managing the size of log streams and archival of log data.
- The system logger can dynamically change the entry-to-element ratio of the log stream structure to manage the storage and avoid inefficient use of the structure storage.

  Previously, an estimated average log block size for a structure, specified on the AVGBUFSIZE parameter in the LOGR policy, was used to calculate an entry-to-element ratio that applied for the life of the structure.

  OS/390 Release 3 system logger uses the value specified on AVGBUFSIZE only to set an initial element-to-entry ratio. System logger then dynamically monitors and manages the entry-to-element ratio for structures based on actual structure usage. Dynamic management of the entry-to-element size helps maintain efficient use of coupling facility structure space.
- System logger connect processing has been restructured in OS/390 Release 3 to provide more parallelism. This can result in better performance at connect time, because connects for log streams in different coupling facilities can be processed in parallel.
- Structure rebuilds have support to protect log data during the period of time after a structure rebuild fails and up until the time that the reason for the rebuild failure is corrected and another rebuild initiated.
- System logger can automatically allocate staging data sets during certain structure rebuild failures, even when the customer does not request staging data sets for their logging configuration. This ensures a reliable, non-volatile copy of log data for recovery.
- Support for remote site recovery that allows log stream data at a local or primary site to be duplicated at a remote or secondary site. The remote site can become the primary site if a disaster occurs at the primary site.

  OS/390 Release 3 system logger provides services that allow development of a resource manager application to monitor writes and deletes issued against a log stream. The resource manager application can then be associated with a log stream. The resource manager can be informed of all successful writes to a log stream and potentially modify the range of data to be deleted. In addition, the log block import service (IXGIMPRT) is provided for copying a log block from one log stream to another with identical log block identifier and time stamp as the original.

2.1.1 New Exploiters of the System Logger

With OS/390 Release 3, in addition to SYS1.LOGREC and the operations log (OPERLOG), the following products and services require the OS/390 system logger:

- CICS Transaction Server for OS/390
- IMS/ESA Common Queue Server
- Remote Site Recovery
- Resource Recovery Services

2.1.2 Incompatibilities with the Old System Logger

There are several incompatibilities between the OS/390 Release 3 version of the system logger and the previous version, as follows:

- The size of the couple data set may be larger due to a new record type, DSEXTENT. The OS/390 Release 3 format of the LOGR couple data set cannot be used with systems running prior releases of MVS. This also changes the format of the LOGR couple data set report output. See 2.2.2.1, “Logger Couple Data Set Report” on page 16.
- The OS/390 Release 3 system logger can dynamically allocate staging data sets for log streams during rebuild recovery of the log stream structure. See 2.4.1, “Logger Structure Rebuild Recovery” on page 19.

2.2 System Logger Couple Data Set

The system logger component manages log streams based on the policy information in the LOGR couple data set. The LOGR couple data set must be accessible by all systems in the sysplex. Always define a primary and an alternate LOGR couple data set.

There are two utilities provided to format a couple data set and to define a policy for the couple data set. These utilities are:

IXCL1DSU Use to format the LOGR couple data set.

IXCMIAPU Use to add, update, list, or delete policy data on the LOGR couple data set.

Note: See *Setting Up a Sysplex* for more detailed information.

2.2.1 LOGR Couple Data Set for Release 3

The activation of all the OS/390 Release 3 system logger new functions requires reformatting of the LOGR couple data set using the IXCL1DSU utility. If you have log streams that exceed 168 simultaneous DASD log data sets, you need to increase the number of log data sets available for log streams using the DSEXTENT parameter on the OS/390 Release 3 IXCL1DSU utility initialization control statement. Figure 4 shows the JCL for the utility to format a new LOGR couple data set which specifies a new record type, DSEXTENT(nnnnnnnn).

Where:

DSEXTENT nnnnnnnn defines the number of log data set directory extents for which the LOGR couple data set should be formatted. Each log stream has space for 168 log data sets within the log stream record. The log data set directory extents are used only to increase the log streams data set directory capacity beyond the 168 data set limit.

Default 0

Maximum 99,999

Log streams can now expand in increments of 168 data sets. This allows for a maximum of 16.8 million log data sets within the sysplex for extent processing (99,999 extent records * 168 log data sets per extent record). See 2.3, “DASD Log Data Sets” on page 16.

```
//DEFINE JOB
//FORMAT EXEC PGM=IXCL1DSU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
 /* LOGR Couple data set - primary */
   DEFINEDS SYSPLEX(WTSCPLX1) DSN(SYS1.LOGR3.CDS1) VOLSER(TOT001)
     DATA TYPE(LOGR)
        ITEM NAME(LSR) NUMBER(50)
        ITEM NAME(LSTRR) NUMBER(10)
        ITEM NAME(DSEXTENT) NUMBER(20)
 /* LOGR Couple data set - alternate */
   DEFINEDS SYSPLEX(WTSCPLX1) DSN(SYS1.LOGR3.CDS2) VOLSER(TOT002)
     DATA TYPE(LOGR)
        ITEM NAME(LSR) NUMBER(50)
        ITEM NAME(LSTRR) NUMBER(10)
        ITEM NAME(DSEXTENT) NUMBER(20)
```

Figure 4. LOGR Couple Data Set Format Utility

Note: If you have no log streams that require greater than 168 log data sets in their data set directory, then there is no need to format data set directory extent records in the LOGR couple data set.

The LOGR couple data set policy includes the following:

- Log stream definitions
- Coupling facility list structure definitions
- Data containing the current state of a log stream, for example, whether a log stream is currently connected to the coupling facility structure

2.2.2 Converting to Release 3

If you add a DSEXTENT record type to the LOGR couple data set, it must be reformatted. Once you have formatted the new LOGR couple data sets, primary and alternate, it can be maintained and updated using either:

- The IXCMIAPU utility to update the LOGR policy manually
- The IXGINVNT system logger service when you want your application program to update the LOGR policy dynamically

Note: The new LOGR couple data set with the DSEXTENT record type can only be used on OS/390 Release 3 or higher systems.

Add the new couple data set as the active alternate using the following command:

```
SETXCF COUPLE,TYPE=LOGR,ACOUPLE=(SYS1.LOGR3.CDS1)
```

Note: All systems in the sysplex should now be at the OS/390 Release 3 level for the OS/390 system logger to use the LOGR couple data set in its new format.

Then switch the alternate to become the primary, and make the second new couple data set the new alternate.

2.2.2.1 Logger Couple Data Set Report

Figure 5 shows the IXCMIAPU utility program report for the definitions of the OPERLOG log stream. The report output shows the definitions for the log stream that were made in the LOGR policy. With OS/390 Release 3, this report also shows the number of DASD log data set directory extent records in use.

```
LOGR Inventory Record Summary

Type                        Formatted          In-use
____________________        _________          ______

LSR (Log Stream)                 20               2

LSTRR (Structure)                20               2

   LOGSTREAM NAME(SYSPLEX.OPERLOG)  STRUCTNAME(SYSTEM_OPERLOG) LS_DATACLAS(SHARE33)
             LS_MGMTCLAS() LS_STORCLAS() HLQ(IXGLOGR) MODEL(NO) LS_SIZE(512)
             STG_MGMTCLAS() STG_STORCLAS() STG_DATACLAS() STG_SIZE(0)
             LOWOFFLOAD(60) HIGHOFFLOAD(80) STG_DUPLEX(YES) DUPLEXMODE()

   STRUCTURE NAME(SYSTEM_OPERLOG) LOGSNUM(1)
             MAXBUFSIZE(65532) AVGBUFSIZE(400)
             LOGSTREAMS CURRENTLY DEFINED TO THIS STRUCTURE(1)
```

Figure 5. Output from the IXCMIAPU Utility for the Operations Log

When defining the log stream in the LOGR couple data set, the following definitions are for the system logger-managed DASD log data sets:

NAME This is the log streamname used in the creation of the DASD log data set naming convention. See 2.3.2.1, “DASD Log Data Set Naming Convention” on page 17.

HLQ This is the high level qualifier for the DASD log data sets. The default is IXGLOGR. See 2.3.2.1, “DASD Log Data Set Naming Convention” on page 17.

STG_DUPLEX (YES) specifies to use DASD staging data sets for a backup copy of the log blocks for each system in the coupling facility log stream. See 2.4, “Staging Data Sets” on page 18.

LS_SIZE Specifies the size, in 4K blocks, of the DASD log data sets for the log stream being defined. See 2.3.1, “Choosing a Log Data Set Size” on page 17.

2.3 DASD Log Data Sets

The system logger requires that a coupling facility list structure be defined for each log stream. When an application writes data to a log stream, it is first written to the coupling facility list structure for that log stream. This allows multiple instances of an application running on different systems to merge their data by writing to the same log stream.

The system logger uses VSAM linear data sets to store log stream data that has been moved from the coupling facility when the coupling facility structure space allocated for the log stream reaches its installation-defined threshold. These data sets become the DASD log data sets.

It is recommended that DASD log data sets be managed by system-managed storage (SMS). You can manage log stream data sets by either:

- Modifying automatic class selection (ACS) routines
- Defining the SMS data class, storage class, and management class explicitly in the log stream definition using the IXCMIAPU utility or the IXGINVNT service

A log stream can have data in multiple DASD log data sets; as a log stream fills log data sets on DASD, the system logger automatically allocates new ones for the log stream. The system logger increments the sequence number in the log stream data set name as new data sets are added for a particular log stream. See 2.3.2.1, "DASD Log Data Set Naming Convention." Before OS/390 Release 3, the maximum number of DASD log data set was 168. With OS/390 Release 3, log streams can expand their DASD log data sets in increments of 168 by obtaining one of the DSEXTENTs defined in the LOGR couple data set policy. Depending on the number of extents defined, a log stream can be virtually unlimited in size. See 2.3.2, "Choosing a DSEXTENT Value."

2.3.1 Choosing a Log Data Set Size

The size of the log data sets can be as large as you can afford to make them. This minimizes the number of log data sets required to represent a log stream. It also minimizes the number of times the system logger must allocate and switch to a new log data set when an old one becomes full. The LS_SIZE parameter is used to define the size of a log data set. See Figure 5 on page 16.

By default, each log stream is limited to a maximum of 168 log data sets. With OS/390 Release 3, this log stream limit is increased by reformatting the LOGR couple data set and adding the DSEXTENT parameter.

2.3.2 Choosing a DSEXTENT Value

The number of log data set directory extent records that should be formatted in the LOGR couple data set is dependent on the log stream requirements. Each extent defined represents 168 log data sets. An extent is allocated to one and only one log stream. You should format at least 15% more log data set directory extent records than will be in use at any point in time.

The system logger issues a shortage message that indicates a shortage once the number of in-use extents exceeds 85% of the number formatted. A critical shortage message is issued once the number of in-use extents exceeds 95% of the number formatted.

If this is the first time you are formatting a LOGR couple data set with DSEXTENT records, you might chose to format a reasonable number and monitor the usage either through periodically running a LOGR couple data set report and checking the percentage in use or through watching for the shortage message, IXG261E.

2.3.2.1 DASD Log Data Set Naming Convention

The DASD log data set names are constructed from installation-defined values. The log data set format is:

```
HLQ.logstreamname.Annnnnnn
```

The HLQ and logstreamname are shown in Figure 5 on page 16. The last qualifier is an automatically generated sequence number whose first character is

A. As each log data set becomes allocated, the sequence number is updated by 1. Using the definitions from Figure 5 on page 16, the first log data set allocated is:

```
IXGLOGR.SYSPLEX.OPERLOG.A0000001
```

2.3.3 Using Log Data Sets

The data set extent (DSEXTENT) records, specified in the LOGR couple data set, are a free pool to be used by all applications creating log streams. Each defined log stream has a data set extent or permanent extent of its own which is not part of the free pool. When a log stream has all 168 data sets in use and full, an extent is obtained from the free pool which provides an additional 168 data sets. The first data set allocated from the first free pool extent that is allocated by the OS/390 system logger has the following name (when no previous log data sets have been deleted):

```
IXGLOGR.SYSPLEX.OPERLOG.A0000169
```

2.3.3.1 Log Data Set Monitoring Messages

The messages in Table 1 are related to monitoring of the number of log data sets and the number of directory extents that are allocated.

Table 1. OS/390 System Logger Messages

Message ID	Message explanation
IXG261E	Issued when usage of log data set directory extents is over 85%.
IXG262A	Issued when usage of log data set directory extents is over 95%.
IXG257I	Issued when the number of log data sets in a log stream′s data set directory get above 90% of the total number allowed.
IXG301I	When issued with a return code of X′08′ and a reason code of X′085C′, this indicates that an offload has failed due to a log data set directory full condition. If an offload fails because of a lack of extent records, system logger will retry the offload after additional log data sets have been made available.

2.4 Staging Data Sets

The primary use for staging data sets is to avoid a single point of failure for log data. When an application in each system is writing a log block into a log stream, the OS/390 system logger places the log block into the couple facility and by default places a copy of the log block into a dataspace on the system that initiates the write. In the case where the coupling facility is in an LPAR on the same CPC as a dataspace and the CPC is lost, then both copies of the log data are lost. To avoid this situation, or avoid having a single point of failure, you have the option to specify the use of staging data sets using the STG_DUPLEX(YES) specification in the LOGR policy for the log stream.

Note: Staging data sets are allocated as VSAM linear data sets. You should make them SMS-managed.

2.4.1 Logger Structure Rebuild Recovery

With OS/390 Release 3, the OS/390 system logger can dynamically allocate temporary staging data sets if dataspaces are in use for a log stream structure and a rebuild fails. This happens when there is a log stream structure failure followed by the rebuild failure and a backup copy of the log data is in the dataspaces. A staging data set is allocated on all systems and the log data is copied from the dataspaces to the staging data sets on each system. All user connections to the log stream are broken. This ensures that a non-volatile copy of the log data is maintained and protected between the failure and the correction.

2.5 Deleting Log Data Sets

The OS/390 Release 3 system logger provides support such that log data can remain in the log stream for a specific length of time. This eliminates having to remember to delete the log data. Log data can only be deleted for log streams that are connected and being written to by an application.

Note: Log data can only be deleted from a log stream on a log data set basis. All the data in a log data set must be deleted before the log data set can be deleted.

Two new keywords are added to the DEFINE LOGSTREAM specification. They are:

RETPD Specifies a *retention period*, which is the number of days that log data may stay in the log stream before being eligible to be automatically deleted. Once the retention period has expired for an entire log data set, the log data set is eligible to be deleted. Automatic deletion using AUTODELETE(YES) occurs for log data sets whose retention period has expired.

RETPD=0 is the default. The retention period can be a value between 0 and 65,536.

AUTODELETE Specifies when the system logger can automatically delete log data.

(NO) AUTODELETE(NO) specifies that the OS/390 system logger can delete a log data set when both the retention period has expired and an application uses the IXGDELET service to delete the log data set.

This is the default.

(YES) AUTODELETE(YES) specifies that the OS/390 system logger should automatically delete a log data set when the retention period expires or the IXGDELET service deletes the log data set.

Warning: If you specify AUTODELETE(YES) and RETPD=0, log data is eligible for deletion as soon as it is created. These keywords can only be used when the active primary LOGR couple data set is at an OS/390 Release 3 level.

Figure 6 shows the utility program used to define log streams in the LOGR policy contained in the LOGR couple data set. A retention period of 30 days is defined and log data sets with data 30 days or older are automatically deleted by the OS/390 system logger.

```
//DEFINE JOB
//FORMAT EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
     DATA TYPE(LOGR)
        DEFINE LOGSTREAM NAME(SYSPLEX.OPERLOG)
               RETPD(30) AUTODELETE(YES)
```

Figure 6. Defining a Retention Period

The RETPD and AUTODELETE keywords may be updated in the LOGR policy by using the UPDATE LOGSTREAM specification shown in Figure 7.

```
//UPDATE JOB
//FORMAT EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
     DATA TYPE(LOGR)
        UPDATE LOGSTREAM NAME(SYSPLEX.OPERLOG)
               RETPD(100)
```

Figure 7. Updating a Retention Period

2.5.1 Log Stream Data

In Figure 8, there are six log blocks in the log data set. If the log stream retention period is one day, Bklid=1 is older than or equal to the retention period and is eligible for deletion. However, no log data in this log data set would be deleted from the log stream, even though log block Blkid=1 is eligible for deletion. The OS/390 system logger ensures that all the log data for a given log data set is expired before deleting any log data in the log data set. The OS/390 system logger only needs to compare the date of block Blkid=6 for the retention period for the log stream to determine that it cannot delete any log data in the log data set.

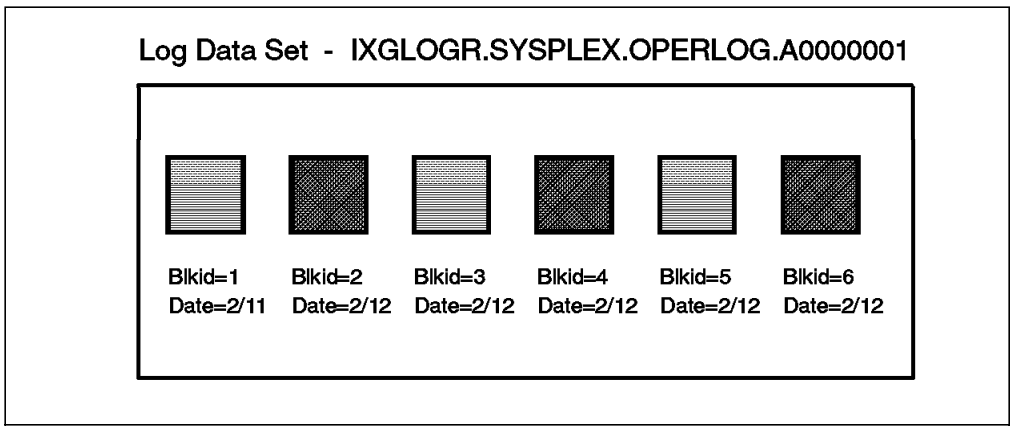


Figure 8. Log Data Set Log Stream Records

Note: Retention period processing for log data sets is not the same as for data sets (RETPD on a JCL DD statement), because data set retention period processing indicates that a data set should not be deleted until its retention period expires and the retention period starts once the data set is created. This

support applies a retention period to the data within the data set, not to the data set itself.

2.5.2 Deleting Log Stream Data

With the use of a retention period, one of the effects on a log stream may be to force structure-resident log data to be offloaded to log data sets, even though the log data may have been deleted through a IXGDELET service request. This happens because the retention period delays the physical deletion of the log data until the retention period expires. This delay of the physical deletion of the log data results in the log data being placed into what is known as the *inactive* section of the log stream.

Thus, with OS/390 Release 3, all log streams are defined as having an active section and an inactive section as follows:

Active An active section is defined to be the log data in the log stream which has not been deleted through an IXGDELET request. When log data is first written into the log stream, it resides in the active section of the log stream and is considered active log data.

Once active log data is deleted through an IXGDELET request, the log data is either physically deleted because it is expired log data, or moved to the inactive section of the log stream because it is unexpired log data.

Inactive The inactive portion of the log stream is defined to be the log data in the log stream that has been deleted through an IXGDELET request, but based upon the retention period defined for the log stream, the log data should be retained within the log stream.

Inactive log data remains in the inactive section of the log stream until it expires, at which point it is physically deleted.

The length of the inactive section of the log stream can be as large as necessary in order to maintain the required quantity of unexpired log data.

There is no way to delete data from the inactive section of the log stream; the retention period option on the log stream definition must be used for this purpose.

2.5.3 Viewing Inactive Log Stream Data

An application can use the IXGBRWSE service to view log data. The OS/390 Release 3 system logger browse service has a new option called VIEW=(ACTIVE|ALL) that can be used to define the sections of the log stream that the browse session has access to view. The default is VIEW=ACTIVE. The application now has control over whether it has access to just the active log data in the log stream, or to all the log data in the log stream, both active and inactive log data.

2.6 Coupling Facility Structure Storage Management

With OS/390 Release 3, the OS/390 system logger dynamically manages a structure′s entry-to-element ratio by periodically checking to determine if the actual usage is significantly different than the structure allocation and, if so, it attempts to alter the structure′s entry-to-element ratio to match the current usage.

The entry-to-element ratio for a log stream coupling facility structure is determined from the buffer size keyword AVGBUFSIZE. The AVGBUFSIZE is the average log block size written into a log stream that maps into a structure. The OS/390 system logger uses the AVGBUFSIZE specification to determine the entry-to-element ratio to specify when it connects to the structure.

Choosing an AVGBUFSIZE is difficult, because it is difficult to know the size of every record an application writes to its log. Logger applications such as CICS, OPSLOG, and LOGREC provide no recommendations because the average log block write size varies by installation. Previous to OS/390 Release 3, once the coupling facility structure was connected to by the OS/390 system logger, the entry-to-element ratio could not be changed.

However, with OS/390 Release 3, the system logger can dynamically change this ratio. The AVGBUFSIZE value from the structure definition in the LOGR couple data set is used as an initial recommendation. Previously, this value could not be changed without first deleting the structure definition and then redefining the structure definition with the new average buffer size and redefining all the log streams for that structure. The OS/390 Release 3 system logger dynamically manages a structure′s entry-to-element ratio by periodically checking to determine if the actual usage is significantly different than the structure allocation. It it is significantly different, it alters the structures entry-to-element ratio dynamically. The logger polls a structures actual ratio every thirty minutes to determine how to adjust the ratio.

2.6.1 Log Stream SMF Recording

The entry-to-element ratio changes are added to the OS/390 system logger SMF record, type=88. This can assist in monitoring the effectiveness of the entry-to-element ratio changes. This SMF record also has a new section that contains the aggregate structure information in addition to the log stream section, the event section, and the structure section. This new section represents a single system′s view of the overall structure activity during the SMF recording interval.

2.7 Remote Site Recovery

OS/390 Release 3 system logger provides services that allow development of a resource manager application to monitor writes and deletes issued against a log stream. The new services available with OS/390 Release 3 are: The resource manager application can then be associated with a log stream. The resource manager can be informed of all successful writes to a log stream and potentially modify the range of data to be deleted. In addition, a new log block import service (IXGIMPRT) is provided for copying a log block from one log stream to another with identical log block identifier and time stamp as the original. Event notification facility (ENF) also allows new events to assist a resource manager in managing the log stream.

Log stream data at a local (or primary) site can be shadowed (or duplicated) at a remote (or secondary) site, as shown in Figure 9 on page 23. The remote site can become the primary site in case a disaster occurs at the primary site. The new system logger service, IXGIMPRT, allows a resource manager application to write into a log stream at the remote site. The transmitted log data blocks have the block ID and the GMT time stamp identical to the block ID and GMT time

stamp assigned at the local site. The application must provide its own method or transmission program to move the log blocks to the remote site.

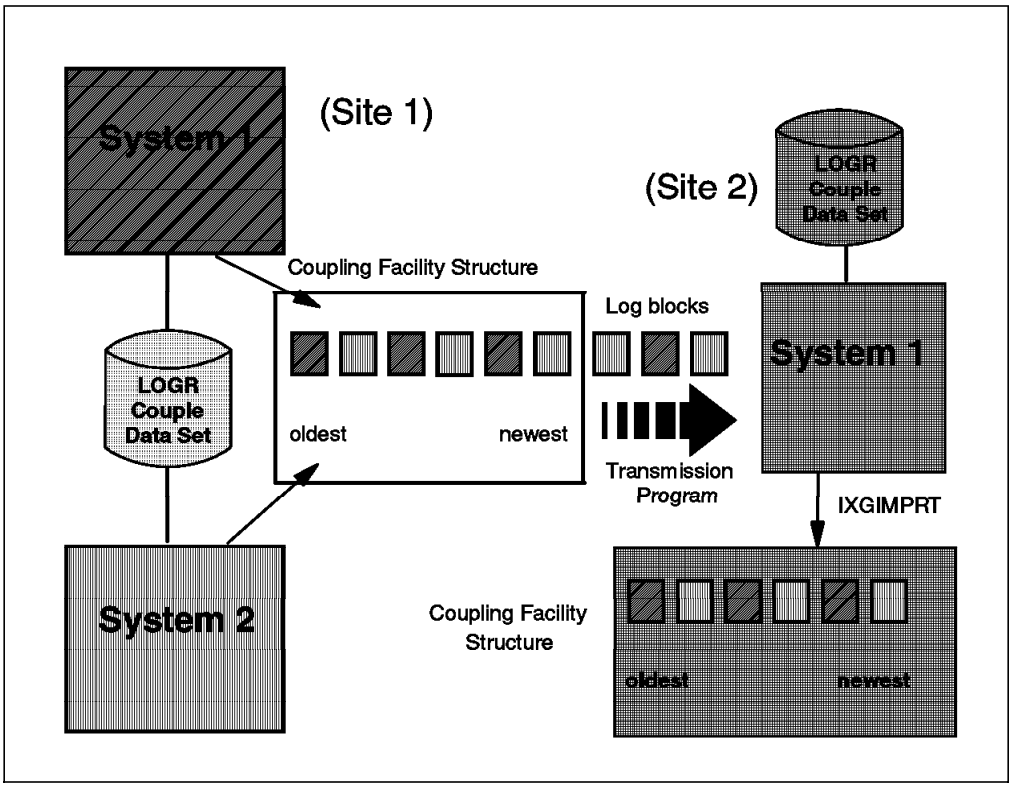


Figure 9. Remote Site Recovery Overview

2.7.1 New System Logger Services

The OS/390 system logger provides new services for applications connecting to log streams. The new services are as follows:

IXGCONN The existing service IXGCONN introduces a new keyword, IMPORTCONNECT, which is used to classify a log stream connection for a resource manager. A resource manager exit address is also specified on the IXGCONN macro. The resource manager exit is specified by the RMNAME keyword.

IXGIMPRT The IXGIMPRT service provides a program the ability to write a log block and to explicitly specify the block ID and GMT time stamp to be assigned to the log block. In order to use the new IXGIMPRT function, the corresponding connection must be classified as an IMPORTCONNECT-type connection. See 2.7.3, “Resource Manager Connections to a Log Stream” on page 24.

IXGQUERY The IXGQUERY service allows a user to retrieve information about a log stream.

This service can be used by a resource manager to obtain a safe import point. The safe import point is defined as the highest log block ID and its corresponding GMT time stamp processed by the writer offload process. It provides an indication of the highest block ID that can be safely imported at the remote site. See 2.7.4, “Safe Import Point” on page 26.

2.7.2 Resource Manager

A resource manager is an application you can write and associate with a log stream to manage resources and processing for a log stream. Before a resource manager can connect to a log stream, the name of the resource manager must be specified in the log stream definition in the LOGR couple data set. You can specify one resource manager name for a log stream in the log stream definition using the IXCMIAPU utility. Figure 10 shows a utility job to create a LOGR policy.

```
//DEFINE JOB
//STEP1 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=*
//SYSIN DD *

    DATA TYPE(LOGR) REPORT(YES)

      DEFINE STRUCTURE NAME(STRUCTURE1)
             LOGSNUM(32)

      DEFINE LOGSTREAM NAME(STREAM1)
             STRUCTNAME(STRUCTURE1)
             LS_MGMTCLAS(STANDARD)
             LS_DATACLAS(VSAMLS)
             HLQ(LOGGER)
             RMNAME(XMITTER)
```

Figure 10. Utility Job to Create LOGR Policy

Note: If you specify a resource manager name for a log stream in the LOGR policy, the resource manager specified must connect to the log stream. If the resource manager does not connect, system logger does not process any IXGDELET requests to delete log data. This is so that the resource manager does not miss any information about deletes issued against the log stream.

2.7.3 Resource Manager Connections to a Log Stream

The resource manager connects to the log stream it manages using the RMNAME, RMEXIT, RMDATA, and RMEVENTS parameters on the IXGCONN service. The connect request must be issued from the resource manager address space. The RMEVENT parameter on the connect request specifies the events that you want to invoke the resource manager user exit. When you specify RMEVENT=LBWRITE, successful write requests to the log stream invoke the resource manager user exit. When you specify RMEVENT=LBDELETE, successful delete requests to the log stream invoke the resource manager user exit. You may specify both parameters for the RMEVENT keyword.

A resource manager can connect to a log stream from a given system. The resource manager can connect to multiple log streams.

2.7.3.1 Resource Manager User Exit

When a write or delete request occurs against the log stream, system logger gives control to the resource manager exit, passing a parameter list. The resource manager exit runs in the resource manager address space. The resource manager exit is called as follows:

Write request The resource manager exit is called after the write request completes. If staging data sets are in use for the connection, the exit is called after the write to the staging data set completes.

Delete request The resource manager exit is called before the delete request is processed. This allows the resource manager exit to accept, reject, or override the delete request on behalf of the log stream.

The resource manager exit is always invoked before the request completion is reported to the system logger application that issued the request.

When the resource manager exit gains control, it receives information in the resource manager exit parameter list (RMEPL) mapped by the IXGRMEPL macro. This information includes:

- The name of the log stream associated with the resource manager.
- An indication whether the request that gave the resource manager exit control is a write or delete request.
- The data specified in the RMDATA parameter on the resource manager's IXGCONN request, if specified.
- A pointer to the buffer containing the log data written to the log stream for write invocation. For delete invocation, the information includes the block identifier(s) involved with a IXGDELET request.

For example, through the resource manager user exit, a resource manager might be notified of a write or delete request issued against a log stream. The resource manager can then perform further processing to accept, reject, or override the delete request with a different log block identifier. When a resource manager provides remote site recovery functions, it needs to transmit all log stream changes to the remote site.

At the remote site, the "receiver" program uses the IXGIMPRT service to write data to a log stream, specifying a log block identifier and GMT time stamp for each log block and IXGDELET service to delete log blocks that are deleted from the log stream. The import service can be used to create copies of log data from one log stream to another, preserving the GMT time stamp and log block identifier assigned when the log block was written to the source log stream. The *source log stream* is the original log stream that the log stream is importing blocks *from*. The log stream you import blocks *to* is the *target log stream*.

In order to use the IXGIMPRT service, the connection to the log stream must be an import connection, issued with AUTH=WRITE IMPORTCONNECT=YES. Note that when you specify AUTH=WRITE IMPORTCONNECT=YES for a connection, you cannot issue the IXGWRITE request against the connected log stream.

2.7.4 Safe Import Point

Following is an example of when the safe import point is used:

- Suppose the resource manager has transmitted log blocks 1 through 6 to the remote site but has yet to import any of them. (Assume no log blocks have yet been offloaded for the log stream at the local site.)

  An offload operation at the local site occurs because the high offload threshold value is attained, resulting in log blocks 1 and 2 being written to a DASD log data set. The safe import point is set to block ID 2.

  1. An indication is sent by the remote site recovery resource manager to the remote site to import log blocks 1 and 2. At the remote site, log blocks 1 and 2 are imported.
  2. Next, the failure of the primary site occurs when both the coupling facility and the MVS image simultaneously fail.
  3. A rebuild is initiated and is handled by another MVS image. After the rebuild completes the log stream contains log blocks 1, 2, 4, 5 and 6.
  4. The remote site recovery resource manager is notified of the rebuild via an ENF 48 signal.
  5. The resource manager should discard all log blocks with a block ID greater than 2 at the remote site, and upon subsequent notification that the safe import point has changed, begin its browse session at the first log block greater than 2 and browse up to the new safe import point. These log blocks can be then sent to the remote site and safely imported.
  6. Next, an offload occurs resulting in log blocks 4, 5, and 6 (but not 3) being offloaded to DASD.
  7. The new safe import point is set to log block id 6.
  8. The browsing application, since it has been notified of the rebuild (through an ENF) first tells its remote instance to discard all log blocks greater than block ID 2 and then it must reinitiate its browse from log block 2 through 6 and send that data to the remote site.
  9. It may also wish to inform its remote instance that certain gaps are expected and do not need to be recovered.
  10. If a rebuild has not occurred between changes of the safe import point, then all log blocks up to and including the most current safe import point can be unconditionally imported into the log stream.
  11. The safe import point indication is an integer and always increases for a defined log stream. Its value is set to zero when a log stream is defined to the MVS System Logger.

Chapter 3. TSO/E Generic Resource Support

One problem with TSO/E in the past was that a user of TSO/E had to know the system name he wanted to log on to. This also meant that in a Parallel Sysplex environment with a number of TSO/E systems, there was no possibility for workload balancing between those systems.

So, for example, let us consider QMF as an end user tool for DB2 query processing and the use of IBM Application System (AS) for project planning. In the past, these systems did not fit very well into a Parallel Sysplex environment, while on the other side, DB2 itself with data sharing did. You can see the way traditional logon works in Figure 11.

In this example we assume we have two users named Paul and George, both working on TSO/E. User George does normal logon processing, giving the application name and user ID to VTAM and logging on to TSO/E system SC52TS. User Paul tries to reconnect to his lost session on system SC50TS.

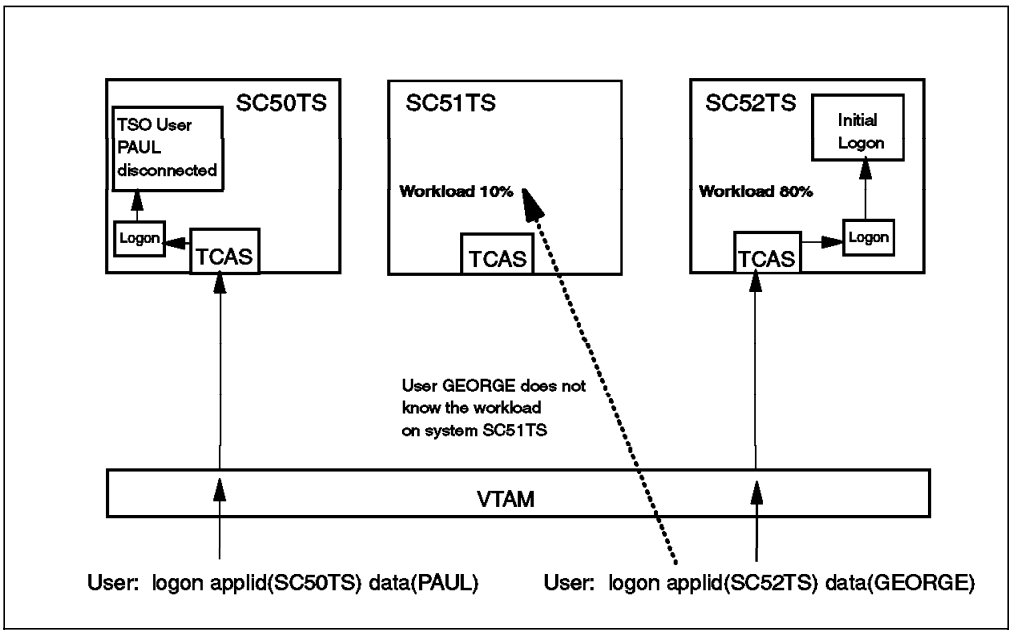


Figure 11. TSO/E Traditional Logon

Both users have to specify the application names of the TSO/E systems. But with a traditional logon procedure, there is no way for user George to log on to system SC51TS automatically, even though system SC51TS at the moment of logon has the lowest workload. As a TSO/E user, user GEORGE does not know this system is available for work.

For doing manual balancing of TSO/E users, system programmers could have given user IDs on different systems to those users. For example, users JOHN, PAUL, GEORGE, and RINGO could be routed to only SC50TS, while JIM, JOE, and MIKE could be routed to only SC51TS and so forth.

But what happens if only users JOHN, PAUL, and RINGO are logged on to one system at a specific timeframe? They are all logged on to system SC50TS. At that time SC51TS has no workload (we do not consider other subsystems at this moment), because users JIM, JOE and MIKE are not at work.

That means there is no real workload balancing for TSO/E. For administrators who have to make the decisions of which users must log on to a specific TSO/E system, there is additional work to do.

Without TSO/E generic resource support, if one of three systems (SC50TS) where user Paul wants to log on goes down, he will be unable at that moment to do more work on TSO/E, as shown in Figure 12.

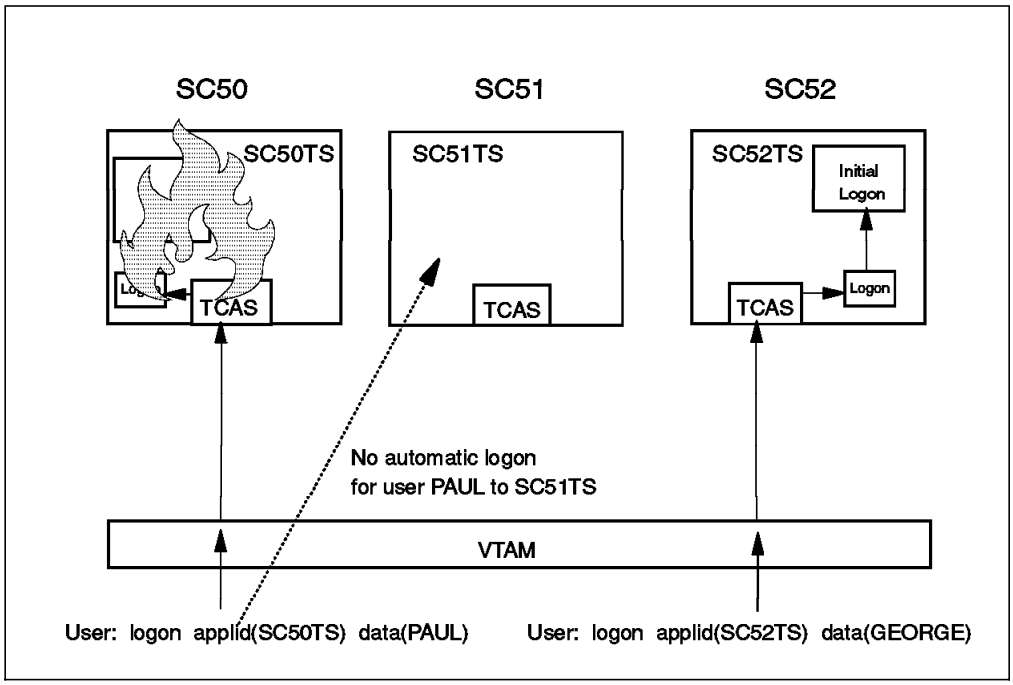


Figure 12. TSO/E System Down

In contrast, if generic resource is implemented, he will be automatically routed to system SC51TS in this situation.

3.1 TSO/E Generic Resource Support

With OS/390 Release 3, the solution to this problem is implemented. With TSO/E Generic Resource Support (TSO/GR) in a Parallel Sysplex environment, TSO/E on one system can be known by the same generic name as a TSO/E on another system in the same sysplex. With this option, workload balancing and high availability for TSO/E is given by using VTAM generic resource to cause work to be placed on the appropriate system. This support should also reduce the need to understand the topology of the TSO/E application by end users. This support should improve the usability of the MVS system.

The components that are involved to realize TSO/GR are:

- VTAM, for which generic resource initial support was introduced in MVS/SP Version 5.2
- JESXCF

  JESXCF provides the basic function required by TSO/E in the way that it uses a subsystem interface call to JES to obtain the information necessary to route a distributed method request to a target system determined from the information extracted by the JES subsystem interface routine. TSO/E gains

access to this function using a macro interface called IXZXTSOI. This macro is a BCP function. It performs several functions:

  - It returns the system name to a caller where the TSO/E user is running.
  - It calls a method on the system upon which the TSO/E user is running. It passes information to the method and returns information to the caller once the method has been dispatched and completed.
  - It provides a mechanism that allows a method to be invoked on all active systems within the systems and provides a consolidated response to the caller.

  The support is provided in JES2 and JES3.

- Workload manager, for which you can find more description in Chapter 4, “Workload Manager in OS/390 Release 3” on page 35.

If a particular TSO/E system is down, a session request using the the generic name can still be satisfied as another TSO/E system within the Parallel Sysplex can be selected as the session partner. Workload manager makes efficient use of system resources which are available and having a lower workload compared to other systems at the moment of logon processing.

If you plan to use TSO/E generic resource, then you must define it for a TSO/E system in member TSOKEYnn of SYS1.PARMLIB, as shown in Figure 13.

```
USERMAX=30,
RECONLIM=59,
BUFRSIZE=1000,
HIBFREXT=6600,
LOBFREXT=3300,
CHNLEN=4,
CONFTXT=NO,
SCRSIZE=9760,
GNAME=SCSTSO        <== NEW
```

Figure 13. Content of SYS1.PARMLIB Member TSOKEYnn

The name of the parameter is GNAME and its value is the name you choose when doing the logon process.

If you, after having started VTAM, start TSO/E with the MVS START command, include the use of TSO/E generic resource in the following way:

```
S TSO,,,(GNAME=SCSTSO)
```

SCSTSO is an example value of the parameter GNAME.

The generic resource name may have any 1-to-8 EBCDIC characters that must be unique from any real resource name in the network, and may not be “TSO.”

There is no default value, so if you do not specify one in member TSOKEY00 or on the MVS START command, you have no generic resource active on your system(s).

The logon string for a logon to one of the TSO/E systems within a Parallel Sysplex could then look like:

```
logon applid(scstso) data(paul)
```

Behind this generic name could be, for example, three TSO/E systems called SC50TS, SC51TS, and SC52TS.

As an operator, you can use the DISPLAY command to see which TSO/E system within the sysplex has TSO generic resource activated as follows:

```
D NET,ID=(SCSTSO)
```

where SCSTSO is the generic name for a number of TSO/E systems.

The output on the screen can be seen in Figure 14.

```
IST097I DISPLAY ACCEPTED
IST075I NAME = SCSTSO, TYPE = GENERIC RESOURCE
IST1359I MEMBER NAME        OWNING CP   SELECTABLE  APPC
IST1360I USIBMSC.SC50TS     SC50M          YES       NO
IST1360I USIBMSC.SC55TS     SC55M          YES       NO
IST1360I USIBMSC.SC49TS     SC49M          YES       NO
IST1360I USIBMSC.SC62TS     SC62M          YES       NO
IST1360I USIBMSC.SC47TS     SC47M          YES       NO
IST1360I USIBMSC.SC42TS     SC42M          YES       NO
IST1360I USIBMSC.SC48TS     SC48M          YES       NO
IST1360I USIBMSC.SC43TS     SC43M          YES       NO
IST314I END
```

Figure 14. Screen Output of Display GNAME Status

As in the past, with TSO/E Generic Resource, a particular TSO/E user can only be logged on to one system in the sysplex at a given time. This is a restriction which is part of the support provided by JES.

An attempt to logon to more than one system in a given sysplex with one user ID while not being in the state of reconnection results in TSO/E error message IKJ56467I. This also happens if the RECONNECT option is not specified on the logon panel (type S at that option).

3.2 Reconnection Support

Another problem that occurs if you lose connection to your TSO/E system has also been solved with TSO generic resource support.

Normally, the user address space is not terminated if the system programmer has specified a reconnect interval. If the user does logon processing to TSO/E before the reconnect interval has expired, he will be reconnected to his original address space. So he can do work from that point where he lost connection, and has minimal losses.

If the defined reconnect interval expires before the user logs on again with RECONNECT option S in the TSO/E logon panel, the original user address space is terminated.

The parameter for the reconnect interval is RECONLIM and it can be found in TSOKEYnn member of SYS1.PARMLIB (see Figure 13 on page 29). The value is set in minutes and can have a range between 0 and 32,767.

The problem that is solved with TSO/E generic resource support in OS/390 Release 3 is how to get a user reconnected to his original address space after losing connection. If a user loses connection, the user address space enters reconnect state, as usual. When the user logs on again to a TSO/E system via generic resource, the selection algorithms normally might assign the user to one of the other TSO/E systems within the sysplex. But an attempt to reconnect to a TSO/E system other than the original would be unsuccessful because the original address space does not exist on that system.

The way reconnection would be successful in that case in a sysplex environment is described in the following section. Let us take a closer look at TSO/E user PAUL who had a session on SC50TS, and then logged on using a generic resource name and lost his connection.

1. User PAUL tries to reconnect to his lost TSO/E session within a given sysplex using the generic resource name SCSTSO as an application id parameter.
2. VTAM then talks to the workload manager to let it make a decision as to which system to use for this expected new TSO/E user. After looking for the actual workload at that moment, workload manager decides to put a new TSO/E address space on system SC52TS.
3. VTAM invokes the TSO/E logon processor to set up the user address space for the “new” user. The user ID and password specified by the user has to be validated. During that process, TSO/E scans the terminal status block (TSB) chain to verify whether the user is already logged on or not.
4. In our sysplex environment, the TSB search fails on system SC52TS because a user address space does not exist on that system. As a result, TSO/E invokes JESXCF, using the IXZXTSOI service, to determine whether the user address space still exists anywhere in the sysplex.
5. Next, JESXCF calls a routine provided by TSO/E on that system where the user address space still exists if the user is known to JESXCF. If the user is not known to JESXCF, the call fails with a return code.
6. The routine scans the TSB chain for the matching user ID PAUL, recognizes the reconnect application name from the TSBRECAP field, and at that point it has two choices:
   - Return the application name to the TSO/E system that is handling the logon processing via JESXCF
   - Issue a return code with the indication that the appropriate user is not in the state to be reconnected.
7. If the reconnect application name is obtained successfully, TSO/E replaces the contents of TSBRECAP in the TSB of the new address space on system SC52TS to which the terminal has been temporarily assigned with the name obtained from the TSO/E system holding the old (disconnected) address space.
8. When TSO/VTAM receives control from TSO/E for a reconnect request, the reconnect application name in the TSB associated with the newly created address space is compared with the local terminal control address space (TCAS) LU name. If these two names differ, then VTAM performs a CLSDEST PASS, enabling the TCAS on system SC50TS to recognize the pass of the terminal session over the sysplex, eventually reconnecting the terminal session to the address space of user PAUL.

Refer to Figure 15 on page 32 for the logon flow.

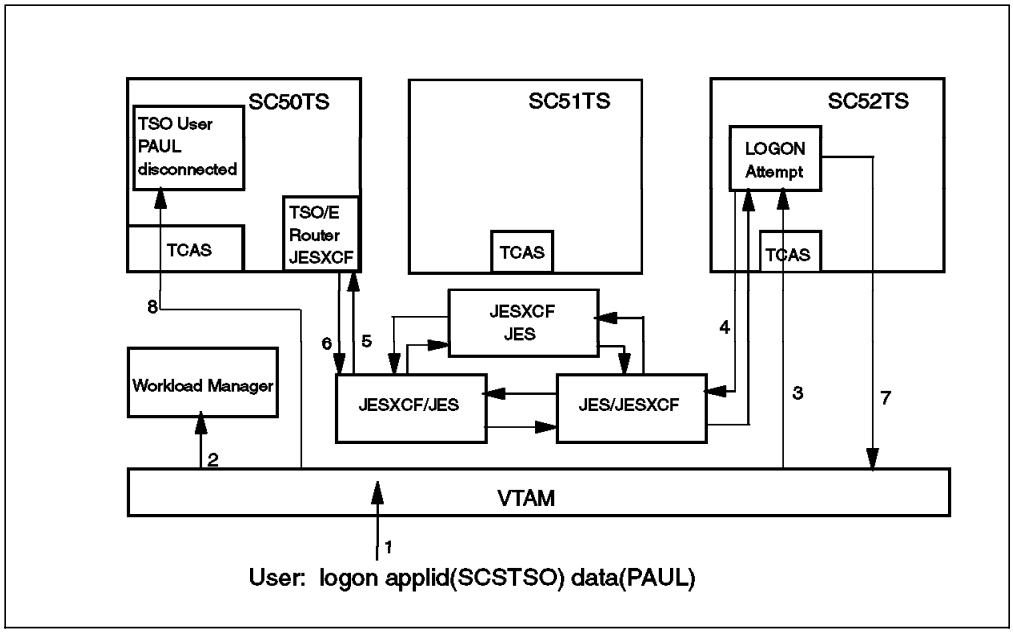


Figure 15. TSO/E Generic Resource Reconnection Support

3.3 Software Requirements

To use TSO/E generic resource, you must have at least the following software components (at specific levels):

- OS/390 Release 3 Base (including JESXCF and workload manager).
- OS/390 Release 3 JES2 or JES3.
- VTAM Version 4 Release 4.
- TSO/E Version 2 Release 5.0 with APAR OW23828. This APAR provides the base level TSO/E support for TSO/E logon reconnect.

3.4 Migration Considerations

If you migrate to TSO/E generic resource, you have to consider other system components also. These components are:

- Global resource serialization (GRS)
- Virtual telecommunication access method (VTAM)

The things you have to watch in GRS are described in 3.4.1, “GRS Requirements” on page 33.

In a sysplex with MVS images at a level before OS/390 Release 3, the IXZXTSOI service, used by TSO/E to communicate with JESXCF, does not allow communications with systems that are not at that specified level.

For the installation of TSO generic resource support the following migration path is suggested:

1. Installation of OS/390 Release 3 (without JES function), one system at a time until the entire sysplex is running at Release 3 level.
2. Installation of OS/390-JES2 Release 3 or OS/390-JES3 Release 3, one system at a time until the entire sysplex is running at JES2 or JES3 level of OS/390 Release 3.
3. Activate the TSO generic resource function.

3.4.1 GRS Requirements

First, take a look at GRSRNLnn member of SYS1.PARMLIB and enter the following resource name list definition statements (RNLDEF) to the system inclusion resource name list shown in Figure 16.

```
/*********************************************************************/
/*  SYSTEM INCLUSION RESOURCE NAME LIST - RNLDEF STATEMENTS         */
/*********************************************************************/

RNLDEF RNL(INCL) TYPE(GENERIC)
QNAME(SYSIKJUA)  RNAME(SYS1)

RNLDEF RNL(INCL) TYPE(GENERIC)
QNAME(SYSIKJBC)
```

Figure 16. TSO/GR Modifications of GRSRNLnn Member of SYS1.PARMLIB

The meaning of the QNAME parameters is as follows:

SYSIKJUA The TSO/E data set SYS1.UADS has to become a global resource. Its type has to be generic.

SYSIKJBC The TSO/E data set SYS1.BRODCAST has to become a global resource. Its type has to be generic.

You can find more information in *OS/390 MVS Planning: Global Resource Serialization*.

With the operator command shown in Figure 17, you can look at the RNL information.

```
D GRS,RES=(SYSIKJUA,*),RNL=I
 ISG343I 13.48.45 GRS STATUS 390
      LIST TYPE QNAME    RNAME
      INCL GEN  SPFEDIT
      INCL GEN  SYSDSN
      INCL GEN  SYSIKJBC
      INCL GEN  SYSIKJUA SYS1
      INCL GEN  SYSZVOLS
 QNAME    QNAME    QNAME    QNAME
 SYSIKJUA
```

Figure 17. MVS Display Command to show Resource Name List Information

Chapter 4. Workload Manager in OS/390 Release 3

This chapter describes the new capabilities of OS/390 Workload Manager (WLM) in OS/390 Release 3. A brief overview of the workload manager is presented together with the new capabilities.

WLM was first announced with MVS/ESA Version 5.1, and had only small improvements with MVS/ESA 5.2.2 and OS/390 Release 2.

OS/390 Release 3 brings important enhancements to WLM in three areas:

- Increased efficiency in managing different types of workloads (batch, TSO, and transactions).
- New workload types, such as DB2 SQL stored procedures, DSOM, TSO/E, and APPC.
- Improvements to help administer and operate WLM.

This chapter discusses the following topics:

- Introduction

  This section gives a conceptual view of WLM and lists the new services provided by WLM: more resources under the control of WLM and new entities now managed by WLM/SRM. Building on this, WLM is now able to manage new types of work.

- New Services

  This section is logically divided into two parts:

  - Resources now controlled by WLM such as I/O
  - New entities now managed by WLM, such as Business Unit of Work and Server Address Spaces

- New Workloads

  This section describes how WLM assists in meeting the business objectives of the following types of work:

  - DB2 Stored SQL Procedures
  - DSOM
  - TSO
  - APPC

- WLM Internal Changes

  This section lists some customer and IBM internal requirements satisfied in this release in order to improve the administration and operation of WLM.

- Migration Considerations

  This section includes related topics such as: changes in OS/390 MVS for WLM, and migration and compatibility considerations.

4.1 Introduction to Workload Manager

MVS workload management addresses the sysplex needs for:

- Managing workload distribution
- Load balancing
- The distribution of computing resources to competing workloads

Workload management introduces a sysplex-wide view of performance administration, monitoring, and management of workloads running under MVS. The MVS workload manager runs in a sysplex, with a workload manager function couple data set (WLM CDS) shared among all the systems. All the systems are informed of any change in the active service definition. The workload manager couple data set allows high availability: in case of a failure in the path or in the disk itself, the alternate data set is automatically activated. The MVS workload manager uses XCF services to communicate. There is no requirement for a coupling facility hardware unit; ESCON channel-to-channel (CTC) connections can be used.

Each processor in a sysplex uses a common service definition, or policy, which is stored in the shared workload manager couple data set, and may be refreshed by the service administrator or authorized system operator. Activating a new policy on one system in the sysplex results in all other systems activating the same new policy simultaneously.

4.1.1 Service Class

A service class is similar in concept to a performance group. It represents a grouping of work with similar resource requirements and performance requirements, which can be server address spaces, batch jobs, or CICS/IMS transactions. Each service class can have one or more *periods*. Each service class period has a *goal*.

4.1.2 Workloads

The workload manager is the OS/390 BCP component that controls the execution of different types of workloads, such as transactions or batch programs, and tries to meet an installation-defined service level or goal by making decisions on resource assignment and, when operating in a sysplex, placing the workload in the member of the sysplex that is most suitable to meet the defined goals. The workload manager and the SRM control the following:

- The way work is categorized into workloads
- The way work is managed prior to execution
- The way work is managed during execution
- The methods for obtaining performance reporting and analysis

Workloads are defined in terms of *business service level goals*, as shown in Figure 18 on page 37.

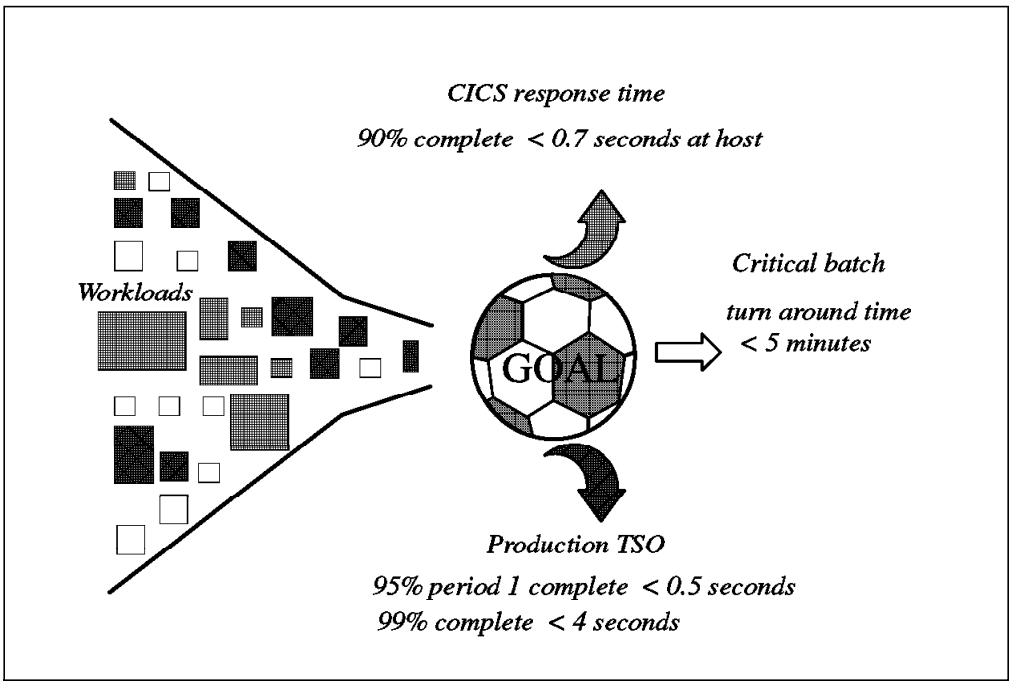


Figure 18. Workloads in Terms of Business Goals

4.1.3 Service Level Goals

WLM classifies work into service classes. Each service class comprises one or more periods and each period has a *goal*. The goals may be expressed in four ways, as shown in Figure 19.

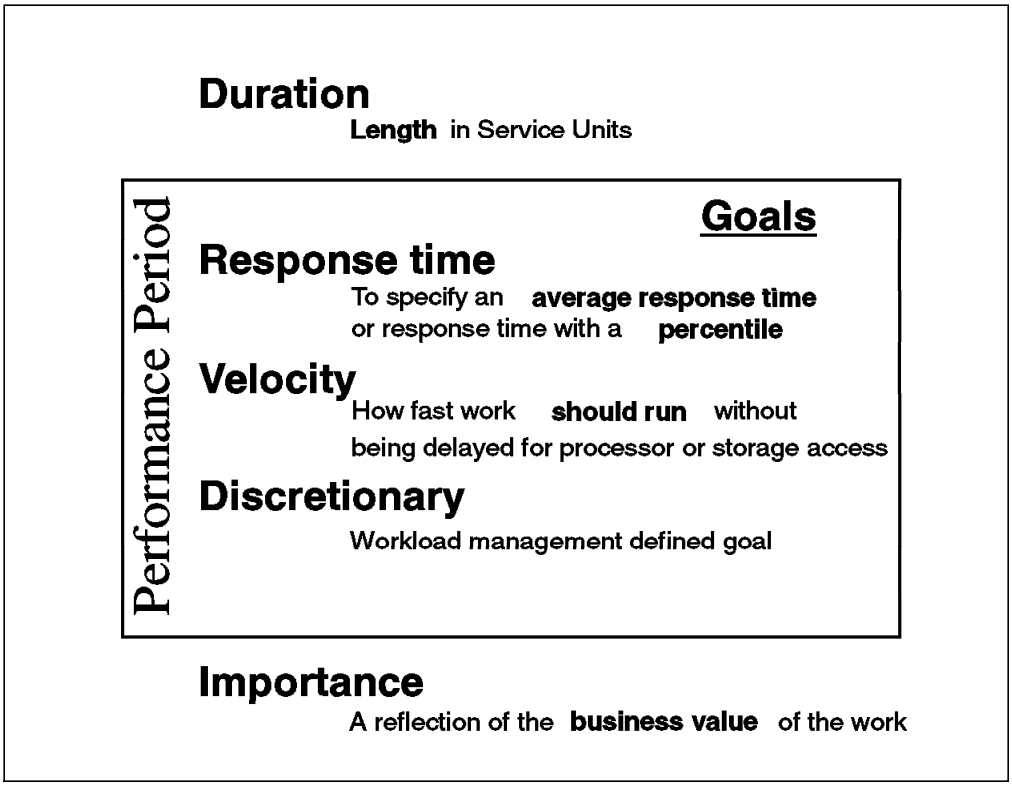


Figure 19. Workload Manager Definitions

4.1.3.1 Importance

The *importance* is the relative importance of the service class goal. Importance is used when work is not meeting its goal. It helps determine which service class should donate resources to a higher importance service class that is not achieving its goal.

Each goal has an associated importance level. The importance is expressed by a number between 1 (maximum importance) and 5 (minimum importance).

4.1.3.2 Performance Period

Each performance period has an *importance* that indicates how vital it is to meet the service goal.

The goals for each performance period may be expressed as:

Response time average Specifies what you want the average response time to be.

Response time percentile Specifies a response time and the percentage of total transactions you want to be below the specified response time.

Velocity Specifies the resource consumption rate.

Discretionary Lets the system do its best.

4.1.3.3 Duration

The duration specifies the length of a period in service units. If the work does not complete when the number of service units has been used, the work moves to the next performance period. Do not specify a duration for the last period. Multiple performance periods are not available for work in the IMS and CICS subsystem work environments.

4.1.3.4 Performance Index (PI)

When we define goals for a workload, we also need some feedback about whether the goals are being met or not. The achievement of each goal is measured by a ratio called the *performance index* (PI):

- A value less than 1 means that the system is achieving the goal.
- A very large performance index indicates that the service class is badly missing the goal.

The definition of the ratio depends on the type of goal:

Type of Goal	Performance Index Calculation
Average response time	Actual response time / Goal response time
Response time percentile	Actual percentage / Goal percentage
Velocity	Objective Velocity / Actual Velocity
Discretionary	Set to 0.81

Table 2. Performance Index Calculation

The performance index indicates how well a service class is doing in meeting its goal. The calculation is sysplex-wide (global performance index) and internal to each system in the sysplex (local performance index). Each service class period may have two types of PIs:

Sysplex PI A PI that represents global performance across all the systems in the sysplex.

Local PI A PI that represents performance in each local system of the sysplex. A service class may have more than one local PI if that service class has work executing on multiple systems in the sysplex.

Through services provided by SRM, performance data is periodically exchanged between sysplex systems in goal mode. The information is sent across the systems and allows SRM to construct an approximate view of the status of the sysplex through the calculation of sysplex PIs. The aggregate view enables SRM algorithms such as policy adjustment to make trade-offs within each individual system. This exchanged data includes:

- Response time information for each service class period, to allow each local SRM to calculate the sysplex PIs for each service class period
- CPU service consumption information for each resource group, to enforce globally the capping and protection algorithms
- Information identifying the active service policy

The SRM uses the PI value in the following ways:

- When the PI is equal to one, it means that the service class period is exactly meeting its goal.
- When the PI is greater than one, it means that the service class period is missing its goal.
- When the PI is less than one, it means that the service class period is exceeding its goal.

RMF reports the actual response time, the velocity attainment, the performance index, and the goals.

4.1.3.5 Donor and Receiver Algorithm

In order to meet the goals and service rate group requirements, the policy adjustment routine determines:

Receiver A receiver is the service class period SRM is considering helping. SRM helps only one receiver during each policy interval, although it may access multiple receivers before finding one to help.

Donor A donor is a service class period that donates resources to the receiver. Multiple donors may donate multiple resources to a single receiver during one policy interval.

Periodically (several times per second), the following cycle is done:

1. Select all service class periods whose global performance index is greater than 1.
2. Order the service class periods by importance so the most important are on top of the list and make a second list within this by performance index with the worse performers at the top.
3. Search the list for the first service class that is delayed for a resource that WLM has control over.
4. Search for a donor.

A donor is a significant user of a resource needed by a receiver and:

- Is less important than the receiver or
- Is more important than the receiver, but is over-achieving its goal

If receiver and donor are found, the receiver's priority is increased, and the donor's priority is decreased for the resource, and the cycle ends.

If SRM is unable to find a donor-receiver couple in the list, or the list is empty, the process is repeated in each member of the sysplex using local performance indexes.

Note: Before OS/390 Release 3, the only resources that could be donated and received were processor storage and CPU. With OS/390 Release 3, I/O is a resource for the donor/receiver algorithms. See 4.3, "Sysplex I/O Priority Management" on page 44.

The donor and receiver are rarely a single address space. WLM usually refers to a receiver service class period, not a receiver address space.

WLM also ensures that the reallocation of resources from the donors to the receivers will do more good than harm. The term used is:

Net value A receiver is only helped by a specific donor if there is projected to be sufficient *net value* to the resource allocation. This net value assessment makes sure that using a donor to help a receiver results in more projected benefit to the receiver than projected harm to the donor.

4.2 OS/390 Release 3 Enhancements Overview

New work manager delays are available as RMF expands WLM-related reports of Monitor III and the postprocessor to display these values. Also, DASD response time values on service class level are shown. In addition, the following items are enhancements to WLM for OS/390 Release 3:

In support of managing resources:

- Sysplex I/O priority management
- Business unit of work management
- Server address space management

In support of work environments that now exploit workload management:

- DB2 stored SQL procedures support
- DSOM sysplex workload management
- DCE sysplex workload management
- TSO sysplex workload management
- APPC/MVS support for generic resources

4.2.1 Sysplex I/O Priority Management

Previous releases of SRM and WLM only assign the following resources to workloads:

Physical CPU, storage, I/O

Logical Tasks, multiprogramming level, control blocks, and so on

For better goal attainment, OS/390 Release 3 WLM has added a new resource, input/output (I/O), to the previously managed physical resources of CPU and storage.

This support introduces sysplex-wide goal-oriented management of I/O priorities driven by WLM knowledge of goals for work, and the business importance of those goals. The I/O priority for each request is separated from the dispatching priority associated with the requesting dispatchable unit, allowing independent algorithmic adjustments.

I/O priorities can be determined dynamically, based upon the business value of that I/O to the user.

I/O priorities are maintained at a WLM service class period level, and are synchronized across the systems in a Parallel Sysplex. The I/O priority associated with each I/O request is passed to the DASD control units.

This support does not eliminate shared device contention, but should improve the ability of the system to manage that contention when it occurs, allowing for more effective utilization of shared devices. For more details, see 4.3, “Sysplex I/O Priority Management” on page 44.

4.2.2 Business Unit of Work Management

OS/390 Release 3 extends the *enclave* concept introduced in the MVS/ESA SP 5.2 release as a means for managing a large number of SRBs scheduled to an address space. This support addressed distributed DB2 with a large number of tasks within an address space and an inability to manage the execution of the SRBs toward WLM-defined goals for work.

The new *extended enclaves* are called *business units of work*. In the rest of this redbook the terms *enclave* and *business unit of work* are synonyms except when referring to MVS/ESA V5.2.0 enclaves.

This support improves the ability of the system to manage client/server workloads by enabling WLM to manage the individual business units of work within a server address space. WLM and SRM can make decisions between dispatchable units associated with different business units of work, providing an additional level of manageability and control solely through the existing WLM view of goals for work. This allows control of resource consumption for units of work that span multiple address spaces, and address spaces that serve multiple units of work simultaneously. For more details, see 4.4, “Business Unit of Work Management” on page 50.

4.2.3 Server Address Space Management

This support provides a mechanism for dynamic management of application server address spaces. Servers can be created and destroyed as needed, without any external controls. New algorithms are provided to assess the value of having a given quantity of address spaces, balancing the importance of work in the execution backlog against the availability of system resources.

4.2.4 DB2 Stored SQL Procedures Support

WLM uses server address space management to adjust the number of address spaces for the different service classes using that function.

With DB2 4.1, implementation of stored SQL procedures allows all requests for SQL program calls, whether via distributed DB2 or local attachment, to be funnelled to a single DB2 address space (SPAS). In the new SPAS, a variable number of tasks are attached. The number of tasks is defined in the DB2 startup parameters. When a stored SQL procedure request arrives, the request is queued by DB2 and then subsequently routed to one of the available tasks within the SPAS.

This support provides facilities for queueing, scheduling, and control of DB2 stored SQL procedures and allows WLM to manage the backlog of stored SQL procedure requests, maintain knowledge about server address spaces under control of WLM, and provide the necessary control mechanisms to initiate execution of that work at the appropriate time.

4.2.5 DSOM Sysplex Workload Management

WLM uses server address space management to adjust the number of address spaces for the different service classes using that function.

4.2.6 DCE Sysplex Workload Management

This support integrates DCE into the Parallel Sysplex workload management philosophy and environment. With the necessary corequisite enhancements to DCE, the multisystem, multi-address space nature of MVS is applied to server applications in an MVS/DCE Parallel Sysplex configuration.

This provides the capability to:

- Distribute DCE connectionless RPC requests across systems in a multisystem sysplex, applying WLM sysplex-wide knowledge of available capacity.
- Dynamically create and destroy DCE server address spaces, adapting to demands of the workload based on the business importance of that work and the availability of system resources.

4.2.7 TSO Sysplex Workload Management

TSO users can now be logged on to any member in the sysplex. With the help of VTAM generic resources, WLM places TSO sessions on the most appropriate member of the sysplex.

This support extends the existing WLM support of VTAM generic resources and allows the following:

- TSO user sessions can be transparently distributed across a subset of MVS system images within a sysplex. Logon requests can be distributed across a VTAM-eligible list of MVS system images by using WLM sysplex-wide knowledge.
- TSO users need not be aware of the sysplex topology.
- Support for user logon reconnect, which directs secondary logon requests (after a session failure) back to the proper MVS system image.

4.2.8 APPC/MVS Support for Generic Resources

APPC also uses VTAM generic resources. WLM also decides on session placement. The VTAM generic resources capability for APPC/MVS was introduced in MVS/ESA SP 5.3.0. The initial design assumption of a single generic resource name per address space is removed with this release.

This support allows application servers that may be spread across multiple systems, to take advantage of the Parallel Sysplex growth potential and high availability characteristics.

4.2.9 OS/390 Release 3 Evolution

Figure 20 shows the services of WLM up to OS/390 Release 2 (white dots), the new users and services (underlined in the figure), and the new supported features (black dots).

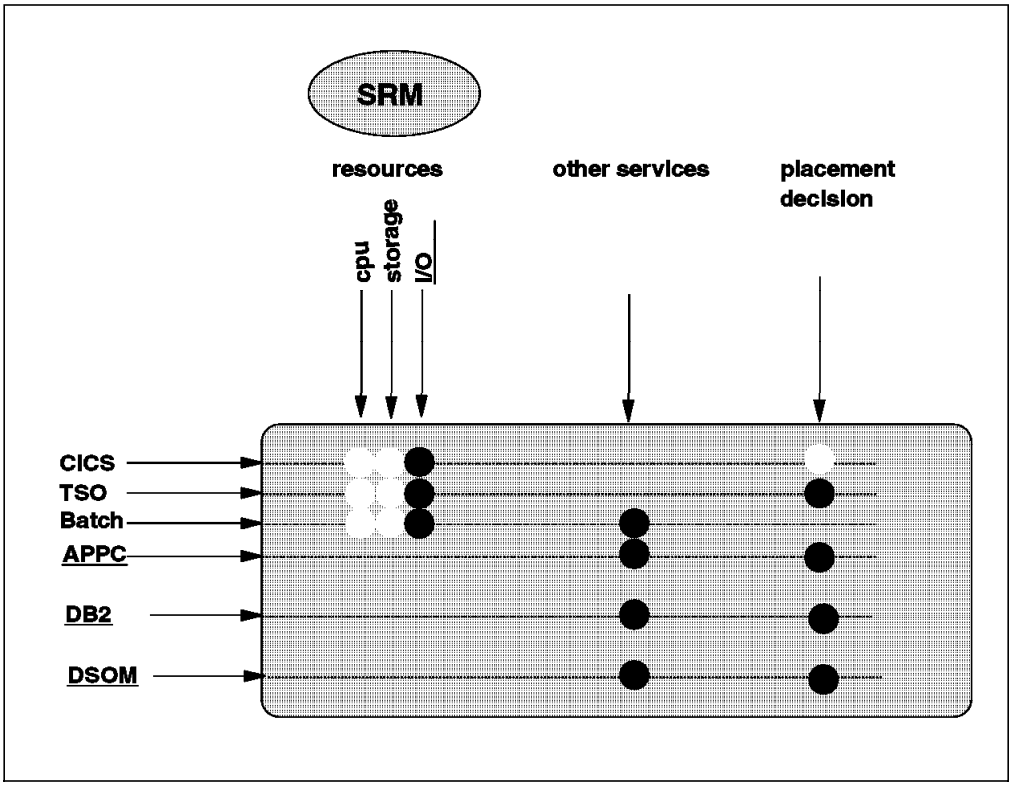


Figure 20. WLM Evolution

Note:

- I/O management by WLM allows better control over current users such as CICS and IMS transactions, TSO, and batch.

- Placement decisions are mostly (but not exclusively) taken at VTAM generic resource resolution time.

4.3 Sysplex I/O Priority Management

In past releases of MVS and OS/390, CPU and storage were the resources used by SRM for goal mode management. The I/O subsystem did not undergo any major redesign since the announcement of XA architecture.

OS/390 Release 3 WLM introduces changes for the whole logical path of the I/O.

One important feature of OS/390 Release 3 WLM is the ability to set priorities in the UCB queue. Queueing at the UCB level happens when a program attempts an I/O and the device is being used by another program of the same MVS image.

Previously, the I/O supervisor (IOS) would organize the queue according to the specification of the IOQ parameter in IEAIPSxx: it could be in arrival order (IOQ = FIFO) or according to priorities (IOQ = PRTY). If PRTY was specified, a *fixed* priority could be specified (IOP=xx) in IEAIPSxx. By default, the priority was that of the address space that attempted the I/O.

With MVS/ESA V5.1, the situation changed slightly: when WLM was in *compatibility* mode, the user had a choice of FIFO or priority queueing. However, in *goal* mode, FIFO queueing is not selectable any more, as shown in Figure 21. On the other hand, IOP is not honored. Therefore, the UCB queue is managed in priority mode and the priority is equal to the address space dispatching priority.

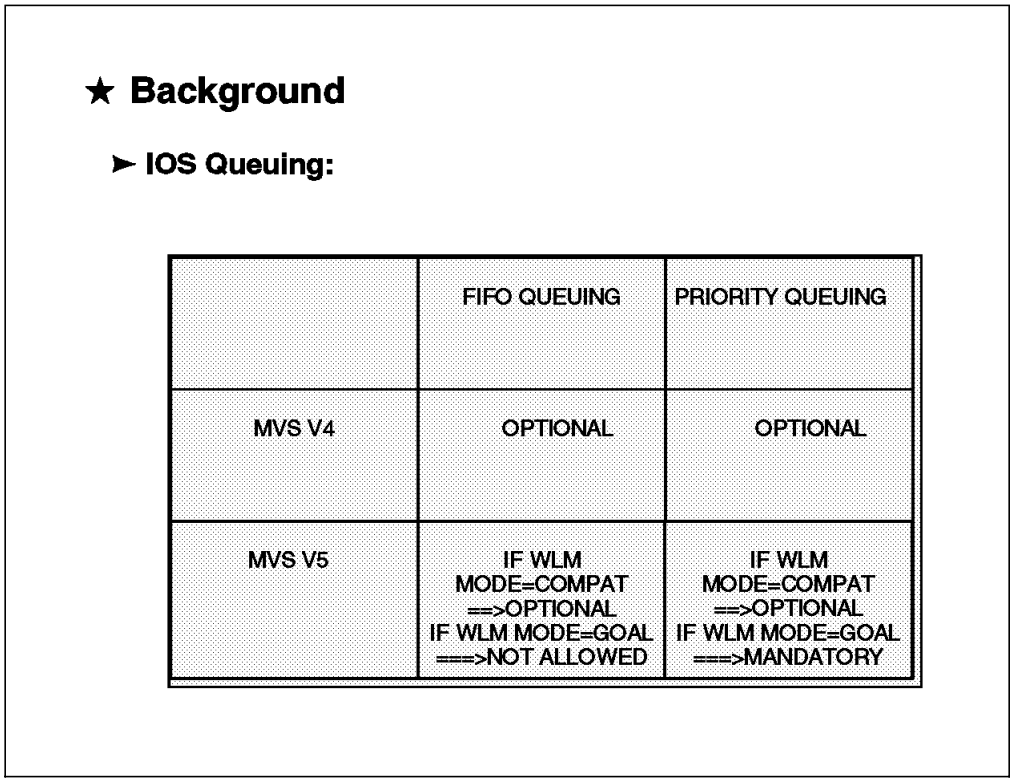


Figure 21. I/O Queueing in MVS

Related I/O features: Two other features relate to queueing, as follows:

I/O interrupt priority With XA architecture, there are two ways of I/O post processing:

- Via an I/O interrupt
- Via test pending interrupt (TPI)

With TPI, interrupts may have a priority. Only paging I/Os take advantage of the interrupt priority.

I/O scheduling in DB2 Version 3 Allows execution of synchronous I/O and prefetch read under the user's address space, thus taking advantage of MVS I/O queueing facilities.

4.3.1 I/O Management

Figure 22, in the upper left corner shows the different software layers involved in an I/O request. Every layer has its own programming interface, and the user may choose to work at any layer.

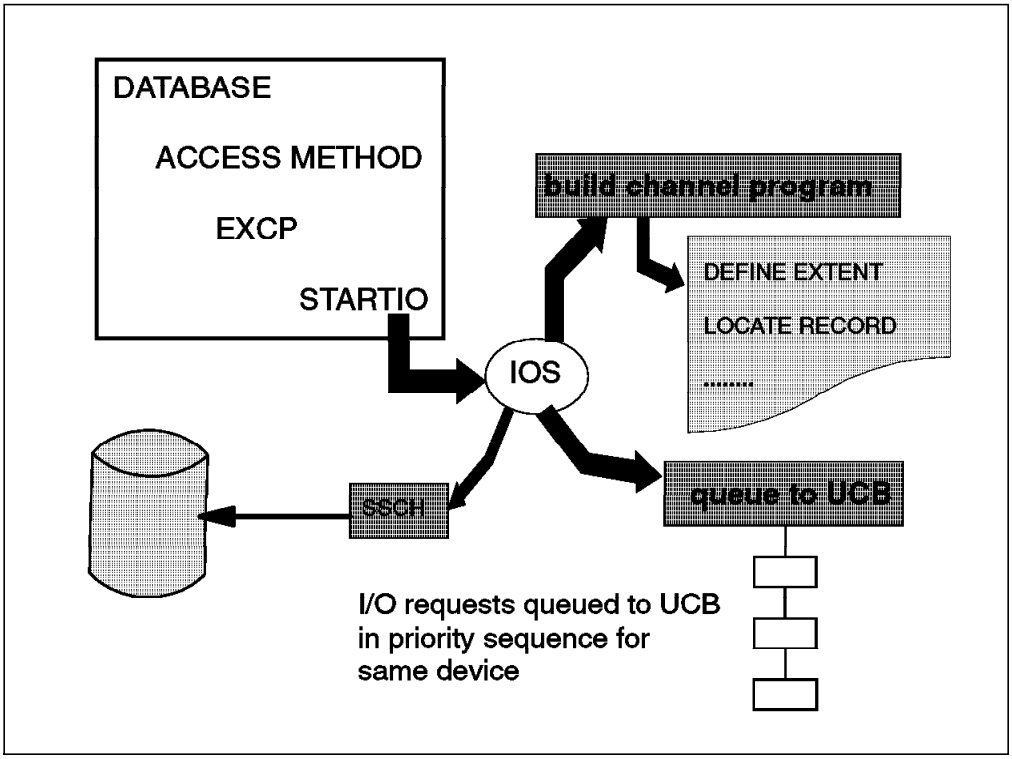


Figure 22. I/O Drivers Background

Very briefly, the different layers provide the following services:

Database The database manager is responsible for data integrity when many simultaneous transactions access the same data; for example, serialization at record level, logging, two phase commit, and so on.

Access method The user works with records. The access method manages buffers. Serialization is at the data set level.

EXCP The user works with blocks, and builds a model of the channel program.

STARTIO This macro invokes IOS.

IOS has, among others, the following functions:

- Page fix the buffers
- Build channel program from model
- Add channel program prefix (DEFINE EXTENT and LOCATE RECORD CCWs)
- Build IOQE
- If UCB Busy, enqueue IOQE. When UCB available:
  - Execute Start Sub-channel instruction (SSCH)
  - Process I/O interrupt
  - Post the requester

4.3.2 Changes to the I/O Process with Release 3

Sysplex I/O priority management in OS/390 Release 3 focuses solely on DASD devices. As stated in the design issues, DASD is the predominant source of device contention for workloads.

The goals of this support are:

- To separate work dispatching priority from I/O scheduling priority.
- To manage contention at the sysplex level.
- The SRM provides algorithms for assigning I/O priorities to the various workloads, based upon information gathered through sampling techniques and data obtained from the IOS component.
- Collected information and the resulting I/O priority adjustments are distributed across systems in a sysplex using existing WLM cross-system communications (XCF) services.
- For each I/O request, IOS sets the request priority based upon a value established by SRM for the requesting work unit. For supported devices, the priority value is then passed on to the I/O subsystem by the I/O device support logic for management purposes within the control units.

The following changes have been introduced in the I/O process:

- IOS enqueues requests on the UCB queue by I/O priority.
- The I/O priority of an enclave or an address space is adjusted by SRM. Neither the user nor the WLM administrator have control on I/O priority.

4.3.2.1 Assigning a Priority to I/O Requests

Each I/O request is performed by or on behalf of an MVS dispatchable unit, either a TCB or an SRB. The dispatchable units are associated with:

Address spaces ASCBs for TSO and batch address spaces.

Enclaves ENCBs for Distributed DB2, the new DCE, DSOM, and DB2 stored SQL procedures environment enclaves.

Each ASCB (for address space-oriented work) and ENCB (for enclave-oriented work) is associated with an SRM service class period, which becomes the focal point of SRM management algorithms.

For the purposes of managing I/O, the SRM controls a small fixed number of I/O priority levels as follows:

- I/O priorities are managed in the range of X′F8′ through X′FF′, where:

  X′FF′ Reserved for high priority system address spaces.

  X′FE′ Reserved for started tasks.

  X′F8′ Reserved for address spaces with a discretionary goal.

- The range of priorities between X′F9′ and X′FD′ is dynamically assigned to address spaces and enclaves, as determined by the SRM algorithms.

Each service class period maps to an I/O priority level. This mapping is dynamically adjusted based upon how each service class period is meeting its goals and how the I/O contributes to the success or failure. The assigned I/O priority is kept by the SRM in each ASCB and ENCB. When IOS starts the I/O and places the I/O request on the UCB queue for the target device, as shown in Figure 22 on page 45, the priority of the request (from the associated ASCB or ENCB) is saved in the I/O queue element and is used to place the request in priority sequence with respect to other pending requests for the same device.

4.3.2.2 Execution Requirements for I/O Priority Management

When the I/O request is to be executed, the following conditions must be met:

- The target device must support I/O priority queueing.
- The device is available.
- The highest priority request queued to the target UCB is used.
- A data manager exit for DFSMS, for example modifies the channel program to be executed by rebuilding the define extent parameter area to include a new extension containing the I/O priority field, and modifies the define extent CCW in that channel program to point to the new parameter area.
- The I/O priority associated with the request is transferred to the control unit for management purposes.

Note: The supporting DASD control units must be able to support a configuration consisting of MVS systems in WLM goal mode, MVS systems in WLM compatibility mode, and non-MVS systems incapable of supplying I/O priorities. For MVS systems in compatibility mode or MVS systems not at the OS/390 Release 3 level, the new define extent parameters are not passed to control units supporting the new architecture. For I/O requests not having an associated I/O priority, the supporting control units are responsible for setting a default priority. The use of a default priority effectively results in FIFO queueing of requests within the control units. For coexistence with systems operating in WLM goal mode, the default I/O priority assigned by supporting control units must be within the range of priorities in use by WLM.

4.3.3 I/O Priority Management Decisions

To manage I/O priorities, SRM needs information on the utilization of each device. When SRM wants to increase an address space or enclave I/O priority, it does it with a donor / receiver logic (see 4.1.3.5, “Donor and Receiver Algorithm” on page 39). So the SRM must be able to determine if:

- A unit of work can significantly improve its goal achievement by raising its I/O priority (potential receiver)

- Lowering the I/O priority of unit of work A (potential donor) can improve the achievement of unit of work B (receiver).

SRM needs two types of information to make decisions on I/O priority management, as shown in Figure 23:

Device usage SRM can make decisions by comparing the level of device usage across the range of I/O priorities, and projecting the amount of device usage that may occur if I/O priorities are altered.

The device usage time is the sum of device connect time and device disconnect time for each I/O request, as measured by the I/O subsystem.

Note: These timings are obtained by IOS from the Channel Measurement Block (CMB) at the completion of each I/O request, and are accumulated by IOS in the ASCB or ENCB associated with the requesting dispatchable unit.

Device delay SRM can make decisions by comparing the level of device contention across the range of I/O priorities, and projecting the amount of contention that may occur if I/O priorities are altered.

This device delay time is the sum of the time each request was delayed by IOS (IOS queue time), the time delayed in the I/O subsystem (pending time), and control unit delays (queue time), as shown in Figure 23.

Note: The I/O subsystem and control unit delays are obtained by IOS from the Channel Measurement Block (CMB) at the completion of each I/O request, and are accumulated by IOS in the ASCB or ENCB associated with the requesting dispatchable unit. The IOS queue time is obtained by timed sampling of the IOS queues by the SRM sampling routine running out of the WLM address space.

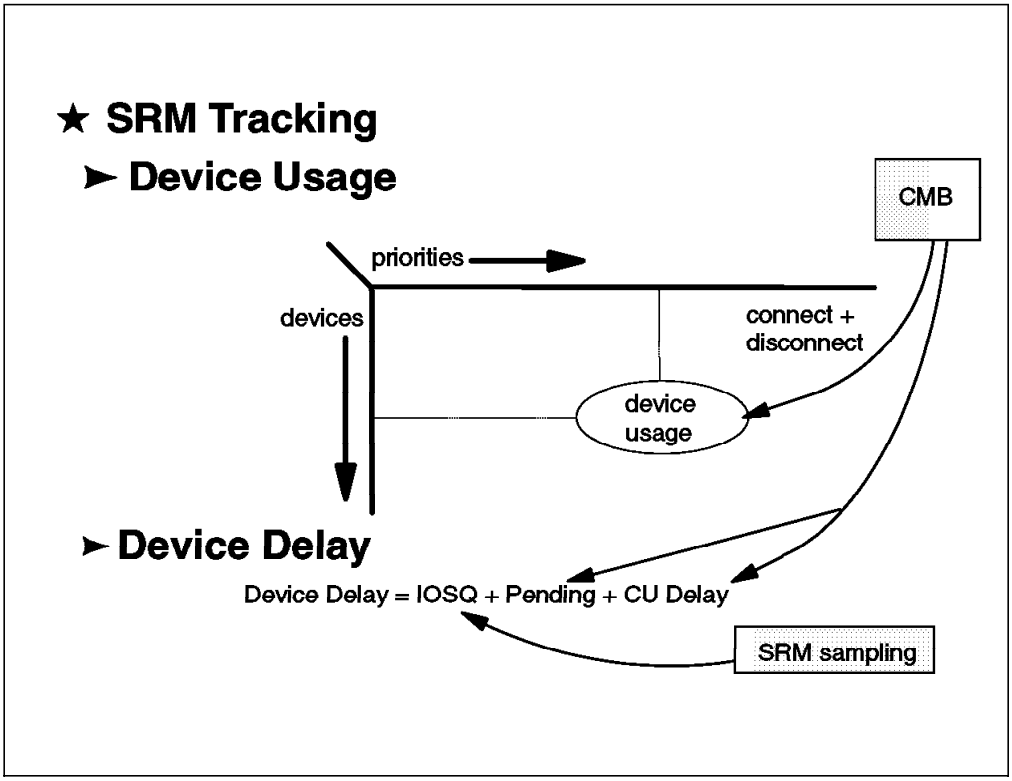


Figure 23. SRM Tracking

4.3.4 Managing Service Class Delays

SRM makes decisions based on service class period achievement. I/O priority adjustments are driven by SRM detection that:

- The goal for a service class period is not being achieved.
- The service class period is the highest importance period that is not being achieved.
- The service class period is the one with I/O delays that make up a significant portion of the total measured or estimated delays.

SRM consolidates information from each ASCB and ENCB from the whole sysplex. The consolidation takes place every 10 seconds. To make an adjustment to the I/O priority of a service class period, all the following conditions must be met:

1. The service class period has the highest importance among all the service class periods needing help.
2. I/O delays are a significant part of total delays.
3. A donor service class period can be found.

If the donor is not using the same set of devices as the receiver, lowering its priority has no effect on the receiver, so a *clustering algorithm* has been included to determine users of similar sets of devices.

4.3.4.1 Changing I/O Priority Considerations

A change in I/O priority for a service class must be propagated to the whole sysplex. The system making that decision has the responsibility for taking the following actions:

- New work on the local system (jobs and enclaves) inherits the updated I/O priorities as they are created by updating the SRM mapping of service class periods to I/O priorities.
- Cause all future I/O requests from previously active work units to use the updated priorities by updating the I/O priority field within the ASCB for each active address space and within the ENCB for each active enclave.
- Propagate the previous two actions to the SRM on each system in the sysplex by distributing the revised mapping of service class periods to I/O priorities using XCF services.

4.3.5 Reporting I/O Priority Delays with RMF

The RMF Workload Activity report contains additional fields to report the percentage of time the service class period has been delayed due to I/O. The assigned priorities are visible through the RMF Monitor II interface, which shows a snapshot of the assigned priority at the time the display is generated.

Workload activity reporting by RMF is extended to include reporting, by service class period basis, by address space, and by enclave for the collected I/O delay.

4.4 Business Unit of Work Management

The primary means by which resource consumption is controlled is at address space level, and since address spaces can be swapped in or out, they can be given more or less processor storage, and their priorities can be adjusted, giving more or less access to the CPU for associated local SRBs and TCBs. This approach is effective for traditional workloads such as TSO and batch, where the unit of work executes entirely in the address space. An extension to this is work executed in cross-memory, where accounting of the consumption is done in the calling address space (for example DB2).

These address space-oriented controls are, however, ineffective when applied to situations where:

- Execution of a given unit of work crosses address space boundaries (as in Figure 26 on page 53) by switching dispatchable units
- A given address space executes multiple units of work concurrently

OS/390 Release 3 solves this problem by extending the enclave concept. The new extended enclaves are called business units of work, and extend the enclave support to TCBs. This concept is intended to support the new type of MVS work: the OpenEdition and client/server processing application platforms. In the remainder of this chapter, the terms enclave and business unit of work are synonymous, except when referring to MVS/ESA SP 5.2 enclaves.

4.4.1 Business Unit of Work Enclaves

A new business unit of work (SRM transaction) is started whenever work is begun that has not declared itself to be a continuation of an existing transaction. The enclave create service, IWMECREA, is extended to support two new types of TCB enclaves. A new keyword parameter, TYPE, indicates the type of enclave to be created:

Independent An enclave that represents a new transaction.

Independent indicates that a new enclave is being created, which may have SRBs, TCBs, or both. An independent enclave is one which itself comprises a business unit of work that is classified and reported on as an entity.

Dependent A enclave that represents the continuation of an existing transaction and is being created for TCB management purposes, where the enclave represents a continuation of the business unit of work associated with the current home address space.

This type of enclave is created, for example, to have TCB management and service accumulation on behalf of a batch job or TSO user address space, in some other address space.

An enclave type of MONENV may also be created, which indicates that the enclave is being created for TCB management purposes, where the enclave represents a continuation of the business unit of work associated with a WLM monitoring environment performance block (PB). This type of enclave is created, for example, to have TCB management and service accumulation on behalf of a CICS or IMS transaction in some other address space.

Dependent enclaves derive their service class from the original transaction, and service consumed is considered part of the originating transaction for period switch and reporting purposes.

Note: Independent is the default when an enclave is created.

4.4.2 Resource Consumption Before MVS/ESA SP 5.2

Up to and including MVS/ESA V5.1.0, the only recipient of resource consumption was the home address space. This meant that, in a client/server configuration, the server address space, even working on behalf of a client, would be charged for its resource consumption and be managed according to its own goals and not to a client′s goals. The drawback of this is that the server can execute tasks on behalf of clients with different goals, but all tasks execute subject to the priority and the goals of the server address space.

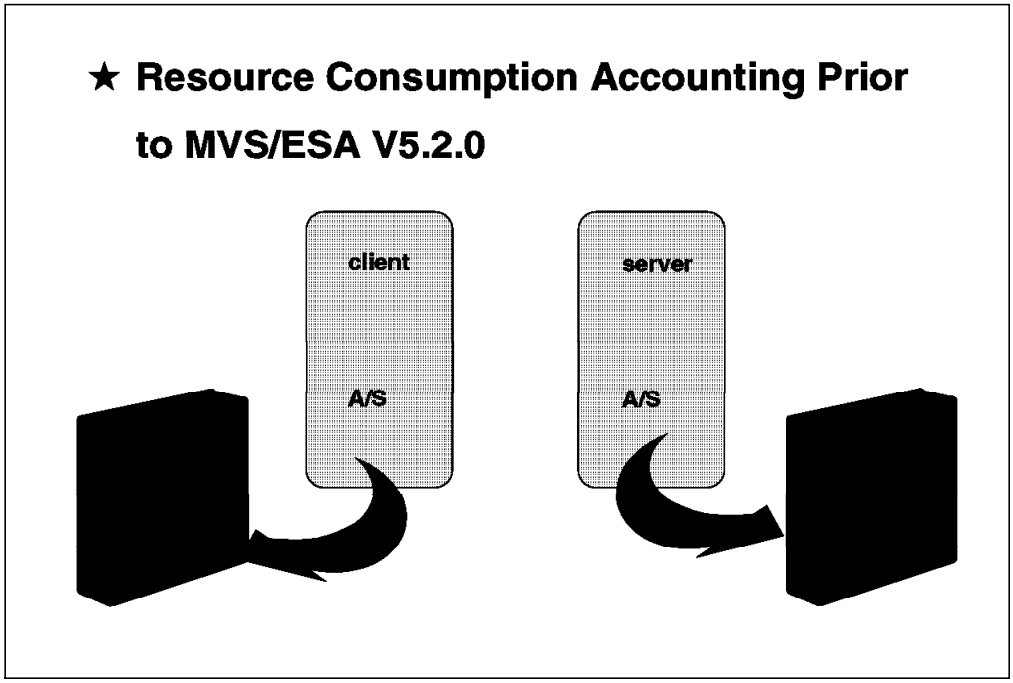


Figure 24. Resource Consumption in MVS/ESA SP 5.1

4.4.3 Enclaves With MVS/ESA SP 5.2

A partial solution to this problem was given in MVS/ESA V5.2.0 with the introduction of enclaves, as shown in Figure 25:

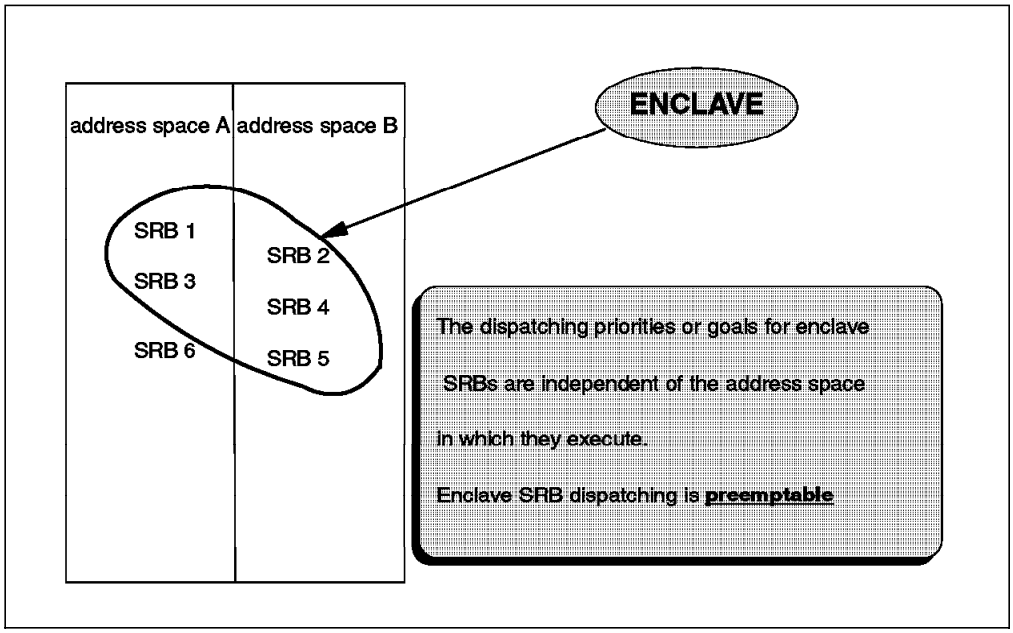


Figure 25. Enclaves With MVS/ESA SP 5.2

Enclaves were announced in MVS/ESA V5.2.0. An enclave is a set of SRBs belonging to the same or to different address spaces. Contrary to previous local SRBs who inherited their priority from the home address space, an enclave has its own goal independent of the home address space. Service is accumulated

for the whole enclave. Another characteristic of the enclave is that the SRBs are preemptible.

4.4.4 Enclaves With OS/390 Release 3

Whereas enclaves in MVS/ESA V5.2.0 could include only SRBs, in OS/390 Release 3 they can include both TCBs and SRBs. In Figure 26, enclave 1 includes SRB A1 of address space A and TCB B2 and TCB B3 of address space B. Enclave 2 includes SRB A2 from address space A, SRB B1 and TCB B1 from address space B and SRB C2 from address space C.

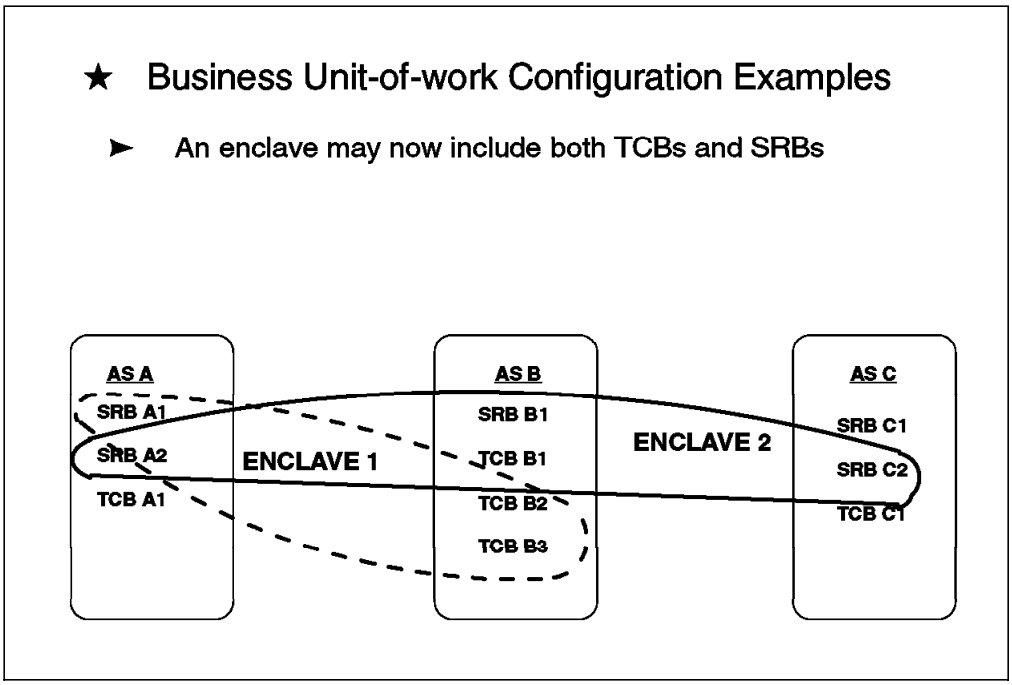


Figure 26. Enclave Examples

Note: A dispatchable unit (TCB or SRB) can belong to only one enclave at a given instant.

The rules that govern the business units of work are:

- TCBs and SRBs of an address space may belong to different enclaves. In Figure 26, address space B has two TCBs that belong to enclave 1 and one TCB and one SRB belonging to enclave 2.
- Resource consumption is accumulated for both the enclave and the address space. A dependent enclave is a logical continuation of the transaction already active in a client's address space. Therefore, CPU and MSO service for a dependent enclave is included in the SMF 30 record of the owning address space, and in the SMF 72 record for the address space's transaction. MSO service for the enclave is calculated based on the frame count of the owning address space, not on frame usage in the address space(s) where the enclave is executing. So you probably want to assign a response goal to the enclave, as it corresponds to a business unit of work and a velocity or a discretionary goal of the server address space.
- Business units of work have performance period capability. This means that the system can react to long-running or resource-consuming units and vary their goal on the fly.

- Services are provided to dynamically connect/disconnect a TCB or an SRB to or from an enclave. This allows the existence of a server address space with a permanently created TCB (see 4.5, “Server Address Space Management” on page 55), with the TCB dynamically switching from one business unit of work to another.
- Association of a TCB with an enclave triggers association of all daughter tasks.

Note: For an independent enclave, CPU service is included in the SMF 30 record of the owning address space, and in the SMF 72 record for the enclave′s service class or performance group period. MSO service is not calculated for an independent enclave. For both dependent and independent enclaves, IOC service is included in the SMF 30 and 72 records associated with the address space where the enclave work is executing. SRB service for enclaves is always zero.

4.4.5 New Business Unit of Work Services

To support business units of work, WLM provides the following set of services to work managers:

IWMECREA The creation of a business unit of work is done using the existing WLM IWMECREA service. A new parameter (TYPE) lets you specify if the business unit of work is new (TYPE=INDEPENDENT) or is a continuation of an existing business unit of work. See 4.4.1, “Business Unit of Work Enclaves” on page 51.

IWMEJOIN The new service IWMEJOIN is provided that permits a TCB to be associated with a previously existing enclave. Once joined, management of that TCB is governed by the service class period associated with the enclave, and system resource consumption under that TCB is accumulated to the enclave (rather than the address space in which the TCB exists).

IWMELEAV The new service IWMELEAV is provided that permits disassociation of a TCB from its enclave. Once departed, the TCB once again becomes associated with the address space in which it used to exist, and is governed by the service class period associated with that address space.

IWMESQRY The new service IWMESQRY is provided and it determines if the active dispatchable unit is currently associated with an enclave. This service is used to determine if:

- A function that requires execution under an enclave needs a new enclave to be created
- An existing enclave can merely be joined

4.4.6 Business Unit of Work Exploiters

Any work manager using server address spaces is a potential user of business unit of work management. Two identified users are:

- DB2 Stored SQL Procedures
- DSOM

4.5 Server Address Space Management

Some work managers, such as DB2 or DSOM, provide services through other address spaces. DB2 does this for SQL stored procedures. The services may be provided by one or more address spaces. Each address space may process transactions in single thread (one at a time) or in multithread (more than one in parallel); the processing mode is up to the work manager and is beyond the scope of WLM.

WLM with OS/390 Release 3 can take advantage of both *business unit of work* and *server address space management* to accomplish a service level objective for the transactions.

The design of server address space management addresses the following questions:

- If the objectives are not met, can an additional server improve the performance index?
- If there is a resource constraint (CPU or storage) in the system, how do you reduce the activity of server address spaces?
- When should the number of server address spaces be decreased?
- Will the creation of a new server address space adversely impact the performance of other more important goals?
- How do you report resource consumption and response time of units of work that span more than one business unit of work. See also 4.4.4, “Enclaves With OS/390 Release 3” on page 53.

The work managers, DCE, DSOM, and DB2 stored SQL procedures, use WLM services to permit workflow from their network attachment address spaces, through WLM, and into server address spaces for execution. As shown in Figure 27 on page 56, the three components that participate in meeting the objectives are:

Work manager A subsystem or work manager routes transactions to WLM, identifies the server JCL to WLM, and provides shell services for applications.

WLM Creates or deletes server address spaces

Directs work into the server address spaces

Decides when a new server address space has to be created

Reports on goal achievement

SRM Monitors the performance of *transaction environments*, gathers performance information, and takes decisions for SRM or WLM actions

The server address spaces managed by WLM, shown in Figure 27 on page 56, are created as started tasks, to serve work associated with an application environment. The workload requirements for a given application may determine that multiple server address spaces should be activated. Figure 27 on page 56 shows five server address spaces to support the workload. The address spaces making up an application environment are identical in terms of capabilities, because they are all created using the same JCL procedure, and are

interchangeable. An application request can be processed by any of the address spaces comprising the application environment, with identical results.

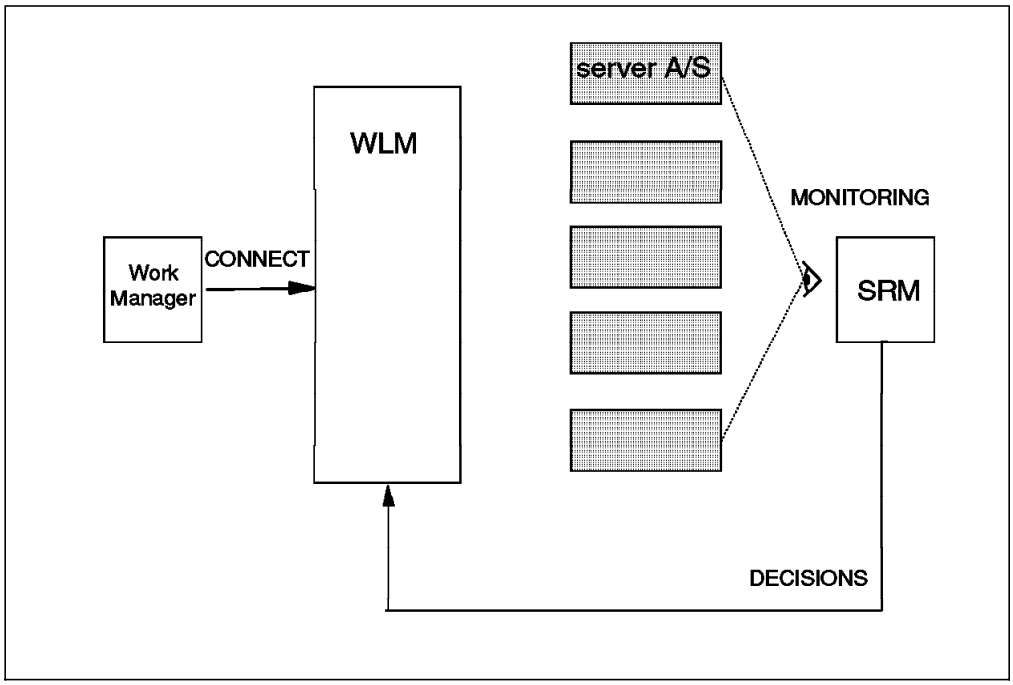


Figure 27. Server Address Space Management Components

The functions of the server address spaces are determined by the set of resources allocated during address space creation, as defined in the startup JCL. This can include definitions of:

- The accessible application program
- Compiler runtime libraries
- Connections to file systems
- The initial program to be given control

Within each address space an environmental *shell* is provided by the associated work manager. This shell may be either the job step program (as with DB2 stored SQL procedures), or run time services invoked by the executing application programs.

To use server address space management, the following steps are necessary:

- The WLM administrator must *define* the application environment to WLM (see 4.5.1, “Defining Application Environments” on page 57).
- At subsystem initialization, the work manager must *connect* to WLM (see 4.5.1.1, “Work Manager Connect to WLM” on page 59).
- The work manager, WLM, and the server address space must cooperate to start a new transaction (see 4.5.2, “Transaction Environment” on page 59).
- SRM and WLM monitor the execution of the transaction (see 4.5.3, “Controls and Adjustments” on page 62).
- WLM generates information for RMF (see 4.5.3.3, “RMF Reporting” on page 63).

- The operator may control the execution of the application environment (see 4.5.4, “Operator Commands” on page 64).

4.5.1 Defining Application Environments

To benefit from server address space management, you must define an *application environment* by giving it a name. This name is used in:

- Work manager external definitions
- WLM administrative definitions
- System programmer definitions

The application environment name is used to bind everything together. The workload supporting a given work manager may be partitioned into multiple application environments.

Figure 28 shows an example of two application environments. The ability to span more than one member of a sysplex depends on the work manager. DSOM address spaces may run anywhere in the sysplex. However, DB2-stored SQL procedures have to run on the same image as the DB2 instance that started them, since all its address spaces depend on the same subsystem instance (either DB2A or DB2B).

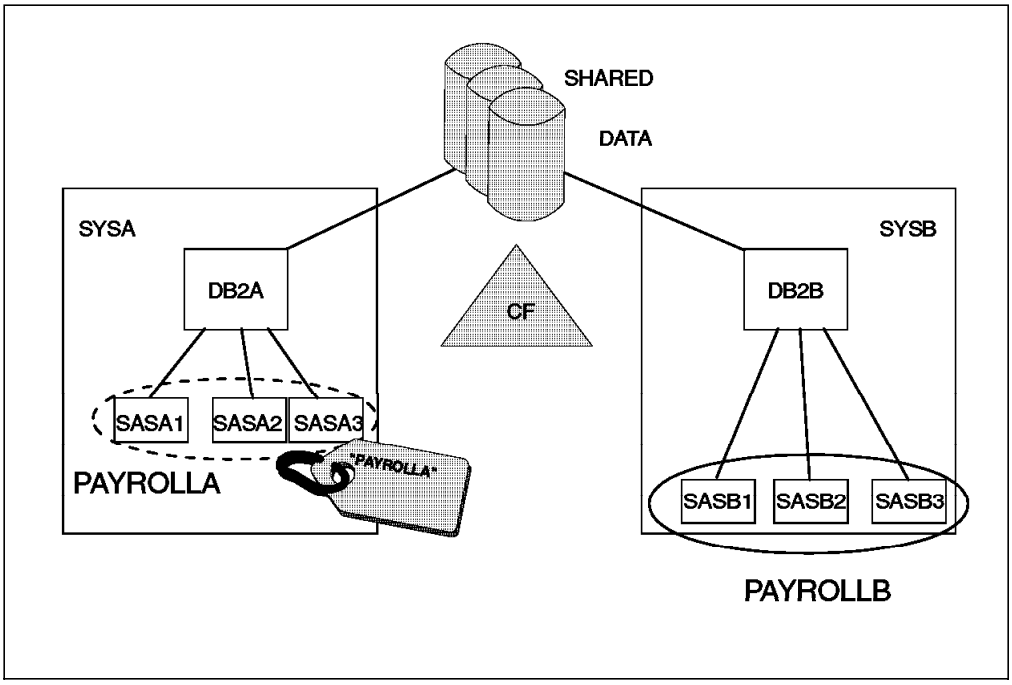


Figure 28. Application Environment

The WLM administrator defines an application environment to WLM through a set of panels. The application environment definition is included in a policy. The following data must be provided:

Name Identifies the application environment to WLM. It must be sysplex-wide unique. For example, PAYROLLB (Figure 28).

Work manager identifier Identifies the work manager by type (for example DB2) and by subsystem name (for example DB2B).

Started procedure name Identifies the member within SYS1.PROCLIB containing the JCL to start a server address space.

Start parameters Identifies optional parameters to be passed to a server during address space creation (for example, user ID for security checking).

WLM management options Special options governing WLM control of address spaces.

Figure 29 is the WLM primary definition menu, and it now contains option 9 for defining application environments.

```
  File  Utilities  Notes  Options  Help
-------------------------------------------------------------------------
  WLM                         Definition Menu                  LEVEL003
Command ===> ____________________________________________________________

Definition data set . .  : none

Definition name . . . . . ________ (Required)
Description . . . . . . . ________________________________

Select one of the
following options. . . . __ 1.  Policies
                           2.  Workloads
                           3.  Resource Groups
                           4.  Service Classes
                           5.  Classification Groups
                           6.  Classification Rules
                           7.  Report Classes
                           8.  Service Coefficients
                           9.  Application environments
```

Figure 29. Selection of Application Environment Panel

After selecting option 9, the panel shown in Figure 30 is displayed to enter the application environment fields.

```
   Application-Environment  Notes  Options  Help
-------------------------------------------------------------------------
                    Create an Application Environment
Command ===> ____________________________________________________________

Application Environment  . . . PAYROLL_____________________ Required
Description  . . . . . . . . . Payroll application_____________
Subsystem Type . . . . . . . . DB2_ Required
Procedure Name . . . . . . . . PAYROLL_
Start Parameters . . . . . . . 'SSN=&IWMSSNM'_____________________
                               ____________________________________
                               ___________________________________

Select one of the following options.
1   1.  No limit
    2.  Single address space per system
    3.  Single address space per sysplex
```

Figure 30. Application Environment Definition Panel

Once the application environment is created, any modifications or changes or new environments can be selected from the following panel. This panel is entered from option 9 in Figure 29 once an application environment is created.

```
  Application-Environment  Notes  Options  Help
------------------------------------------------------------------------
                 Application Environment Selection List       Row 1 to 1 of 1
Command ===> ____________________________________________________________

Action Codes: 1=Create, 2=Copy, 3=Modify, 4=Browse, 5=Print, 6=Delete,
              /=Menu Bar

Action  Application Environment Name      Description
  __    PAYROLL                           Payroll application
```

Figure 31. Application Environment Selection Panel

4.5.1.1 Work Manager Connect to WLM

For work to be routed to server address spaces managed by WLM, the work manager connects to WLM via the IWMCONN service, specifying the QUEUE_MANAGER(YES) option and through two parameters: the subsystem type (SUBSYS parameter) and the subsystem name (SUBSYSNM parameter). This sets the scope of WLM work queueing and address space management functions which operate on a subsystem instance basis.

On receiving the connect request, WLM sets up the appropriate control blocks to be able to queue transactions and create server address spaces for the application environments managed by the work manager.

The connection of a server address space to WLM results in a physical association of that space to a specific work manager subsystem instance. The server address space only processes work originating from that specific work manager and having the same application environment name. This association lasts for the life of the server address space. Each server can have one and only one WLM connection, so only one application environment may be served by each server address space.

4.5.2 Transaction Environment

The association between active servers for each application environment and the related work manager also enables performance optimizations to be implemented by retaining the following across work requests handled by each server:

- Application programs
- File allocations
- Data base connections

Because various work requests (transactions and jobs) for a given application environment may have vastly different characteristics, resource consumption patterns, and requirements, for better control by SRM, each application environment is logically divided into a set of *transaction environments* consisting of the WLM service classes associated with work requests targeted to that application environment.

The different transactions in an application environment may have different response times or service requirements. So it is natural to associate transactions with service classes. Each unique combination of application environment and service class defines a transaction environment, as shown in Figure 32 on page 60.

WLM creates one queue per transaction environment. A server address space can fetch work from only one queue. As work arrives across the IWMQINS interface, it is organized by a unique combination of work manager, application environment name, and WLM service class name. Each of these unique combinations therefore represents a queue of work to be managed by WLM.

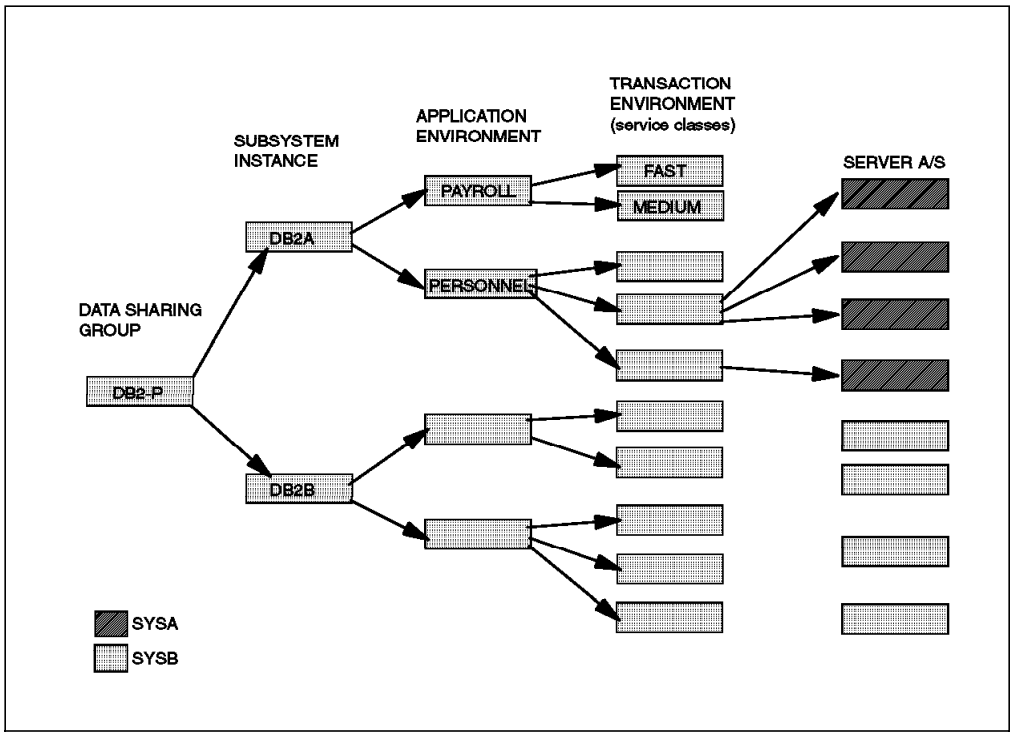


Figure 32. Server Address Space Concepts Hierarchy

4.5.2.1 Queueing Work to Application Environments

The server address space associations, shown in Figure 32, are defined in the following ways:

Subsystem group to subsystem instance: This is defined at subsystem definition.

Subsystem instance to application environment: This is defined at WLM application environment definition. The application environment name (label) must have a matching definition in the subsystem instance.

Application environment to transaction environment: When a transaction arrives at WLM, the subsystem indicates the application environment. WLM determines the service class. If a transaction environment already exists for the coupled application environment and service class, WLM queues the transaction. Otherwise WLM creates a transaction environment and starts a server address space.

The work manager address space queues work requests to WLM using the IWMQINS service and provides to WLM the following information:

- The name of the application environment
- The enclave token associated with the work request
- An optional user ID to be used for resource access control during work unit processing
- User data for locating data areas owned by the work manager

Transaction environment to server address space: At server address space creation time, WLM *binds* the server to a transaction environment. The server is not aware of the transaction environment. When the server selects work, WLM provides a transaction from the appropriate queue. If the queue is empty, WLM suspends the server. If work arrives late, WLM resumes the server.

4.5.2.2 Transaction Flow

Figure 33 describes the flow of a transaction when it arrives at the work manager, together with the associated time intervals.

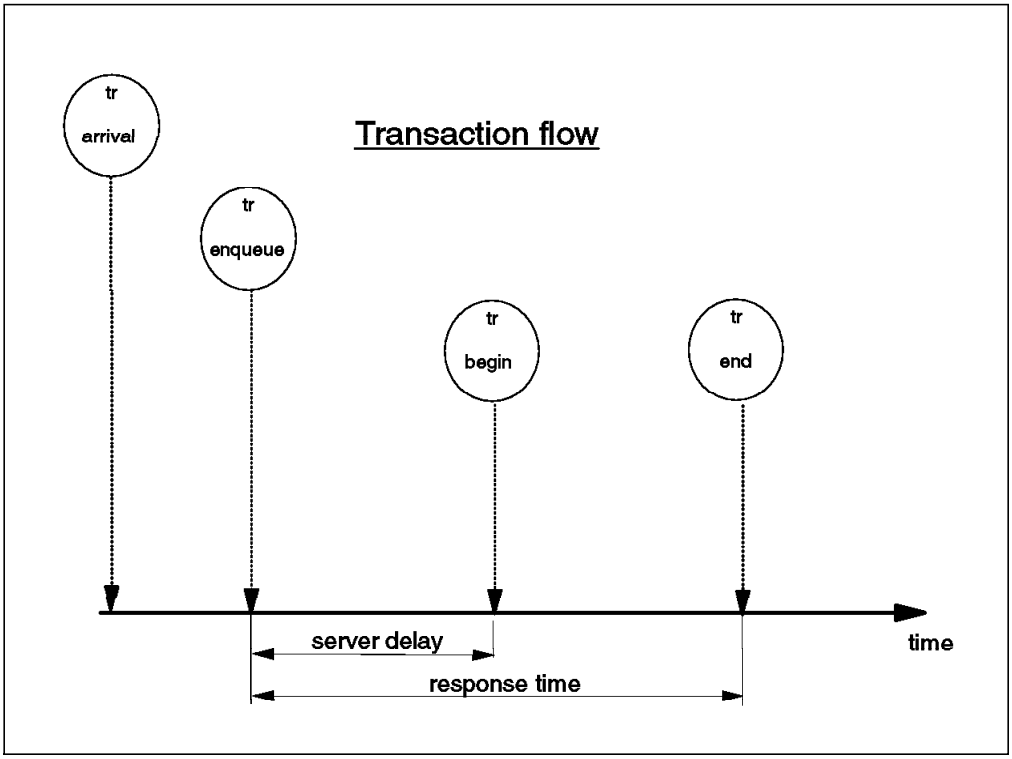


Figure 33. Transaction Flow

When a transaction reaches the system:

1. The work manager queues the transaction to WLM using the IWMQINS service.
2. The WLM manager queues the transaction to its transaction environment if it exists. If a transaction server appropriate for this transaction does not exist:
   a. WLM creates the transaction environment.
   b. WLM starts a server address space.

c. WLM waits for the server to connect using the IWMCONN service specifying the SERVER_MANAGER(YES) parameter.

3. Eventually a transaction server selects the transaction using the IWMSSEL service.
4. WLM creates a business unit of work on behalf of the server using the IWMSTBGN service, which is also used to create a client-based security environment.
5. When the transaction ends, statistics are collected, including CPU time[1], for the transaction and server address space delay. Server address space delay is the time the transaction has been sitting in the queue waiting to be selected by a server.

   Upon completion of each work request, the IWMSTEND service deletes the business unit of work relationship and security environment associated with the execution task. The system resources consumed under the execution task, from IWMSTBGN to IWMSTEND, are accounted for in the business unit of work enclave.
6. The execution task is then free to process another work request, with service accumulated to the server address space and with the security environment of the server address space.

A more detailed illustration can be found in 4.7, “Stored SQL Procedures and Server Address Space Management” on page 66.

4.5.3 Controls and Adjustments

SRM adjusts the number of active server address spaces per transaction environment based on resource constraint on the system and goal achievement for the service class periods. Adjustments are performed independently on each system in the sysplex. Though done independently, they are not unrelated, given that a key element of the decision making, the performance index, is sysplex-wide. So the failure to meet the goals on one system may favor routing work to another, thus contributing to load balancing

4.5.3.1 Reducing Server Address Spaces

To reduce the number of active server address spaces, SRM and WLM take three types of actions, depending on the stability of the conditions that cause the following actions on a server address space, as shown in Figure 34 on page 63:

Swap When a shortage of CPU or storage exists in the system, SRM may decide to reduce the MPL of server address spaces (as it does with domains) depending on how well the related service class period is achieving its goal and the quantity of MPL delay compatible with achieving the goals.

Unbind If the difference between the number of address spaces serving a transaction environment and the MPL is consistently large, SRM will ask WLM to unbind address spaces from the transaction environment. The server address space then becomes “free” and can later be bound to the same or another transaction environment of the same application environment.

---

[1] Only CPU time and service are collected for enclaves. SRB time has no meaning, because non-preemptable SRBs cannot be part of enclaves.

Destroy This is an action that is taken in a very conservative way by WLM because SRM does not know the cost for the work manager to create or destroy an address space. In some cases, this cost may be considerable.

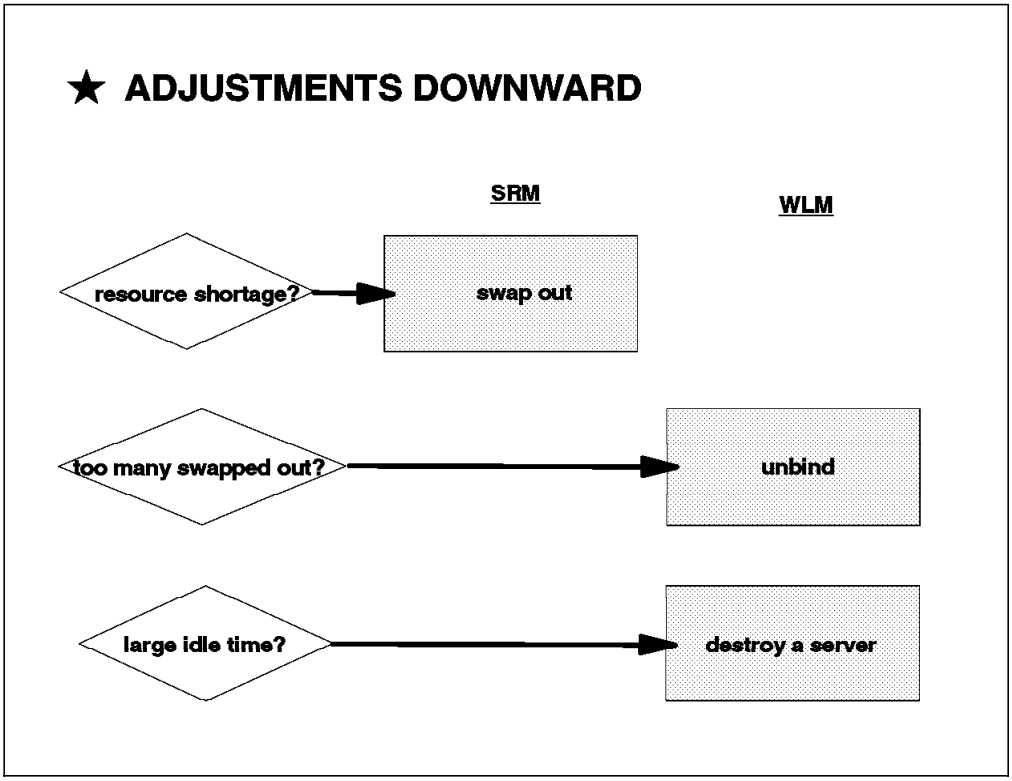


Figure 34. Reducing the Number of Server Address Spaces

4.5.3.2 Increasing Server Address Spaces

A decision to increase the number of server address spaces for a transaction environment is driven by detection that the goals of a service class period are being missed, and an observation that server address space delays are a significant portion of the total observed delays for the service class period.

To increase the number of servers, WLM first tries to assign a previously unbound address space belonging to the same application environment. If this fails, WLM creates a new address space. However, to avoid flooding the system, address space creation is done in a paced way.

4.5.3.3 RMF Reporting

RMF workload activity reports contain additional fields to display the percentage of time that each service class period was delayed due to the unavailability of a server address space. This delay information is recorded by RMF in extensions to the existing RMF Type 72 SMF records. No new WLM display commands or RMF reports are introduced to show the number of server address spaces in use.

RMF Monitor III can be used to display aggregate resource consumption for a collection of server address spaces.

4.5.4 Operator Commands

Two new WLM operator commands are provided to control the availability of server address spaces for application environments. The following command is used to suspend further initiation of work for a named application environment and to cause all associated server address spaces, once idle, to be terminated:

```
VARY WLM,APPLENV=name,QUIESCE
```

This command is useful when you need to:

- Perform maintenance on application program libraries statically allocated to the server address spaces
- Update the JCL procedure for an application environment
- Suspend execution due to repeated application failures or errors

The reverse of the quiesce option is to resume the application environment:

```
VARY WLM,APPLENV=name,RESUME
```

To activate a new application environment, it has to be defined in a policy, which has to be activated as follows:

```
VARY WLM,POLICY=name
```

The following command is provided to cause a purge of resources managed privately within each server address space supporting the specified application environment and for the address spaces to terminate, resulting in a total refresh of that environment:

```
VARY WLM,APPLENV=name,REFRESH
```

It is useful to refresh LE/370 modules in the server address spaces.

The following command is provided to show the status of one (or all) WLM-defined application environments. This command would be used by the system operator to determine if an environment was previously quiesced, or to find out if the application environment is not being managed by WLM.

```
DISPLAY WLM, APPLENV=name
```

4.6 DB2 Stored SQL Procedures Support

Any client server application may use WLM server address space management to integrate its service goals into the global policy of the sysplex. To do so, it must interface with WLM as described in the previous paragraphs. Currently only DB2 and DSOM have made the implementation, but more applications will join the group in the near future.

A *stored procedure* is an application program that is stored at the DB2 server and that can be invoked by a client using an SQL CALL statement. Such an application can contain:

- Conditional logic, provided by standard third-generation programming languages supported by LE/370, such as COBOL, PL/I, C, and Assembler, or by fourth-generation languages, such as VisualGen, that map to these languages.

- One or more SQL statements providing access to local and remote servers, as well as access to non-relational data sources such as VSAM files and IMS data.

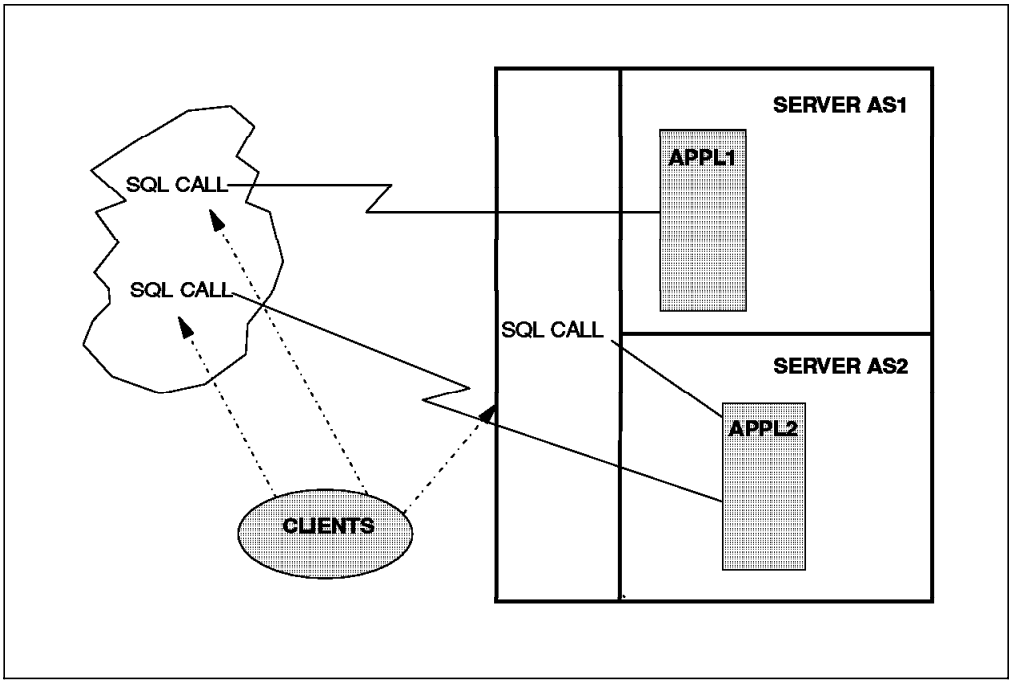


Figure 35. Schematic View of DB2 Stored SQL Procedures

In short, a stored procedure provides almost all the function of a standard DB2 application, and can be initiated by a client. Improved application performance, security, and maintenance are the primary benefits associated with stored procedures.

A stored procedure is initiated with a single SQL CALL statement from a client, but can contain multiple SQL requests for local and remote data. This approach can significantly reduce the number of network send and receive operations, thereby improving the application's elapsed time, and the DB2 CPU time consumed by an application. And, because the stored procedure uses the local DB2 Call Attach interface, there is no added overhead beyond that associated with a local DB2 application. For applications that remotely issue several SQL statements, this savings can be significant.

Stored procedures also give the application designer greater control over sensitive data managed by the server. When applications are run on end-user workstations, sophisticated users can use interactive debugging tools to "walk through" the application logic at run time, modifying the contents of program variables and SQL statements, thereby affecting the integrity of the overall transaction. Stored procedures alleviate this problem by allowing the application designer to encapsulate the processing of sensitive data on the server.

Finally, stored procedures can help to simplify application maintenance by reducing the number of changes required to client applications as a result of changes to database design on the server or changes to application logic. And, if portability of the client application to other platforms is likely to be an issue in the future, localizing much of the application logic on the server is an effective way of preparing for client platform independence.

Stored SQL procedures CALLs from remote systems reach WLM through the distributed data facility (DDF) component of the local DB2.

4.7 Stored SQL Procedures and Server Address Space Management

WLM server address space management has been described in 4.5, “Server Address Space Management” on page 55. This section describes the specific use of the facility by DB2 stored SQL procedures and serves as a short explanation of the facility.

- At application definition time, the administrator:
  - Identifies the work manager (for example, DB2T)
  - Assigns a name to the application environment (for example, PAYROLLA)
  - Specifies the name of the procedure that starts the server address spaces
  - Identifies optional parameters to be passed to the server (for example, RADV user IDs)
  - Gives WLM management options such as maximum and minimum number of server address spaces
- Before starting to process a transaction, DB2 needs to CONNECT to WLM, as shown in Figure 36. WLM then prepares the control blocks that anchor the application environments.

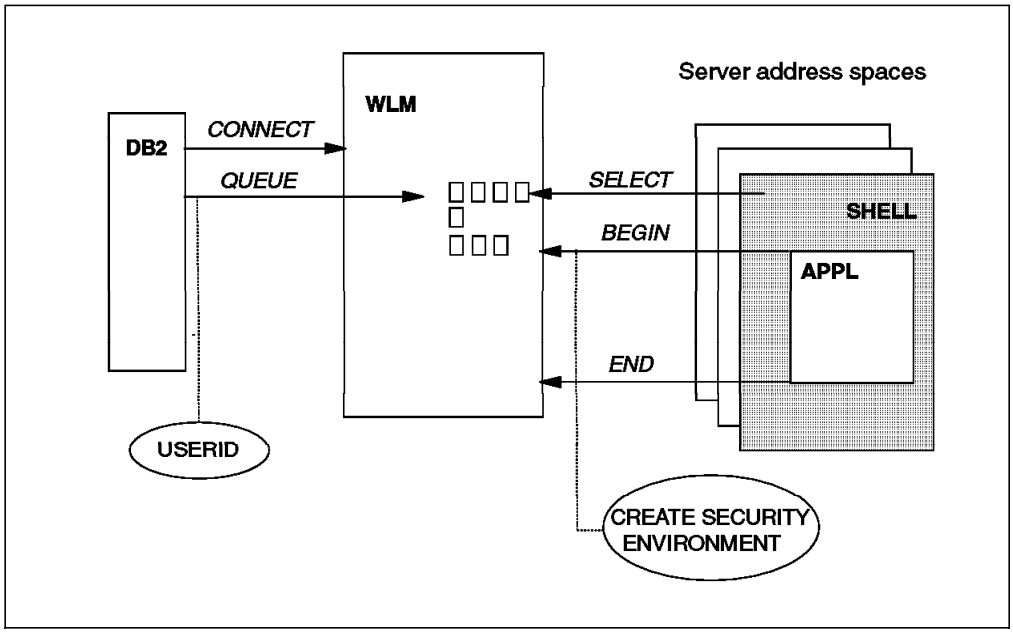


Figure 36. WLM Server Address Space Exploitation

- When a transaction arrives, DB2 *may* create an enclave, depending upon the origin of the transaction. (For more information about enclaves, see 4.4, “Business Unit of Work Management” on page 50).
  - If the origin is DDF, an independent enclave is created.
  - If the origin is TSO, JES, or APPC, a dependent enclave is created which inherits the attributes of the originating space.

  - If the origin is CICS or IMS, no enclave is created.

- Then DB2 queues the transaction with an IWMQINS service call. As optional data, DB2 passes the user ID for later security checking.
- WLM enqueues the transaction to a transaction environment:
  - If no address space serves the transaction environment, a server address space has to be started.
  - The server address space is started by the procedure defined in step 1. The main program is the shell program (DSNX9STP). The start procedure passes a parameter to the shell to specify if it will work in single or multithread.
  - The shell connects to WLM using the IWMCONN service. Authorization is checked at this point.
- Eventually a server address space issues a IWMSSEL service to select a transaction from the transaction environment queue.
- Before passing control to the application program, the shell invokes the begin execution process of using the IWMSTBGN service of WLM. At this point, the security environment is established. Resource consumption starts to be charged to the transaction at this point.
- The application program ends and returns control to the shell, which invokes the end execution processing using the IWMSTEND service of WLM, and resource consumption ends.

4.7.1 WLM Compatibility and Goal Mode

It is possible to take advantage of WLM support of multiple server address spaces for DB2 stored SQL procedures work, while remaining in WLM compatibility mode. To do this, managing the quantity of server address spaces for each of the DB2 application environments must be done manually. The installation must determine how many spaces to start and must manually terminate spaces no longer required (to save system resources). RMF can be used to analyze server address space delay information to determine if the proper quantity of spaces are in use.

Starting the server address spaces is through use of operator start and cancel commands, or through use of automated operations facilities such as NetView or AOC.

Two options are provided with WLM goal mode and stored SQL procedures management as follows:

Automatic With WLM automatic management of stored SQL procedures in effect, the quantity of server address spaces is totally under control of WLM. Spaces are created as needed and terminated when no longer needed, with no system operator awareness. There is no mechanism provided for customer intervention in this WLM process. WLM controls the number of spaces. A system operator or automated operations packages may start and/or cancel server address spaces being managed by WLM even while in goal mode. Servers not started by WLM are used as if they were started by WLM, and if unnecessary, they are terminated.

Manual Manual operation in WLM goal mode is the capability to manually control the number of address spaces serving stored SQL procedures. The installation must determine the number of address spaces needed to serve each application environment, start that number of spaces, and monitor the performance in case the workload demands have changed.

4.7.2 Other Exploiters of Server Address Space Management

DSOM exploits WLM server address space management to dynamically adjust the number of address spaces.

4.8 Sysplex-Wide TSO and WLM

Since the announcement of sysplex there has been a strong requirement to be able to distribute the TSO workload in the sysplex. In OS/390 Release 3, TSO benefits from VTAM generic resources, thus giving WLM the opportunity to distribute sessions among different members of the sysplex.

TCAS provides the external definition of the generic resource name to be used for a given MVS image. VTAM then services session logon requests, with the assistance of WLM decision-making algorithms.

4.8.1 VTAM Generic Resources

Initial WLM support for VTAM generic resources was introduced in the MVS/ESA SP 5.2.0 release. With OS/390 Release 3 and with support provided by the VTAM, TSO/E, JES2, and JES3 products, WLM support is extended to permit sysplex distribution of:

- TSO LU_2 sessions
- APPC/MVS LU_6.2 sessions

Any VTAM application program running on a VTAM APPN node that is connected to an MVS coupling facility structure identified by the STRGR start option can be known by a generic resource name. All VTAM nodes having access to the common generic resource coupling facility structure, along with the structure itself, make up a generic resources configuration. Note the following points:

- A generic resources configuration must be part of a sysplex environment.
- All VTAMs in the sysplex environment must be running under MVS/ESA Version 5 Release 1 or a subsequent release of MVS that supports coupling facility services.
- All VTAMs in a generic resources configuration must be connected to an MVS coupling facility in which the generic resource structure may be allocated. VTAM connects to the generic resource structure when it is defined in the active CFRM policy.
- The generic resource coupling facility structure stores the data needed by VTAM to implement the generic resources function. This data includes mappings that show:
  - For each generic resource name, a list that identifies the generic resource members currently using that name. This is called a generic resource mapping.

– For each LU in session with one or more generic resource members, the application program network name and the generic resource name for each generic resource member. This is called a partner LU mapping.

Note: VTAM Version 4 Release 2 or later is required for all VTAMs in a generic resources configuration, and all VTAMs must be defined as network nodes or end nodes. If an application program is both a generic resource and uses multinode persistent sessions, all VTAMs in the sysplex must be Version 4 Release 4 or later.

To implement generic resources, the network must be APPN. An APPN network comprises two types of host nodes:

- One or more NNs (Network Nodes), which manage the LOGONs from the terminals
- One or more ENs (End Nodes) where the applications will run

Generic resources are defined in the table ISTGENERIC,[2] which resides in the Network Node and is backed by a structure in the coupling facility. ISTGENERIC maps generic LU names to actual LU names.

If an application program wants to be the final destination of a LOGON to a generic resource, it must *register* to the generic resource with macro

```
SETLOGON    OPTCD=GNAMEADD,NIB=genericname
```

When the SETLOGON GNAMEADD occurs, VTAM updates its mapping of real LU to generic LU associations within its coupling facility structures. The VTAM NN services network requests to locate the VTAM instance to satisfy session logon requests.

Only one generic name may be specified, which means that at a certain time an application program (represented by an ACB) cannot be registered to more than one generic resource at a time.

In a mixed release environment, only systems at OS/390 Release 3 can participate in generic resource definition. The downlevel systems can run TSO, but they require a specific logon.

When a user logs on to a generic resource, VTAM finds in its table the specific LU names that map to the generic name. If more than one candidate maps to the generic name, VTAM either runs its own exit or invokes WLM to decide session placement.

4.8.2 TSO Session Placement

When you log on to TSO, you log on to the TCAS application (full name is IKTCAS). TCAS is known to VTAM by its LUname. In Figure 37 on page 70, the LUnames for the TCASs in each system are respectively TS1, TS2, TS3 and TS4. After the user has logged on to TCAS, TCAS creates an address space, activates an LU in that address space with the LUname, TTTnnnn (where TTT is the LUname of TCAS and nnnn is the first available LU number). Then TCAS OPENs the ACB of LUnnnn and uses CLSDST PASS (close destination pass) to close its

[2] Prior to VTAM 4.3, the structure name had to be ISTGENERIC which limited to one the number of structures per coupling facility. In VTAM 4.3, the name of the structure can be set by the administrator, allowing you to have multiple network nodes in the sysplex managing different generic resource tables.

own session with the terminal and does a session BIND between the terminal and the application TTTnnnn.

4.8.2.1 TSO Generic Resource Name

To specify a generic resource name, this is accomplished by specifying GNAME=genericname in TSOKEYnn member of SYS1.PARMLIB.

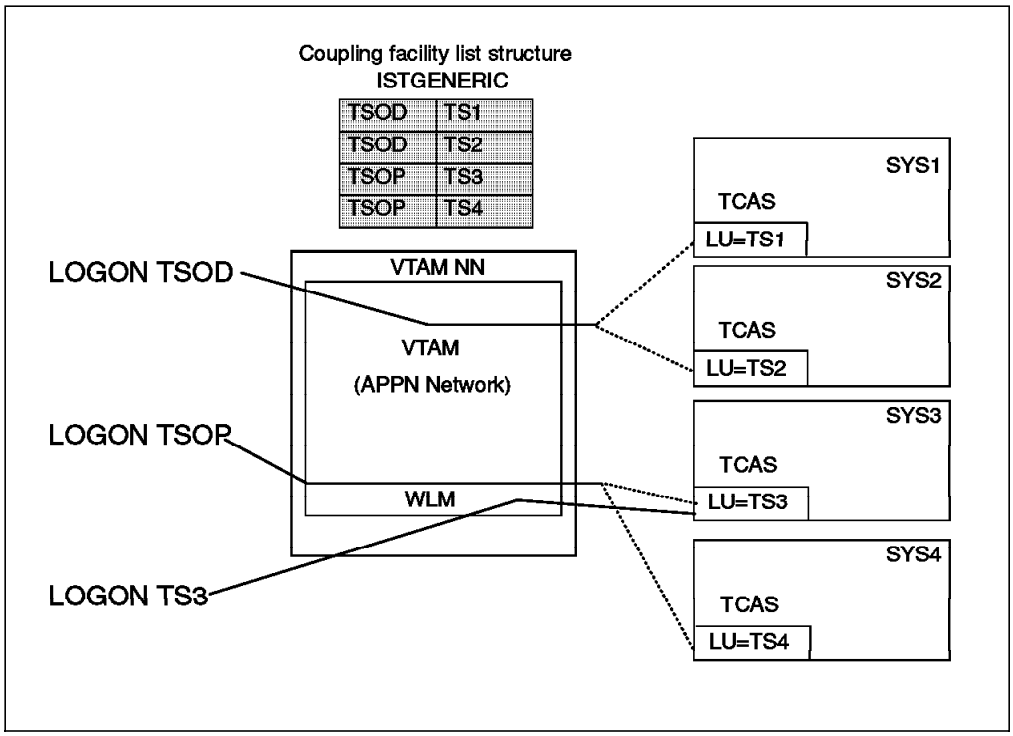


Figure 37. Generic Resource Topology

In Figure 37, the sysplex contains two generic resources:

- TSOD mapped by LUs TS1 and TS2
- TSOP mapped by LUs TS3 and TS4

A user, shown in Figure 37, logs on to specific node TS3, and that user gets a TSO session in SYS3.

A user, shown in Figure 37, logs on to generic resource TSOD. This user gets a session either in system SYS1 or SYS2.

Another user, shown in Figure 37, logs on to resource TSOP and gets a session in either system SYS3 or SYS4.

VTAM and WLM are both involved in the placement process. The process consists of a series of filtering operations followed by a selection by WLM (see Figure 38 to Figure 41).

4.8.2.2 Construct Candidate List

A session request for a generic name, such as a logon request, may result in multiple candidate real LUs. For this case, VTAM NN constructs a list of eligible real LUs and then trims the list to exclude those LUs being above the TSO MAXUSER limits. VTAM then makes an initial decision, selecting the best real LU from the list to equalize the number of sessions assigned to each system. The initial list contains all the LUs registered to the generic name.

As shown in Figure 38, a user does a logon and VTAM issues a SETLOGON causing VTAM to make an initial decision of selecting the best real LU from the list to equalize the number of sessions assigned to each system using MAXUSER limits.

VTAM then gives WLM an opportunity to make a better choice, by using the WLM service, IWMGRSEL. The WLM selection exit first attempts to eliminate obviously bad choices by checking whether the TSO workload is badly missing its goals or subject to resource constraints. The decision is restricted to eligible MVS images that are not badly missing goals and not resource-constrained. If all MVS images are in bad shape, then the choice must be made among bad choices.

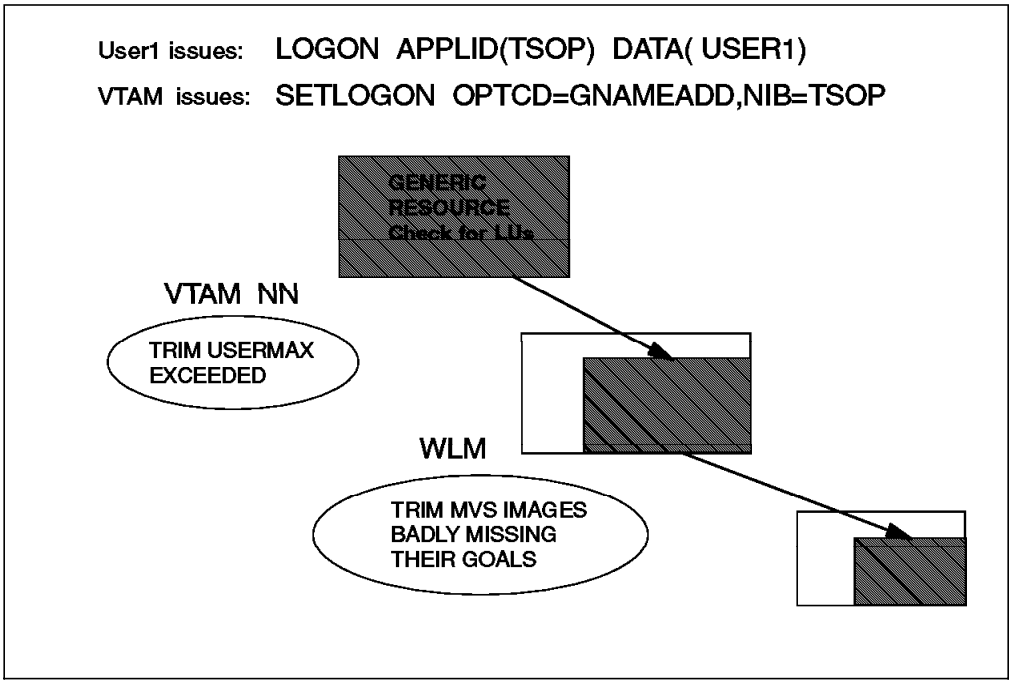


Figure 38. Construct Candidate List (1/4)

4.8.2.3 Determine CPU Capacity of Candidates

Now a calculation is done by WLM to determine what the raw CPU cost is estimated to be for recent sessions routed to each eligible MVS image, and what the TSO users demand is at the moment in terms of both the long and short term cost of the TSO workload, along with the number of users present. This raw CPU cost represents the expected capacity required by the current session and other latent demand.

This information is used to relate the CPU capacity used at each importance level, making it possible to see the highest importance level the new session

would be competing with on each MVS image. There may be available (unused) capacity, or it might compete with discretionary work or even higher importance work.

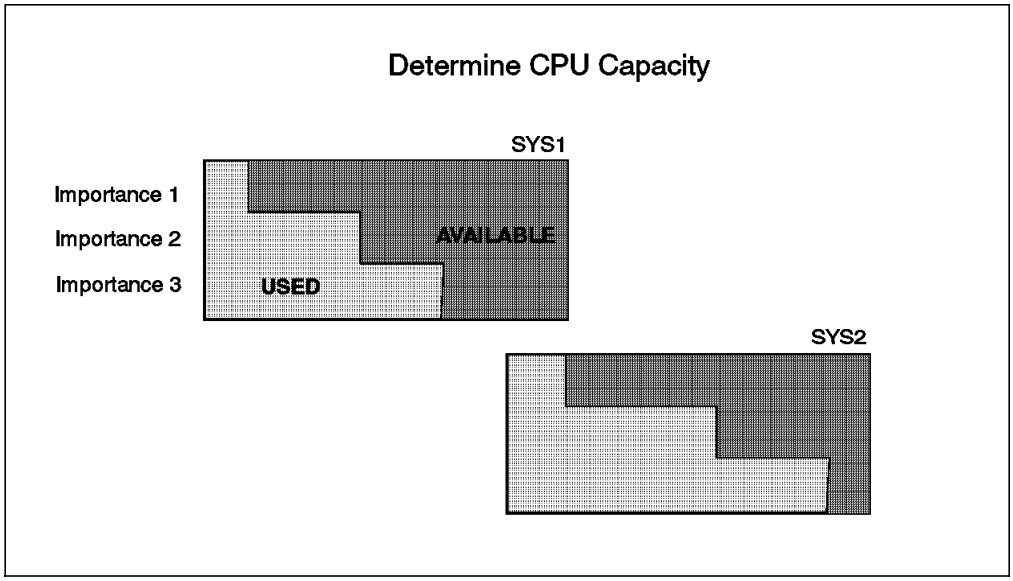


Figure 39. Determine CPU Capacity of Candidates (2/4)

4.8.2.4 Determine TSO Ranking of Candidates

The new list is divided into three categories according to goal achievement, as shown in Figure 40 on page 73. Each system is ranked according to the importance level derived above and whether TSO on that image is:

- Meeting its goals
- Occasionally missing its goals
- Badly missing its goals

Preference is given to systems where there is available capacity and TSO is meeting goals.

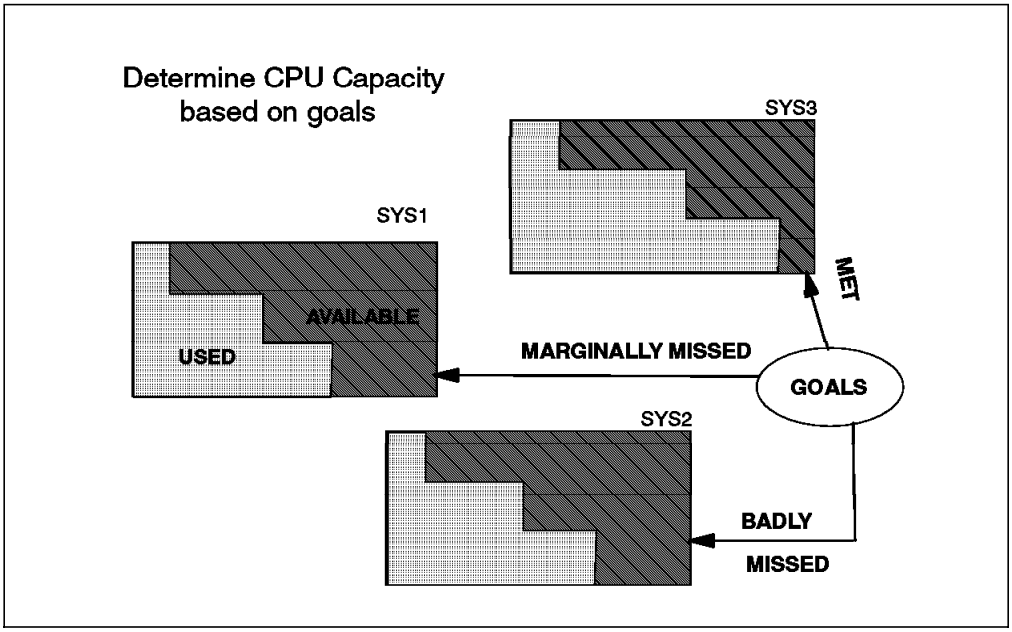


Figure 40. Determine Ranking Based on Goals (3/4)

4.8.2.5 Determine by TSO Performance Index

Performance index values break any ties if multiple systems have the same TSO ranking, as shown in Figure 40. Then, WLM selects the system with the lowest performance index (see Figure 41).

Note: Once the choice is made, there is a generic resource user exit that has the final decision on where the session is placed.

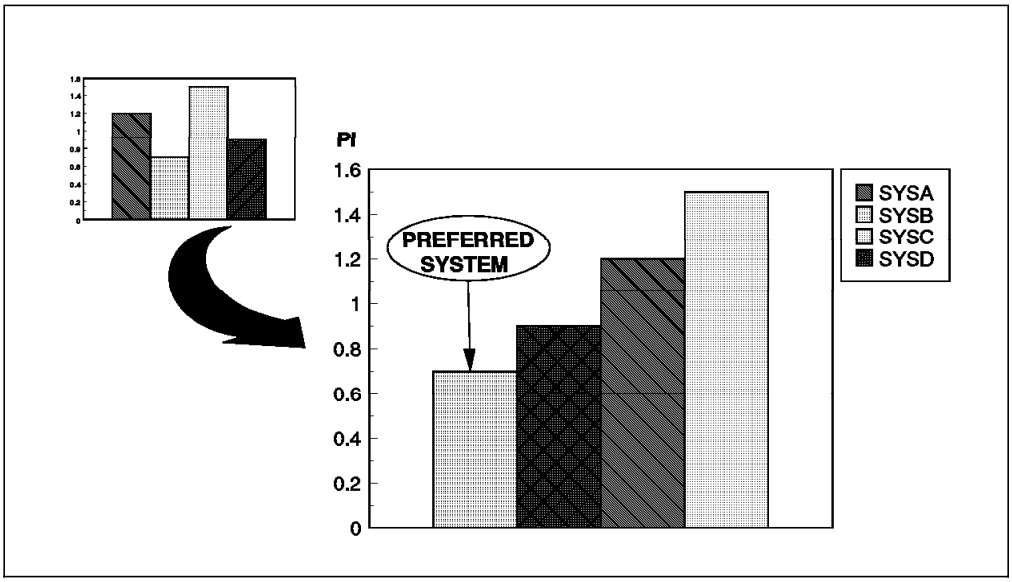


Figure 41. Determine Placement Based on Performance Index (4/4)

4.8.3 TSO/E Generic Resource Considerations

This section contains some considerations in support of generic resources such as:

- The support for DISCONNECT, RECONNECT, and SEND
- Session placement when WLM is in compatibility mode
- Compatibility and externals

4.8.3.1 User DISCONNECT from a TSO Session

The operation of generic resources can be totally transparent to the end user. This means that end users do not need to know in what system their sessions are established.

Given that the LOGON procedures are identical and that the same data is accessible on all systems that resolve to the same generic resource, when you DISCONNECT from TSO either on purpose or because of problems with the terminal, you need to RECONNECT to the same MVS image. TSO and VTAM make use of existing JES knowledge of TSO users within a sysplex. Each TSO user address space exists as a job in the JES job queue, and the JES job-related data structures indicate the system on which the address space exists. A new interface is provided by JES2 and JES3, permitting TSO to determine if a user is logged on, and if so, where. This service is used during RECONNECT processing and allows TSO to direct the request to the same MVS image.

4.8.3.2 Session Placement with WLM in Compatibility Mode

When WLM is in compatibility mode, session placement is made by VTAM with no WLM assistance as follows:

- VTAM attempts to equally distribute sessions over all the systems
- VTAM eliminates systems at USERMAX

In a sysplex with a mixture of compatibility and goal modes, the decision depends upon the WLM mode in effect on the systems that VTAM determines to be eligible for a given placement decision. If all systems in the VTAM-eligible list of real LUs are in WLM goal mode, the WLM session placement algorithms and rules are in effect. If any one system in the VTAM-eligible list of real LUs is in WLM compatibility mode, the VTAM balanced-number-of-sessions decision is used.

4.9 APPC/MVS Support For Generic Resources

APPC/MVS support for generic resources allows placement of LU_6.2 sessions to benefit from WLM session placement decisions. See the section 4.8.1, “VTAM Generic Resources” on page 68 for the details of VTAM generic resource support.

4.9.1 APPC/MVS Generic Resource Name

To specify a generic resource name, use a new parameter, GRNAME=genericname, in the APPCMPxx member of SYS1.PARMLIB for the LUADD statements to be managed.

The APPC-managed LUs support three operating environments:

- Conversations managed by APPC/MVS (ASCH LUs)

- Conversations managed by an alternate scheduler (such as IMS)
- Conversations to be routed to an APPC server (NOSCHED LUs)

For each APPC/MVS LU having a generic resource name defined, the APPC/MVS initialization process includes registering the associated generic names with VTAM. This is done using the VTAM SETLOGON GNAMEADD service, which identifies the ACB for the real LU and the generic resource name to be associated with that real LU. As the SETLOGON GNAMEADD occurs, VTAM on each individual system causes the VTAM network node (NN) to be updated to reflect the real LU to generic LU associations.

4.9.2 APPC/MVS Session Placement

A session request for a generic name may result in multiple candidate real LUs. For this case the VTAM NN constructs a list of eligible real LUs. VTAM then makes an initial decision, selecting the best real LU from the list to equalize the number of sessions assigned to each. VTAM then gives WLM an opportunity to make a better choice, by calling the WLM selection service routine, IWMGRSEL.

WLM selection of the target real LU depends upon the type of LU backing the generic LU name and the functional level of the various software products installed. When the latest levels of software, MVS 5.3 and VTAM 5.1, are present, the WLM choice for ASCH LUs and NOSCHED LUs is driven by the number of sessions bound to each LU and the capacity of each LU′s system.

This is also the mechanism for alternate scheduler LUs, except where the alternate scheduler participates as a WLM work manager. For this case, such as IMS 5.1, the decision-making logic factors in the actual performance of each IMS as compared to the performance goals of the transactions that that IMS runs.

4.9.3 APPC/MVS and WLM Compatibility Mode

For a sysplex operating in WLM compatibility mode, where all systems are in compatibility mode, APPC session placement decisions are made solely by VTAM with no WLM assistance. For this environment, VTAM attempts to distribute the sessions equally across the systems having the requested generic resource defined.

For a sysplex with a mixture of WLM modes, the outcome depends upon the WLM mode in effect on the systems that VTAM determines to be eligible for a given placement decision. If all systems in the VTAM-eligible list of real LUs are in WLM goal mode, the WLM session placement algorithms and rules are in effect. If any one system in the VTAM-eligible list of real LUs is in WLM compatibility mode, the VTAM balanced-number-of-sessions decision is used.

4.10 WLM Internal Changes

The following new usability enhancements make WLM easier to use and control:

RESET command audit record The SRM RESET command processor has been changed to generate a new SMF record, thereby leaving a footprint showing that performance characteristics of a job have been manually altered.

Record policy change in SYSLOG A new message, IWM001I, is written to syslog when a WLM policy is activated through the WLM administrative application or by the VARY WLM,POLICY=name operator command. This allows users to determine exactly when policies have been activated, and by whom.

There are no changes to the text of message IWM001I, nor is there a need for revisions to the message explanation.

Reduce volume of SMF 99 records Changes will be made to the content of the existing SMF 99 records and to the frequency at which records are generated, to reduce overall quantity of data generated by SRM.

Type 99 records are used for WLM problem determination.

Abbreviate the QUIESCE command The syntax of the QUIESCE option of the SRM RESET command has been made simpler, permitting the system operator to use “Q” for the option.

```
E jobname,A=asid,{PERFORM=nnn       }
                 {SRVCLASS=classname}
                 {QUIESCE|Q         }
                 {RESUME            }
```

Printed policy readability The FILE pulldown from the WLM administrative application can be used to print a service definition. This produces a script input file which you can process to see a clear view of relationships in the service definition.

PERFORM value in WLM classification The WLM classification rule definition panels (within the WLM administrative application) have been updated to support the performance group number (PERFORM=) for a job. This is a migration aid, so that existing JCL libraries and JES subsystem definitions can be used, unaltered, as input to WLM job classification.

Another qualifier is the procedure name. This has been added to classify DB2 stored SQL procedures according to application environment.

Display service definition time/day The DISPLAY WLM command response has been extended to include the date and time that the current service definition was installed. The additional information allows system operators to determine when the last service definition update took place, so they will know whether intended updates were installed via the WLM administrative application.

```
IWM025I  13.43.33  WLM DISPLAY 839
  ACTIVE WORKLOAD MANAGEMENT SERVICE POLICY NAME: CICSWORK
  ACTIVATED: 1997/08/02  AT: 17:05:31  BY: HEISIG    FROM: SC52 <----
  DESCRIPTION: CICS primary policy
  RELATED SERVICE DEFINITION NAME: CICSpol
  INSTALLED: 1997/08/02  AT: 17:02:49  BY: HEISIG    FROM: SC52 <----
  WLM VERSION LEVEL: LEVEL003
```

4.11 Migration Considerations

When migrating to the new WLM by installing OS/390 Release 3, there are several important considerations regarding:

- The coexistence of Release 3 with previous releases in the same sysplex
- Co-requisite changes in MVS and in other products to support WLM

4.11.1 OS/390 Release 3 Coexistence Considerations

The format of the WLM couple data set changes and may need to be reformatted when using the new functions (in particular server address spaces). A toleration PTF is provided for old releases to accept the new format. However, none of the new functions are usable in old releases.

Before installing OS/390 Release 3 in a multi-system sysplex, you must install the following compatibility PTF on each pre-OS/390 Release 3 system in the sysplex:

On this release:	Install this PTF:
MVS SP5.1	UW33066
MVS SP5.2 or 5.2.2	UW33067
OS/390 R1 or R2	UW33068

If one of the following conditions exists, you must reallocate the WLM couple data set, and reinstall and activate the service definition:

- You want to use some of the OS/390 R3 workload management functions.
- You plan to migrate all the systems in the sysplex to OS/390 R3.

The OS/390 Release 3 version of the workload management application works only with a reallocated WLM couple data set, even if the new workload management functions are not used.

If you want to make changes to the service definitions and use any of the OS/390 Release 3 workload management functions, you must install the following PTFs on earlier systems:

On this release:	Install this PTF:
OS/390 R2	UW32488
OS/390 R1	UW32487
MVS SP 5.2.0	UW32486
MVS SP 5.1.0	UW32485

4.11.1.1 Goal Mode Compatibility

You can install OS/390 Release 3 with no changes to the workload management service definition or to the WLM couple data sets; workload management runs as it did before installing OS/390 Release 3.

Note: The only exception to this is where the pre-OS/390 Release 3 service definition has more than 100 service classes defined. In this case, the service definition must be modified so that there are no more than 100 service classes.

Note: OS/390 Release 3 changes the maximum number of allowable service classes from 999 to 100.

Chapter 5. SYSOUT Application Programming Interface (SAPI)

A new interface called the SYSOUT application programming interface (SAPI), SSI function code 79, provides an enhanced interface to the JES spool. SAPI is intended for application development or user-supplied authorized assembler programs to have on-demand SYSOUT viewing, printing, and modification that compliments the traditional role in JES printing. SAPI allows access to JES SYSOUT data sets independent of the normal JES-provided print or network access. Users of this function are programs operating in an address space external to JES in the sysplex.

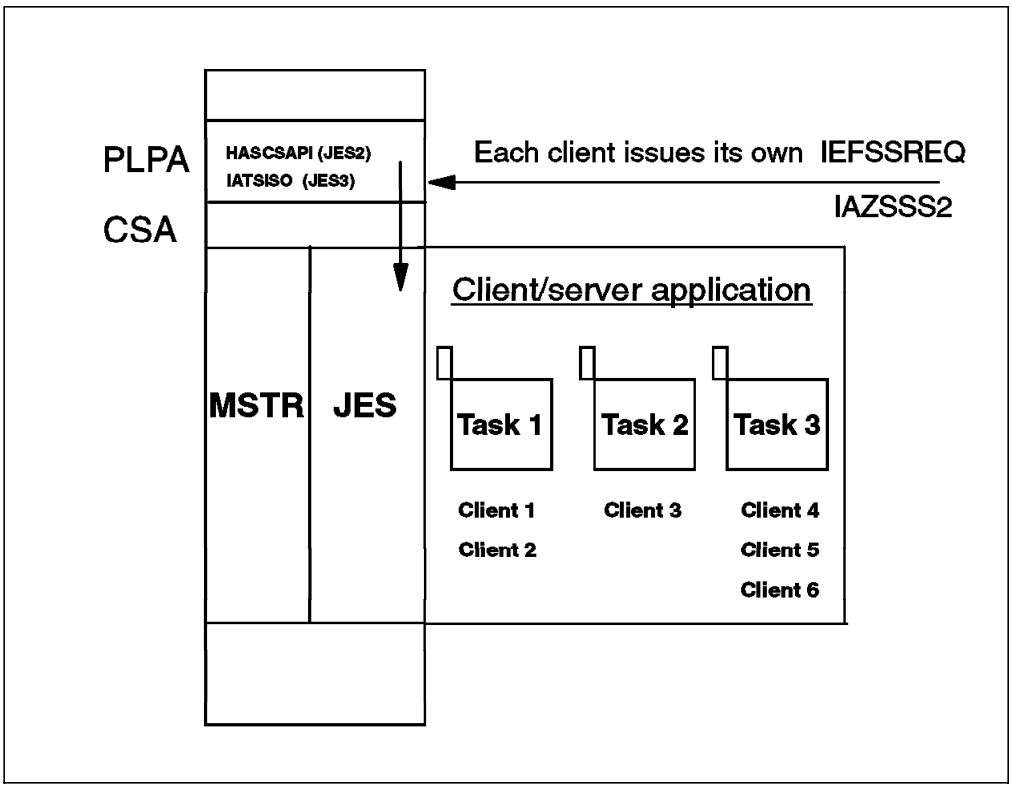


Figure 42. Overview of SAPI Processing

The SAPI interface allows greater processing capability than does the original Process SYSOUT (PSO) interface. The PSO interface call allowed a user-supplied authorized assembler program to access JES SYSOUT data sets. The programs using this interface are called *external writers*. They operate in an address spaces external to JES.

Note: The PSO interface, SSI function code 1, remains unchanged in OS/390 Release 3.

5.1 SAPI Functions

The SAPI support addresses many outstanding PSO requirements. SAPI provides the following enhancements to applications, and these enhancements are the major differences between the two interfaces:

- The SAPI interface allows JES to function as a server for applications needing to process SYSOUT data sets residing on the JES spool, as shown

in Figure 42. SAPI supports multiple, concurrent requests from the application's address space. Each issuer of the IEFSSREQ macro is referred to as an *application thread*.

- Support for three request types of processing using the SSI:

  PUT/GET The request type selects a data set for processing and returns the previously retrieved data set. 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.

  COUNT Returns the number of data sets, including line, page, record and byte counts, that matches the selection criteria. JES counts the number of schedulable elements (OSEs/JOEs) matching the input selection criteria and returns the count to the application thread. 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 Modifies the data set attributes of all data sets matching the selection criteria. 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 the SSOB functional extension. BULK MODIFY can request:

  – Class update - the class of each selected data set is changed to the specified new class.

  – Delete processing - each selected data set is deleted.

  – Destination update - the destination of each selected data set is changed to the specified new destination.

  – Release processing - each data set is moved to the WRITER queue in JES3, and marked non-held in JES2.

    Release processing is applicable only to data sets on the JES3 Output Service HOLD queue, or for those data sets with dispositions of HOLD or LEAVE for JES2.

  Processing for a BULK MODIFY request occurs for each data set matching the application thread's selection criteria.

  Note: It is important to understand that job boundaries can be crossed.

- Supports selection of SYSOUT data sets through an expanded selection criteria. There are 27 selection fields in the SAPI support versus eight in the PSO support.
- Supports the use of wildcards for data set selection through jobname, forms, writer name, destination, creator user ID, and prmode.

- Supports a greater number of SYSOUT data set characteristics. Information returned by JES to the application is 59 characteristics for SAPI versus 16 for PSO.
- Supports scheduler work block (SWB) processing by returning a SWB token for SJFREQ processing, or returns a SWBTU block address for SWBTUREQ processing by the application, or both.
- Support is identical for held and non-held SYSOUT.
- Support is through a common JES2 and JES3 interface with some minor exceptions so that the application need not be concerned with the JES it is interfacing with.

For a complete description of SAPI processing and SSI function code 79, see *OS/390 MVS Using the Subsystem Interface*.

5.2 SYSOUT Processing Using SAPI

The SAPI functional extension to the SSOB is a data area mapped by the IAZSSS2 macro. The application is expected to implement the following steps to read JES spool data:

1. GETMAIN storage for the SSOB and the functional extension and initialize the entire functional extension (IAZSSS2) to zeros. Field SSOBINDV of the SSOB header must be set to point to the SAPI functional extension.

   The IAZSSS2 mapping macro defines the fields in the SSOB functional extension on the IEFSSREQ request for SAPI processing. The definition of fields in this macro are clearly identified as input, output, and disposition fields.

2. Fill in the eye catcher and length field

   Figure 43 on page 82 shows the GETMAIN areas and the eye catcher and lenght field name.

3. Fill in the optional selection flags and fields in IAZSSS2. These are the input fields passed to JES for selecting a SYSOUT data set.

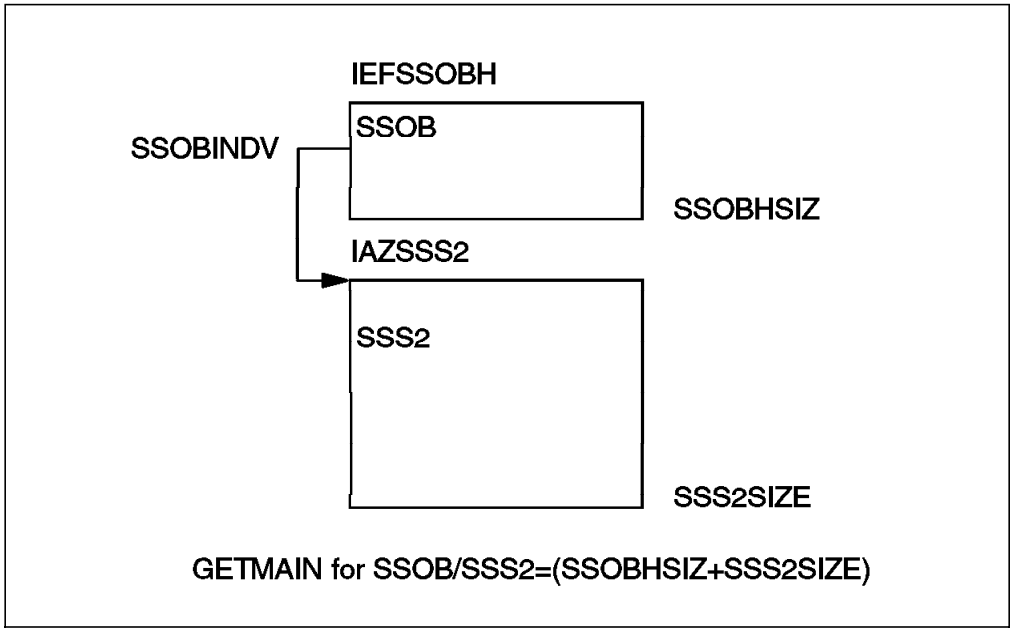


Figure 43. GETMAIN and Setup of SSOB/SSS2 Pair

Figure 44 on page 83 begins the applications request to JES for SYSOUT data sets. Steps 4 through 10 are shown in Figure 44 on page 83, and are explained in the step numbered text that follows in this section.

(4). - Issue SYSOUT PUT/GET Request

4. Issue IEFSSREQ macro to invoke JES processing for the desired type of processing, as shown in Figure 44 on page 83, by setting one of the following request types:
   - Set SSS2TYPE to SSS2PUGE for PUT/GET processing.
   - Set SSS2TYPE to SSS2COUN for COUNT processing.
   - Set SSS2TYPE to SSS2BULK for BULK MODIFY processing.

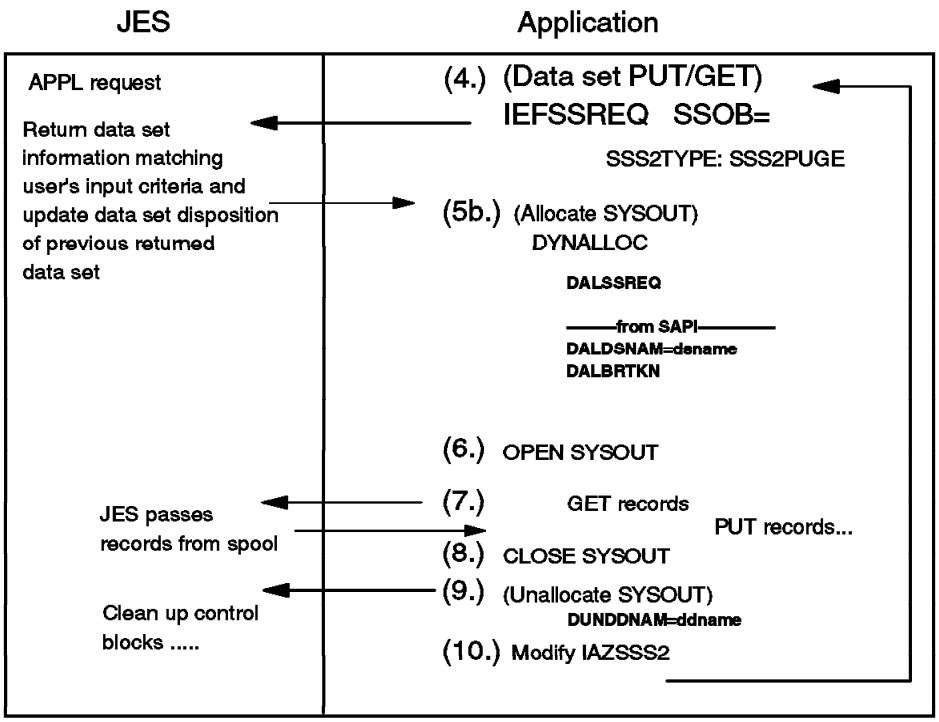


Figure 44. Overview of an Application Selecting SYSOUT

Considerations when making a request type in step 4

The following considerations apply to any of the three types of SAPI calls:

- Each unique SSOB/SSS2 pair supplied as input on the IEFSSREQ request is viewed as a separate thread by JES.

  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 JES considers to be the owner of that specific thread) fails, then JES 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 JES has disconnected from that SSOB/SSS2 pair.

  The SSOB functional extension field SSS2JEST is the binding value that JES uses to associate a specific SSOB/SSS2 pair to its thread. The owner of a thread is the TCB that makes the FIRST request and receives a token in field SSS2JEST. After initially setting SSS2JEST to X'00' as part of the application thread's original initialization of the SSS2, the application thread cannot modify the SSS2JEST field.

- The “output section” of the SSS2 is initialized once by the application thread. The application thread does so by clearing the entire SSS2 with binary zeroes prior to initializing any input fields and then issuing the first IEFSSREQ request. Subsequently, JES manages all the output section fields. An application thread can only change the contents of this output section after an IEFSSREQ request has been made with the SSS2CTRL flag set. JES considers such a subsequent request as a new thread because as a result of the SSS2CTRL bit being set on the prior IEFSSREQ call, JES disassociates all JES-maintained resources held.
- Destination fields can include a single, maximum 8-character destination or a destination in the format of *node.userid*. For the latter case you must have an NJE-defined destination as the node.
- Output field SSS2RET2 indicates which of the input selection fields were not used by JES in the selection of work.
- The SSI Function Code 54 call (Request Subsystem Version Information) can be used to determine the appropriate SYSOUT class to use when modifying the data set’s SYSOUT class through the SAPI BULK MODIFY call.
- JES provides a minimum amount of validity checking of an input SSS2 before a final call (SSS2CTRL) is processed. This validity checking includes:
  - Ensuring that a valid SSS2 eye catcher is present
  - Ensuring that a valid version number is present
  - Ensuring that a valid request type is present
  - Ensuring that a valid length is present
  - Ensuring that a valid disposition, if applicable, is present

  Note: If an incomplete selection criteria is passed to JES, JES may select data sets that you did not plan to process.

For PUT/GET requests, once the application thread receives a data set from the JES, 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 a DALBRTKN text unit. JES 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 JES-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.

Note: The allocation/unallocation of the data set must occur once per returned data set.

The PUT portion of this 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.

A difference between SAPI and PSO during unallocation is that SAPI does not process any of the unallocation text units as occurs in Process SYSOUT.

The SSS2 provides specific disposition fields for JES to use during the subsequent SAPI PUT/GET call to provide for disposition processing. From a JES processing point of view, the disposition processing for the previous data set occurs prior to the processing of the selection of the next data set, but both are occurring within the same IEFSSREQ call by the application thread.

You must provide at least SAF UPDATE authority for the JESSPOOL resource class to the application thread in order to correctly issue the SAPI PUT/GET call.

When a PUT/GET request obtains a data set, you need to set a disposition field to inform JES how to do disposition processing.

SSS2DSP1 This field contains the data set disposition flags. If the flag has all bits off, the data set is deleted. When the SSS2DKPE flag is set, the data set is kept. If the SSS2RHLD flag is set, the data set is kept and made non-selectable for this application.

(5). - Completion of SYSOUT Call - PUT/GET

5. At the completion of a SAPI PUT/GET IEFSSREQ request, and depending on whether data sets were selected or not, the application thread can do the following:

   a. If no data set is returned for the PUT/GET request (SSS2EODS is set) by JES:

      1) The application thread can end the SAPI GET/PUT processing by turning on SSS2CTRL in SSS2MSC1 and issuing an IEFSSREQ to inform JES that it can the PUT/GET processing is completed. (Shown in step 12.)

         or it can

      2) WAIT for a data set to arrive into the JES spool by using an ECB passed in SSS2ECBP. (This is shown in step 11, and you can go to step 4 when it is POSTed.) The application thread must provide a pointer to an ECB in field SSS2ECBP if the application thread wants JES to post it when newly created work has characteristics matching the thread′s selection criteria. This occurs after JES 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.

         Note: This ECB address may be set on all PUT/GET requests or done separately at end of data (go to step 11).

         Note: For JES3 only, once the application thread begins PUT/GET processing, a COUNT or BULK MODIFY request cannot be initiated prior to receiving an SSS2EODS response to a PUT/GET request.

      (5b). - Data Set Returned

   b. If a data set is returned for the PUT/GET request and the thread wants to access the data set, the data set needs to be allocated through DYNALLOC SYSOUT.

      JES has updated output fields in the IAZSSS2, some of which have to be copied to the DYNALLOC text units, as follows:

      1) Set DALSSREQ to the JES subsystem name

      2) Set DALDSNAM from the dsname in SSS2DSN

3) Update the DYNALLOC text unit pointer list with the text unit address from the SSS2BTOK field.

   Note: The text unit pointed from the SSS2TOK field is DALBRTKN and it has been entirely initialized by the JES. The application merely needs to provide its address in the text unit list for DYNALLOC.

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 JES, 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. JES then provides to 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.

Note: The use of either setting causes JES to perform additional processing overhead to satisfy this request. Thus, the application thread should not request the SWB information unless needed by the application. Examples of this additional overhead are spool I/O to read the stored SWBTU blocks, SJF services that JES needs to invoke to prepare the environment, additional GETMAINs needed to satisfy the request, on so on.

(6). - Open the SYSOUT Data Set

6. OPEN the SYSOUT data set by issuing an MVS OPEN macro.

(7). - Read SYSOUT Data Set

7. Obtain records from the SYSOUT data set by issuing a GET request.

(8). - Close the SYSOUT Data Set

8. Issue the CLOSE macro to close the SYSOUT data set.

(9). - Unallocate the SYSOUT Data Set

9. Issue an Unallocate request for the SYSOUT data set (override text units are ignored, do not supply them). The PUT portion of the SAPI GET/PUT processing occurs when the application thread subsequently issues a following IEFSSREQ macro to select another data set.)

(10). - What to do next

10. Modify the IAZSSS2 to do data set disposition on the unallocated data set, and:

    - Select another data set by setting the selection criteria if desired and go to step 4.

      or

    - If you want to end processing, set SSS2CTRL in flag SSS2MSC1 and go to step 12, as shown in Figure 45 on page 87.

Note: Step 4 must be the next request to do data set disposition on the previously closed data set and to request the next data set.

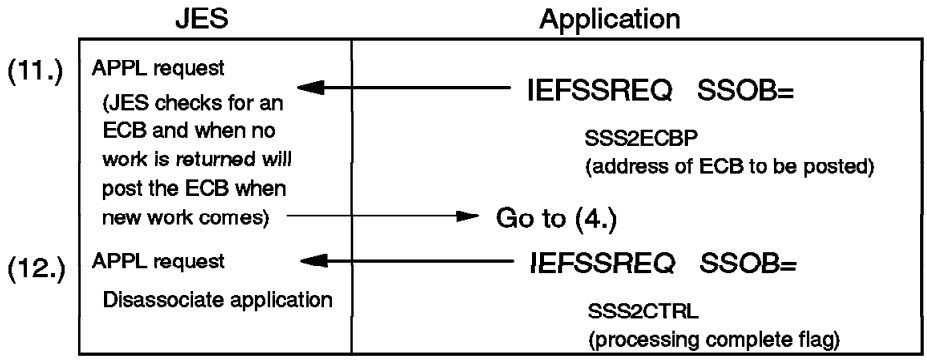


Figure 45. Processing When No Data Set is Returned

COUNT and BULK MODIFY are “immediate” SAPI requests. There is no posting of the ECB after either request has been processed by JES. If a thread wants to end processing after either request, it issues IEFSSREQ with the SSS2CTRL flag set.

See Appendix A, “Sample SYSOUT Application Program (SAPI)” on page 195 for a sample SAPI application.

5.2.1 Wildcard Support

The SAPI SSS2 input selection criteria may include wildcards. The supported wildcards are:

***** Matches zero or more characters.

? Matches one character.

The following fields support the use of wildcards:

SSS2JOBN Job name

SSS2CREA Owning user ID

SSS2PRMO Process modes

SSS2DEST Destination

SSS2PGMN User writer name

SSS2FORM Form numbers

Chapter 6. OS/390 Release 3 SDSF

The release of SDSF that supports OS/390 Release 3 is called OS/390 Release 3 SDSF and is an optionally priced feature available only through OS/390.

New function is added to improve the systems management and usability of the product. Also added is support for a workstation interface to SDSF. This workstation interface is provided by an IBM business partner, Phoenix Software International.

This release of SDSF can be installed with the OS/390 level of the BCP (5645-001), and the following levels of JES2:

- MVS/ESA SP-JES2 Version 5 Release 1 (5655-068)
- MVS/ESA SP-JES2 Version 4 Release 2 (5695-047)

Figure 46 shows which SDSF releases are installable with the versions of JES2 that are currently supported.

SDSF Release ↓	JES2 4.2.0	JES2 4.3.0	JES2 5.1.0	JES2 5.2.0	JES2 OS/390 Release 1	JES2 OS/390 Release 3
1.4	X	X	X			
1.5	X	X	X	X	X	
1.6	X	X	X	X	X	
OS/390 R3	X	X	X	X	X	X

Figure 46. SDSF and JES2 Compatibility

The functional enhancements available with OS/390 Release 3 SDSF are:

- Support for more JES2 devices, readers and punches, through the primary option menu.
- A security assist through a conversion tool to convert ISFPARMS to RACF profiles.
- Changes to the ARRANGE command for column widths.
- Overtype extension for fields previously modified only by operator commands.
- The ability to select the date format through pop-ups.
- Allowing user control for the default log, OPERLOG or SYSLOG.

6.1 New Device Support

OS/390 Release 3 SDSF implements new tabular displays that allow operators to display and control JES2 readers and punches. The displays support action characters for such things as starting, stopping, and halting the devices, and overtypes to control their characteristics.

The displays eliminate the need for operators to learn and remember the JES2 command syntax, and greatly reduce the amount of typing required. Also, they reduce the need to issue display commands that contribute to the size of the log and consoles.

The displays can be invoked through a command or by a choice from the primary option menu, as shown in Figure 47.

```
V1R7M0 -----------------  SDSF PRIMARY OPTION MENU ---------------
COMMAND INPUT ===>                                  SCROLL ===> PAGE
 Type an option or command and press Enter.
    LOG       - Display the system log
    DA        - Display active users in the sysplex
    I         - Display jobs in the JES2 input queue
    O         - Display jobs in the JES2 output queue
    H         - Display jobs in the JES2 held output queue
    ST        - Display status of jobs in the JES2 queues
    PR        - Display JES2 printers on this system
    INIT      - Display JES2 initiators on this system
    MAS       - Display JES2 members in the MAS
    LINE      - Display JES2 lines on this system
    NODE      - Display JES2 nodes on this system
    SO        - Display JES2 spool offload for this system
    PUN       - Display JES2 punches on this system
    RDR       - Display JES2 readers on this system
    ULOG      - Display user session log
    END       - Exit SDSF
```

Figure 47. OS/390 Release 3 SDSF Primary Option Menu

Note: This support continues the SDSF direction of providing a central point of control for JES2 devices.

The punch and reader displays consist of a row for each device defined to JES2, and are shown in Figure 48 and Figure 49 on page 91.

```
SDSF PUNCH DISPLAY                                    LINE 1-4 (4)
COMMAND INPUT ===>                                      SCROLL ===> PAGE
NP   PUNCH    STATUS   SFORMS   JOBNAME  JOBID    OWNER    REC-CNT REC-PRT
     PUN1     ACTIVE   STD      ROGERS   JOB00386 ROGERS   1,280
     PUN2     DRAINED  STD
     PUN3     DRAINED  STD
     PUN4     DRAINED  STD
```

Figure 48. OS/390 Release 3 SDSF Primary Option Menu

Device characteristics that are modifiable are displayed as overtypeable fields.

```
SDSF READER DISPLAY                                        LINE 1-4 (4)
COMMAND INPUT ===>                                           SCROLL ===> PAGE
NP   READER   STATUS   JOBNAME  JOBID    OWNER    REC-CNT    REC-PROC   C HOLD
     RDR1     ACTIVE   ROGERS   JOB38263 ROGERS                          A NO
     RDR2     DRAINED                                                   A NO
     RDR3     DRAINED                                                   A NO
     RDR4     DRAINED                                                   A NO
```

Figure 49. OS/390 Release 3 SDSF Primary Option Menu

Action characters can be used to perform actions against the devices, such as start and stop.

6.2 Security Assist Conversion Tool

With SDSF, there are two methods available to provide security, internal parameters (ISFPARMS) or SAF. The SAF support was added with SDSF 1.3.0, and it offers many advantages, including improved auditability and granularity. However, the complexity of converting from ISFPARMS to SAF has been a major obstacle.

The first offering to reduce this complexity was a redbook, *Enhanced Exploitation of RACF 1.9.2 SDSF Conversion*, GG24-4085. This redbook provided a tool for converting from ISFPARMS security to SAF/RACF security. With OS/390 Release 3 SDSF, this tool is integrated as part of the SDSF product.

Note: This tool requires ISPF and RACF to be installed.

A conversion EXEC, ISFACR, invokes an ISPF pop-up, as shown in Figure 50.

```
          ISFPARMS to RACF Conversion Utility

     Select conversion steps in order.

         1.  Define profile
         2.  Convert ISFPARMS to profile descriptions
         3.  Review profile descriptions
         4.  Convert descriptions to RACF commands
         5.  Review RACF commands



     F1=Help  F12=Cancel
```

Figure 50. ISFPARMS to RACF Conversion Utility Menu

All the panels for this conversion utility are pop-ups. PF keys can be used for editing the data sets used in the conversion.

Selecting **Option 1** from the menu in Figure 50 displays the pop-up shown in Figure 51 on page 92.

```
                SDSF Security Conversion Assist Profile
ISFPARMS input data set
===> 'SYS1.PARMLIB(ISFPRM00)'
Profile description data set
===> 'ROGERS.IN.SDSF'
CLIST library
===> 'ISF.V1R7M0.AISFEXEC'
RACF commands data set
===> 'ROGERS.SDSFRACF.CLIST'

Prefix for generated GROUP names
===> ISF
Owner group name for resource profiles
===> ISF
JES name for use in RACF resources
===> JES2

F1=Help  F4=Edit assem input  F5=Edit descriptions  F6=Edit RACF
F12=Cancel
```

Figure 51. ISFPARMS Conversion Profile (Option 1)

The data set names, RACF GROUP names and prefixes, and CLIST library shown in Figure 51 are the defaults chosen by SDSF (with ROGERS being a user ID of our SDSF session). You must create two data sets before pressing Enter from this screen as follows:

CLIST library — We used ISF.SISFEXEC which is the data set where the REXX utilities are that do the conversion for each of the steps shown in Figure 50 on page 91.

ISFPARMS input data set — You can specify an ISFPARMS input data set that is in either assembler format or statement (dynamic) format. If the input ISFPARMS data set is in statement format, that ISFPARMS is converted to assembler format and written to a data set, and is used later in generating the RACF commands.

userid.IN.SDSF — This data set must be created and is a sequential file with a minimum record length of 80. This data set is created when you select **Option 2** from Figure 51.

userid.SDSFRACF.CLIST — This data set must be created and is a sequential file with a minimum record length of 133. This data set is created when you select **Option 2** from Figure 51.

The PF keys shown in Figure 50 on page 91 and Figure 51 allow editing of the data sets described in the previous list after they have been created.

To continue, from Figure 50 on page 91 select **Option 2** and you will receive the pop-up shown Figure 52 on page 93.

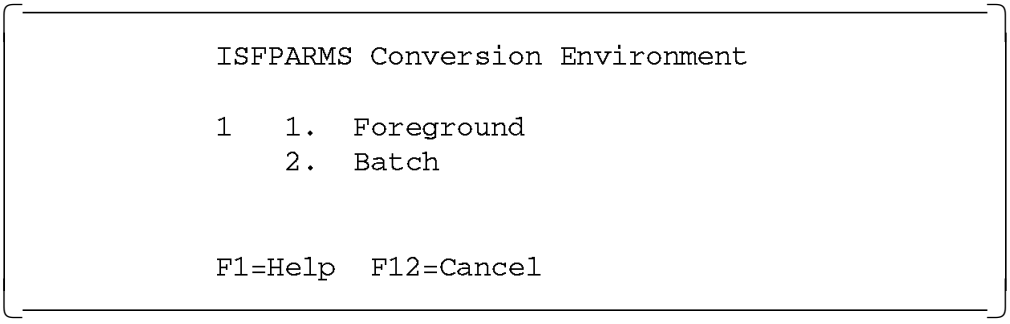


Figure 52. Convert ISFPARMS to Profile Descriptions (Option 2)

You can select foreground or batch and when you press Enter, the profile descriptions are created and presented on the screen. You can use PF5, as shown in Figure 50 on page 91 and Figure 51 on page 92, or select **Option 3** from Figure 50 on page 91.

After reviewing the profile descriptions and returning to the primary conversion menu, select **Option 4** to convert the descriptions to RACF commands. The presents the pop-up shown in Figure 53.

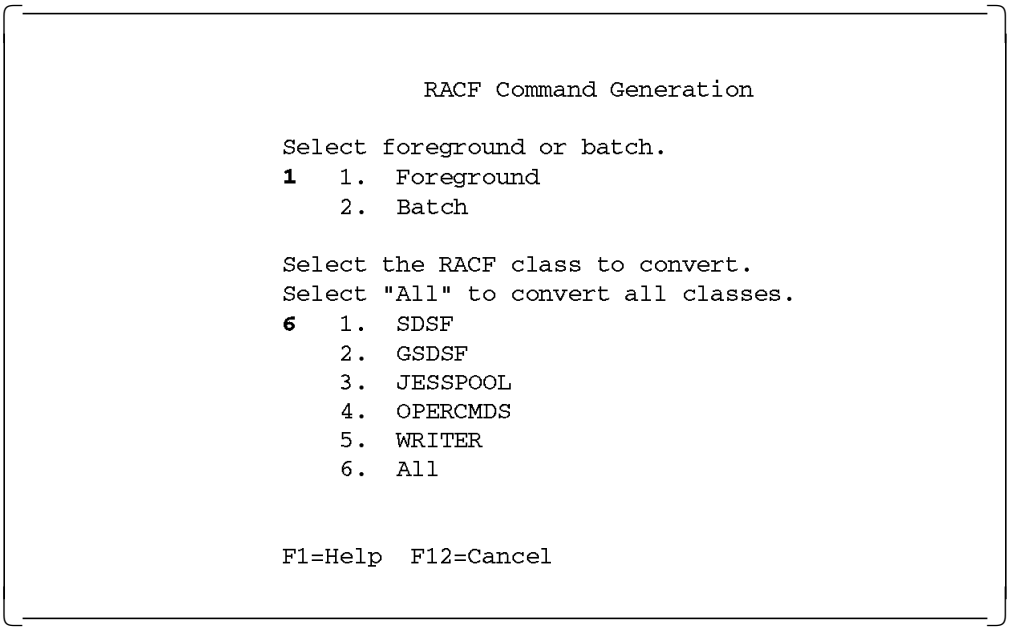


Figure 53. Convert Profile Descriptions to RACF Commands (Option 4)

You can run in foreground or in batch. You have the option of deciding which classes to convert. **Option 6** causes conversion for all appropriate RACF classes (SDSF, GSDSF, JESSPOOL, and OPERCMDS). The pop-up is primed with selections for foreground and RACF class SDSF.

After the profiles are generated and you are back at the the pop-up shown in Figure 50 on page 91, you can select **Option 5** to review the generated profiles.

Note: If you already had profiles defined in the selected classes, the utility that generates the new profiles deletes any old profiles in the selected classes. An example of some generated profiles are shown in Figure 54 on page 94.

```
RDEL OPERCMDS JES*.ROUTE.JOBOUT
RDEL OPERCMDS JES*.**
RDEL OPERCMDS MVS.CANCEL.ATX.*
/*PERMIT  MVS.CANCEL.TSU.* CLASS(OPERCMDS) ID( TRAUNER) DELETE*/
/*PERMIT  MVS.CANCEL.TSU.* CLASS(OPERCMDS) ID( SYS1) DELETE*/
RDEL OPERCMDS MVS.MCSOPER.*
RDEL OPERCMDS MVS.**
RDEL OPERCMDS *.**

/* Remove profiles in class RACFVARS */
/* Commands for  GROUP  profiles */
ADDGROUP  ISF OWNER(SYS1) SUP(SYS1 ) DATA('ISFPARMS GROUP OWNER#GROUP')
/* Commands for  GLOBAL  profiles */
RDEFINE  GLOBAL JESSPOOL OWNER(ISF) UACC(NONE ) DATA('JESSPOOL') AUDIT(
RALTER  GLOBAL JESSPOOL NOWARNING
RALTER GLOBAL JESSPOOL  ADDMEM(*.&RACUID.*/ALTER)
RALTER GLOBAL JESSPOOL  ADDMEM(*.*.&RACUID*/ALTER)
/* Commands for  JESSPOOL  profiles */
RDEFINE  JESSPOOL *.* OWNER(ISF) UACC(NONE ) DATA('SDSF') AUDIT(ALL)
RALTER  JESSPOOL *.* NOWARNING
```

Figure 54. RACF Profiles Generated by Option 5

6.3 ARRANGE Command Enhancements

OS/390 Release 3 SDSF enhances the ARRANGE command to allow end users to control the width of a *single* column on tabular displays. The Arrange pop-up, which previously allowed changing of the default column order, is extended to also control the default *multiple* column lengths, as shown in Figure 55. This allows a user to customize the columns on his SDSF screens to make more efficient use of the space on the SDSF panels.

Users can change the length of any single column with the ARRANGE command, or change the lengths of one or more columns with the Arrange pop-up.

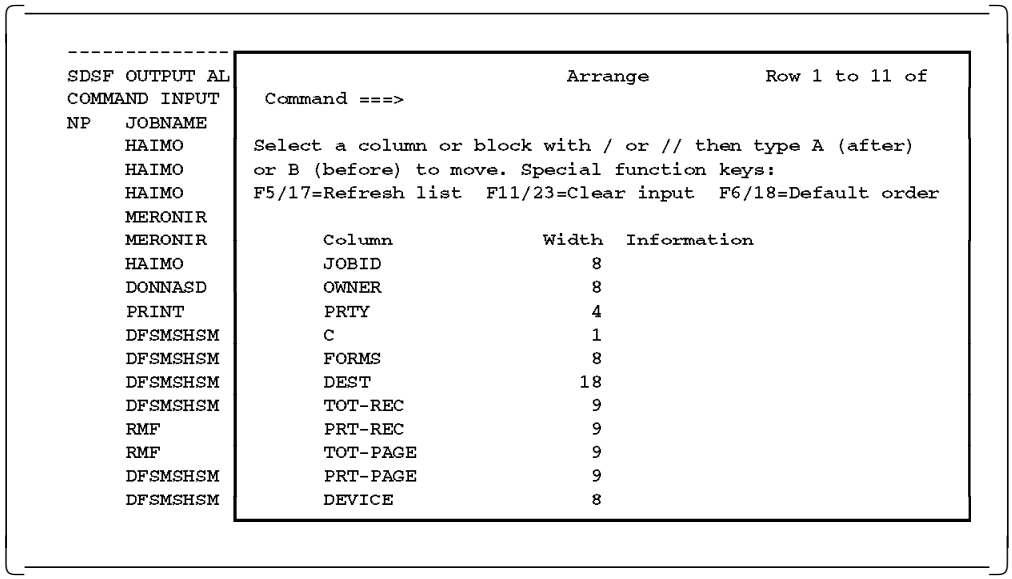


Figure 55. Arrange Pop-up to Change Column Widths

To change the column widths, just overtype the column length information shown in Figure 55.

Note: The lengths specified with Arrange are saved across sessions.

The Arrange support compliments the Overtype Extension support of this release, which allows users to display a pop-up for the maximum length of an overtypeable field, as shown in Figure 56 and Figure 57 on page 96.

In Figure 56, it is desired to change the priority of all the data set shown. This is now done by placing a **+** in the priority column of the first output job and the block repeat, **//=**, action characters in the *NP* column for the next output job. The last output job shown has the end block action character, **//**.

```
  Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF OUTPUT ALL CLASSES ALL FORMS     LINES 3,047,506  LINE 1-26 (991)
COMMAND INPUT ===>                                      SCROLL ===> PAGE
NP    JOBNAME  JOBID    OWNER    PRTY C FORMS    DEST    TOT-REC   PRT-REC  TOT-
      RCHIANG  TSU16383 RCHIANG     + A STD      U11         104
//=   SMF88RPT JOB06027 CICSRS4   144 A STD      LOCAL        72
      HERZOG1  JOB05338 HERZOG    144 A STD      LOCAL       256
      RECPTFS  JOB01225 RCONWAY   144 A STD      LOCAL        41
      LOGRSMF  JOB06028 CICSRS4   144 A STD      LOCAL        72
      SMF88RPT JOB01162 CICSRS4   144 A STD      LOCAL        72
      HERZOG1  JOB01257 HERZOG    144 A STD      LOCAL       256
      LOGRSMF  JOB01328 CICSRS4   144 A STD      LOCAL        72
      RECPTFS  JOB01542 RCONWAY   144 A STD      LOCAL        41
      OS33PSF  JOB18770 DAV       144 A STD      LOCAL         4
      OS33PSF  JOB18770 DAV       144 A STD      LOCAL
      OS33RAC  JOB15289 DAV       144 A STD      LOCAL         5
      OS33RAC  JOB15265 DAV       144 A STD      LOCAL        10
      OS33BDT  JOB14489 DAV       144 A STD      LOCAL        10
      OS3PSF   JOB12855 DAV       144 A STD      LOCAL         4
      OS3PSF   JOB12855 DAV       144 A STD      LOCAL
      OS3PSF   JOB12858 DAV       144 A STD      LOCAL         4
//    OS3PSF   JOB12858 DAV       144 A STD      LOCAL
```

Figure 56. Overtype Extension

Now press Enter, and Figure 57 on page 96 is displayed.

```
  Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF OUTPUT ALL CLASSES ALL FORMS     LINES 3,047,506  LINE 1-26 (991)
COMMAND INPUT ===>                                      SCROLL ===> PAGE
NP   JOBNAME  JOBID                                    T-REC   PRT-REC  TOT-
     RCHIANG  TSU16383      Overtype Extension          104
//=  SMF88RPT JOB06027                                  72
     HERZOG1  JOB05338    Column PRTY                  256
     RECPTFS  JOB01225    Maximum length 4              41
     LOGRSMF  JOB06028                                  72
     SMF88RPT JOB01162    Type a value.                 72
     HERZOG1  JOB01257    ===> 144                     256
     LOGRSMF  JOB01328                                  72
     RECPTFS  JOB01542                                  41
     OS33PSF  JOB18770    F1=Help  F12=Cancel           4
     OS33PSF  JOB18770
     OS33RAC  JOB15289 DAV       144 A STD      LOCAL     5
     OS33RAC  JOB15265 DAV       144 A STD      LOCAL    10
     OS33BDT  JOB14489 DAV       144 A STD      LOCAL    10
     OS3PSF   JOB12855 DAV       144 A STD      LOCAL     4
     OS3PSF   JOB12855 DAV       144 A STD      LOCAL
     OS3PSF   JOB12858 DAV       144 A STD      LOCAL     4
//   OS3PSF   JOB12858 DAV       144 A STD      LOCAL
```

Figure 57. Overtype Extension

Overtype the priority shown in Figure 57, and when you press Enter, the priority of all the output jobs is changed to the new specified priority.

6.4 Selecting Data Formats

The date format can be set by using the Options action bar, which displays the pop-up shown in Figure 58.

Select **Option 16** and press Enter.

```
  Display  Filter  View  Print  Options  Help
-----------------------------
SDSF OUTPUT ALL CLASSES ALL F  16  1. Set action character display...
COMMAND INPUT ===>                 2. Find limit...
NP   JOBNAME  JOBID    OWNER       3. Change include SYSIN to ON
     HAIMO    TSU20609 HAIMO       4. Set bookshelf...
     HAIMO    TSU20609 HAIMO       5. Set display values to ON
     HAIMO    TSU20609 HAIMO       6. Set screen characteristics...
     MERONIR  JOB20761 MERONI      7. Set delay for responses...
     MERONIR  JOB20895 MERONI      8. Set console name...
     HAIMO    TSU20972 HAIMO       9. Set search characters...
     DONNASD  JOB21085 DONNAS     10. Assign PF keys...
     PRINT    JOB21086 DONNAS     11. Change show PF keys to ON
     RONN     TSU18521 RONN       12. Set language for help and tutorial..
     MERONI   TSU20730 MERONI     13. Set cursor to OFF
     EJES$ENV JOB19899 HAIMO      14. Set confirmation to OFF
     PRINT    JOB21087 DONNAS     15. Operlog limit for filter...
     PRINT    JOB21091 DONNAS     16. Set date format...
     PRINT    JOB21150 DONNAS
     BUILDMCS JOB20155 HAIMO      144 B STD      LOCAL
```

Figure 58. Setting the Date with Options Action Bar

When you press Enter, Figure 59 on page 97 is displayed. You then select the proper date format by selecting the date order, and you can now also select the separator between the day, month, and year.

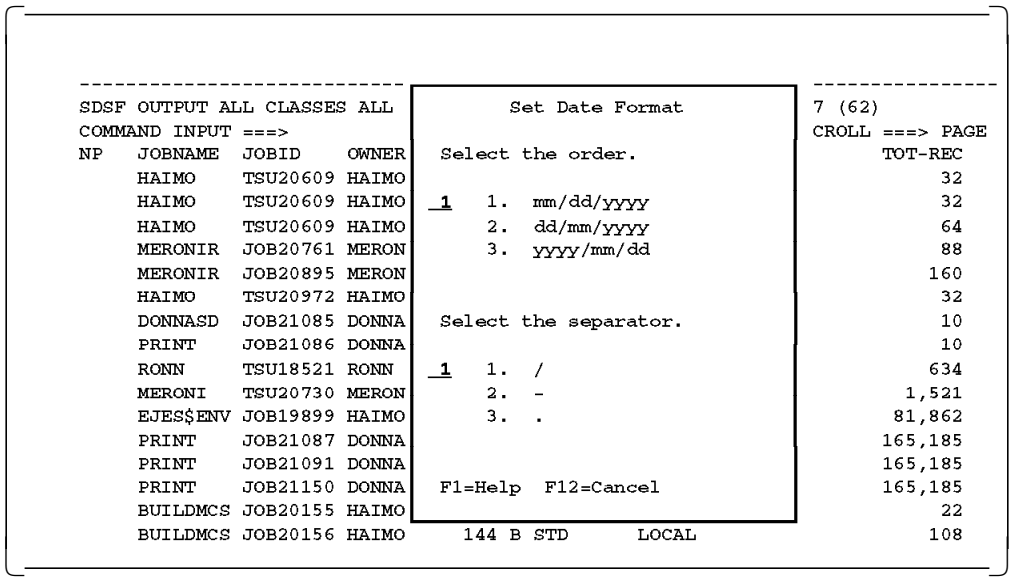


Figure 59. Setting the Date Format

The date format can also be set by operator command as shown:

```
SET DATE YYYYMMDD .
SET DATE YYYYMMDD /
```

This first command sets the date as 199./06.30, and the second command sets the date as 1997/06/30.

6.5 Setting the Log Default

The log default can be set by using the Options action bar which displays the pop-up shown in Figure 60 on page 98. In previous SDSF releases, if the OPERLOG and SYSLOG were both active, when a user issued LOG on the command line, the default log displayed was the OPERLOG.

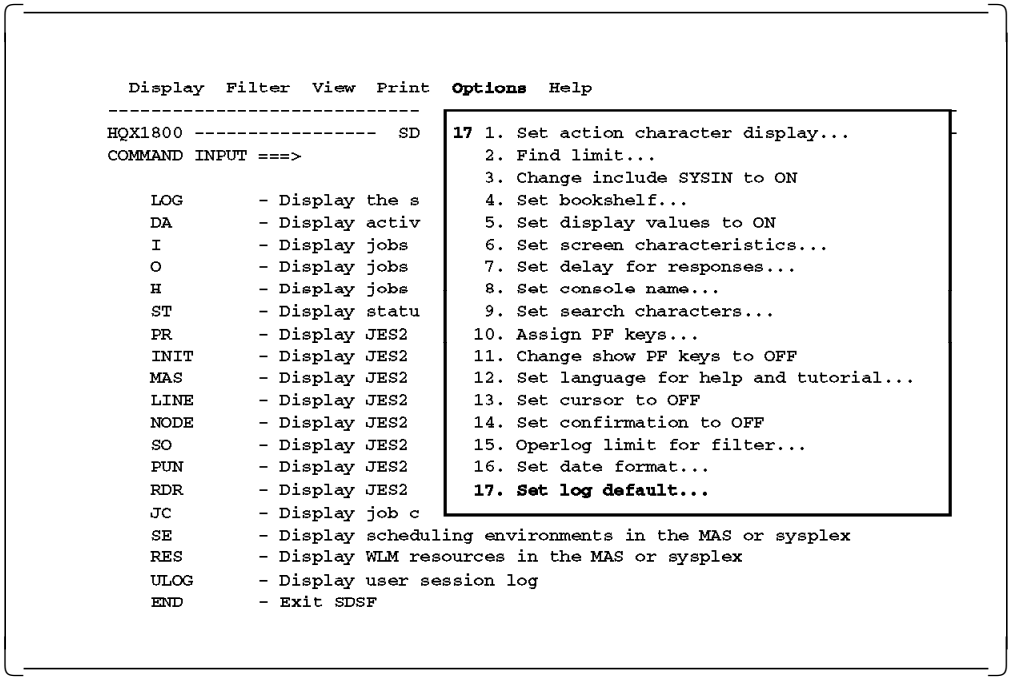


Figure 60. Setting the Log Default

If you select **Option 17** and press Enter, Figure 61 will be displayed. You can now choose what your user default will be when you Enter LOG on the command line.

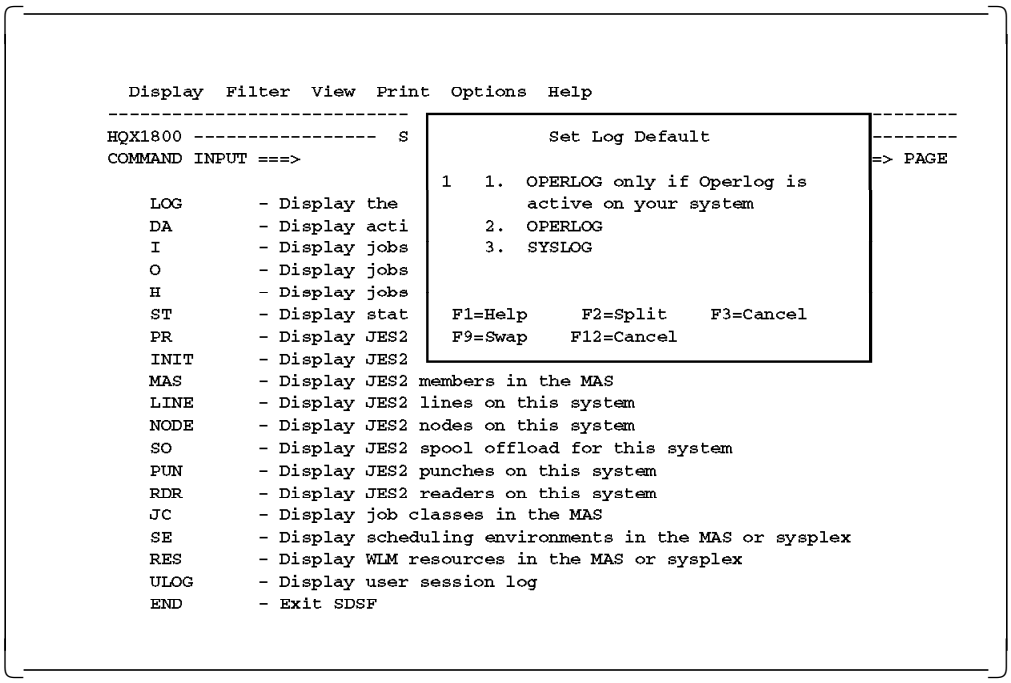


Figure 61. Setting the Date Format

Chapter 7. A Sample Functional Subsystem (SFSS)

With OS/390 Release 3, a sample functional subsystem is shipped in SYS1.SAMPLIB. This sample code is provided to assist in the writing of a user-written functional subsystem. This sample FSS is meant to complement the documentation in the publication *OS/390 Using the Functional Subsystem Interface*.

An FSS is a logical output-device processing component. An FSS communicates with the MVS job entry subsystem (JES) to take output data sets (SYSOUT) from the JES spool, read in the data and attributes, and write the data appropriately to an output device or other logical output means. JES/FSS communication for controlling the device is also supported. This overview is shown in Figure 62. A current exploiter of the FSS is Print Services Facility (PSF). PSF is the application code, called the functional subsystem application (FSA), in the FSS address space shown in Figure 62.

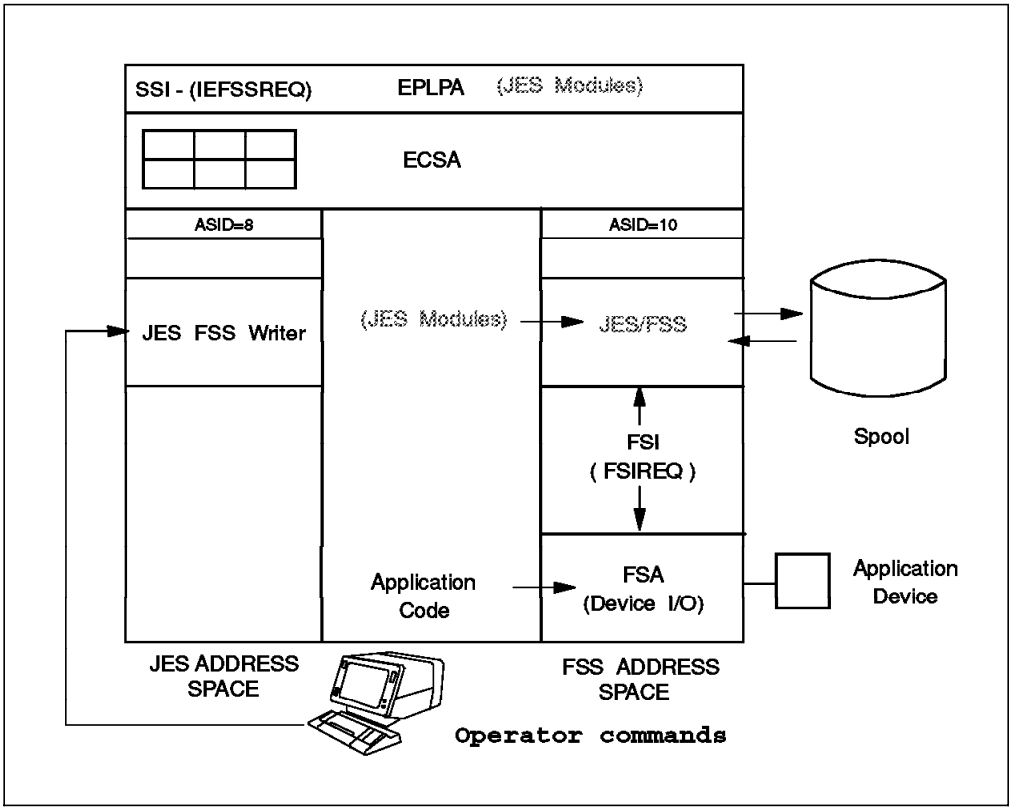


Figure 62. Overview of Functional Subsystem (FSS) Processing

7.1 Using the Functional Subsystem Interface (FSI)

he application code talks to the JES code in the FSS address space by using the Function Subsystem Interface (FSI) via the FSIREQ macro. This interface is documented in *OS/390 Using the Functional Subsystem Interface*.

The application code selects SYSOUT data sets using a GETDS request through the FSI. The JES code passes the request to the JES address space and a data set is selected and sent back to the FSS address space.

The JES code in the FSS address space reads the spool when the application code makes a request through the FSI. The spool data is passed to PSF using the FSI.

The FSA and JES use the FSIREQ macro to invoke FSI services. The FSIREQ macro allows JES to issue orders to the FSA and the FSA to issue requests to JES. The macro requests using the FSIREQ access services are:

GETDS The GETDS service requests access to a JES spool data set and its characteristics. JES selects a data set from the output queue.

GETREC The GETREC service obtains one or more records from the accessed data set. JES code in the FSS address space reads the spool.

FREEREC The FREEREC service frees one or more logical records that it previously acquired with a GETREC request.

RELDS The RELDS service releases a data set previously obtained by the GETDS service. JES makes the data set complete for output processing in the JES address space.

CHKPT The CHKPT service requests JES to record checkpoint information for the JES spool data set currently being processed on the FSA device. The checkpoint information is used for restart situations.

7.2 Using the Sample Functional Subsystem

The sample code written in assembler consists of five modules and is a working illustration of how you might implement your own functional subsystem. This sample code is a starting point for applications programmers to develop an FSS which can be used to drive output devices, such as plotters, microfiche writers, or other devices. More importantly, since the FSS selects output data sets from JES, once the data is obtained, any installation processing of the output can be done.

The five modules in SYS1.SAMPLIB are the sample FSS code and they also contain general documentation for the FSS interface example. These modules provide the following functions:

IAZSFSS This module operates the FSS main task and contains the entry point for the load module. This module gets control when JES issues the START command, as shown in Figure 67 on page 104.

IAZSFSA This module operates the FSA subtasks. It executes under the subtask created when an ATTACH is done by IAZSFSS, and it initiates the application or FSA. Multiples of these subtasks are supported.

IAZSFSD This module operates an output device or can be used to support any distribution of the output data sets. It gets control from IAZSFSA. The code shipped in this module selects SYSOUT from JES and writes it back to the spool as a SYSOUT data set. This part of the module should be replaced with an installation method of using the SYSOUT data set.

IAZSFSJ This module handles communication between JES and the FSS or an FSA.

IAZSFSE This module provides recovery for the FSS and FSA tasks.

7.2.1 Sample FSS Devices

The sample FSS must be activated through the use of a device. This device must be defined to JES. The device definition is identical to the PSF-controlled devices. Even if a real device is not being supported, a device must be defined. The SFSS supports multiple devices per address space. The FSS-to-FSA connections and communications, in the sample FSS, are based on the standard task structure of a mother task (the FSS) running multiple daughter tasks (the FSAs). The FSS does an ATTACH for the FSA tasks when JES issues a START FSA order, as shown in Figure 67 on page 104.

7.2.1.1 JES2 Device Definitions

The JES2 initialization definitions for an SFSS are identical to those for PSF. The FSS and PRTnnnnn statements are required as follows:

```
FSS(wtriaz)  PROC=wtriazf
PRT(prt1)  FSS=wtriaz,MODE=FSS
```

Other parameters may be used on the FSS(fssname) and PRTnnnnn statements if required.

7.2.1.2 JES3 Device Definitions

The JES3 initialization definitions for an SFSS are identical to those for PSF. The FSSDEF and DEVICE statements are required as follows:

```
FSSDEF,FSSNAME=wtriaz,PNAME=wtriazf
DEVICE,JNAME=prt1,MODE=FSS,FSSNAME=wtriaz
```

Other parameters may be used on the FSSDEF and DEVICE statements if required.

7.2.1.3 SFSS Procedure

An SFSS procedure must exist in SYS1.PROCLIB. In our example of the JES initialization definitions, the procedure name is WTRIAZF. The execution program is IAZSFSS, as shown in Figure 63:

```
//IAZSFSS  EXEC PGM=IAZSFSS,REGION=0M,TIME=1439
//PARMLIB  DD  DSN=SYS1.PARMLIB(WTRIAZF0),DISP=SHR  <-- Optional
```

Figure 63. Sample FSS Procedure

The PARMLIB DD is optional and it specifies procedure options. There are three ways to specify the options:

1. These options may be placed on the EXEC statement as follows:

   ```
   PARM='opt1=val1,opt2=val2,......'
   ```

2. All parameter options have defaults.
3. The operator may set parameter options by using the MODIFY command.

Note: The parameter options are documented in module IAZSFSS.

7.2.1.4 Parameter Options

The IAZSOPT macro in SYS1.SAMPLIB is used to generate the options table when the SFSS is activated. Figure 64 shows the parameter options with the defaults underlined.

```
AMODE=31|24
CONERR-RETRY=NONE|FSS|FSA|BOTH
CONNECT-WTOR=NONE|FSS|FSA|BOTH
ESTAE-SDUMP=YES|NO
ESTAE-WTOR=YES|NO
MSG-SUPRESS=NONE|TRIVFSS|TRIVFSA|TRIVIAL|INFO1|INFO2
MSG1-SCROLL=NO|YES
SJF=YES|NO
NOTIFY=YES|NO
SEPARATORS=YES|NO
SMF=YES|NO
SPIN=GROUP|NONE|DATASET
SPIN-CLASS=Z|x
WAITS-CA=(0MS,0MS)|(xMS,yMS)
WAITS-SNA=(0MS,0MS)|(xMS,yMS)
DEVNAME-LEN=4|3
```

Figure 64. Parameter Options

7.2.2 Installing the Sample FSS

Installing the SFSS is very simple. The following steps are necessary:

1. Assemble the source supplied in SYS1.SAMPLIB (the five modules).

```
//ASMIT PROC MEMBER=,LIB='SYS1.SAMPLIB',
//                  OBJLIB='MYFSS.OBJLIB'
//ASM  EXEC PGM=ASMA90,PARM=('XREF(SHORT)',NOTEST,RENT,NODECK,
//                  OBJECT,BATCH,NOTERM),REGION=2500K
//SYSIN DD DSN=&LIB(&MEMBER),DISP=SHR
//SYSLIB DD DSN=&LIB,DISP=SHR
//             DD DSN=SYS1.MACLIB,DISP=SHR
//             DD DSN=SYS1.AMODGEN,DISP=SHR
//SYSLIN DD DSN=&OBJLIB(&MEMBER),DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD DSN=&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(2,2))
//ASMIT PEND
//*
//IASSFSA EXEC ASMIT,MEMBER=IAZSFSA
//IASSFSD EXEC ASMIT,MEMBER=IAZSFSD
//IASSFSE EXEC ASMIT,MEMBER=IAZSFSE
//IASSFSJ EXEC ASMIT,MEMBER=IAZSFSJ
//IASSFSS EXEC ASMIT,MEMBER=IAZSFSS
```

Figure 65. Sample Assembly Job for SFSS

2. The SFSS load module (IAZSFSS) must contain all five CSECTs. It must be APF-authorized and re-entrant. The following is a sample LINKEDIT job:

```
//SAMPFSS EXEC LKED,PARM='LIST,LET,XREC,AC=1,RENT,REUS'
//SYSLMOD DD DSN=MYFSSAPF.LINKLIB,DISP=SHR
//SYSIN DD DSN=MYFSS.OBJLIB(IAZSFSS),DISP=SHR
//             DD DSN=MYFSS.OBJLIB(IAZSFSA),DISP=SHR
//             DD DSN=MYFSS.OBJLIB(IAZSFSD),DISP=SHR
//             DD DSN=MYFSS.OBJLIB(IAZSFSJ),DISP=SHR
//             DD DSN=MYFSS.OBJLIB(IAZSFSE),DISP=SHR
//             DD *
 ORDER IAZSFSS
 PAGE IAZSFSS,IAZSFSA,IAZSFSD,IAZSFSJ,IAZSFSE
 NAME IAZSFSS(R)
 //*
```

Figure 66. Sample LINKEDIT Job for SFSS

The IAZSFSS module must be first in the load module IAZSFSS.

Note: The load module name IAZSFSS must be placed in the program properties table (PPT) with key=1 to ensure subpools 0-127 are in key 1.

3. Create a cataloged procedure as shown in 7.2.1.3, “SFSS Procedure” on page 101.
4. Define the device to JES as shown in 7.2.1.1, “JES2 Device Definitions” on page 101 and 7.2.1.2, “JES3 Device Definitions” on page 101.

7.3 Starting the Sample FSS

The sample FSS is normally started by the operator when activating a device. A typical example follows:

JES2 The operator issues a $SPRT1 command. This activates a PCE for the writer that issues the MVS START command:

START WTRIAZF.WTRIAZF,,,(JES2,0004),SUB=JES2

JES3 The operator issues a *X WTR,OUT=PRT1 command. This activates an FCT that issues the MVS START command:

START WTRIAZF.WTRIAZF,,,(JES3,1,,42)

The JES2- and JES3-issued START commands are shown in Figure 67 on page 104. This command starts the FSS address space and the IAZSFSS load module gets control.

An FSIREQ for FSA-CONNECT is issued to the JES by module IAZSFSS. The results of the FSA-CONNECT are verified. JES 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. It initializes the FSA and responds back to the JES address space.

7.3.1 Operator Starts the FSS

The operator starts the FSS by issuing the following commands:

JES2 $sprt1

JES3 *x wtr,out=prt1

JES then issues the MVS start command, as shown in Figure 67 on page 104.

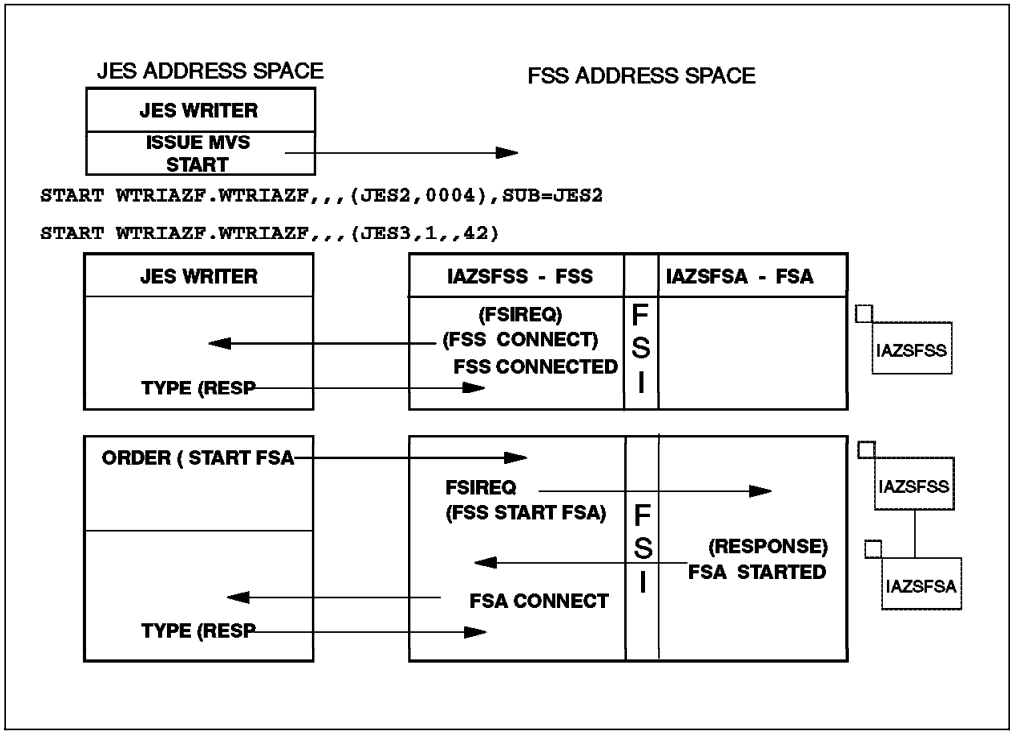


Figure 67. Sample Functional Subsystem (SFSS) Processing (Part 1)

Figure 68 shows the sample FSS address space TCB structure when JES2 is the primary job entry subsystem.

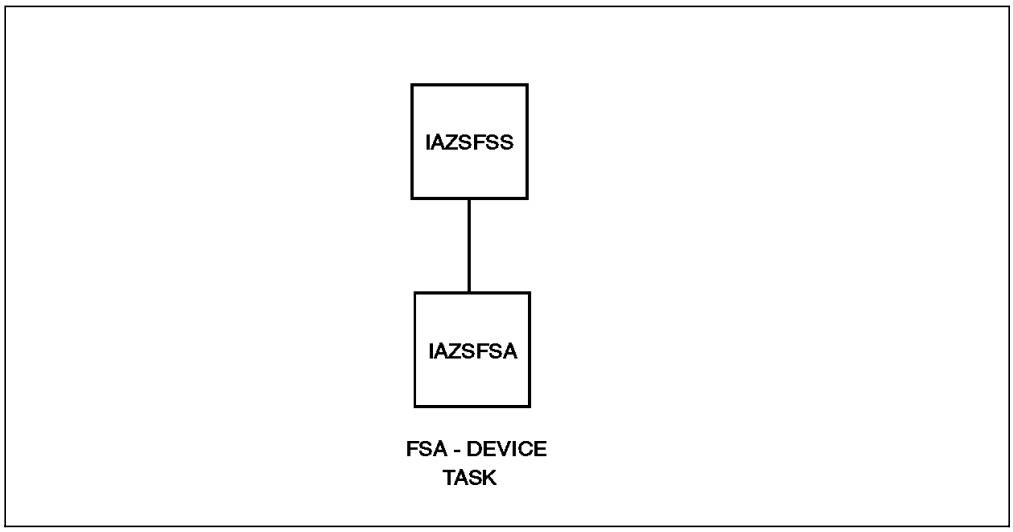


Figure 68. TCB Structure in SFSS Address Space (JES2)

Figure 69 on page 105 shows the JES2 start-up messages.

```
$SPRT1
$HASP000 OK
START  WTRIAZF.WTRIAZ,,,(JES2,0004),SUB=JES2
$HASP100 WTRIAZF  ON STCINRDR
IEF695I START WTRIAZF  WITH JOBNAME WTRIAZF  IS ASSIGNED TO USER STC
   , GROUP SYS1
$HASP373 WTRIAZF  STARTED
IEF403I WTRIAZF - STARTED - TIME=18.07.35
*SFSS001I WTRIAZ   FSS=006000, FSA=009000, FSD=00D000, FSJ=00E000
*SFSS002I WTRIAZ   FSID=00040000, ASID=004C, SUB=JES2, ROUT=  2
 SFSS051I WTRIAZ   OPTS:  AMODE=31,CONERR-RETRY=BOTH,
 SFSS051I WTRIAZ   OPTS:  CONNECT-WTOR=BOTH,DEVNAME-LEN=4,
 SFSS051I WTRIAZ   OPTS:  ESTAE-SDUMP=NO,ESTAE-WTOR=NO,
 SFSS051I WTRIAZ   OPTS:  FSA-TERM=0,MSG-SUPPRESS=NONE,
 SFSS051I WTRIAZ   OPTS:  MSG1-SCROLL=NO,NOTIFY=YES,
 SFSS051I WTRIAZ   OPTS:  SEPARATORS=YES,SJF=YES,SMF=NO,
 SFSS051I WTRIAZ   OPTS:  SPIN=GROUP,SPIN-CLASS=T,
 SFSS051I WTRIAZ   OPTS:  WAITS-CA=(OMS,OMS),WAITS-SNA=(OMS,OMS)
```

Figure 69. Sample SFSS Start-up Messages (JES2)

Figure 70 shows the sample FSS address space TCB structure when JES3 is the primary job entry subsystem.

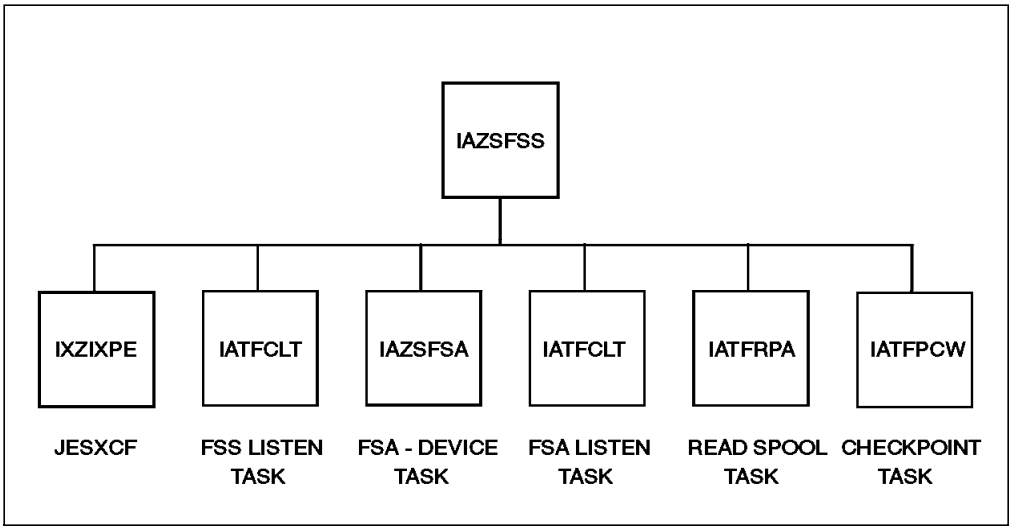


Figure 70. TCB Structure in SFSS Address Space (JES3)

Module IAZSFSA passes control to IAZSFSD for each important area of FSA processing logic:

Start FSA Order through FSS from JES, initialize driver

Stop FSA Order through FSS from JES, terminate driver

Start device Order from JES, start selecting FSA work

Stop device Order from JES, stop selecting FSA work

Device set up A new data set has been obtained from JES using a GETDS request

Data set put A new data record has been obtained from JES using a GETREC request

Data set close All data records have been obtained from JES using the GETREC request

Data set SMF An SMF Type 6 record is being written

These FSS areas of processing are not supported but could be added to the logic of IAZSFSA and IATSFSD.

Data set SJF A means to add to the table of OUTPUT attributes that are retrieved via SJF

Operator interv Order from JES for operator intervention

Synch Order from JES, Synch ORDER

Set Order from JES, Set ORDER

JES issues the ORDER to start the device, as shown in Figure 71. Once the START DEVICE has been responded to from the FSA, the FSA may start selecting output data sets from JES using the GETDS. A GETDS request is made from IAZSFSA and JES responds with a data set.

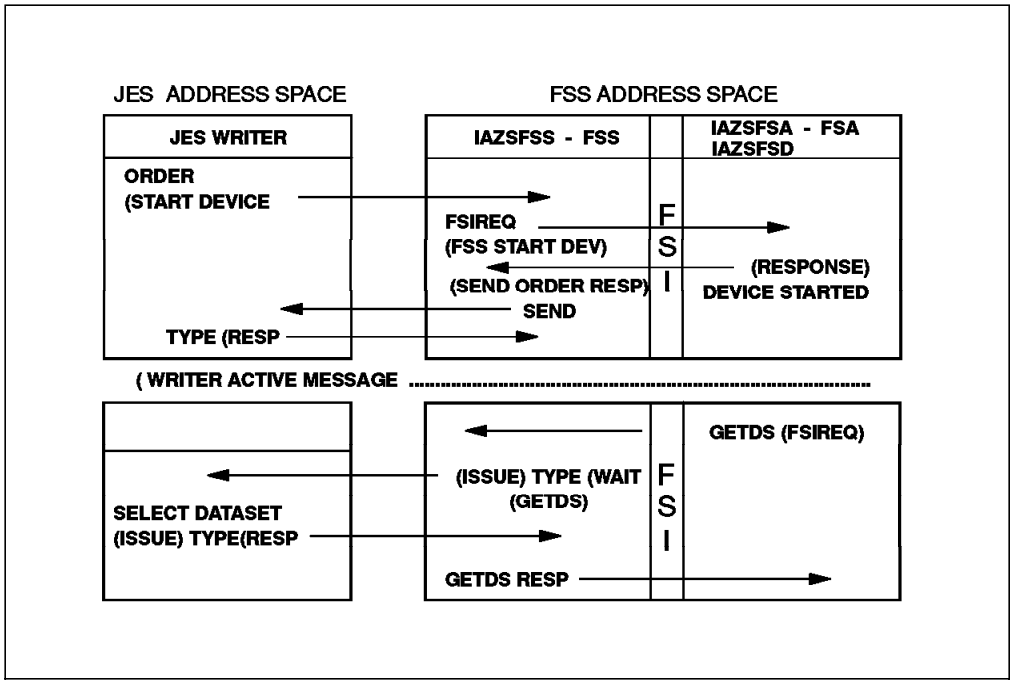


Figure 71. Sample Functional Subsystem (SFSS) Processing (Part 2)

As shown in Figure 72 on page 107, 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 JES address space.

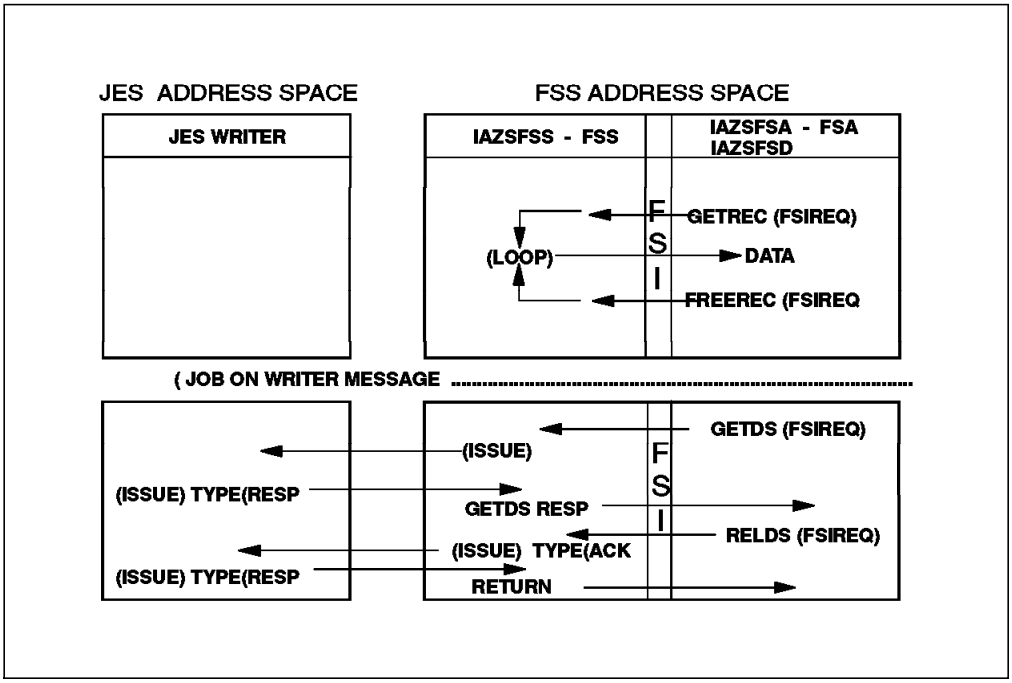


Figure 72. Sample Functional Subsystem (SFSS) Processing (Part 3)

Chapter 8. IP PrintWay and NetSpool

The new IP PrintWay and NetSpool elements of OS/390 transmits output data sets from the job entry subsystem (JES) spool to printers in a TCP/IP network. IP PrintWay, acting as a line printer client (LPR), converts output data sets from EBCDIC to ASCII, if requested, and transmits the data set to any line print daemon (LPD) in a TCP/IP network. IP PrintWay can transmit data to LPDs running in a printer or to LPDs running on a host system, including LPDs running on MVS, AIX, OS/2, OS/400, or UNIX systems.

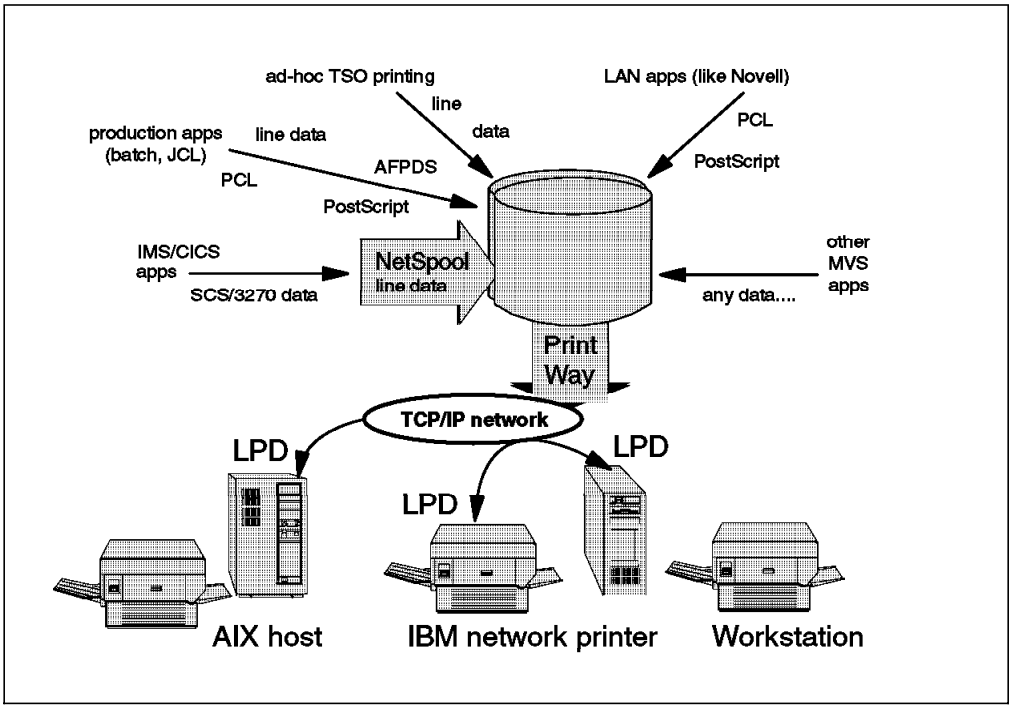


Figure 73. IP Printway and NetSpool

Note: The term LPD refers to any print server that observes the protocol defined in Request for Comment (RFC) 1179 and amendments.

IP PrintWay is now part of the IP PrintWay/NetSpool feature of OS/390 Release 3. IP PrintWay and NetSpool also are stand-alone features of PSF/MVS.

NetSpool converts data received from VTAM applications into S/370 line data and places the data on the JES spool. After NetSpool has created an output data set on the JES spool, JES, IP Printway, or PSF/MVS can print the data set or transmit it to another location for printing. NetSpool can be configured for your installation so that you do not have to change existing VTAM applications. That is, existing VTAM applications send print requests to NetSpool in the same manner as they currently send print requests to SNA-network printers.

NetSpool can be used with IP PrintWay to distribute VTAM application output (such as CICS and IMS) to a TCP/IP network, without modification of your applications.

Note: The IP PrintWay and NetSpool elements are recommended rather than the Network Print Facility (NPF).

8.1 IP PrintWay and NetSpool Functions

Some of the benefits IP PrintWay and NetSpool provide your installation include:

- IP PrintWay allows distribution of data for use with workstation applications. Data sets can be transmitted to a print queue on a workstation which allows you to use workstations not only for running applications but also for printing data.

  With NetSpool, multiple VTAM applications can simultaneously direct output to the JES spool for printing on a single shared physical printer or for routing to the target printers.

- Routing flexibility - MVS job submitters can specify routing information, such as the name of the target print queue and the IP address or name of the printer′s host system, in Job Control Language (JCL). Or, an installation can specify routing information for each target printer in an IP PrintWay routing data set created by your installation.

  To facilitate the use of NetSpool with IP PrintWay, you can define NetSpool logical printers in the same routing and options data sets used by IP PrintWay, instead of in the NetSpool print-characteristics data set.

- IP PrintWay supports the existing translation and formatting options of the MVS LPR command and also some additional options that affect the transmission of data sets. An installation can define sets of these options in an IP PrintWay options data set. An MVS job submitter can request a set of options in JCL, or the installation can associate a set of options with each printer in the routing data set.

- By temporarily storing the print data on the JES spool, IP PrintWay can retry transmission of data sets and also can verify that the transmission of data is successful before deleting data sets from the JES spool. In addition, IP PrintWay can retain data sets on the JES spool for a specified amount of time after either successful or failed transmissions.

  Netspool data integrity is improved by placing VTAM application print output on the JES spool. NetSpool enables you to use the security, checkpoint/restart, and reprint capabilities provided by JES for the VTAM applications.

- Netspool allows you to specify Advanced Function Presentation (AFP) parameters, for example, page definition, form definition, and character sets, to take advantage of AFP formatting when printing on PSF-controlled printers.

8.2 TCP/IP Remote Printing Services

TCP/IP provides client and server support for remote printing. The remote printing application allows you to spool data sets remotely to a line printer daemon (LPD). The line printer client or requestor (LPR) sends the spooled data set to a specified print server host and to a specified printer.

If an LPD running in a host system has queueing capability, it performs print spooling functions as well as print driver functions. Systems supporting LPDs include OS/390, MVS/ESA, AIX, OS/2, OS/400, or UNIX.

The line printer requestor (LPR) function and the line printer server (LPD) function are part of the IBM TCP/IP Version 3 Release 2 of the OS/390 base

feature. LPR is available as a TSO command, and the LPD server is implemented as an MVS started task.

8.2.1 TCP/IP and IP PrintWay

IP PrintWay transmits data to LPDs, also known as print servers, running in a printer or a host system. Figure 74 on page 112 shows the steps that occur from the time IP PrintWay selects data sets from the JES spool until IP PrintWay transmits the data sets to an LPD and deletes the data sets from the JES spool.

1. A job running on the MVS system creates output data sets on the JES spool. The output data sets may contain S/390 line data or they may contain formatted data, such as PCL, PostScript, or MO:DCA-P data.

   NetSpool is an example of an application that creates output data sets on the JES spool. The MVS spool data creator specifies the SYSOUT class, destination, and form name that corresponds to an entry in the IP PrintWay routing data set. Each routing entry contains the same routing information that can be specified directly through JCL parameters or dynamic allocation text units.

   In addition, in a JES2 environment, the MVS job submitter can specify the following information on the OUTPUT JCL statement for each output data set:

   - Name of the print queue
   - IP address or name of the printer's host system
   - The name of a set of transmission options
   - Retry parameters in the event of an unsuccessful transmission
   - The period of time to retain the data set on the JES spool after transmission

2. IP PrintWay, on MVS systems, runs as a functional subsystem writer, also called a functional subsystem application (FSA), as shown in Figure 74 on page 112. 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. An FSS extends the scope of JES processing. The IP PrintWay FSA selects output data sets from the JES spool, according to work-selection criteria defined for it through the JES parameters or operator commands. The work-selection criteria must correspond to the information set up in the PrintWay routing data set.

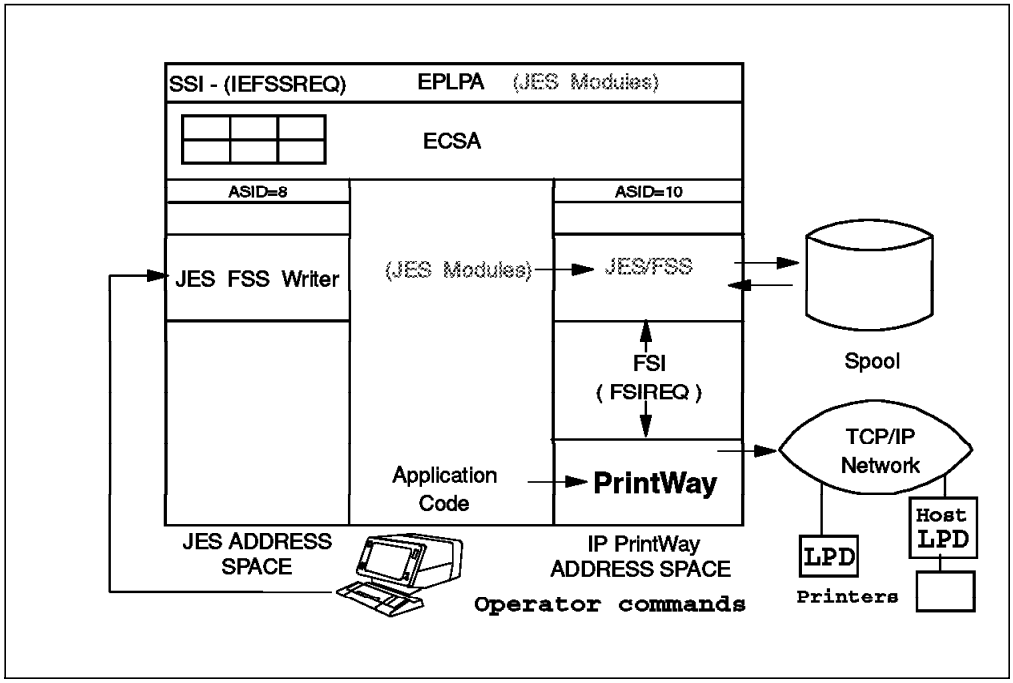


Figure 74. IP PrintWay Functional Subsystem

An installation can define several IP PrintWay FSAs to run in one functional subsystem address space (FSS), as well as several FSSs, to handle a high volume of data.

3. IP PrintWay routes each selected data set to an LPD, using routing information specified either in the JCL or in an optional IP PrintWay routing data set. The IP PrintWay routing data set is a VSAM data set created and maintained by the installation. It contains a routing entry for each target print queue. Each routing entry specifies the same routing information as an SYSOUT creator can specify for spool data sets.

   - IP PrintWay optionally translates spool data sets from EBCDIC to ASCII, using options specified in the IP PrintWay options data set. The IP PrintWay options data set is a VSAM data set created and maintained by the installation, which contains sets of options called options entries.
   - IP PrintWay maintains a transmission queue to keep track of data sets being processed. This transmission queue contains the status of each transmission, routing information, and option information. Using IP PrintWay ISPF panels, the system administrator can monitor the status of transmissions, reroute data sets to another print queue, and change the transmission options, if necessary.

     IP PrintWay transmits data sets to the target print queue using TCP/IP protocol. IP PrintWay also transmits printing options, such as the number of copies to print as specified for the source data set.

     IP PrintWay retries an unsuccessful transmission the number of times requested in either JCL or the routing entry. After successfully transmitting each data set, or after completing the requested number of transmission attempts, IP PrintWay either deletes the data set from the JES spool or retains the data set on the JES spool for the period of time specified in JCL or the routing entry.

8.2.2 Printing Choices

Figure 75 on page 114 summarizes some of the available choices for output printing in an OS/390 environment.

NetSpool NetSpool captures print data from SNA applications and puts the data on the JES spool. SNA applications like CICS and IMS can send print data through LU0, LU1, or LU3 sessions. A 3270 or SCS data stream is converted by NetSpool into a line print data stream and is placed on the JES spool for printing to:

- A remote LPD server using IP PrintWay
- A PSF print driver (to IPDS (Intelligent Printer Data Stream) printers)
- JES-managed printers (not shown in Figure 75 on page 114)

PSF/MVS PSF/MVS prints data sets from the JES spool to IPDS printers. PSF/MVS can format and print output data sets selected from the JES spool on printers in a TCP/IP network. PSF/MVS prints only on printers that support IPDS and are directly attached to TCP/IP or attached using the i-data 7913 IPDS Printer LAN Attachment.

IP PrintWay IP PrintWay takes data off the JES spool and transmits the data sets through the TCP/IP LPR/LPD protocol to servers and printers.

LPD server TCP/IP for MVS configured for the remote print server (LPD). You configure the TCP/IP for MVS LPD server by creating an LPD configuration data set to allow remote TCP/IP users to use printers on the NJE network which supports (JES, RSCS, and Power).

You start the LPD server as an address space in your MVS system. The LPD server allows users in the TCP/IP network to address JES-controlled printers and punches. A client from any TCP/IP host can use the local line printer requestor command (LPR) to print a local file on a JES-controlled printer. The printer may be a local JES system printer, or it may be a printer accessed through an NJE network.

LPR client A client, such as a TSO/E user, can use the LPR client function using the LPR command from any TCP/IP host, to print a local file on a JES-controlled printer or on remote TCP/IP printers.

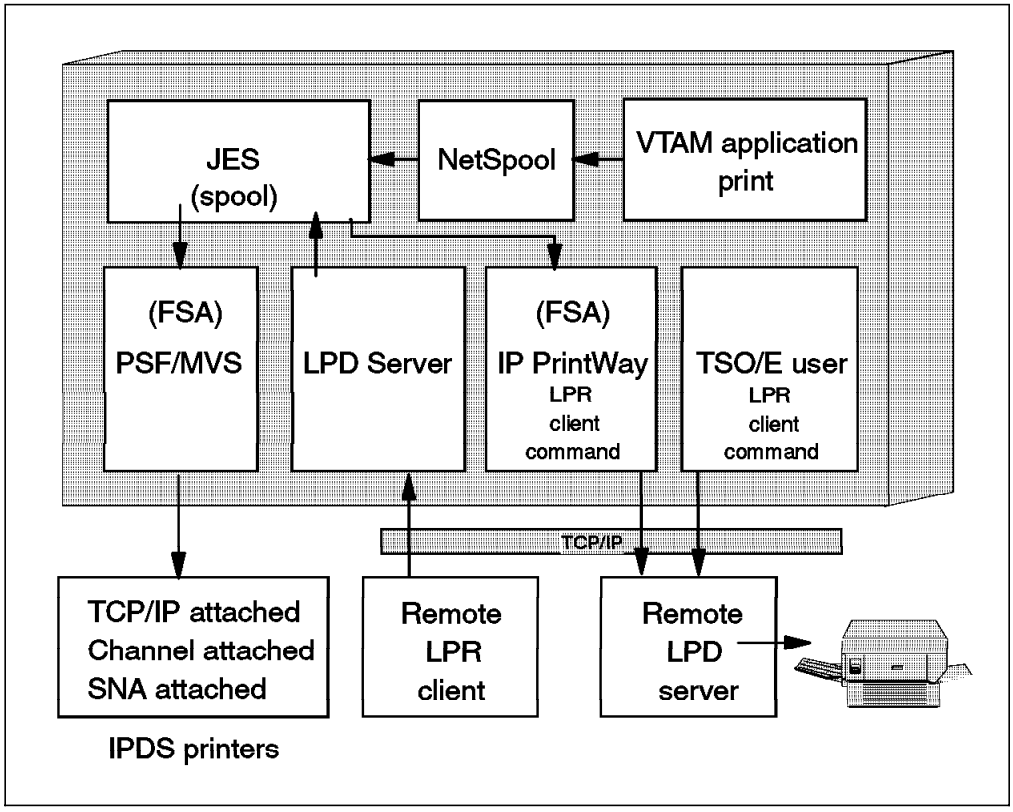


Figure 75. Printing with PSF/MVS, IP PrintWay, and NetSpool

8.2.3 IP PrintWay LPR Command

IP PrintWay has implemented a new IP PrintWay-specific LPR client command. This LPR command eliminates many of the restrictions with the current MVS TCP/IP LPR command. The LPR command is used to send the output to a selected printer, as shown in Figure 76 on page 115. The benefits provide improved job throughput with a large number of concurrent LPR/LPD sessions. IP PrintWay transmits data sets in the same order as they are acquired from the spool data set, ensuring FIFO order on the printer. A queue manager is currently started automatically in the FSS address space with an internal scan interval of 15 seconds.

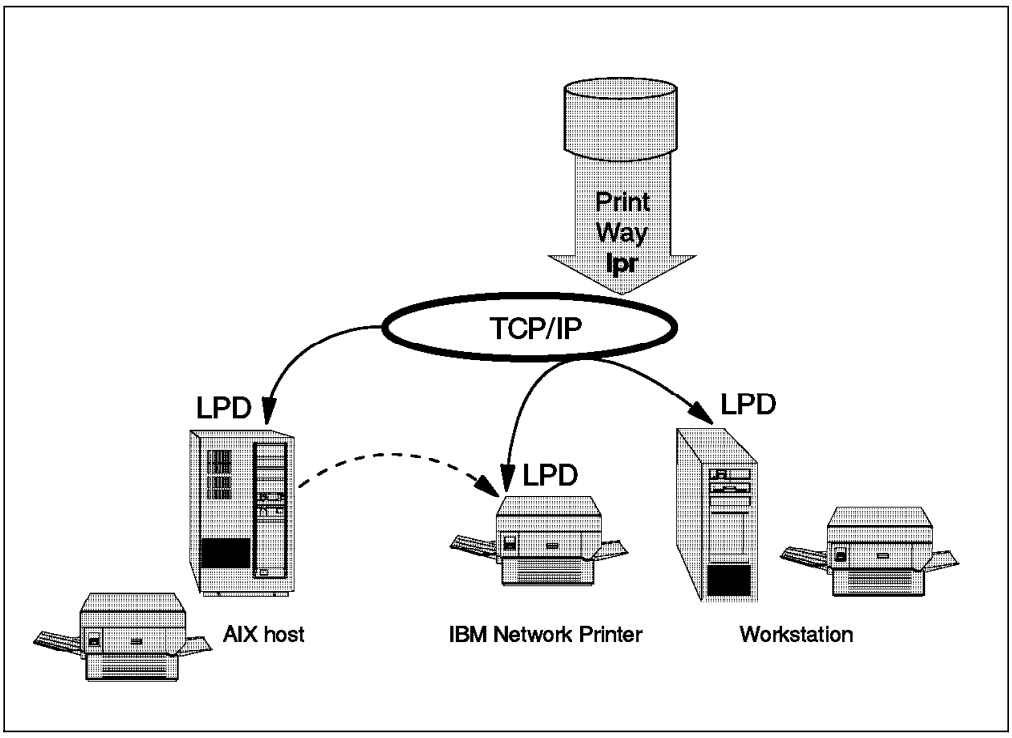


Figure 76. IP Printway and the LPR Command

Note: Use the options data set to specify the LPR options for the processing of data sets. See 8.3.1, “IP PrintWay Data Sets.”

8.3 Setting Up IP PrintWay

IP PrintWay executes as a functional subsystem (FSS) and is a feature of PSF/MVS. To define the IP PrintWay environment, the following definitions are required:

- IP PrintWay data sets
- TCP/IP
- JES initialization definitions
- An IP PrintWay procedure in SYS1.PROCLIB

8.3.1 IP PrintWay Data Sets

IP PrintWay provides sample JCL in the ANF.SANFSAMP data set ANFDEAL member that deletes or allocates the VSAM data sets needed for IP PrintWay. This sample JCL should be used to define the routing, options, and queue data sets. The JCL requires modification to run and if you change the data set names, also change the data set names in the IP PrintWay sample startup procedure, shown in Figure 89 on page 131.

The options, routing, and queue VSAM key-sequenced data sets are shown in Figure 77 on page 116.

ISPF panels can be used to create, browse, modify, or delete entries in the routing and options data sets. The alternative to ISPF panels is a macro, ANFGPWFL, that creates a batch job for creating, modifying, and deleting routing and options entries. See the *IBM IP PrintWay Guide* for the details on using ISPF or the macro.

The ANF.SANFSAMP data set ANFMIJCL member provides JCL to allocate and initialize the IP PrintWay message-log data set. After modifying this JCL, you can use it to allocate space for and initialize the PrintWay message-log data set.

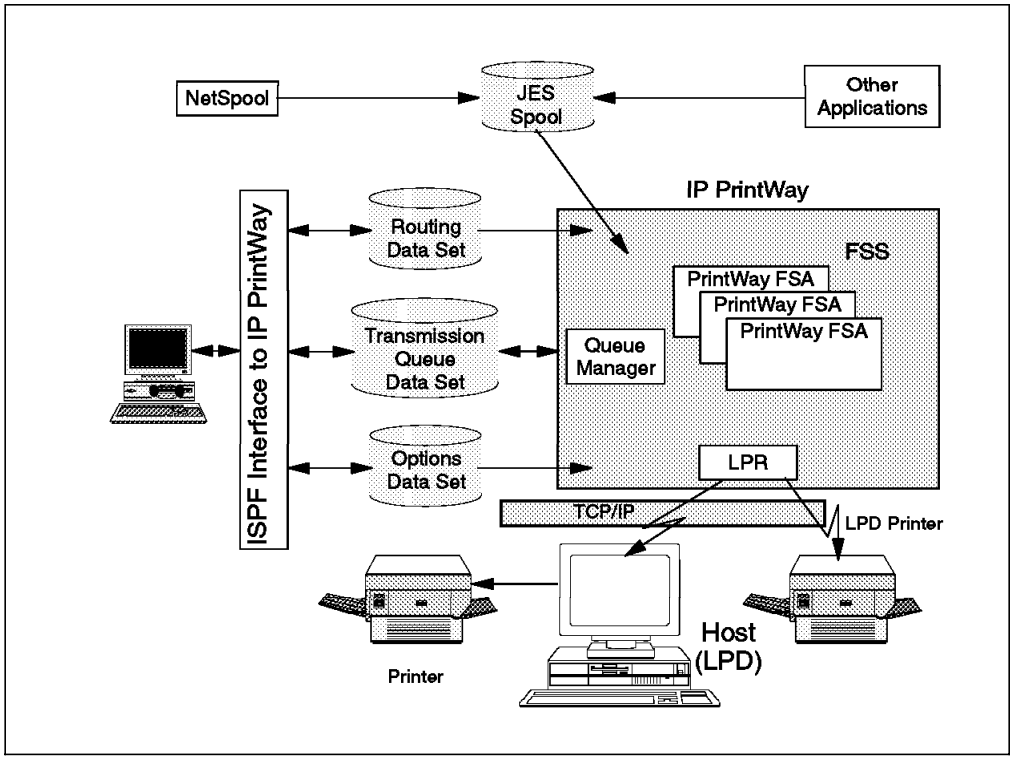


Figure 77. Printing with IP PrintWay

The options and routing IP PrintWay data sets are optional. The *queue* data set is *required*. These data sets provide the following functions:

Routing The IP PrintWay routing data set contains a routing entry for each target print queue. Each routing entry contains the name of the target print queue, the IP address or host name of the printer's host system, retry and retention information, and the name of an options entry which contains transmission options.

When a JES spool data set is created that is to be processed by IP PrintWay, the creator must specify the class, destination, and form name information for the data set that corresponds to an entry in the IP PrintWay routing data set in the JCL. IP Printway uses the class, destination, and form name information to locate a routing entry. The routing entry contains in addition to the "matching" information, the "routing and processing data" -- the name of the options entry, the name of the print queue on the target system, and the host name or IP address of the target system.

Note: The routing data set is not required if all users use the OUTPUT statement to specify the IP address and PRTQUEUE as follows:

```
//R1      OUTPUT  DEST='IP:9.12.14.63',
//        PRTQUEUE='lpt2'
```

For the other new keywords on the OUTPUT statement, see 8.5.1, "IP PrintWay JCL Keywords" on page 133.

Note: JES3 does not support the OUTPUT statement JCL parameters required to specify routing information for IP PrintWay. Therefore, in a JES3 environment, an installation must create an IP PrintWay routing data set.

Options

IP PrintWay, as an option, can convert data streams from EBCDIC to ASCII. This option is specified in the IP PrintWay options data set which contains the LPR options for a routing entry. The name of an option entry is specified in a routing entry. One options entry might define options for data streams that require conversion to ASCII format, and another might define options for data streams that do not require conversion, such as PCL, PostScript, or MO:DCA-P data streams.

Note: If routing information is specified in the job submitter's JCL, it is recommended to have one options data set for all IP PrintWay FSSs.

Queue

IP PrintWay selects output data sets from JES and places them on the transmission queue. The transmission queue contains the status of each transmission, routing information, and option information.

IP PrintWay initially creates a queue entry for each data set to be transmitted. IP PrintWay retains the queue entry until either IP PrintWay deletes the data set from the JES spool or the IP PrintWay FSA processing the data set ends normally or abnormally and releases the data sets it is processing back to JES so that another IP PrintWay FSA can process them.

Using the PrintWay ISPF panels, an administrator or operator can:

- Monitor the status of transmissions
- Reroute data sets to another print queue
- Change the transmission options

Note: The transmission-queue data set is a required data set. If more than one PrintWay FSS is defined, you should allocate a separate transmission-queue data set for each FSS. See Figure 88 on page 130.

Message-log

IP PrintWay writes messages that track data-set transmissions in an IP PrintWay message-log data set. The IP PrintWay message-log data set is a required data set. You can create one message-log data set to be shared by all IP PrintWay functional subsystems (FSSs). Alternatively, you can create separate data sets for each FSS.

The IP PrintWay message-log data set is organized as a circular queue. When a message extends to the end of the data set, the next message starts at the beginning. In this way, the size of the message data set remains stable, with old messages being automatically overlaid by newer messages.

The size of this data set should be considered because when it fills, it starts using entries beginning back at the top. The messages in this data set are not written to the SYSLOG or OPERLOG.

8.3.1.1 Maintaining the Routing Data Set

Using an ISPF interface, the primary options menu is shown in Figure 78, and it contains dialogs that can be used to create the routing and options data set entries.

Before your installation can use the PrintWay ISPF panels, you must enable the panels by performing these functions:

- Provide ISPF users with access to the PrintWay ISPF programs and panels.

  To assist you in enabling the PrintWay ISPF panels, PrintWay provides a REXX EXEC in ANF.SANFEXEC(ANFINIT). The ANFINIT REXX exec contains code to perform the following functions:

  – Provide access to the PrintWay ISPF programs and panels
  – Define default names for the PrintWay data sets accessed from the panels
  – Invoke the PrintWay main ISPF panel

  Once you have modified ANFINIT, ISPF users can invoke it as a stand-alone process, or you can modify the main ISPF panel so that users can invoke ANFINIT from the main ISPF panel.

  Note: All ISPF users can, by default, use the IP PrintWay ISPF panels to update the routing, options, and transmission-queue data sets. To prevent access or to allow read-only or read-write access for appropriate users, you can use the SAF interfaces to restrict access to the IP Printway data sets.

  If you have protected the IP PrintWay data sets, the PrintWay panels display different text for different users. The panels display only the actions that a user is allowed to perform. For example, if a user has read-only access to the PrintWay options data set, that user is allowed only to browse entries in the data set and is not allowed to add, delete, or modify entries.

- Provide ISPF users with access to required C run time and TCP/IP libraries.
- Define default names for the PrintWay data sets accessed from the panels.

Note: IP PrintWay can be running when you update the routing or options data set.

On the IP PrintWay primary option menu, selection **P** (PRINTERS), shown in Figure 78, brings up a menu that allows you to create the routing and options data set entries.

```
SC504  ------------------- IP PrintWay Primary Options Menu ------------------
OPTION ===> P
                                                              USERID - ROGERS
Type an option or command and press Enter.                    DATE   - 97/03/13
                                                              TIME   - 14:28
     P PRINTERS     - IP PrintWay Printer Configuration
     Q QUEUE        - IP PrintWay Queue Operations

     S SET DEFAULTS - IP PrintWay ISPF Defaults
     D DIAGNOSIS    - IP PrintWay Diagnosis Functions

     E EXIT         - Exit IP PrintWay
```

Figure 78. IP PrintWay Primary Option Menu

The IP PrintWay printer configuration menu, shown in Figure 79 on page 119, contains dialogs that can be used to create the routing and options data set entries, options 3 and 6.

```
SC5O4 ---------------- IP PrintWay Printer Configuration ---------------------
OPTION ===>
                                                                 USERID - ROGERS
Type an option or command and press Enter.                       DATE   - 97/03/13
                                                                 TIME   - 14:28
   1  SELECT ROUTES  - Select routing entries to list
   2  LIST ROUTES    - List all routing entries
   3  ADD ROUTE      - Add new routing entry

   4  SELECT OPTIONS - Select options entries to list
   5  LIST OPTIONS   - List all options entries
   6  ADD OPTION     - Add new options entry
```

Figure 79. IP PrintWay Printer Configuration

8.3.1.2 ISPF Dialog for Routing Entries

Selecting **option 3** allows creation of routing entries in the routing data set that are used to route print data sets to a printer in the TCP/IP network. The routing entry ISPF panel is shown in Figure 80 on page 120. The routing entry controls what to do with a data set, when it cannot be printed but has been selected from JES, by specifying a retry time and limit, and a retain time. You can optionally specify an options entry to be associated with the routing entry. The following fields are required:

- DEST
- CLASS
- FORMS

Note: These fields are used to select IP PrintWay processing controls for print requests. Data sets on the JES spool must exactly match a routing entry for the DEST, CLASS, and FORMS to be processed by IP PrintWay. Note also that on the IP Printway FSS writer work selection parameters you do not have to specify all of these parameters. For example, the class is enough for a data set to be selected by the IP Printway FSS writer. If IP PrintWay does not find a matching routing entry, it issues the following message into the message-log data set.

```
   ANFM131I Data set:  spool.data.set.name No
routing entry found, data set held
```

8.3.1.3 Routing Entry Specifications

Routing entries are created using the ISPF panel shown in Figure 80 on page 120. The routing entry data used to create the entry is as follows:

DEST The destination name must be used by the user in his JCL. See 8.5, “Requesting IP PrintWay Printing” on page 133.

CLASS This is the output class that is used by JES for scheduling output data sets to the IP PrintWay FSS. This class must be specified by the user in the JCL for the data set.

FORMS The forms name specified here must match of the forms of the data set scheduled by JES to the FSS.

Note: There must be an exact match of DEST, CLASS, and FORMS for the data set to be processed by the IP PrintWay.

Print Queue Name This name for some IP printers is a fixed name.

Host Name/IP Address This can be the host name of the IP printer or the actual IP address of the printer. The IP address can be used in the user JCL for routing to the printer.

Options Name This is the name of the options entry in the option data set. When the option entry is created, it contains the LPR options for processing the data set at the IP printer.

8.3.1.4 Default Routing Entry

A default routing entry should be created. This is necessary because if you specify the IP host name and print queue name, but none of the remaining JCL parameters, IP PrintWay uses the values from the default routing file entry for the remaining parameters, including the name of an options entry.

The destination (DEST) of the default routing entry should be DFLTNTRY. The default entry contains default values for the OUTPUT statement keywords if they are not present in the JCL of the submitter. See 8.5.1, “IP PrintWay JCL Keywords” on page 133.

```
ADD    ---------------------- IP PrintWay Routing Entry ----------------------
COMMAND ===>
                                                              USERID - ROGERS
DEST ===> DFLTNTRY   CLASS ===> _   FORMS ===> ________      DATE   - 97/03/13
                                                              TIME   - 14:28

Retry        : Time     ===> 00:10:00 (HH:MM:SS)
             : Limit    ===> 4      (0-999)
Retain Time  : Success  ===> 00:00:00 (HH:MM:SS)
             : Failure  ===> 00:00:00 (HH:MM:SS)

Print Queue Name        ===> ____________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________

Host Name/IP Address    ===> ____________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________

Options Name            ===> ____________________

NetSpool Printer Name   ===> ________
NetSpool LU Class       ===> ____________________________________________________
_____________________________________________________________________________
_____________________________________________________

NetSpool Default Page Format Entry ===> ________
NetSpool End-of-File Rules Entry   ===> ________
```

Figure 80. Adding a Default Routing Entry

8.3.1.5 Listing Routing Entries

Figure 81 shows the IP PrintWay routing list using the ISPF panels. The first entry shown is the default routing entry. You can display the entry by using the action character to browse it, or you can change or update the entry by using the edit action character.

```
DATE 97/03/12 TIME 14:39 - IP PrintWay Routing List ---------- Row 1 to 9 of 9
COMMAND ===>                                                  SCROLL ===> PAGE
Actions: B-Browse  C-Copy  D-Delete  E-Edit  O-Browse Options  U-Edit Options
A DEST     C FORMS    Print Queue  Host Name/IP Address         Options Name
  DFLTNTRY
  ITSO3130 J STD      afccu2       9.12.0.140                  MVSDATA
  I3130P   J STD      afccu2       9.12.0.140                  POSTSCRIPT
  LPT2     J STD      lpt2         9.12.14.63                  MVSDATA
  LPT2P    J STD      lpt2         9.12.14.63                  POSTSCRIPT
  RAHKISPB J STD      PASS         9.84.253.244                MVSBIN
  RAHKISPC J STD      PASS         9.84.253.244                MVSCC
  RAHKISPO J STD      PASS         9.84.253.244                POSTSCRIPT
  RAHKISP2 J STD      PASS         9.84.253.244                MVSDATA
****************************** Bottom of data ********************************
```

Figure 81. IP PrintWay Routing List Pane Showing Default Routing Entry

8.3.1.6 Maintaining Options Data Set

Each routing entry can name an options entry in the IP PrintWay options data set for IP PrintWay to use when transmitting data sets. An options entry contains LPR options that are used by the LPR component of IP PrintWay, the names of installation exits, and other transmission options.

If you do not name an options entry, IP PrintWay uses default LPR options described in 8.3.1.4, “Default Routing Entry” on page 120 and does not invoke the begin data set, end data set, or record exit routines. Otherwise IP PrintWay calls the named exits before transmitting each data set that uses the options entry. Using these exits, you can modify most LPR options; add one or more records to the beginning of a data set; add, replace, or delete records in a data set; and add one or more records to the end of a data set.

You specify the name of the options entry to use in each routing entry. You can specify the same options entry in more than one routing entry. Also, the job submitter can specify the name of the options entry to use on the PRTOPTNS JCL parameter.

Some LPR options affect the data processing that the LPR component of IP PrintWay performs before it transmits data sets to the LPD. Other LPR options specify how the LPD prints the data on the target printer.

8.3.1.7 Overview of LPR Options

Following is a brief overview of the LPR options:

BINary|NOBinary

EBCDIC-to-ASCII Translation: IP PrintWay can translate data streams from EBCDIC to ASCII before transmitting the data sets. (BINARY and NOBINARY options)

Translation Tables: When you request translation to ASCII for double-byte data streams, you can specify a DBCS language option which tells the IP PrintWay LPR which translation table to use.

For both single-byte and double-byte data streams, you can also specify the name of a customized translation table. (TRANSLATETABLE option)

SOSI option

DBCS Shift-Out and Shift-In Translation: When you request translation to ASCII for double-byte data streams, you can specify how IP PrintWay is to translate the EBCDIC shift-out (X′0E′) and shift-in (X′0F′) characters in the data stream. You select the type of SOSI translation required by the target printer. IP PrintWay supports the following types of SOSI translation:

- Translate EBCDIC SOSI characters to ASCII SOSI characters. (SOSI ASCII option)
- Leave EBCDIC SOSI characters unchanged. (SOSI EBCDIC option)
- Translate EBCDIC SOSI characters to ASCII space characters. (SOSI SPACE option)
- Remove EBCDIC SOSI characters. (SOSI NONE option)

For the JIS78KJ or JIS83KJ language options, IP PrintWay supports only the SOSI ASCII option. IP PrintWay always translates the EBCDIC SOSI characters to the corresponding ASCII SOSI characters.

CC|NOCc

Carriage Controls: When you request translation to ASCII, the IP PrintWay LPR determines whether or not the data set contains carriage controls from the record format of the data set. You can instead specify the type of carriage-control processing IP PrintWay is to perform for all data sets by using the options entry:

- Interpret the first character as a carriage control. (CC option)
- Do not interpret the first character as a carriage control. (NOCC option)

TOpmargin|NOTOpmargin

Page Formatting: When you request translation to ASCII, you can specify the following page-formatting options:

- Number of blank lines for a top margin. (TOPMARGIN option)
- Number of lines to print on a page. (LINECOUNT option)
- Whether or not to insert a header at the top of each page. (HEADER and NOHEADER options)

The IP PrintWay LPR uses these page-formatting options only when processing data sets that do not contain carriage-control characters.

If you specify certain filter values in the FILTER option, the IP PrintWay LPR ignores these page-formatting options. (A FILTER specifies the type of data processing to be performed by the LPD on the target system, and whether or not the IP PrintWay LPR is to process carriage-control characters or format pages.)

POstscript|NOPostscript

PostScript Processing: The IP PrintWay LPR inspects each data set to determine whether or not the data stream is a PostScript data stream (%! in the header). For all PostScript data sets, the IP PrintWay LPR verifies that all LPR options are compatible. You can specify the following LPR options related to the processing for PostScript data sets:

- You can request that the IP PrintWay LPR not inspect each data set to determine whether or not it is a PostScript data set. (NOPOSTSCRIPT option)
- You can request that IP PrintWay add the header (%!) required by some systems to recognize a PostScript data set. IP PrintWay adds this header only to data sets that do not already contain the characters. (POSTSCRIPT option)
- You can request that the IP PrintWay LPR convert non-PostScript data sets to PostScript data sets for printing with print lines parallel to the long edge of the paper. (LANDSCAPE option)

  Note: You can specify the LANDSCAPE option only for non-PostScript data sets.

BUrst|NOBurst

Banner Page: The IP PrintWay LPR can send information to the LPD for printing on a banner page (BURST option). You can specify a class, user name, and job name for printing on the banner page. (CLASS, USER, and JOB options)

When you omit the class, user name, and job name, the IP PrintWay LPR transmits the following information to the LPD:

- The name of the MVS system. (CLASS)
- The user′s name. (USER)
- The data set name. (JOB)

Parameters Processed by the LPD - You can specify the following parameters to be passed to the LPD:

- Job information (JOB option)
- Title (TITLE option)
- Name (NAME option)
- Filter codes (FILTER option)
- Width of a line (WIDTH option)
- Number of spaces to indent each line (INDENT option)

USERPORTS

User Ports: You can specify that the LPD permits the MVS port address to be outside the range allowed by RFC 1179; that is, outside the range of 721 to 731, inclusive. Some LPDs permit this, others do not. (USERPORTS option)

LPDSIZE

LPD Buffer Size: You can specify a buffer size for an LPD that has a buffer too small for large data sets that may be transmitted to it. (LPDSIZE option)

IP PrintWay retains data sets that exceed the buffer size on the JES spool, if the JCL or routing entry specifies a retention time period for failed transmissions. You can then reroute the data set to another LPD with a larger buffer size.

LINETerm

Line Termination Control: You can specify a unique line-termination control, if the target printer expects a control other than line-feed. (LINETERM option)

TIMeout

Time Out: You can specify a time out value for TCP/IP transactions, if the IP PrintWay default of 10 minutes is not suitable for your installation. (TIMEOUT option)

Note: IP PrintWay transmits each data set it accepts for transmission from the JES spool individually. To increase the probability that all a job′s data sets print together, IP PrintWay transmits the data sets in a job at the same time. However, IP PrintWay cannot prevent other programs from interleaving print requests to the printer which may cause a job′s data sets not to print all together.

You can request through an options entry that PrintWay transmit each data set within a job as soon as it is selected from the JES spool. IBM recommends that you specify this type of transmission grouping for long-running jobs that spin off IP PrintWay data sets. For example, NetSpool creates only one job for each NetSpool logical printer. This job contains all data sets created for that logical printer. The job completes only when the NetSpool program ends.

Note: You can also specify parameters that NetSpool uses for data-set allocation in the IP PrintWay options data set, instead of in the NetSpool print-characteristics data set. These parameters are used only by the NetSpool program.

8.3.1.8 ISPF Dialog for Options Entries

You can use the IP PrintWay ISPF panels to list, browse, add, copy, modify, and delete entries in the PrintWay options data set. You can also use ANFGPWFL macro instructions to add, modify, and delete options entries.

Note: Do not use VSAM editing functions to change any records in the PrintWay options data set.

Figure 82 on page 125 shows an IP PrintWay options panel.

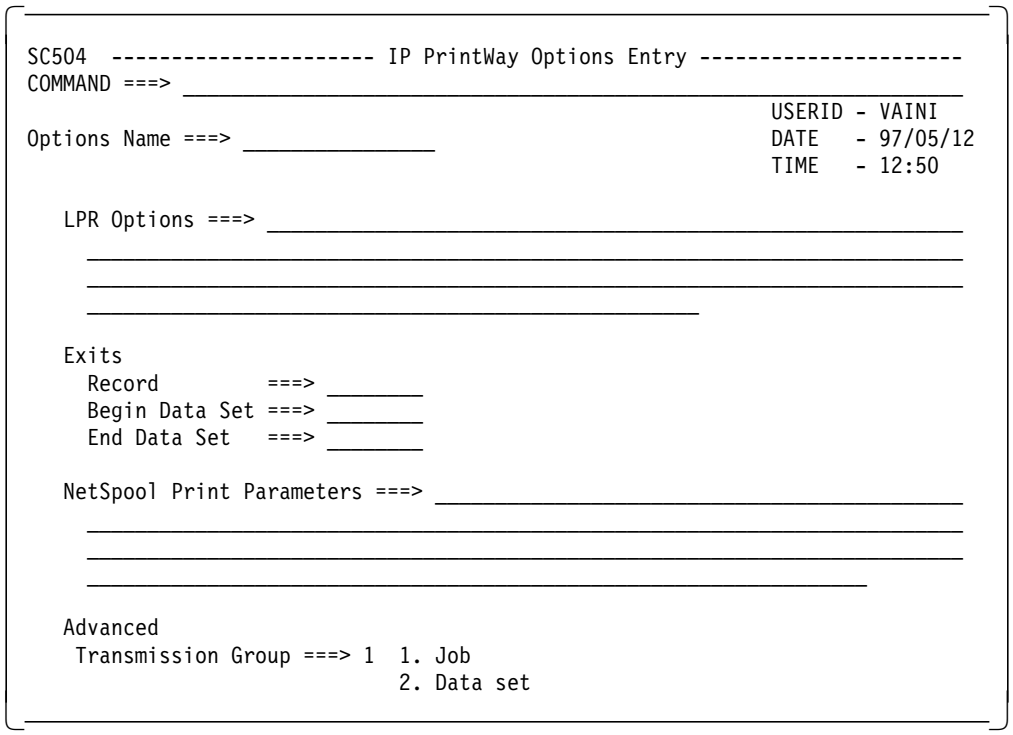


Figure 82. IP PrintWay Options List Entry Panel

The ISPF panels checks the validity of many of the values you specify in the fields of an options entry. However, to verify the LPR options, submit a data set that uses each options entry. If PrintWay finds an error, it issues a message to the PrintWay message-log data set.

The options panel entries are:

Options Name

The name of the options entry.

The options entries are referenced from:

- The Options Name field of a IP PrintWay Routing Entry panel
- The PRTOPTNS parameter of an ANFGPWFL TYPE=ROUTING macro instruction
- The PRTOPTNS parameter of an OUTPUT JCL statement

LPR Options

A list of parameters that are used by the PrintWay LPR when transmitting data sets.

This field can have up to 255 characters. Separate the LPR options by blanks.

Exits: Begin Data Set

The name of the PrintWay Begin Data Set Exit routine. PrintWay calls the exit you specify before processing any records in a data set.

Exits: End Data Set

The name of the PrintWay End Data Set Exit routine. PrintWay calls the exit you specify after processing all records in a data set.

Exits: Record

The name of the PrintWay Record Exit routine. PrintWay calls the exit you specify before processing each record in a data set.

NetSpool Print parameters

A list of parameters for use only by the NetSpool program. Refer to IBM NetSpool Guide for a list of the parameters that you might want to specify here.

PrintWay does not check these parameters for validity.

This field can have up to 255 characters. Separate the NetSpool parameters by blanks.

Transmission Group

If the Job option is selected, PrintWay selects and processes all data sets within the same print job before transmitting them. This increases the likelihood that multiple data sets of the same job print together.

If the Data Set option is selected, PrintWay transmits each data set as soon as it is ready to be transmitted.

NetSpool - Select the Data Set option if NetSpool is creating any of the data sets. This is because one NetSpool job contains all data sets created for a logical printer while NetSpool is running, until the NetSpool program terminates.

JES3 - In a JES3 environment, PrintWay ignores this option. In JES3, PrintWay can select only one data set at a time from the JES spool; therefore, PrintWay cannot group all data sets within a job together before transmission.

Figure 83 shows an option data set listing.

```
DATE 97/03/27 TIME 17:29 - IP PrintWay Options List ---------- Row 1 to 6 of 6
COMMAND ===>                                                  SCROLL ===> PAGE
Actions: B-Browse  C-Copy  D-Delete  E-Edit  R-List Routes
A Options Name     LPR Options                      Record   Begin DS End DS
  MVSBIN           binary
  MVSCC            nobinary CC
  MVSDATA          nobinary noheader lineterm 0D25
  MVSDATA1         nobinary noheader
  MVSNOCC          nobinary nocc lineterm 0D25 nohea
  POSTSCRIPT       po binary timeout 60
********************************** Bottom of data ****************************
```

Figure 83. IP PrintWay Options List

8.3.1.9 Viewing the Message-Log

IP PrintWay issues a message to the IP PrintWay message-log data set when a data set is received from JES, when it is successfully or unsuccessfully transmitted to the target system, and when it is released to JES. The data set may also contain other IP PrintWay messages, including messages written by IP PrintWay installation exits.

The messages in this data set wrap around to the beginning when the data set becomes full. The time-stamp preceding each message indicates when IP PrintWay wrote the message. IP PrintWay writes a string containing equal signs, ========, at the end of the last message written. You can search, for example in the ISPF browse or view for this string to find the latest message.

Your installation can use the IP PrintWay message exit to suppress unwanted messages from the message-log data set. The IP PrintWay message exit may also be used to redirect messages to other targets, for example to system log or consoles.

See Appendix B, “Sample ANFUXMSG Exit” on page 217 for an example of the ANFUXMSG exit for rerouting the message-log messages to system log.

8.4 JES Definitions for IP PrintWay

IP Printway executes in an FSS address space. This requires the FSS definitions to be made in the JES initialization streams. The printer devices (FSAs) that are defined to JES are not the actual printers in the TCP/IP network. You define a printer to JES that controls access to the FSS address space. Multiple printers may be defined to JES for a single FSS address space, as shown in Figure 84 on page 128. When multiple printers are defined to JES to control the same FSS, these printers must be defined in the IP PrintWay procedure, as shown in Figure 89 on page 131.

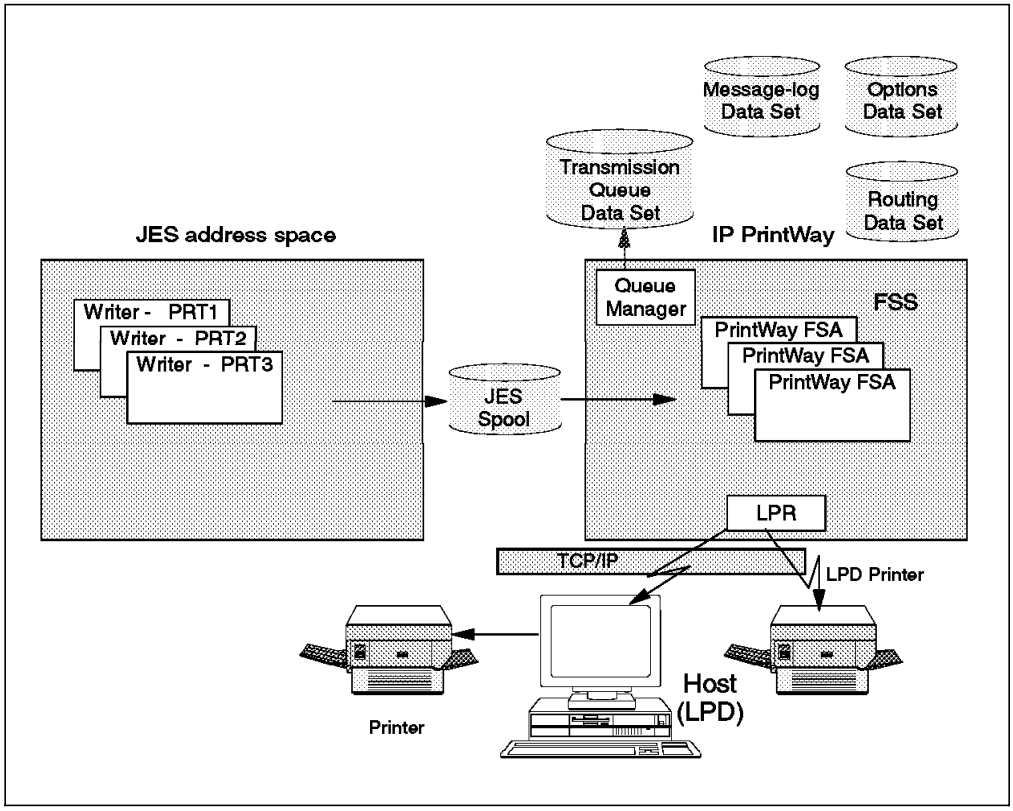


Figure 84. Printing with IP PrintWay

8.4.1 JES2 IP PrintWay Definitions

The FSS and PRTnnnn statements are used to define the IP PrintWay functional subsystem. For the three printers shown in Figure 84, the following initialization statements are required:

```
FSS(PRINTWAY) PROC=PRINTWAY
PRT1 FSS=PRINTWAY,MODE=FSS,START=NO,TRKCELL=YES,WS=(R,Q),CL=JK,
     ROUTECDE=TCPIP
PRT2 FSS=PRINTWAY,MODE=FSS,START=NO,TRKCELL=YES,WS=(R,Q),CL=JK,
     ROUTECDE=TCPIP
PRT3 FSS=PRINTWAY,MODE=FSS,START=NO,TRKCELL=YES,WS=(R,Q),CL=JK,
     ROUTECDE=TCPIP
```

Figure 85. JES2 IP PrintWay Definitions

In JES2 you can also dynamically define a functional subsystem (FSS) with the $ADD operator command:

```
$ADD  FSS(PRINTWAY),PROC=PRINTWAY
$ADD  PRT1,FSS=PRINTWAY,MODE=FSS,TRKCELL=YES,WS=(Q),Q=JK,ROUTECDE=TCPIP
$ADD  PRT2,FSS=PRINTWAY,MODE=FSS,TRKCELL=YES,WS=(Q),Q=JK,ROUTECDE=TCPIP
$ADD  PRT3,FSS=PRINTWAY,MODE=FSS,TRKCELL=YES,WS=(Q),Q=JK,ROUTECDE=TCPIP
```

8.4.2 JES3 IP PrintWay Definitions

The FSSDEF and DEVICE statements are used to define the IP PrintWay functional subsystem. For the three printers shown in Figure 84 on page 128, the following initialization statements are required:

```
FSSDEF,TYPE=WTR,FSSNAME=PRINTWAY,PNAME=PRINTWAY,TERM=YES
DEVICE,DTYPE=PRTAFP1,JNAME=PRT1,DGROUP=TCPIP,DGRPONLY=YES,
  JUNIT=(,SC50,UR,,,SC49,UR),FSSNAME=PRINTWAY,DYNAMIC=NO,
  WC=(J),WS=(CL,D)
DEVICE,DTYPE=PRTAFP1,JNAME=PRT2,DGROUP=TCPIP,DGRPONLY=YES,
  JUNIT=(,SC50,UR,,,SC49,UR),FSSNAME=PRINTWAY,DYNAMIC=NO,
  WC=(J),WS=(CL,D)
DEVICE,DTYPE=PRTAFP1,JNAME=PRT3,DGROUP=TCPIP,DGRPONLY=YES,
  JUNIT=(,SC50,UR,,,SC49,UR),FSSNAME=PRINTWAY,DYNAMIC=NO,
  WC=(J),WS=(CL,D)
```

Figure 86. Additional JES3 IP PrintWay Definitions

Note: Make sure you define **DGRPONLY=YES** for the IP PrintWay printers.

8.4.3 Scheduling Output to IP PrintWay

Output is not scheduled to a printer unless the output data set characteristics exactly match the following characteristics in an entry in the IP PrintWay routing data set:

- SYSOUT class
- Destination
- Forms

8.4.3.1 Writer Selection Parameters

There are really three options for selecting the writer selection parameters. You can schedule output:

- By SYSOUT class
- By destination selection parameters for the printer.
- By using class and destination

See Figure 85 on page 128 for JES2 definitions and Figure 86 for JES3 definitions. These figures show class and destination for the writer selection.

8.4.3.2 Scheduling by Destination Only

If you want to schedule to the writers by destination only, a SYSOUT class must still be defined in the IP PrintWay routing data set.

Note: SYSOUT class would not have to be defined as a writer selection parameter.

The destination shown in the definitions is TCPIP. The end user can specify on a SYSOUT DD statement:

```
//PRINT  DD   SYSOUT=J,DEST=TCPIP
```

The IP PrintWay routing data set must contain an entry for each SYSOUT class a user may use, as shown in Figure 87 on page 130. SYSOUT classes J and K could be used on the DD statement.

```
DATE 97/06/14 TIME 14:39 - IP PrintWay Routing List ---------- Row 1 to 9 of 9
COMMAND ===>                                                  SCROLL ===> PAGE
Actions: B-Browse  C-Copy  D-Delete  E-Edit  O-Browse Options  U-Edit Options
A DEST     C FORMS    Print Queue  Host Name/IP Address         Options Name
  DFLTNTRY
  TCPIP    J STD      afccu2       9.12.0.140                  MVSNOCC
  TCPIP    K STD      afccu2       9.12.0.140                  MVSNOCC
****************************** Bottom of data ********************************
```

Figure 87. IP PrintWay Routing List

Note: With JES2, SYSOUT classes J and K must be defined to the printer even though the writer selection parameter Q is not specified. With JES3, no classes have to be defined to the writer.

The following DD statement would not be processed by IP PrintWay because there is no routing data set entry with class T.

```
//PRINT  DD   SYSOUT=T,DEST=TCPIP
```

8.4.4 Multiple IP PrintWay FSS Address Spaces

Multiple printer FSS address spaces may be used to send output to the same TCP/IP printers. Multiple IP PrintWay address spaces require separate transmission queues, as shown in Figure 88.

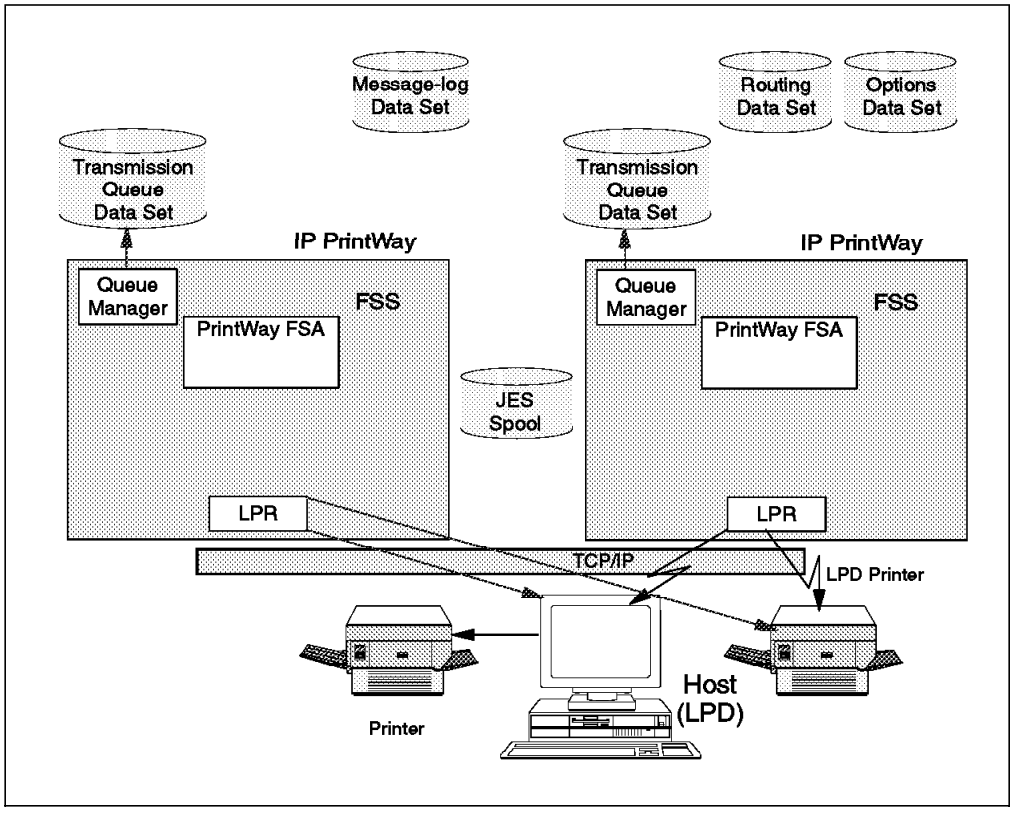


Figure 88. Multiple IP PrintWay FSS Address Spaces

8.4.5 IP PrintWay Procedure

Before JES can start an IP PrintWay FSS, you must create a cataloged procedure for the FSS. The name of the procedure is specified on the JES FSS initialization statement. The same startup procedure can be used to start more than one PrintWay FSS as long as each FSS shares the same routing, options, and transmission-queue data sets.

The procedure shown in Figure 89 has been modified from the sample shown in the ANF.SANFPROC data set ANFWPROC member.

```
//PRINTWAY PROC  HLQ=ANF,HLQU=PRINTWAY,
//             TCPHLQ=TCPIP
//* ----------------------------------------------------
//* This is a sample procedure for starting IP PrintWay
//* ----------------------------------------------------
//IEFPROC  EXEC  PGM=ANFFIEP,REGION=1M,TIME=NOLIMIT,PARM=(8000)
//*-----------------------------------------------------
//*  NOTE: The PARM=(8000) specifies the default hiperspace
//*        allocation for each FSA, which is 32M
//*        (8000-4K blocks).
//*        WARNING: This is allocated from the system
//*                 page space and may  affect system
//*                 performance if the size is too large
//*                 or many FSAs are running.
//*-----------------------------------------------------
//STEPLIB  DD DSN=&HLQ..SANFLOAD,DISP=SHR
//         DD DSN=CEE.V1R4M0.SCEERUN,DISP=SHR
//         DD DSN=&TCPHLQ..SEZALINK,DISP=SHR
//ANFROUTG DD DSN=&HLQU..ROUTING,DISP=SHR
//ANFOPTNS DD DSN=&HLQU..OPTIONS,DISP=SHR
//ANFQUEUE DD DSN=&HLQU..QUEUE,DISP=SHR
//SYSTCPD  DD DSN=SYS2.&SYSNAME..TCPIP.DATA,DISP=SHR
//ANFMMSG  DD DSN=&HLQU..MSGFILE,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//PRT1     CNTL
//PRT1     PRINTDEV TRACE=NO
//PRT1     ENDCNTL
//PRT2     CNTL
//PRT2     PRINTDEV TRACE=NO
//PRT2     ENDCNTL
//PRT3     CNTL
//PRT3     PRINTDEV TRACE=NO
//PRT3     ENDCNTL
```

Figure 89. Sample IP PrintWay Procedure

8.4.5.1 Program Properties Table for IP PrintWay

As part of MVS system configuration, you must define the program properties of IP PrintWay to the MVS system. To do so, update the MVS Program Properties Table (PPT) by adding the entry shown in Figure 12 to the SCHEDxx member of SYS1.PARMLIB.

Note: You can use the SET SCH command to dynamically modify the contents of the PPT. The SET SCH command causes the system to replace the current PPT definitions with the IBM-supplied default PPT definitions, and the PPT definitions from one or more SCHEDxx members that you specify on the command. The

new PPT definitions take effect immediately (that is, without requiring a re-IPL of the system).

```
/*                                                                  */
    PPT    PGMNAME(ANFFIEP)  /* Add program name ANFFIEP to the PPT   */
           KEY(1)            /* Protection key 1                      */
           NODSI             /* Does not require data set integrity   */
           NOPREF            /* No preferred storage required         */
           NOSWAP            /* Non-swappable                         */
           SYST              /* System task                           */
           NOPRIV            /* Not a privileged job                  */
           CANCEL            /* Cancellable                           */
           PASS              /* Requires password protection          */
           AFF(NONE)         /* No affinity to a particular processor */
```

Figure 90. Program Properties Table for IP PrintWay

The values shown for the PPT entry are the recommended values. Modification of these values may result in abends or other unexpected results.

8.4.5.2 Configuring the TCPIP.DATA Data Set for IP PrintWay

The TCPIP.DATA data set contains TCP/IP configuration statements for TCP/IP applications. IP PrintWay uses the following configuration statements in the TCPIP.DATA data set:

TCPIPJOBNAME This statement specifies the name of the TCP/IP program in your installation. For IP Printway, specify the same name that you specified on the EXEC statement of the IP PrintWay procedure.

If you do not specify a name in either the TCPIPJOBNAME statement or the EXEC statement, the name of the TCP/IP program must be TCPIP, which is the TCP/IP default name.

DATASETPREFIX This statement specifies the high-level qualifier for TCP/IP data sets. IP PrintWay uses this high-level qualifier when searching for TCP/IP translation tables.

If you not define a DATASETPREFIX statement, you can define a high-level qualifier using the EZAPPRFX installation job. If you do not specify a high-level qualifier in either the DATASETPREFIX statement or the EZAPPRFX job, the high-level qualifier of your TCP/IP translation data sets must be TCPIP.

If you have a unique TCPIP.DATA data set for use by IP PrintWay, you can identify that data set in the IP PrintWay startup procedure by using a //SYSTCPD DD statement or by naming the data set using the jobname of the IP PrintWay startup procedure as the high-level qualifier.

To find the TCPIP.DATA data set, IP PrintWay follows the standard TCP/IP search sequence, using the first value found for each statement. If an allocation fails, or if the data set either does not exist or is not available, IP PrintWay searches the next data set in the sequence.

8.5 Requesting IP PrintWay Printing

You can request output to be processed by IP PrintWay in two ways. You can use the routing data set entry, or use JCL keywords to specify the IP PrintWay options.

The two options for routing output using IP PrintWay from the JCL of the submitter are:

1. The print data set creator specifies the following JCL parameters on the SYSOUT DD statement or on a dynamic allocation request:

   - Output class
   - Output destination
   - Output form name

   These parameters must correspond to an entry in the IP PrintWay routing data set. They must exactly match an entry, and at least one parameter must be included in the IP PrintWay FSS writer work selection criteria.

   Note: This option is supported by both JES2 and JES3.

   DD statement example: You use the DEST keyword and SYSOUT class to specify a print destination which is defined in a routing entry for IP PrintWay. The FORMS information will be set by JES to the default value STD because it is not specified on the allocation request.

   ```
   //PRINT  DD   SYSOUT=J,DEST=ITSO3130
   ```

   Note: A matching routing entry, if found in the IP PrintWay routing data set (shown in Figure 81 on page 121), is used to process the print data set.

   When a match is found, IP PrintWay selects an LPD server and printer from its routing data.

2. The job submitter can specify the routing information for output data sets by specifying the JCL keywords on an OUTPUT JCL statement, shown in 8.5.1, “IP PrintWay JCL Keywords.” Any of the following parameters can be used:

   - The name of the print queue
   - The IP address
   - The name of the printer’s host system
   - The name of a set of transmission options

   In addition to the routing information, you may specify:

   - Retry parameters in the event of an unsuccessful transmission
   - The period of time to retain the data set on the JES spool after transmission

   Note: This option is supported by JES2 only.

   Note: This option is not supported by the OUTDES service or DYNAMIC allocation to dynamically define processing options for a system output (SYSOUT) data set.

8.5.1 IP PrintWay JCL Keywords

You may use the following IP PrintWay JCL keywords to specify an IP PrintWay print data set. When you use these keywords, you do not need to set up a routing entry.

```
DEST='[nodename.]IP:host'
FORMS=form_name
PRTOPTNS=options_name
PRTQUEUE='print_queue'
RETAINF='hh:mm:ss' | FOREVER
RETAINS='hh:mm:ss' | FOREVER
RETRYL=nnn
RETRYT='hh:mm:ss'
CLASS=class
```

Figure 91. IP PrintWay JCL Keywords

Note: The DEST=IP, PRTOPTNS, PRTQUEUE, RETAINF, RETAINS, RETRYL, and RETRYT parameters are ignored in a JES3 environment.

You can specify the preceding JCL parameters on OUTPUT JCL statements, as follows:

- **DEST='**[**nodename.**]**IP:host'**: Host specifies the 1- to 115-character IP address or name of the printer's host system. When you specify the IP subparameter, IP PrintWay requires that you also specify the PRTQUEUE parameter.

  When you specify the IP subparameter, the nodename is optional. IP PrintWay ignores it, but JES may use it as a work-selection criterion.
- **PRTOPTNS=options_name** specifies the 1- to 16-character name of the IP PrintWay options entry. The options_name must match the name of an options entry in the IP PrintWay options data set.
- **PRTQUEUE='print_queue'** specifies the 1- to 127-character alphanumeric name of the target print queue. When you specify the DEST=IP parameter, this parameter is required.
- **RETAINF='hh:mm:ss'** | **FOREVER** specifies the period of time for which IP PrintWay retains the data set on the JES queue after a transmission fails and all requested retries have been attempted.
- **RETAINS='hh:mm:ss'** | **FOREVER** specifies the period of time for which IP PrintWay retains the data set on the JES queue after a successful transmission.
- **RETRYL=nnn** specifies the maximum number of times that IP PrintWay is to retry an unsuccessful transmission. Specify a number from 0 to 999.
- **RETRYT='hh:mm:ss'** specifies the time interval between retries.
- **CLASS=class** specifies the SYSOUT class.

Following is an example of an OUTPUT JCL statement with the IP PrintWay keywords:

```
//PW      OUTPUT  DEST='IP:9.12.14.63',
//        PRTOPTNS=POSTSCRIPT,
//        PRTQUEUE='lpt2',
//        RETAINF='01:00:00',
//        RETAINS='0',
//        RETRYL=4,
//        RETRYT='00:10:00',
//        CLASS=J
```

Note: Specifying DEST="IP:9.12.14.63" is not allowed on a JCL DD statement.

If you specify IP host name and print queue name, but none of the remaining parameters, IP PrintWay selects a default entry in your routing file and uses the values you have specified in that entry for the remaining parameters, including the name of an options entry.

8.6 IP PrintWay Transmission Queue

The IP PrintWay transmission queue contains an entry for each data set on the JES spool being processed by the IP PrintWay FSAs within one or more FSSs. A transmission-queue entry indicates the status of the transmission of the data set and contains routing information and transmission options.

IP PrintWay initially creates a queue entry for each data set to be transmitted. IP PrintWay retains the queue entry until either IP PrintWay deletes the data set from the JES spool, or the IP PrintWay FSA processing the data set ends normally or abnormally and releases the data sets it is processing back to JES so that another IP PrintWay FSA can process them.

IP PrintWay retains queue entries and deletes queue entries automatically, based on retry and retention parameters specified either in the routing entry or in the JCL for each data set, as follows:

- If the transmission of a data set is successful, IP PrintWay retains the queue entry until the retention period specified for successful transmissions has expired and then automatically deletes the queue entry.
- If the transmission of a data set fails, IP PrintWay retains the queue entry until the retention period specified for failed transmissions has expired and then automatically deletes the queue entry.

When all IP PrintWay FSAs stop processing, the transmission-queue should be empty. If any entries do remain, delete the entries or reallocate the data set before restarting IP PrintWay. If you reallocate the data set, you must also reinitialize it.

While a data set is on the IP PrintWay transmission queue, you can use IP PrintWay ISPF panels to browse, modify, hold, reset, and delete the entry for the data set. You can, for example, reroute a data set to a different print queue, or retransmit a data set to the same print queue.

Note: While data sets are on the IP PrintWay transmission queue, you cannot use JES commands to modify or purge them.

You can manage the IP PrintWay transmission queues by using the ISPF panel interface as shown in Figure 92 on page 136.

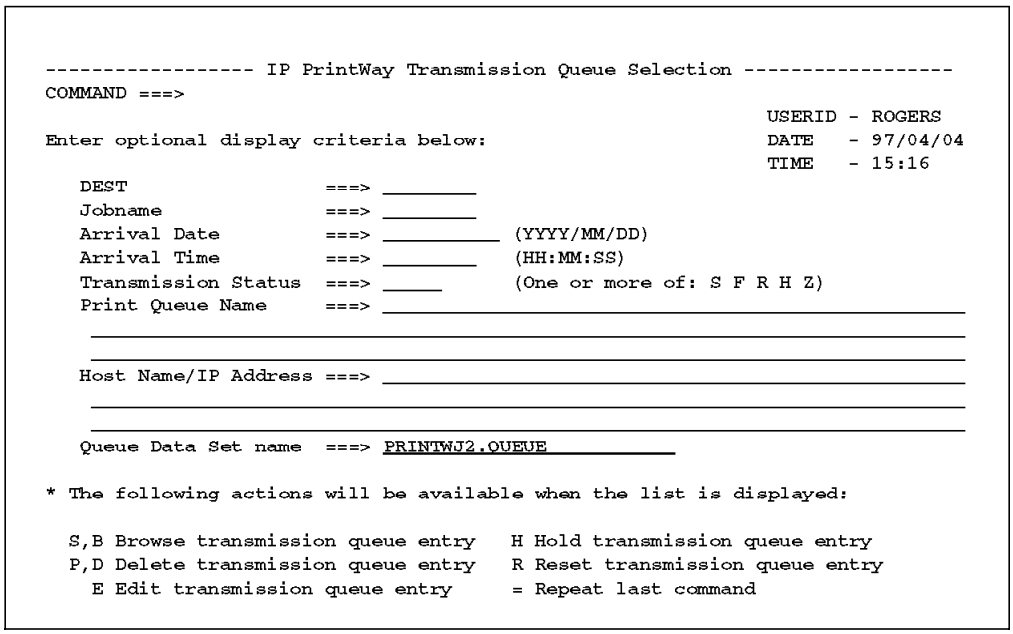


Figure 92. ISPF IP PrintWay Transmission Queue Selection Panel

You must display a list of transmission-queue entries before you can perform other functions, such as browsing or modifying an individual entry. You can either list all transmission-queue entries, or you can select the entries you want to list.

Once you have entered the selection criteria on the panel shown in Figure 92, a list of transmission queue entries will be displayed as shown in Figure 93.

```
DATE 97/03/26 TIME 16:04 - IP PrintWay Transmission Queue ---- Row 1 to 3 of 3
COMMAND ===>                                                 SCROLL ===> PAGE
Actions: B-Browse  D-Delete  E-Edit  H-Hold  R-Reset  =-repeat
A DEST     Jobname  S Arrival Date/Time   Print Queue  Host Name/IP Addr
  I3130P   ROGERSII Z 1997/03/26 16:04:53 afccu2       9.12.0.140
  ITSO3130 ROGERSBB Z 1997/03/26 16:24:53 afccu2       9.12.0.140
  ITSO3130 ROGERSBB Z 1997/03/26 16:24:54 afccu2       9.12.0.140
****************************** Bottom of data ************************
```

Figure 93. ISPF IP PrintWay Transmission Queue

On the IP PrintWay Transmission Queue panel you can take actions against the queue entries. For example, the result of a browse action against an entry on the queue is shown in Figure 94 on page 137.

```
BROWSE ----------------- IP PrintWay Transmission Queue Entry ----------------
COMMAND ===>
                                                              USERID - ROGERS
DEST                ...: I3130P                               DATE   - 97/03/26
Jobname             ...: ROGERSII                             TIME   - 17:40
Job ID              ...: JOB03633
Data Set Name       ...: WTSCPLX1.ROGERS.ROGERSII.JOB03633.D0000103.?
Transmission Status ...: Z (Queued for first attempt)


Last Activity : Date     ...: 0000/00/00  Time     ...: 00:00:00
Next Activity : Date     ...: 1997/03/26  Time     ...: 15:11:06
Retry         : Time     ...: 00:00:00    Retries  ...: 0   of 0
Retain Time   : Success  ...: 00:00:00    Failure  ...: 24:00:00

LPR Options          ...: po binary timeout 60



Print Queue Name     ...: afccu2



Host Name/IP Address ...: 9.12.0.140
```

Figure 94. ISPF IP PrintWay Transmission Queue Entry

When you browse an entry on the IP PrintWay transmission queue, you can view fields, but you cannot change any of them. If a field in an entry appears blank when you browse it, it means that IP PrintWay uses a default value for the field.

Other actions you may take against the queue entries include:

- Holding a transmission-queue entry or all transmission-queue entries in a list
- Resetting a transmission-queue entry or all queue entries in a list
- Modifying a transmission-queue entry:
  - Rerouting a data set
  - Changing lpr options
  - Changing retention periods
  - Changing the time between retries
  - Changing the retry limit
- Deleting a transmission-queue entry

For additional IP PrintWay details, see *IBM IP PrintWay Guide*.

8.7 Printing Using NetSpool

NetSpool is a feature of PSF/MVS. NetSpool creates JES spool files from print requests made by VTAM applications, as shown in Figure 95 on page 138. Once the print data sets are on the JES spool, they may be scheduled to:

- JES-controlled printers
- PSF/MVS-controlled printers
- Printers in a TCP/IP network when selected from the JES spool by IP PrintWay

Note: PSF/MVS is not required when using NetSpool.

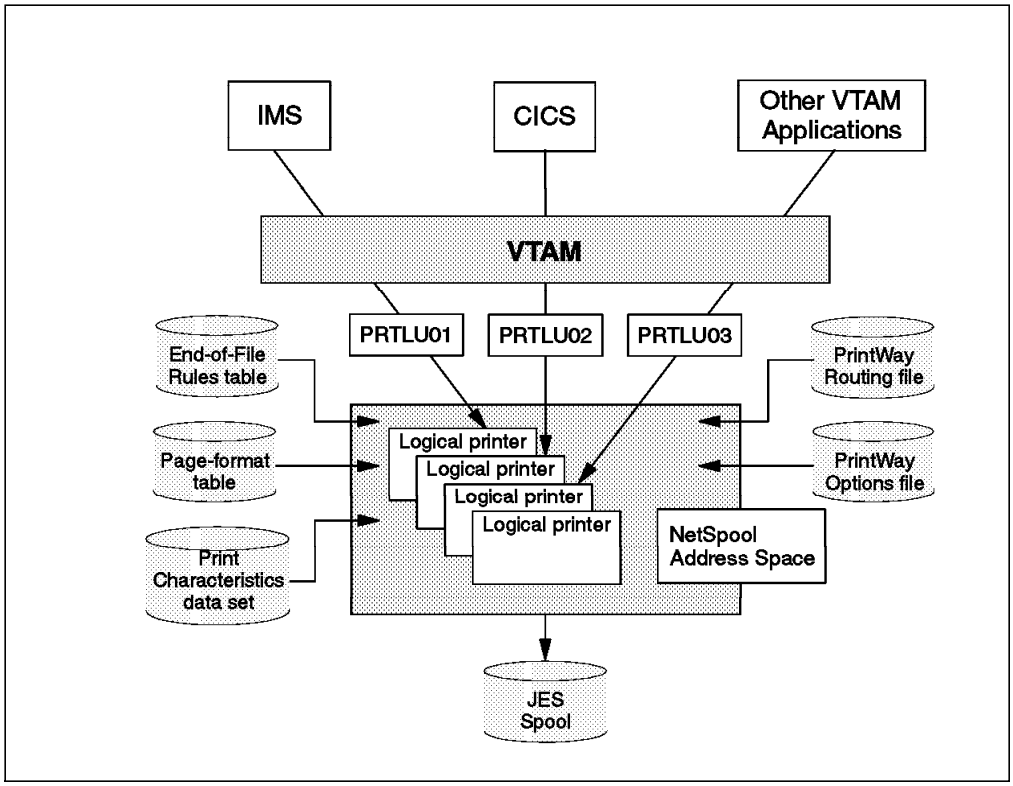


Figure 95. Overview of NetSpool Processing

8.7.1 NetSpool Logical Printers

NetSpool is a VTAM application that simulates SNA-network printers, converts data received from other VTAM applications through the simulated (logical) printers into S/390 line data, and places the data on the JES2 or JES3 spool. After NetSpool has created an output data set on the JES spool, JES, PSF/MVS, or IP PrintWay can print the data set or transmit it to another location for printing. NetSpool can be configured for your installation so that you do not need to change existing VTAM applications. That is, existing VTAM applications send print requests to NetSpool in the same manner as they currently send print requests to SNA-network printers.

8.7.1.1 NetSpool VTAM Data Stream Support

NetSpool accepts the following types of VTAM data streams:

- SNA character string (SCS) data over an LU type 1 session
- 3270 data over an LU type 3 or LU type 0 session.

NetSpool converts the SCS or 3270 data stream received as input into a series of variable-length print records, each starting with an ASA carriage-control character. The Transparent (TRN) control in SCS data streams identifies the start of a transparent data stream. NetSpool supports transparent data by including TRN controls and transparent data in the S/370 line-data output.

NetSpool provides the following two installation exits, which allow you to customize the processing of transparent data for all or selected logical printers:

- The Beginning of File exit (APIPPTD1) allows you to add transparent data to the beginning of a data set.
- The Transparent Data Control exit (APIPPTD2) allows you to inspect transparent data whenever it occurs in the data stream. Based on the transparent data, you can leave the transparent data unchanged, modify the transparent data, or delete the transparent data.

NetSpool also supports DBCS data in both SCS and 3270 data streams. In an SCS data stream, NetSpool supports the following controls, which identify DBCS strings:

- Shift Out and Shift In controls
- Set Attribute controls, with the Character Set attribute

In a 3270 data stream, NetSpool supports the following controls and orders, which identify DBCS strings:

- Shift Out and Shift In controls
- Set Attribute orders, with the Character Set attribute
- Start Field Extended orders with the Character Set attribute
- Modify Field orders with the Character Set attribute

For SCS data streams, NetSpool does not support:

- Function Management (FM) Headers: NetSpool cannot handle FM headers because it does not expect FM data requests that it receives to be formatted. Any request containing an FM header is rejected with a sense code of function error, indicating a non-supported function.
- Some SCS commands (for example, NetSpool) ignore Set Attribute code points with attributes of Color, Highlighting, or Field Outlining.

For 3270 data streams, NetSpool supports only:

- Orders in the basic function set, as described in SNA - Sessions Between Logical Units
- Shift Out and Shift In controls
- Set Attribute (SA), Set Field Extended (SFE), and Modify Field (MF) orders, with the Character Set attribute. NetSpool ignores other attributes of the SA, SFE, and MF orders.

NetSpool rejects other orders in a 3270 data stream with an SNA sense code of function error, indicating a non-supported function. Note that the Write Structured Fields order is not included in the basic function set and therefore is not supported.

Encrypted or compressed data sent to NetSpool is decrypted and decompressed as it arrives at the MVS system on which NetSpool is running. The data is no longer encrypted or compressed while it is processed by NetSpool and after it is placed on the JES spool.

For a more complete description of Netspool, see *IBM NetSpool Guide*.

Chapter 9. Dynamic Linklist Enhancements

OS/390 Version 1 Release 2 introduced enhancements to LNKLST definitions. You can use LNKLST statements in PROGxx to define a LNKLST set and activate it at IPL. The LNKLST set is a set of ordered data sets to be used as the LNKLST concatenation. Instead of specifying LNK=xx at IPL, you can specify PROG=xx to activate the LNKLST set. You can also define SYSLIB statements in PROGxx. SYSLIB allows you to define alternate data sets for the system defaults used in the LNKLST and LPALST concatenations.

OS/390 Version 1 Release 3 introduces dynamic linklist enhancements that provide for changing the LNKLST without IPL. Data sets can be added to or deleted from the LNKLST set. The support involves:

- Defining a LNKLST set that can be used as the LNKLST concatenation (REQUEST=DEFINE).
- Adding a data set to a LNKLST set in its desired position (REQUEST=ADD).
- Deleting a data set from a LNKLST set (REQUEST=DELETE).
- Removing the definition of a LNKLST set (REQUEST=UNDEFINE).
- Testing to determine if a module can be located in a LNKLST set (REQUEST=TEST).
- Obtaining a list of LNKLST sets and users (REQUEST=LIST).
- Updating jobs and address spaces to use the current LNKLST set (REQUEST=UPDATE).
- Query information about support for LNKLST services (REQUEST=QUERYDYN).
- Activating a LNKLST set - making it the “current” set through an operator command.

Defining a LNKLST set involves determining what libraries are to comprise the set. Activating a set involves opening a new DCB for this LNKLST. LLA must also update its LNKLST information when a new LNKLST is activated. The system will recognize the new DCB when it is passed through contents supervisor macro service, for example, LINK and LOAD.

9.1 Dynamic Linklist Modification

Program and operator command support is available for the “dynamic LNKLST” services. Overall, this means that data sets can be added to the LNKLST, removed from the LNKLST, data can be placed into additional extents within LNKLST data sets and that data will be successfully processed.

Dynamic linklist introduces the concept of a LNKLST set. This is a named entity consisting of an ordered set of data sets that is to comprise the LNKLST. The first three data sets in the LNKLST set are defined by the system to be the LINKLIB data set, the MIGLIB data set, and the CSSLIB data set as defined (or defaulted) by the SYSLIB statement in PROGxx. Those data sets are automatically made a part of any LNKLST set that is defined. If any of those data sets is not available at IPL-time, a wait state will result.

You can have more than one LNKLST set defined. Only one is “current” but previously current LNKLST sets can still be “active” or “in use.” The concept of an address space-related LNKLST is also introduced. When an address space or job starts, then the current LNKLST set is associated with that job. The job will continue to use its associated LNKLST set until it terminates (or until the system is told explicitly to change that association). Once a new LNKLST set is made available (that is, is made the current set), any subsequently starting job or address space will be associated with that new LNKLST set. An active, but not current, LNKLST set remains allocated until there are no longer any jobs or address spaces still associated with it. Note that this has no connection with any JOBLIB, STEPLIB, or TASKLIB associated with the job. JOBLIB, STEPLIB, and TASKLIB contribute to the module search order, but are not considered part of the address space′s associated LNKLST.

9.2 Dynamic Linklist and Library Lookaside

When a LNKLST set is activated, LLA is refreshed. Assuming LLA is managing the LNKLST, it will then manage the current LNKLST set as “the” LNKLST. There may of course still be older, “in use” LNKLST sets. LLA will continue to manage the data sets within those LNKLST sets, but only on an individual data set basis. While LLA remains active, it will manage all the data sets that it had been managing plus any new ones that are made a part of a newly activated LNKLST set. It will not drop management of a data set that it had been managing as part of the LNKLST, but that is no longer a part of the current LNKLST set. This means that address spaces running with an “old” LNKLST will still get benefit from LLA management of the data sets that comprise their LNKLST, but not as much benefit as if they were using the current LNKLST.

If LLA is stopped and then restarted, it will of course only manage the data sets that it is told to manage upon the restart - namely individual data sets that are specified, plus the data sets in the current LNKLST set (if LLA is managing the LNKLST). If upon restart you still want LLA to manage the data in data sets that are part of older LNKLST sets but not part of the current LNKLST set, then you must provide LLA with the names of those individual data sets.

9.3 SETPROG Operator Command for LNKLST

You use the SETPROG command to:

- Define a LNKLST set of data sets for the LNKLST concatenation.
- Add data sets to or delete data sets from the LNKLST set.
- Remove the definition of a LNKLST set from the system.
- Test for the location of a specific module in the LNKLST concatenation.
- Activate a LNKLST set as the LNKLST concatenation for the system.
- Update an address space for jobs to use a LNKLST set.

PROGxx parmlib member is used to define one or more LNKLST sets. You can use PROGxx to activate one of the LNKLST sets as the LNKLST concatenation at IPL. (You can also activate the LNKLST concatenation through LNKLSTxx, but IBM recommends that you use PROGxx.) SETPROG LNKLST allows you to dynamically modify the LNKLST concatenation after IPL.

Figure 96 shows the SETPROG LNKLST operator command.

```
SETPROG LNKLST,{DEFINE,NAME=lnklstname{,COPYFROM=lnklstname}{,NOCHECK}           }
               {ADD,NAME=lnklstname,DSNAME=dsname{,VOLUME=volser}{,ATBOTTOM}    }
                                                                 {,ATTOP}       }
                                                                 {,AFTER=dsname}}
               {DELETE,NAME=lnklstname,DSNAME=dsname                            }
               {UNDEFINE,NAME=lnklstname                                        }
               {TEST,NAME=lnklstname,MODNAME=name                               }
               {ACTIVATE,NAME=lnklstname                                        }
               {UPDATE,{JOB=jobname}                                            }
                       {ASID=asid  }                                            }
```

Figure 96. SETPROG LNKLST Operator Command

The operands of the SETPROG LNKLST command are:

DEFINE Specifies that you want to define a LNKLST set (a set of ordered data sets for the LNKLST concatenation).

NAME= The name lnklstname specifies the LNKLST set that you want to specify. Do not use the name CURRENT. The system uses CURRENT to mean the current LNKLST set. Also, all options except TEST, do not use the name IPL. The system uses IPL to mean LNKLST information specified in SYS1.PARMLIB member LNKLSTxx. However, you can specify SETPROG LNKLST TEST(IPL).

COPYFROM= The name lnklstname specifies the name of an existing LNKLST set from which to initialize the LNKLST set you are defining. If you specify CURRENT for the name, the system uses the current LNKLST set.

NOCHECK Indicates that the system does not check to determine if the specified LNKLST set contains SYS1.LINKLIB, SYS1.MIGLIB, and SYS1.CSSLIB before allocating the LNKLST concatenation.

Note: Use NOCHECK with caution. An appropriate use of NOCHECK is after you have modified SYS1.LINKLIB and want to compress SYS1.LINKLIB. For a procedure, see the description of the PROGxx NOCHECK parameter in *OS/390 MVS Initialization and Tuning Reference*.

ADD This operand indicates that you want to add a data set to the specified LNKLST set. You cannot add a data set to either the current or the active LNKLST set. If a data set has been migrated, the request waits until the data set is available. Note that the maximum number of extents in the LNKLST concatenation is 255.

DSNAME= The dsname specifies the 44-character name of a data set or library that you want to add to the specified LNKLST set or delete from the specified LNKLST set. DSN, LIB, and LIBRARY are accepted synonyms for this parameter.

The data set can be a PDS or a PDSE. (IBM recommends that you use PDSEs because of the limitations on the number of extents for a LNKLST concatenation.

Data sets to be added can be SMS-managed or non SMS-managed. After the system determines the volume and the SMS status of the data set, the following actions result in an error when the system tries to allocate the LNKLST set:

- If the data set in the LNKLST set changes status from SMS-managed to non SMS-managed, or from non-SMS managed to SMS-managed.
- If a non SMS-managed data set in the LNKLST set is deleted and moved to another volume.

In either case, to add the data set after the change has occurred, you must first delete the data set from the LNKLST set and add it again.

VOLUME= The operand volser specifies the name of the volume on which the data set resides. The data set must be cataloged. If the volume does not match the name in the catalog, the ADD request fails.

When the data set is cataloged in a user catalog instead of the master catalog, you can use this parameter. If a data set is cataloged in a user catalog, but not in the master catalog, you must specify the VOLSER of the volume on which the data set resides.

ATBOTTOM This operand indicates that you want to place the data set specified on the DSNAME parameter at the bottom of the list of data sets in the LNKLST set.

ATTOP This operand indicates that you want to place the data set specified on the DSNAME parameter at the beginning of the LNKLST set. The system places the LINKLIB, MIGLIB, and CSSLIB data sets in that order at the beginning of every LNKLST set in the LNKLST concatenation. If you use ATTOP, the system always places the data set after the CSSLIB data set.

AFTER= The dsname indicates that the system places the data set specified on the DSNAME parameter after the data set specified by dsname. You cannot use this parameter to place a data set after the

LINKLIB, MIGLIB, or CSSLIB data set in the LNKLST set. Instead, use ATTOP if you want to place the data set immediately after the CSSLIB data set.

Note: ATBOTTOM, ATTOP, or AFTER=dsname indicates where in the LNKLST set you want to place the data set.

DELETE Indicates that you want to delete a data set from the specified LNKLST set.

Note: You cannot delete a data set from either the current or the active LNKLST set.

UNDEFINE Removes the definition of the LNKLST set specified by NAME=lnklstname from the system. You cannot remove the definition of the current LNKLST set, another LNKLST set that is being actively used by a job or address space, or the LNKLST defined at IPL through LNKLSTxx and the LNK parameter of IEASYSxx.

TEST Indicates that you want to locate a specific routine associated with a data set in the LNKLST set. If the system locates the data set, the system indicates the name of the data set. If a data set has been migrated, the request waits until the data set is available.

MODNAME= name specifies the name of a module to be located in the LNKLST set. MODULE and MOD can be used as synonyms for MODNAME.

ACTIVATE Indicates that you want to activate the specified LNKLST set as the current LNKLST concatenation. When you use SETPROG LNKLST to activate the LNKLST set after IPL, jobs or address spaces that are still active continue to use the previous current LNKLST set. To associate a job in an address space to the current LNKLST set after IPL, see UPDATE.

If a data set in the LNKLST set has been migrated before the LNKLST set is activated, the request waits until the data set is available.

When the ACTIVATE request completes, the system issues an event (ENF) signal (event code 52). Depending on the options specified in SMFPRMxx, whenever a LNKLST set is activated, the system records SMF record type 90 subtype 29.

UPDATE Indicates that the system is to update an address space so that a specified job or jobs associated with the address space can use the current LNKLST set. If the job is using another LNKLST set when the current LNKLST set is activated, it will continue to use the original LNKLST set until it completes operations. When the job completes and restarts, the job then uses the data sets defined in the new currently active LNKLST set.

Be careful when you use UPDATE. Updating jobs in an address space while it is fetching a module can cause the fetch to fail or result in fetching a copy that is not up-to-date. The system does not attempt to verify the validity of the data for UPDATE.

JOB= The jobname specifies the name of the job or jobs specified by jobname. You can use wildcard characters (? or *) for jobname. UPDATE updates any job whose name matches the specified criteria. The system compares jobname to the name of any initiated job or jobs that match, or to the name of the address space.

ASID= The asid specifies the address space ID for the job.

9.3.1 SETPROG Command Example

Use the following commands if you want to temporarily add the MY.TEST.LIB data set into the current LNKLST set:

```
SETPROG LNKLST,DEFINE,NAME=TEST.LNKLST.SET,COPYFROM=CURRENT

SETPROG LNKLST,ADD,NAME=TEST.LNKLST.SET,DSNAME=MY.TEST.LIB

SETPROG LNKLST,ACTIVATE,NAME=TEST.LNKLST.SET
```

9.4 DISPLAY PROG,LNKLST Operator Command

Use the DISPLAY PROG,LNKLST command to display information about the LNKLST set. The command provides information about LNKLST sets for the LNKLST concatenation and associated jobs:

```
D PROG,LNKLST{,NAME={lnklstname|CURRENT}                 }
             {,NAMES                                     }
             {,USERS,{CURRENT|NOTCURRENT|NAME=lnklstname}}
             {,ASID=asid                                 }
             {,JOBNAME=jobname                           }
```

Figure 97. Example of SETPROG LNKLST Operator Command

The D PROG,LNKLST command displays information about the LNKLST concatenation and jobs associated with it. The additional operands are:

NAME= CURRENT|lnklstname - Displays the data sets for the specified LNKLST set or concatenation.

If you specify CURRENT, the system displays information for the current LNKLST set that has been activated as the LNKLST concatenation.

NAMES Displays data sets for each LNKLST set defined to the system.

USERS Display list of address spaces.

CURRENT Displays a list of address spaces that use the current LNKLST set.

CURRENT is the default. If you omit this parameter, the system displays a list of address spaces for the current LNKLST set.

NOTCURRENT Displays a list of address spaces that use any LNKLST set besides the current LNKLST set.

NAME= The lnklstname displays a list of address spaces that use the LNKLST set specified by NAME=lnklstname.

ASID= The operand asid displays the LNKLST set in use by the address space for the specified ASID.

JOBNAME= The operand jobname displays the LNKLST set in use by the specified job. The system provides information for any job that matches jobname. The jobname can include wildcard characters (* or ?).

9.4.1 Operator Command Examples

To display information for the LNKLST concatenation (defined as LNKLST1 in PROGxx and activated at IPL), enter the following command:

```
D PROG,LNKLST
```

The output appears in the following format:

```
CSV470I 16.02.30 LNKLST DISPLAY 407
LNKLST SET LNKLST1   LNKAUTH=APFTAB
ENTRY  APF  VOLUME  DSNAME
   1    A   DRV602  SYS1.LINKLIB
   2    A   DRV602  SYS1.MIGLIB
   3    A   DRV602  SYS1.CSSLIB
   4    A   *SMS*   SYS1.CMDLIB
   5    A   *SMS*   SYS1.VTAMLIB
   6    A   *SMS*   SYS1.JES3LIB
```

Another example: To display the LNKLST set associated with the job that matches the jobname NET, enter:

```
D PROG,LNKLST,JOBNAME=NET
```

The output appears in the following format:

```
CSV473I 16.00.51 LNKLST DISPLAY 967
LNKLST SET        ASID  JOBNAME
LNKLST1           0033  NET
```

9.5 CSVDYNL Macro Programming Service

You can use the CSVDYNL macro in an authorized program to change the LNKLST concatenation for associated jobs and address spaces. You can perform the following functions with CSVDYNL:

- Define a LNKLST set by name (REQUEST=DEFINE). A LNKLST set defines the data sets for the LNKLST concatenation.
- Add data sets to the LNKLST set (REQUEST=ADD) or delete data sets from the LNKLST set (REQUEST=DELETE).
- Remove the definition of a LNKLST set (REQUEST=UNDEFINE).
- Test to determine if a module can be located in a LNKLST set (REQUEST=TEST).
- Obtain a list of LNKLST sets and users in the system (REQUEST=LIST).
- Update jobs and address spaces to use the LNKLST set that the system is currently using, called the current LNKLST set (REQUEST=UPDATE).
- Query information about support for LNKLST services (REQUEST=QUERYDYN).

Note: You cannot activate a new LNKLST with the CSVDYNL macro service.

Chapter 10. OpenEdition Services

This redbook provides a brief overview of installation and customization tasks for OpenEdition Services. *OS/390 Release 3 OpenEdition Installation and Customization*, SG24-2087 provides step-by-step descriptions on how to install, customize, and use the OpenEdition product set.

The OS/390 Release 3 support for OpenEdition services enables two open systems interfaces on the OS/390 operating system:

- An application program interface (API)
- An interactive shell interface

With the APIs, programs can run in any environment - including batch jobs, jobs submitted by TSO/E interactive users, and most other started tasks - or in any other MVS application task environment. The programs can request:

- Only MVS services
- Only OpenEdition services
- Both MVS and OpenEdition services

Figure 98 shows how OpenEdition, the shell interface, and the API relate to the rest of the OS/390 operating system.

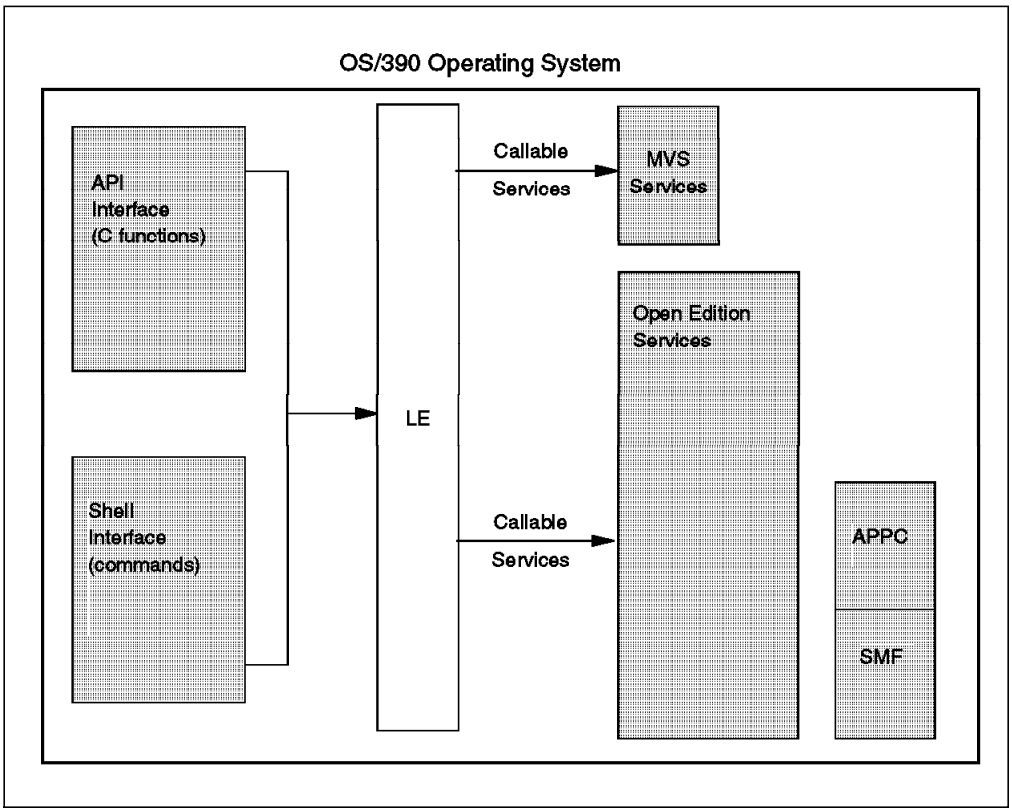


Figure 98. OS/390 Operating System with OpenEdition Services

The shell interface is an execution environment analogous to TSO/E, with a programming language of shell commands analogous to the Restructured eXtended eXecutor (REXX) language. The shell work consists of:

- Programs run interactively by shell users

- Shell commands and scripts run interactively by shell users
- Shell commands and scripts run as batch jobs

OpenEdition requires APPC/MVS support because OpenEdition processes run in address spaces controlled by APPC/MVS transaction initiators. Whether you are currently using APPC or not, you will have to customize it for OpenEdition.

System management facilities (SMF) collects data for accounting. SMF job and job-step accounting records identify processes by OpenEdition user, process, group, and session identifiers. Fields in these records also provide information on OpenEdition resources used by the process. SMF OpenEdition file system records describe file system events such as file open and file close, and file system mount, unmount, quiesce, and unquiesce.

A system programmer defines an OS/390 OpenEdition shell user by assigning the following to the user:

UID An OMVS user ID (UID), which identifies a user of OS/390 OpenEdition services.

GID A group ID (GID), which is a unique number assigned to a group of related users.

These are numeric values associated with a TSO/E user ID and are set in the RACF user profile and group profile when a user is authorized to use OS/390 OpenEdition MVS. The system uses the UID and GID to identify the files that a user owns and the processes that a user runs. As a user, you can control the read, write, and execute access to your files by other users in your group or outside of your group, by setting the permission bits associated with the files.

10.1 OpenEdition Support

OS/390 Release 3 OpenEdition services respond to requests from programs and the shell. OpenEdition services are made up of OpenEdition System Services and OpenEdition Application Services.

- OpenEdition System Services provides:
  - POSIX 1003 .1, .1a, and .1c support
  - Assembler-callable services
  - TSO/E commands to manage the file system
  - OpenMVS ISPF Shell environment
- OpenEdition Application Services interprets commands from interactive users or from programs, called *shell scripts*, and requests MVS services in response to commands. It also provides the **dbx** debugger to help an application programmer debug source programs written in C.

  OpenEdition Application Services provides:
  - A TSO/E command to enter the shell environment
  - A shell environment for developing and running applications
  - The dbx debugger
  - Support for socket applications
  - rlogin (remote login) and inetd functions
  - Direct Telnet based on TCP/IP
  - Support for full-screen applications (cursor support)
  - Communications Server login monitor support.

OpenEdition Application Services now contain the code that was provided in the optional OpenEdition Shell and Utilities and the OpenEdition Debugger features prior to OS/390.

10.2 OpenEdition Services with Licensed Programs of OS/390 Release 3

OS/390 Release 3 OpenEdition services interact with licensed programs associated with the operating system:

- OS/390 Release 3 components, including Advanced Program-to-Program Communication/MVS (APPC/MVS) and system management facilities (SMF)
- C/C++ Compiler, to compile programs
- Language Environment, to execute the shell and utilities or any other XPG4-compliant shell application
- Data Facility Storage Management Subsystem/MVS (DFSMS/MVS)
- Security Server (RACF)
- Resource Measurement Facility (RMF)
- System Display and Search Facility (SDSF)
- Time Sharing Option Extensions (TSO/E)
- Transmission Control Program/Internet Protocol (TCP/IP)
- ISPF dialogs, for OEDIT, or ISPF/PFD for the OpenMVS ISPF Shell
- BookManager READE/MVS, to use the OHELP online help facility

10.3 Installation Concepts for OS/390 Release 3 OpenEdition Services

With OS/390 Release 3, installation of OpenEdition services is simplified:

- OpenEdition System Services (FMID HOMxxxx) has been merged with the BCP, and is now part of the BCP FMID (for example, the FMID is HBB6603 for OS/390 Release 3). In addition, the OMVS address space is started automatically like traditional MVS address spaces such as Dump Services and Global Resource Serialization (GRS).
- Activation of OpenEdition services is available in two levels: *minimum mode* or *full-function mode*. There is also a sockets-only level of OpenEdition System Services.
- The OMVS address space is now started automatically at IPL by means of the OMVS= statement in the IEASYSxx parmlib member. To dynamically reconfigure OpenEdition services, you use the SETOMVS or SET OMVS command. If a full restart of OpenEdition services is required, the system must be re-IPLed.
- The START OMVS and STOP OMVS commands are no longer supported.

In OS/390 Release 2, the OpenEdition Shell and Utilities and OpenEdition Debugger (FMIDs HSUxxxx and HDXxxxx, respectively) were merged with OpenEdition Application Services and made part of FMID HOTxxxx.

10.4 Minimum Mode

If you do not specify OMVS= in the IEASYSxx parmlib member or if you specify OMVS=DEFAULT as shown for sysname SC64 in Figure 99, then OpenEdition services start up in *minimum mode* when the system is IPLed. All BPXPRMxx parmlib statements take their default values. This mode is intended for installations that do not plan to use OpenEdition services.

```
SYS1.PARMLIB(IEASYS00)

***************** Top of Data **************

OMVS=&OMVSPARM.,              OMVS STARTUP

****************** Bottom of Data ***********



SYS1.PARMLIB(IEASYM00)

****************** Top of Data **************

  SYSDEF          HWNAME(SCZP101)
                  LPARNAME(A1)
                  SYSPARM(00)
              SYSNAME(SC63)
              SYMDEF(&OMVSPARM='00')
  SYSDEF          HWNAME(SCZP101)
                  LPARNAME(A2)
                  SYSPARM(00)
              SYSNAME(SC64)
              SYMDEF(&OMVSPARM='DEFAULT')
  SYSDEF          HWNAME(SCZP101)
                  LPARNAME(A2)
                  SYSPARM(00)
              SYSNAME(SC65)
              SYMDEF(&OMVSPARM='65')

****************** Bottom of Data ***********
```

Figure 99. IEASYMxx and IEASYSxx for Openedition Services Start

In minimum mode:

- Most OpenEdition system services are available, but some functions such as FORK(), that require other customization, may not work.
- TCP/IP sockets (AF_INET) are not available.
- A temporary file system is used. It is initialized and primed with a minimum set of files and directories. Any data written to the file system is not written to DASD.

  The temporary file system is initialized with these directories and files:

  / (root directory)

/bin directory
/etc directory
/tmp directory
/dev directory
/dev/null file

- There are no executables in the temporary file system (that is, you will not find the shell and utilities). Do not attempt to install OpenEdition Application Services in the temporary file system (TFS), since no data will be written to DASD.

These are the minimum requirements so that applications can use the OpenEdition services.

In minimum mode you do not need to install or customize SMS, customize APPC, or customize the security product for working with OpenEdition services. Functions such as fork() and spawn() that require APPC customization will not work.

You do not need to define OpenEdition users to the Security Server (RACF). However, you do need to define any user who wants to run an application that uses OpenEdition APIs; for example, you would need to assign a UID to an application that wants to use C pthread functions.

10.5 Sockets-Only OpenEdition System Services

A sockets-only level of OpenEdition System Services is more than the minimum but does not need DFSMS or the shell and other UNIX utilities such as rlogin. For this mode, you need to create a BPXPRMxx member that will:

- Take the default minimum mode and use the temporary file system
- Set up the DOMAIN NAME specifications for the TCP/IP OpenEdition socket connections

Create the BPXPRMxx member as shown in Figure 100 on page 154.

```
 SYS1.PARMLIB(BPXPRM65)

****************** Top of Data ***************************************

MAXPROCSYS (256)     /* DEFAULT=200,  MIN=5,  MAX = 32767 */

MAXPROCUSER (16)     /* DEFAULT=25,   MIN=3,  MAX = 32767 */

MAXUIDS (200)        /* DEFAULT=200,  MIN=1,  MAX = 32767 */

MAXFILEPROC (256)    /* DEFAULT=64,   MIN=3,  MAX = 65535 */

MAXPTYS (256)        /* DEFAULT=256,  MIN=1,  MAX = 10000 */

CTRACE(CTIBPXZK)
FILESYSTYPE TYPE(HFS) ENTRYPOINT(GFUAINIT)
FILESYSTYPE TYPE(UDS) ENTRYPOINT(BPXTUINT)
FILESYSTYPE TYPE(INET) ENTRYPOINT(BPXTIINT)
NETWORK DOMAINNAME(AF_UNIX) DOMAINNUMBER(1) MAXSOCKETS(2000)
        TYPE(UDS)
NETWORK DOMAINNAME(AF_INET) DOMAINNUMBER(2) MAXSOCKETS(10000)
        TYPE(INET)

****************************** Bottom of Data **********************
```

Figure 100. Sample Sockets-Only BPXPRMxx

In the IEASYSxx member, the OMVS= statement must be set to point to this BPXPRMxx member.

10.6 Full-Function Mode

If you specify one or more BPXPRMxx members on the OMVS= statement in the IEASYSxx member as shown for sysname SC63 in Figure 99 on page 152, then OpenEdition services start up in *full-function mode* when the system is IPLed. To use the full function, you need to:

- Customize DFSMS to manage the Hierarchical File System (HFS). You must define a Storage Class and Storage Group, and assign volumes to the Storage Group.
- Set up BPXPRMxx parmlib definitions. Figure 101 on page 155 shows the changes we have applied to the BPXPRM00 member on our system for sysname SC63 .
- Customize APPC/MVS because OpenEdition processes run in address spaces controlled by APPC/MVS transaction initiators.
- Set up your security program.
- Set up the Hierarchical File System (HFS).
- Install the OpenEdition Application Services (the Shell, Utilities, and Debugger).
- Set up the users, defining them to the security product and setting up their user file systems. Figure 101 on page 155 shows the changes we applied to the BPXPRMxx member on our system.

```
****************** Top of Data ***************************************

  MAXFILEPROC(256)                 /* Allow at most 256 open files */

  MAXPTYS(256)                     /* Allow at most 256 pseudo-TTY
                                      sessions active at once      */


  CTRACE(CTIBPX00)                 /* Parmlib member 'CTIBPX00' will
                                      contain the initial tracing
                                      options to be used            */

  STEPLIBLIST('/system/steplib')   /* HFS file /system/steplib will
                                      contain the list of sanctioned
                                      step libraries for set-user-ID
                                      and set-group-ID executables. */

  FILESYSTYPE TYPE(HFS)
              ENTRYPOINT(GFUAINIT)

  ROOT     FILESYSTEM('OMVS.SC63.ROOTA')  /* The ROOT filesystem */
           TYPE(HFS)
           MODE(RDWR)


  FILESYSTYPE TYPE(AUTOMNT)
              ENTRYPOINT(BPXTAMD)

  FILESYSTYPE TYPE(UDS) ENTRYPOINT(BPXTUINT)
    NETWORK DOMAINNAME(AF_UNIX)
            DOMAINNUMBER(1)
            MAXSOCKETS(2000)
            TYPE(UDS)

******************************* Bottom of Data ***********************
```

Figure 101. Sample Full-Function Mode BPXPRMxx

For a more detailed explanation and examples of setup scenarios see *OS/390 Release 3 OpenEdition Installation and Customization*, SG24-2087 and *OpenEdition Planning*, SC28-1890.

10.7 SETOMVS and SET OMVS Operator Commands

You can modify aspects of the OpenEdition configuration by using the SETOMVS operator command, or you can change the BPXPRMxx parmlib members that are in effect using the SET OMVS operator command:

- SETOMVS

  This command allows you to dynamically reconfigure one or more of the OpenEdition system characteristics. A sample SETOMVS invocation is:

  ```
  SETOMVS MAXTHREADS=100,MAXPROCUSER=8
  ```

  Use of the SETOMVS command is limited. It can only be used to change a subset of the BPXPRMxx parameters. BPXPRMxx parameter values that can be changed with this command are listed in *OS/390 OpenEdition Planning*, SC28-1890.

- SET OMVS

  This command lets you specify one or more BPXPRMxx parmlib members to dynamically reconfigure the OpenEdition Services. It allows you to have multiple BPXPRMxx definitions so that you can easily reconfigure a large set of the OpenEdition system characteristics. You can keep the reconfiguration settings in a permanent location for subsequent reference or reuse. A sample SET OMVS invocation is:

  ```
  SET OMVS=(AA,BB)
  ```

  If a parameter is specified more than once, with different values, in the parmlib members, the first value specified is the value used. For example, if you specify SET OMVS=(AA,BB) where AA has a MAXPROCUSER=1 value and BB has a MAXPROCUSER=2 value, MAXPROCUSER=1 is used.

10.8 BPXOINIT

In OS/390 Release 3, BPXOINIT is the started procedure that runs the initialization process. BPXOINIT is also the jobname of the initialization process. (Prior to OS/390 Release 3, the initialization process was created via an APPC allocate and the jobname was OMVSINIT.) BPXOINIT is shipped in SYS1.PROCLIB.

The BPXOINIT address space has two categories of functions:

- It behaves as PID(1) of a typical UNIX system. This is the parent of /etc/rc, and it inherits orphaned children so that their processes get cleaned up, using normal code in the kernel.

  This task is also the parent of any MVS address space that is dubbed and not created by fork or spawn. Therefore, TSO/E OpenEdition commands, batch jobs, and so on, have a parent PID of 1.

- Certain functions that the kernel preforms need to be able to make normal kernel calls. This address space is used for these activities; for example, mmap() and user ID alias processing.

You need to set up BPXOINIT as a started procedure.

The STEPLIB DD statement is propagated from OMVS to BPXOINIT. Therefore, if there is a STEPLIB DD statement in the BPXOINIT procedure, it will not be used if a STEPLIB DD statement was specified in the OMVS procedure.

10.9 Migration Steps for OS/390 OpenEdition

If you already have OpenEdition services up and running, and you are now installing OS/390 Release 3, this is what changes:

- BPXOINIT is now a started procedure, and must be defined as such using the same attributes as the OMVS started procedure definition.
- At IPL time, the OMVS= statement must be specified in the IEASYSxx parmlib member.
- You can no longer START and STOP the OpenEdition services, therefore the S OMVS command must be removed from any COMMNDxx parmlib member.
- Some BPXxxxxx modules have been removed. If any of these modules have been defined to the PROGRAM class, you need to RDELETE the profiles.

- It is no longer recommended that the root file system should be mounted as read-only.
- There is a new FACILITY class profile for server applications.

Note: The changes listed are discussed in more detail in *OS/390 Release 3 OpenEdition Installation and Customization*, SG24-2087, and *OpenEdition Planning*, SC28-1890.

Chapter 11. APPC Enhancements in OS/390 Release 3

Advanced Program-to-Program Communication (APPC) is an implementation of the Systems Network Architecture (SNA) LU 6.2 protocol on a given system. APPC allows interconnected systems to communicate and share the processing of programs.

This chapter contains information for OS/390 APPC/MVS Version 1 Release 3 enhancements. The following summarizes the APPC changes:

- Network-qualified names support:

  If your installation has VTAM Version 4 Release 4 installed, you can enable APPC/MVS LUs to support VTAM network-qualified LU names, which reduces the effort of changing the distributed processing environment in an installation that includes several interconnected networks. Previously, if you had to merge existing networks, you had to rename any LU that did not have a unique network LU name (the 8-byte local LU name) within the networks. Renaming an LU requires changes to several sources of configuration data on several systems, which complicates the tasks required merely to define an LU on other systems.

  With APPC/MVS support of network-qualified names, renaming LUs is no longer necessary when your installation adds networks containing MVS systems. To use this support, you specify the NQN parameter on the LUADD statement to enable an APPC/MVS LU to use the entire network-qualified name for partner LUs. (Without this support, APPC/MVS uses only the network-LU-name portion of an LU name on outbound Allocate calls. In this case, the results of Allocate calls using a network-qualified partner LU name are not guaranteed to be established with the correct partner LU.)

  Once your installation changes LUADD statements to make LUs capable of using the entire LU name, the network ID makes the LU name unique. Then, your installation′s transaction programs (TPs) can use network-qualified partner LU names on Allocate calls, with guaranteed results.

- VTAM generic resource name support of APPC/MVS LUs:

  If your installation has VTAM Version 4 Release 4 installed, you can use APPC/MVS support of VTAM generic resources to:

  – Improve availability of APPC/MVS resources.

    If one LU in the generic resource group or one MVS system is brought down or fails, APPC/MVS work can continue because other group members are still available to handle requests that specify the generic resource name. Work from remote systems is less affected by the removal of any single APPC/MVS LU or MVS system. Additionally, changes in system configuration, capacity, and maintenance have less effect on APPC/MVS work.

  – Provide a single-system image for a multi-system APPC/MVS configuration.

    With generic resource names, transaction programs (TPs) from remote systems can establish conversations with APPC/MVS partner TPs on any MVS system; programmers do not need to know specific partner LU names or update TPs whenever the APPC/MVS configuration changes.

Note that APPC/MVS TPs can use generic resource names only for partner LUs, not for local LUs.

– More easily expand the APPC/MVS configuration.

  Additional APPC/MVS LUs associated with the same generic resource name can provide immediate improvement in performance and availability, with few or no required changes for APPC/MVS TPs or side information.

– Distribute work among two or more active APPC/MVS LUs on a single MVS system or in a sysplex, so that each LU is used as efficiently as possible.

  VTAM and WLM distribute session workload among members of a generic resource group, thus reducing contention for specific LUs, and improving performance of systems and TPs.

• Support to collect API trace data:

  Using the application programming interface (API) trace facility, you can collect data about APPC/MVS and CPI-C calls that an APPC/MVS TP issues. With this data, you can diagnose not only errors that occur during a specific call, but also problems with the conversation flow between the TP and its partners. This diagnostic capability is useful in both testing and production environments.

  APPC/MVS provides the following diagnostic tools to help you debug APPC problems:

  – The API trace facility (introduced in OS/390 Version 1 Release 3)

    Through the API trace facility, you can collect the following data for a particular TP running on a particular LU:

    - Parameters and values specified on calls issued for APPC/MVS and CPI-C services, and values provided on return from those calls
    - The same diagnostic information that the APPC/MVS Error_Extract service provides for calls that return non-zero return codes
    - Parameters and values specified on START and STOP requests for the API trace facility
    - The contents of FMH-5 or FMH-7 records exchanged between conversing TPs

    This API trace data is stored in a pre-allocated data set that you specify when starting the trace. You can view the data only after all tracing activity for the data set has stopped.

    To control API tracing activity, use the ATBTRACE REXX exec (in the SYS1.SBLSCLI0 data set) to start or stop tracing, or to list the status of active API traces. Through the ATBTRACE exec, you specify:

    - The type of trace request: START, STOP or LIST
    - The LU and TP combination that identifies the conversation
    - The pre-allocated data set for storing trace data

    You can invoke the ATBTRACE exec from TSO/E, through JCL for a TP or batch job, or from a high-level language program.

  – The Error_Extract callable service

Error_Extract TP conversation service returns detailed information about errors indicated by return codes. Your TP can display the detailed error information to end users, or use it as input to a debugging program.

- The TP message log

  The TP message log can be used for recovery and problem determination when an error occurs while a TP is processing. You can control the TP message log through parameters from the TP profile and from the APPC/MVS transaction scheduler parmlib member ASCHPMxx.

- System CTRACE and IPCS processing

  Using CTRACE, the application programmer has little control over when and where to cut CTRACE records. Also, CTRACE may contain trace entries from various APPC/MVS applications and may not capture all the parameters passed on APPC/MVS APIs. These make CTRACE difficult to use by APPC/MVS application programmers to identify errors in their APPC/MVS APIs.

- APPC/MVS protected conversations

  APPC/MVS and Resource Recovery Services/MVS (RRS/MVS) OS/390 Version 1 Release 3 introduce a new capability to MVS/ESA by providing system resource recovery services for APPC/MVS conversations. Some knowledge of resource recovery and its role is recommended to fully appreciate the value of APPC/MVS protected conversations. See Chapter 13, “Resource Recovery” on page 167 for background information.

  APPC/MVS protected conversations implements a communication resource manager on the MVS platform and thus enables the MVS platform to be a host for APPC conversations using the two-phase commit protocols.

  From a cooperative application program′s point of view, APPC/MVS protected conversations provide the capability for a cooperative application to have the updates in the cooperative application coordinated. For instance, each transaction program (TP) in the cooperative application can interact with database services and then request a coordinated “commit” or “backout” of resources that have been changed since the last point of consistency. RRS/MVS provides the capability for a cooperative application to invoke the “commit” and “backout” callable services.

  The role of APPC/MVS protected conversations in the two-phase commit protocol used to coordinate the update of local and remote resources is to communicate the progress of the two-phase commit process that keeps the resources of a cooperative application in a consistent state to the distributed resources managers. For a detailed discussion of the two-phase commit protocol used by RRS/MVS and APPC/MVS protected conversations, see *System Network Architecture, LU 6.2 Reference: Peer Protocols* and *Systems Network Architecture Sync Point Services Architecture Reference*.

  You request APPC/MVS protected conversations through a VTAM APPL definition statement for the APPC LU. The SYNCLVL keyword SYNCPT value must be specified for protected conversations. The ATNLOSS keyword must have the value ALL. ATNLOSS specifies that the APPC/MVS-supplied VTAM ATTN exit should be scheduled for all session deactivations.

  You must also define to the system logger a log stream for APPC/MVS use. APPC/MVS writes the results of a log name exchange with a partner LU into this log.

For more details, see *OS/390 MVS Planning: APPC/MVS Management* and *OS/390 MVS Programming: Writing Transaction Programs for APPC/MVS*.

Chapter 12. Security Server Support

The OS/390 Security Server is an integrated optional feature which includes RACF and the OpenEdition DCE Security Server at the OSF DCE 1.1 level. Now, non-DCE applications can use DCE security through the Generic Security Services API. This chapter describes the enhancements to the Security Server in OS/390 Release 3.

12.1 RACF Remote Sharing Facility Enhancements

The RACF Remote Sharing Facility (RRSF), introduced in RACF 2.2, provides the capability to automatically maintain RACF databases on separate OS/390 images. Enhancements to RRSF, available with OS/390 Security Server for OS/390 Release 3, provide the capability for application updates (via the RACROUTE or ICHEINTY macros) to be automatically directed, in addition to command updates and password changes.

When activated, automatic direction of application updates propagate RACROUTE REQUEST=DEFINE and RACROUTE REQUEST=EXTRACT, TYPE=REPLACE requests. If an application invokes ICHEINTY, then ADD, ALTER, DELETE, DELETEA, and RENAME requests are propagated. The synchronization of application updates builds upon the facilities provided by automatic direction of commands. It uses the same mechanisms for activation/deactivation, control and notification. Propagated updates run asynchronously, with the results being returned to a specific user.

The RACF RSF extends functionality beyond the single-host and shared DASD environments while enhancing system performance and availability. The addition of automatic direction of application updates builds upon that functionality, in addition to increasing the usability of the facility.

12.2 TARGET Command Enhancement

Support for multisystem nodes in an RRSF network was introduced via APAR OW13567. This function allows multiple system images sharing a RACF database to be configured as a single remote sharing node. The TARGET command is used to implement RRSF multisystem nodes. With OS/390 Security Server for OS/390 Release 3, the TARGET command is enhanced so that the number of commands an operator has to issue decreases. This support simplifies the management and maintenance of RRSF multisystem nodes.

12.3 Exit For Commands

A new RACF general purpose command exit, IRREVX01, is available with OS/390 Security Server for OS/390 Release 3. When the exit is installed, it receives control for most RACF TSO commands, regardless of where they are issued. The flexibility of the command exit offers customers the ability to tailor RACF commands to better suit their needs, that is, enforcing installation-specific naming conventions. The exit may be activated or refreshed dynamically; an IPL is not required.

The new command exit is invoked after the command has been parsed, before it executes, and again after the command has been executed for all RACF TSO commands except BLKUPD, RVARY, and RACLINK. If the command cannot be parsed successfully, and prompt mode is not in effect, the exit is not called. The exit does not receive control for true RACF operator commands such as RESTART, TARGET, and SIGNOFF.

12.4 Addition of User IDs to the Access List

The automatic addition of a RACF profile creator's user ID to the profile's access list can now be prevented. With OS/390 Security Server for OS/390 Release 3, it is possible for access lists for newly created profiles to be fully determined by profile modeling. This feature is very valuable for customer installations where administrators sometimes define many profiles they do not or should not require access to. Usability is enhanced when creating profiles because it will no longer be necessary to issue an extra command to remove the creator's user ID from the access list. In addition, possible security exposures can be avoided by eliminating automatic access to sensitive data.

A new global option, ADDCREATOR/NOADDCREATOR, has been added to the SETROPTS command. Exceptions to profile modeling are discrete DATASET and TAPEVOL profiles created through RACROUTE REQUEST=DEFINE. These will continue to operate as before so that access-related problems, where immediate UPDATE access is required, can be avoided. For systems that share the same RACF database, the command most recently issued (NOADDCREATOR or ADDCREATOR) will be in effect. For existing RACF databases, the default is set to ADDCREATOR for migration purposes. The RACF panels are updated to support this enhancement.

12.5 Enhanced OpenEdition MVS Support

Users of OS/390 Release 3 Security Server can have the option to cache additional OpenEdition MVS security information in the Virtual Lookaside Facility (VLF). This capability allows RACF to avoid accessing the RACF database when called to create a user security packed (USP) context for MVS OpenEdition users.

A new session type has been added to the System Authorization Facility (SAF) RACROUTE macro instruction, which authorized callers can specify when creating a RACF security context for a user. Specification of this new session type instructs RACF to update the user's statistical information contained in the RACF database on only a daily basis during the creation of a RACF security context. Daily statistical updates are only enabled when a resource manager has specified this new keyword on a RACROUTE REQUEST=VERIFY, ENVIR=CREATE macro instruction.

12.6 Support for Permanent Kernel

The OpenEdition kernel is a permanent component of the BCP with OS/390 Release 3. To enable this change, OS/390 Security Server for OS/390 Release 3 provides a mechanism for the RACF initUSP callable service to create a default OpenEdition security environment for the kernel. The initUSP parameter list is updated to handle situations; that is, if the user ID is not defined to RACF, which

could otherwise prevent the OpenEdition kernel from initializing. This supports the traditional UNIX-based application environment where the kernel is expected to always be operational.

12.7 DCE Support

With the DCE support previously implemented in RACF, a new profile class was created which is used to ″map″ a DCE identity to a RACF identity. Changes to the DCE identity information contained in RACF user profiles now automatically are reflected in the corresponding profile in the RACF DCEUUIDS class. This function relieves the RACF administrator from having to perform an additional error-prone step in profile maintenance when migrating users or cross-linking DCE principals on OS/390. The usability of the RACF ISPF panels is improved when manually managing the DCE data contained in RACF.

12.8 Enhanced Support For SOMobjects

RACF authentication and verification support for SOMobjects was introduced with APAR OW15720. That support has been enhanced with OS/390 Security Server for OS/390 Release 3 so that, in addition to generic profiles, grouping profiles are now allowed. This enhancement permits the grouping of dissimilar resource names, thus simplifying RACF administration of these resources.

12.9 RACF Administration Enhancement

With OS/390 Release 3 Security Server it is possible to administer the data stored in the TCOMMAND field of the TSO segment directly, without requiring any modifications to a user′s logon panel. This enhances RACF usability because administrators now have the ability to dynamically define TCOMMAND data using RACF commands or panels.

Chapter 13. Resource Recovery

A hardware or software failure, human errors, or a catastrophe may corrupt critical computer resources by leaving the data in the resources in an unpredictable state. When incomplete changes to the data in the resources corrupt the resources, the computing system(s) involved must be able to restore the data to a known and predictable state. These critical resources are called *protected resources* or, sometimes, recoverable resources. A protected resource is a resource for which permanent updates are made only in a synchronized and controlled manner, a logical unit of work at a time, through the support of sync-point services.

Resource recovery consists of the protocols and programming interfaces that allow an application program to make consistent changes to multiple protected resources managed by multiple resource managers.

OS/390 Version 1 Release 3 includes the services that, when requested, can coordinate changes to one or more protected resources. These resources may be owned by different resource managers and may reside on different systems. The resource recovery (synchronization) services ensure that all changes are made or none of them are made. Resources that can be protected include:

- Hierarchical databases
- Relational databases
- File systems
- Product-specific resources

The OS/390 resource recovery allows application development choice based on environment, not on restrictions. These application development choices can benefit:

- OS/390 distributed systems
- Heterogeneous distributed systems
- Distributed applications and resource managers
- Object oriented transaction processing
- Non-IBM resource managers for all OS/390 environments

The following describes resource recovery for resources on a single system and for resources distributed across multiple systems.

13.1 Resource Recovery Participants

To understand how to use the OS/390 resource recovery services, you need to understand both the programs that deal with critical resources and how they work together:

Application program The application program accesses protected resources and requests changes to the resources. Work managers, including CICS Transaction Server and IMS Transaction Monitor, typically initiate the execution of an application program.

Resource manager A resource manager controls and manages access to a resource. It is an authorized program that provides an application programming interface (API) that allows the application program to read and change a protected resource. The resource manager, through exit routines that get control in response to resource recovery services events, takes actions that commit or back out changes to a resource it manages.

Often an application changes more than one protected resource, so that more than one resource manager is involved.

13.1.1 Resource Managers

There are three types of resource managers: data resource managers, communications resource managers, and work managers.

Data A data resource manager is one that allows the application to read and change data. Data resource managers include database managers, such as DB2, and record file managers, such as VSAM. To process a sync-point event, a data resource manager would take actions such as committing or backing out changes to the data it manages.

Communications A communications resource manager provides access to distributed resources by allowing an application to communicate with other applications and resource managers, possibly located on different systems. It acts as an extension to the sync-point manager by allowing the local sync-point manager to communicate with other sync-point managers as needed to ensure coordination of the distributed resources the application accesses. Communications resource managers include APPC/MVS and Transactional Remote Procedure Calls (TRPC). To process a sync-point event, a communication resource manager communicates the event to the distributed sync-point managers.

Work A work manager is a resource manager that controls application access to system resources by determining when and in what environment the application runs. Work managers include CICS Transaction Server and IMS Transaction Monitor. To process a sync-point event, a work manager might ensure that the application is in the correct environment to allow the sync-point processing to continue.

Note: The resource manager in resource recovery is different from an OS/390 recovery termination management (RTM) resource manager, which is related to the operating system′s recovery termination management and runs during a task or an address space termination processing.

13.2 Resource Recovery Services

Resource recovery services (RRS) is the sync-point manager. It uses a two-phase commit protocol to coordinate changes to protected resources, so that all changes are made or no changes are made. During its processing, RRS drives resource manager exit routines. For example, if a commit event occurs (for example, when an application requests that changes to several resources be made), RRS drives the commit exit routine for each resource manager involved. The resource managers indicate whether or not they can make the changes, by voting YES or NO. If all resource managers vote YES, RRS notifies the resource managers involved with the unit of recovery to commit the changes, that is, to make the changes permanent in the databases. If any resource manager votes NO, RRS notifies the resource managers to back out all changes. OS/390 RRS provides for distributed sync-point management through communication resource managers, for example APPC.

Resource mangers invoke recoverable resource management services using (SAA CPI-RR) callable services. Recoverable resource management services communicate with resource managers through exits.

OS/390 provides two exit managers: resource recovery services (RRS) and context services. In general, an exit manager is an authorized program that controls the flow of a predefined set of events. When a predefined event occurs, the exit manager gives control to an exit routine owned by a program interested in the event. In this exit routine, the program provides the processing for the event. Note that the exit manager is responsible for detecting the event and to give control to the exit routine(s).

13.2.1 Additional Recovery Services

These services also play key roles in resource recovery:

Registration Registration services coordinates communication between the resource manager and the exit managers. A resource manager must register itself with the system as a resource manager. The resource manager must also set its exit routines with each exit manager. Registration services is itself an exit manager, though it can drive only one exit routine.

Context Context services allows a resource manager to indicate interest in a work context. A context represents the resources for a work request; a context defines the scope of protected resource allocation for which resource managers provide services. A context is usually used by a work manager, which is a resource manager, like IMS, that accepts and manages work, such as transactions, from outside the system.

Context services is an exit manager that provides the data constructs and primitives that resource managers can use as an anchor for a given work request to track specific events related to the work request. For example, when a given context ends, context services drives the end-context exit routines for each resource manager involved.

13.2.2 Additional Recovery Services

Registration services, context services, and resource recovery services (RRS) are implemented as three separate OS/390 components, but it is sometimes useful to think of them as a single function called recoverable resource management services (RRMS), the OS/390 sync-point manager:

Sync-point manager (SPM) The component that implements two-phase commit and resynchronization processing. The subcomponents of the SPM are sync-point services (SPS) and the protected resource managers (the distributed protected resource managers and the local protected resource managers).

Sync-point services (SPS) The component of the sync-point manager that is responsible for coordinating multiple protected-resource managers during sync-point processing. SPS coordinates two-phase commit protocols, resync protocols, and logging.

Two-phase commit protocol The protocol that permits updates to one or more protected resources to be committed or backed out atomically. In the first phase, the initiator sends Prepare, requesting that all agents vote by responding Request Commit or Backout, indicating whether the logical unit of work should be committed or backed out. All agents must vote to commit if the transaction is to be committed. When all the votes are collected, the second phase begins. In this phase, the initiator informs the agents to commit or back out. At various times, the sync-point participants write state information to nonvolatile storage so that the protected resources can be resynchronized if any failures occur during the two-phase commit processing.

Initiator The role of the sync-point services component that begins the coordinated updating of distributed resources.

Agent The role of any participant in a sync-point operation that receives sync-point requests from an initiator. A participant in a sync-point operation can be a sync-point manager or a local sync-point resource.

13.3 Recoverable Resource Management Services

RRMS callable services allow a resource manager to associate itself with a work request. Once the association is established, RRMS services help the resource manager protect the resources it manages.

RRMS provides a two-phase protocol to direct the commit and backout actions by the resource managers.

RRMS can enable resource recovery on a single system or, with a communications manager such as APPC/MVS, on multiple systems. If the resources used by an application program are distributed so that they are on multiple systems, a communications resource manager on one system will coordinate the changes. Each communications resource manager works with the sync-point manager on its system, such as RRS on OS/390.

RRMS, the combination of RRS, context services, and registration services, provides a system programming interface (SPI) that enables a resource manager:

- To register with the system as a resource manager
- To express interest in work requests that access its resources
- To take part in resource recovery for those work requests

Note: Because RRS provides most sync-point processing, technical information (like this hereafter) uses RRS in place of RRMS.

Resource recovery based on the two-phase commit protocol has two functions:

Commit During the commit process, all changes to both local and distributed resources are made permanent.

Backout During the backout process, all pending changes to both local and distributed resources are not made.

In resource recovery, a *unit of recovery* (UR) is the set of changes that are to be made or not to be made. A UR represents an application program′s changes to protected resources since the last commit or backout or, for the first UR, since the beginning of the application. Each UR is associated with a *context*, which can be either of the following:

Native A *native context*, which is the automatically occurring context of the application program and protected resources associated with a work request. A native context is associated with a single application task and owned by MVS. This context always exists as long as the task exists.

Privately-managed A *privately-managed context*, which a resource manager creates when the resource manager that is processing transactions creates a new work request. The resource manager can associate the context with an application′s task that will run for the work request.

The work resource manager owns a privately-managed context it creates, and the resource manager can switch a privately-managed context from one task to another. A privately-managed context is usually used by a work manager, which is a resource manager that accepts and manages work, such as transactions from outside the system. A privately-managed context is created explicitly by a resource manager and ends when the resource manager explicitly ends it or if the owing resource manager ends.

If needed, a work resource manager can disassociate the privately-managed context from a task and later reassociate it with the same task or another task. By changing the associations, the resource manager can have one task that runs for many work requests, many tasks or SRBs that run in series for a single work

request, or both. Note that a context cannot be associated at the same time with multiple tasks.

When a task changes from processing for one work request to processing for another work request, the resource manager should switch the privately-managed contexts associated with the task.

13.3.1 Registration Services

Registration services allows a resource manager to define itself to RRS and context services. As part of registration, a resource manager identifies its exit routines, which enables it to receive notifications of context-related events and participate in two-phase commit.

A resource manager must register every time it is started, regardless of whether the start is caused by failure of the system or the resource manager itself.

Registration is the same whether starting for the first time, restarting after a normal shut down, or restarting after a failure. The sequence is:

1. Call the Register_Resource_Manager service.
2. Call the Set_Exit_Information service one or more times to identify all of the resource manager's exit routines.

In the call to the Register_Resource_Manager service, the resource manager identifies itself with a unique name.

During registration, the resource manager can provide global data. When an exit manager invokes an exit routine, it passes this global data to the routine. The global data should provide the exit routine with an anchor or anchors to the resource manager's data structures.

A registered resource manager becomes unregistered when:

- The resource manager explicitly unregisters itself by a call to the Unregister_Resource_Manager service.
- The resource manager ends. Registration services implicitly unregisters a resource manager at this time.

13.3.2 Context Services

A resource manager expresses interest in a context to cause the system to invoke the resource manager's exit routine when:

- The context ends.
- The context switches from one application's task to another.

When expressing interest in a context, a resource manager can provide context interest data. This data can contain an anchor for the resource manager's data structures for the context. The system also provides a context token that represents the context.

Every task in the system has an associated current context; thus, there is always a context for a given task. When a task is created, context services provides the original (native) context for the task. A call to the Begin_Context service creates a privately-managed context, and a call to the Switch_Context service changes the current context to the privately-managed context. The native context still

exists, but is not current. If a later call to the Switch_Context service disassociates the privately-managed context, the native context again becomes the current context.

13.3.2.1 Context Services Exit Routines

Your resource manager can provide exit routines to be invoked when events occur for its interest in a context. These exit routines are optional.

Note that before context services can drive its exit routines, the resource manager must:

- Register itself through a call to the Register_Resource-Manager service.
- Set the context services exit routines through one or more calls to the Set Exit Information service.

A resource manager can have an exit routine for each context services exit or a single routine for all context services exits.

13.3.2.2 Two-Phase Commit Protocol

When an unit of recovery is ready to commit or back out its changes, the application (or a work manager on behalf of a transaction) invokes RRS to begin the two-phase commit protocol.

The two-phase commit protocol is a set of actions used to make sure that a unit of recovery makes all changes to a collection of resources, or makes no changes to the collection. The protocol verifies the all-or-nothing changes even if the application program, the system, RRS, or a resource manager fails.

The phases of the protocol are:

Phase 1 In the first phase, each resource manager prepares to commit the changes. A resource manager typically prepares by writing the unchanged data image, often called *undo data*, and the changed data image, often called *redo data*, in a resource manager log that it can access during restart.

If the resource manager can then commit the changes, it tells RRS that it agrees to allow the commit to continue. If the resource manager cannot commit the changes, it tells RRS to back them out.

The decision to commit or back out the changes depends on responses from all of the resource managers. If the decision is to commit the changes, RRS hardens the decision, meaning that it stores the decision in an RRS log.

Phase 2 Phase 2 begins the instant the decision is hardened. In this phase, the resource managers commit or back out the changes. Once a commit decision is hardened, the application changes are considered to be committed. If the application, the system, RRS, or a resource manager fails after the decision is hardened, the application changes are made during restart. Before the decision is hardened, a failure of the application, the system, RRS, or a resource manager would cause the changes to be backed out during restart.

13.3.2.3 Commit Example - Local Resource Managers

For a look at the commit function, think of a person who requests an automated teller machine (ATM) to transfer money from a savings account to a checking account. The application program receives the person's input from the ATM. Each account is in a different database. Each database has its own resource manager. The sync-point manager is RRS. Figure 102 shows how the ATM application, resource managers, and RRS work together

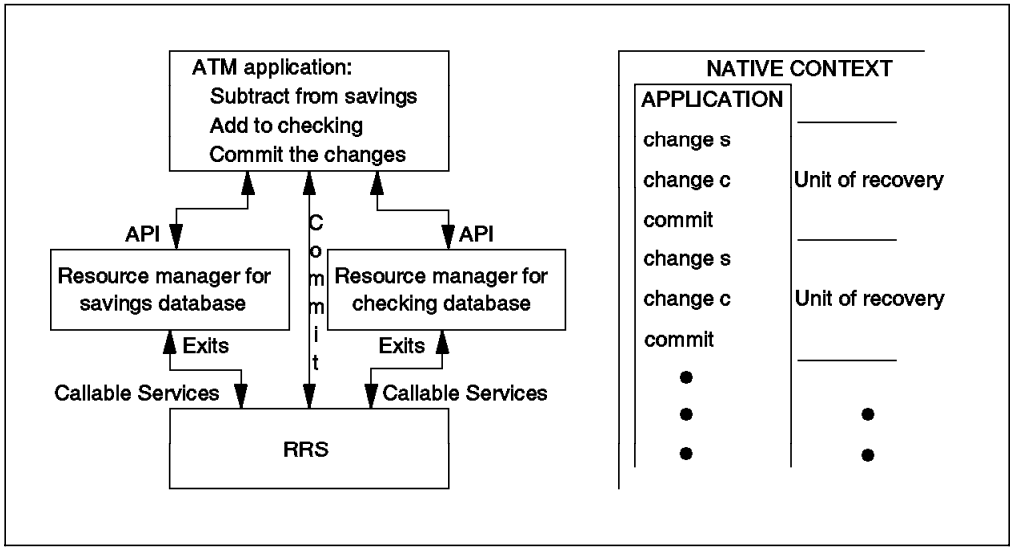


Figure 102. ATM Transaction and Resource Recovery

The actions required to process the ATM transaction are:

1. The ATM user requests transfer of money from a savings account to a checking account.
2. The ATM application program receives the ATM input.

   Figure 103 on page 175 shows, for the same ATM transaction, the sequence of the following actions, with time moving from left to right, in the two-phase commit protocol RRS uses to commit the changes. The top lines in the figure show the state of the UR and the two phases of the two-phase commit protocol. During processing, a UR assumes different states. In each UR state, a resource manager can issue only certain callable services.
3. The ATM application requests the savings resource manager to subtract the money from the savings database. For this step, the application uses the resource manager's application programming interface (API).
4. The ATM application requests the checking resource manager to add the money to the checking database. The application uses this resource manager's API.
5. The ATM application issues a call to RRS to commit the database changes.
6. RRS asks the resource managers to prepare for the changes.
7. The resource managers indicate whether or not they can make the changes, by voting YES or NO. In Figure 103 on page 175, both resource managers vote YES.
8. In response, RRS notifies the resource managers to commit the changes, that is, to make the changes permanent in the databases.
9. The resource managers complete the commit and return OK to RRS.

10. RRS gives a return code to the application program, indicating that all changes were made in the databases.

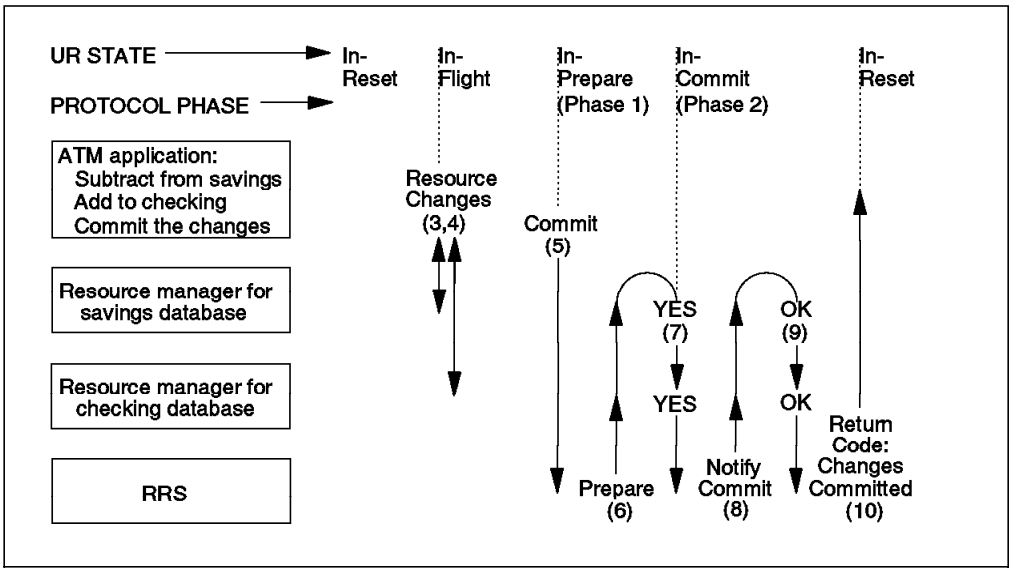


Figure 103. Two-Phase Commit Actions

13.3.2.4 Backout Examples - Local Resource Managers

If, for any reason, the ATM application cannot complete the transfer, the application requests backout in step 5, instead of commit. In this case, the changes are backed out and are not actually made in any database. See Figure 104.

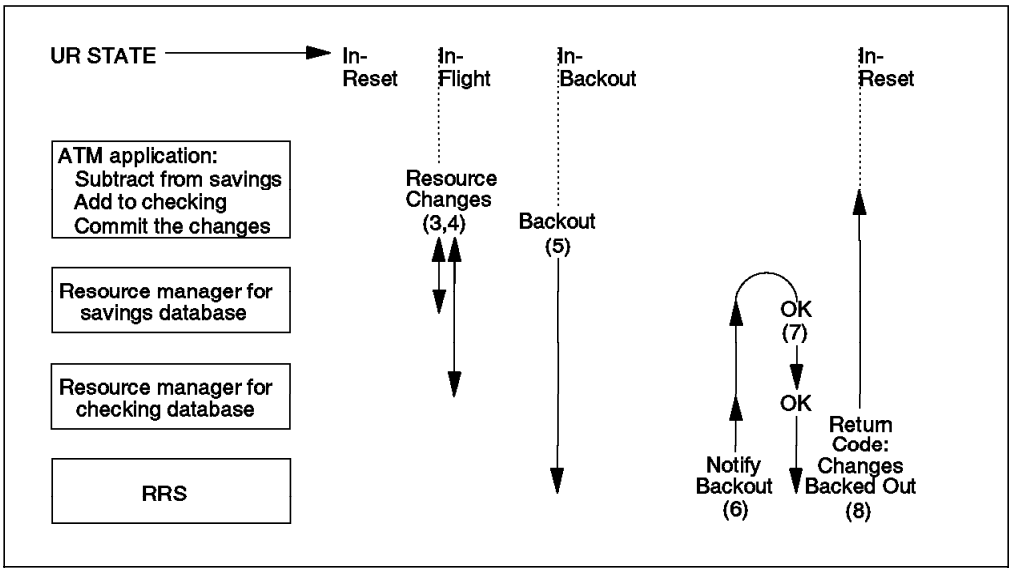


Figure 104. Backout - Application Request

A resource manager can also request backout. If a resource manager cannot make the change to its database, the resource manager votes NO during prepare. If any resource manager votes NO, all of the changes are backed out. See Figure 105 on page 176.

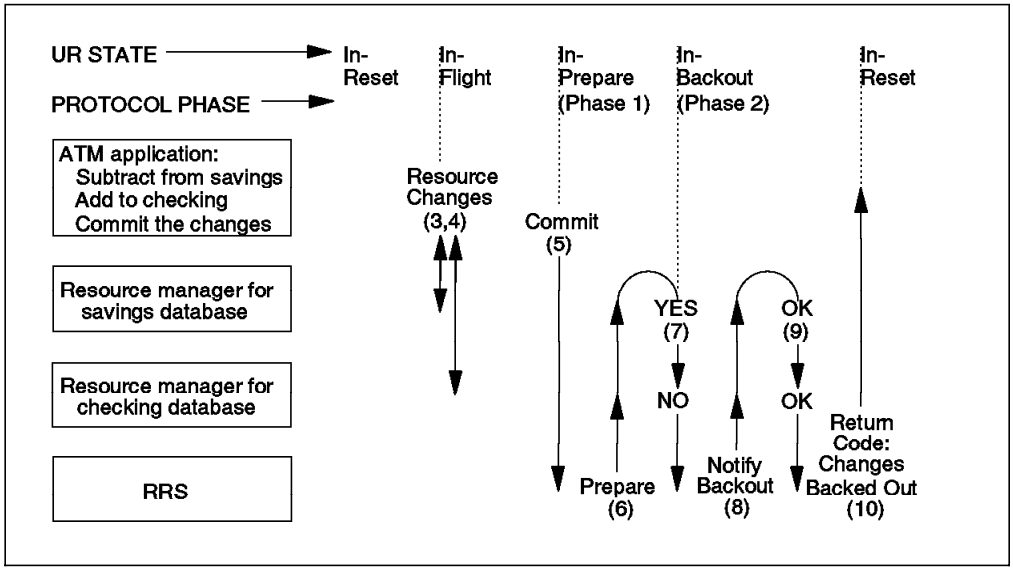


Figure 105. Backout - Resource Manager Votes NO

13.4 Distributed Resource Recovery

The resources that a work request updates can be distributed (reside on more than one system). Figure 106 on page 177 shows the programs that participate in distributed resource recovery, also called distributed sync-point processing.

When resources reside on multiple systems, a part of the application must run on each system. Using the ATM example presented earlier, assume that the resource manager for the savings account database runs on system A, and the resource manager for the checking account database runs on system B. You know from the earlier example that the ATM application has three main responsibilities:

1. Communicate with the ATM user
2. Update the savings account database using the resource manager on system A
3. Update the checking account database using the resource manager on system B

Assume that the part of the application running on system A (APPL-A) always communicates with the ATM user.

When the ATM user requests the transfer of money from savings to checking, APPL-A sees the request, and uses the resource manager on system A to update the savings account database. Completing the transaction means that APPL-A must communicate with the resource manager on system B to update the checking account database. Thus, part of the application (APPL-B) must reside on system B.

APPL-B listens for a signal from APPL-A, and the signal tells APPL-B to use the resource manager on system B to update the checking account database. Another way to implement the application would be for the signal from APPL-A to actually initiate APPL-B.

In either case, APPL-A communicates with APPL-B through a communications resource manager (CRM), such as APPC/MVS. The main function of a CRM is to allow applications to communicate across systems, *but it also allows RRS on one system to communicate with RRS on another system*.

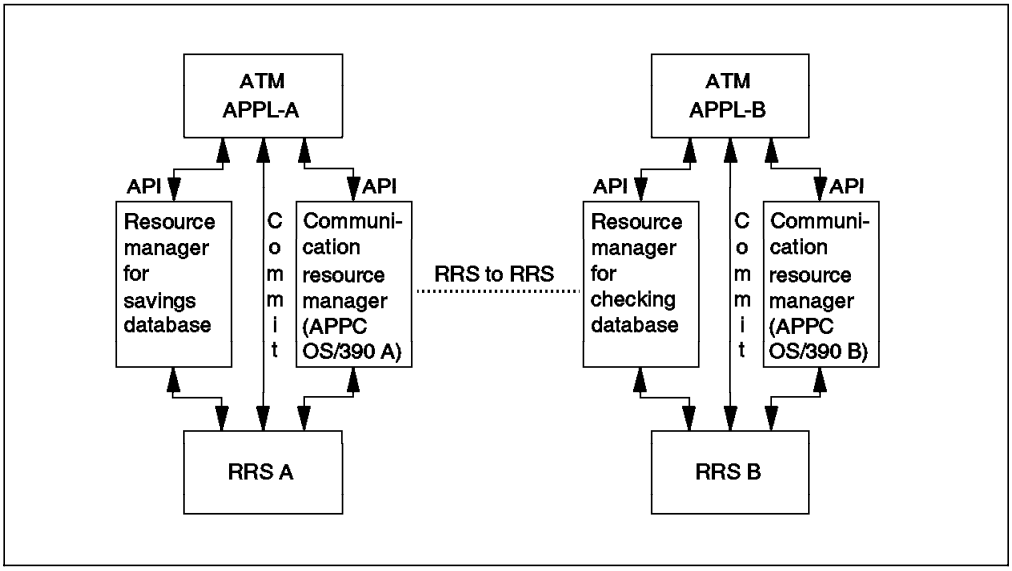


Figure 106. Distributed Resources - Peer-to-Peer

13.4.1 Distributed Resource Recovery: Peer-to-Peer Model

Using the peer-to-peer model, all systems are equal until an application program running on a system issues the commit request that begins the sync-point operation. The fact that any system can initiate the sync-point operation is an important attribute of the peer-to-peer model.

When a work request is distributed rather than local, the two-phase commit processing is slightly different: on the originating system, the UR state becomes in-doubt at the end of phase 1. The state remains in-doubt until all the votes are collected and a commit or backout result is returned to the originating system.

To understand peer-to-peer sync-point processing, remember that RRS is an exit manager; it drives resource manager exit routines in response to resource recovery events. For example, when an application requests commit, RRS drives the PREPARE and COMMIT exit routines for each resource manager involved.

A brief description of the peer-to-peer model, with a more detailed description of what happens when the ATM user mentioned earlier requests a transfer of money from a savings account to a checking account, follows:

1. The ATM user requests transfer of money from a savings account to a checking account. The savings account database resides on one system (system A), while the checking account database resides on another system (system B).
2. The ATM application program (APPL-A) on system A receives the input from the ATM user.
3. APPL-A requests that the savings resource manager on system A subtract the money from the savings database.

APPL-A communicates with APPL-B and APPL-B requests the checking resource manager on system B to add the money to the checking database. To make this request, APPL-A communicates with APPL-B (the part of the ATM application that runs on system B). APPL-B tells the resource manager on system B to add the money to the checking account database.

4. APPL-A calls RRS to commit the database changes. Because the application on system A requests the commit, system A becomes the *initiating system*.
5. RRS system A asks both resource managers to prepare for the changes.
   - RRS on system A collects prepare votes from resource managers on system A. These votes are called *local prepare votes*.
   - RRS on system A tells RRS on system B (the *agent system*) to collect votes from resource managers on system B. These votes are called *distributed prepare votes*.

     To collect these distributed prepare votes:

     – RRS on the initiating system (system A) tells APPC on system A to notify the application on the agent system (system B) of the need to commit the changes. APPC on system A contacts APPC on system B, and APPC on system B tells APPL-B that a commit is needed.

     – APPL-B calls RRS, requesting that the changes be committed.

     – RRS on system B, the agent system, collects distributed prepare votes from the resource managers on system B.
6. Both resource managers indicate to RRS on system A whether or not they can make the changes by voting YES or NO.
   - RRS on system A, the initiating system, determines the outcome of the local prepare votes.
   - RRS on system B, the agent system, tells RRS on system A the result of the distributed prepare votes, using APPC to communicate with RRS on system A.

   RRS on system A, the initiating system, makes the final decision to commit or back out the resource changes, basing the decision on the local prepare votes and the result of the distributed votes. For this example, assume that the overall result is to commit the resource changes.
7. In response, RRS on system A notifies both resource managers to make the changes permanent, which is called committing the changes.
   - RRS on the initiating system (system A) tells the resource managers on system A to commit the changes.
   - RRS on SYSA uses APPC to tell RRS on system B, the agent system, that the final decision is to commit the resource changes.
8. The resource managers complete the commit and return OK to RRS on system A.
   - The resource managers on system A tell RRS on system A that the commit is complete.
   - The resource managers on system B tell RRS on system B that the commit is complete, and RRS on system B uses APPC to tell RRS on system A that the commit is complete on the agent system.

9. RRS on system A issues a return code to APPL-A to indicate that its commit request was successful; the databases have been changed.

Note that the *initiating system* is the system on which the commit request is first issued. In the example, the initiating system is system A. Every other system (such as system B in the example) is an *agent system*. The communications resource manager (CRM) that runs on an agent system is called an *agent CRM*.

At this point, you might want to know more about how RRS uses CRM, for example APPC, to communicate across systems. The basic mechanism is RRS exits:

- APPC registers with RRS as a resource manager, just like any resource manager. Thus, RRS will drive APPC′s PREPARE and COMMIT exit routines when the savings application and the checking application issue their commit requests.

  Note: Unlike the other resource managers, however, APPC has no database to update. Instead, APPC is responsible for communication. Thus, APPC uses its COMMIT and PREPARE exit routines to kick off communications, not to update databases:

  - The PREPARE signal on OS/390_A, for example, is sent to OS/390_B by APPC′s PREPARE exit, which then waits for a response from the application.
  - The REQUEST_COMMIT signal is sent from OS/390_B by APPC′s DISTRIBUTED_SYNCPOINT exit routine, which then waits for a response.
- After APPC′s PREPARE exit on OS/390_A receives the response, the COMMITTED signal is sent by APPC′s COMMIT exit routine, which then waits for a response.
- After APPC′s waiting DISTRIBUTED_SYNCPOINT exit routine on OS/390_B receives the response, the FORGET signal is sent by APPC′s END_UR exit routine to OS/390_A and received by its waiting COMMIT exit.

The initiator is responsible for the overall decision. The agents are responsible for collecting local votes and distributing the decision to the local coordinator (RRS). When APPC/MVS becomes an agent, it informs RRS by taking the *distributed sync-point resource manager* role, which makes it responsible for informing RRS of the commit decision.

RRS on each MVS system provides the two-phase commit protocol for the resource managers on its system. Each RRS collects the prepare votes from the local resource managers, then returns the collective vote to the DSRM.

13.4.2 Distributed Resource Recovery: Client-Server Model

With the client-server model, there can be one client system, which is always the initiating system, and many server systems, which are always agent systems.

The client-server model provides a generic, or flexible, approach to distributed sync-point processing. It is used, for example, by Transactional Remote Procedure Call (TRPC), a communications protocol, and might be suited for many other uses.

Figure 107 on page 180 shows the high-level flow of sync-point processing using the client-server model. The dotted lines link the functions provided on each system.

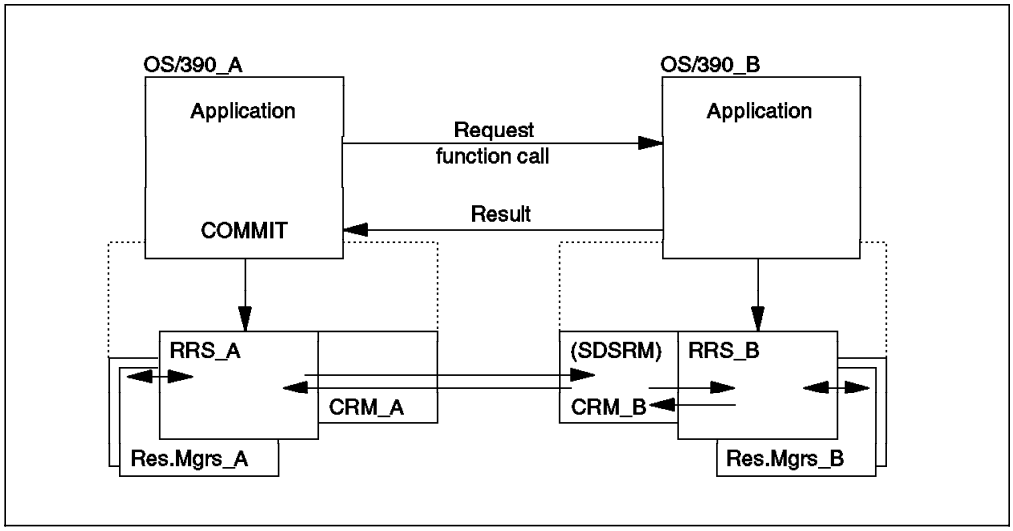


Figure 107. Distributed Resources - Client-Server

In Figure 107, note that the application on OS/390_A, the client system, initiates the sync-point by sending a function call to OS/390_B to request resource changes. When OS/390_B sends a successful result back, the application on OS/390_A calls RRS on OS/390_A to commit the changes. RRS_A sends the request through the communications resource manager (CRM_XA) to OS/390_B. CRM_B takes the server distributed sync-point resource manager (SDSRM) role and, acting as an agent, calls RRS2 on OS/390_B to collect the votes on OS/390_B, then returns the result to RRS_A on OS/390_A.

To understand client-server sync-point processing in more detail, assume a transaction where a user requests the transfer of money from a savings account to a checking account. The transaction requires updates to protected resources on two systems. UR_A represents the changes on the first system, the initiating system, where RM_A manages the savings account database. UR_B represents the changes on the second system, the agent system, where RM_B manages the checking account database. CRM_B on the second system takes the role of the *server distributed sync-point resource manager* (SDSRM).

The actions required to process this sample transaction follow:

1. After making update requests to both RM_A, the resource manager for savings on the initiating system, and RM_B, the resource manager for checking on the agent (server) system, the application issues a call to RRS_A to commit the distributed transaction.
2. RRS_A, RRS on the initiating system, requests the state of the resources managed by RM_A. RRS invokes the resource manager′s STATE_CHECK exit routine to check the state of its resources.
3. If its resources are in the proper state for commit, the state of UR_A changes to in-prepare.
4. RM_A, the resource manager for savings, prepares the resources for commit, then indicates to RRS_A that it is ready to commit the changes.

5. CRM_A, the communications resource manager on the initiating system, sends a PREPARE signal to CRM_B, the communications resource manager on the agent, or server, system.
6. CRM_B, on the agent, or server, system, instructs RRS_B to initiate the prepare phase for UR_B, which represents the requested change to the checking account.
7. RRS_B, RRS on the agent, or server, system, requests the state of the resources managed by RM_B. That is, RRS invokes the resource manager's STATE_CHECK exit routine to check the state of its resources.
8. If the resources are in the proper state for commit, the state of UR_B changes to in_prepare.
9. RRS_B then drives any PREPARE exit routine that the resource manager for checking provides, and the resource manager prepares the resources for commit, then indicates to RRS_B that it is ready to commit the changes.
10. RRS_B collects the votes from any PREPARE exits but, because the results from the initiating system are not known, changes the state of UR_B to in_doubt.
11. RRS_B returns the collective vote to CRM_B. For this example, assume that the collective vote is to commit the changes.
12. CRM_B sends a signal to CRM_A to request that the initiating system commit the changes.
13. RRS_A collects votes from any PREPARE exit routines. For this example, assume that the collective vote is to commit the changes. Because the agent system had returned a commit signal, RRS_A hardens the commit decision. That is, RRS_A writes a record to its log to indicate that the state of UR_A is now in_commit.
14. RRS_A drives any COMMIT exit routine the resource manager for savings provides, telling the resource manager to make the requested resource changes permanent.
15. CRM_A sends a signal to CRM_B to indicate that the initiating system has committed the changes.
16. CRM_B, acting as the SDSRM, calls the Commit_Agent_UR service to return the commit decision to RRS_B. RRS_B hardens the commit decision. That is, RRS_B writes a record to its log to indicate that the state of UR_B is now in_commit.
17. RRS_B then drives any COMMIT exit routine the resource manager for checking provides, telling the resource manager to make the requested resource change permanent.
18. RRS_B then drives any END_UR exit routine that the resource manager provides, and changes the state of UR_B to in_end.
19. RRS_B then returns to CRM_B with the information that the local resources, the resources on the agent, or server, system, are committed.
20. CRM_B calls the Forget_Agent_UR_Interest service to tell RRS_B to delete its log record. The state of UR_B is now forgotten.
21. CRM_B sends a signal to CRM_A to indicate that the initiating system can forget the UR.

22. RRS_A then drives any END_UR exit routine that the resource manager provides, and changes the state of UR_A to in_end.
23. RRS_A then returns the results of the commit request to the application and deletes the log record for UR_A. The state of UR_A is now forgotten.

13.4.3 Distributed Resource Recovery Decisions

Whether a sync-point is local or distributed, following the two-phase commit protocol means that a decision to commit or back out a set of resource changes is, on one level, a mutual decision. That is, the decision to commit or back out is made by the participant designated to make the decision, based on input from the whole set of participants.

There are, however, occasions when a local or distributed resource manager might make a commit or backout decision on its own. This decision is called a *heuristic* decision because it is made by a resource manager that is not designated to make the final decision for the UR as a whole. Installation personnel usually are involved in making a heuristic decision. For example:

- Suppose a system involved in distributed resource processing is hung. Installation personnel use RRS panels to resolve any URs in an in-doubt state. The installation might commit the UR on one system, but the resource managers or installation personnel might back out the UR on another system.
- Database locks are being held too long because one resource manager involved in a UR failed. In this case, a heuristic decision by installation personnel allows the locked resources to be made available for processing before all of the UR is completed.

There are three possible heuristic conditions:

- Heuristic commit (HC) - A heuristic decision to commit some of the protected resources associated with a UR.
- Heuristic reset (HR) - A heuristic decision to back out some of the protected resources associated with a UR.
- Heuristic mixed (HM) - Inconsistent commit or backout decisions for a UR. One of the decisions is heuristic.

Any heuristic condition is a problem because it often means that there have been inconsistent changes to protected resources. Resolving the problem might require manual intervention.

RRS provides Interactive System Productivity Facility (ISPF) panels to allow you to work with RRS. When you use the panels, you can view the following information:

- RRS logs
- UR information
- Resource manager information

Through the panels, you can also take the following actions:

- Determine where a resource manager can restart after a system failure.
- Resolve an in-doubt state for a UR to in-commit or in-backout.
- Remove a resource manager′s interest in a UR.

Thus, the panels provide a way for you to troubleshoot resource recovery. You might use them, for example, if an application, is hung up and you suspect that resource recovery might be the cause of the problem.

13.5 Resource Recovery Services Recoverability

While processing protected resources, the resource manager passes through different states. The resource manager state determines the processing the resource manager can perform:

- Registering - The resource manager registers itself with the operating system′s registration services.
- Setting exit routines - The resource manager sets its exit routines by identifying them and their entry points. Note that the resource manager′s EXIT_FAILED exit routine or the exit manager can unset the exit routines, leaving the resource manager in a registered state, but with its exit routines unset.
- Restarting - The resource manager restarts itself by retrieving and processing any URs that were incomplete from the last time the resource manager was running. Note that the process is the same for starting and restarting.
- Expressing interest in a context - When an application program requests access to a resource, the resource manager expresses interest in the context associated with the application and the work request.
- Expressing interest in a UR - The resource manager expresses interest in the UR being processed by the application program.
- Protecting the resource - The resource manager changes or does not change the resource.

13.5.1 Restarting

The processing steps required to start a resource manager for the first time or to restart it at a later time are basically the same.

When a resource manager restarts, RRS requires it to use the same resource manager name that it used when it was previously running. This requirement exists because RRS uses the resource manager name to identify incomplete URs across failures that involve the resource manager. The name must be unique in a sysplex.

After a resource manager has set its exit routines, it must:

- Check that the logs being used are the same as the logs used previously.
- Obtain any interests in URs that were left as incomplete when it or RRS previously stopped running.
- Respond to any incomplete interests, based on the information returned from RRS and the information in the resource manager logs.

  RRS does not return the incomplete URs in any particular order. If the resource manager expressed protected interest several times in a UR, RRS returns each interest separately with its unique UR interest token.

When a resource manager restarts, it obtains information from RRS to verify that RRS can provide the state of resources at the time the resource manager was

last active. Because RRS keeps information about resources in log streams, the resource manager can verify that the log streams RRS is now using are the same as when the resource manager was last active. If the log streams RRS is using do not match the log streams the resource manager expects, then the resource manager might need to shut down.

To verify that the RRS log streams are current, the resource manager uses RRS services to compare the log name of the current log streams with the log name it expects. Note that the log name is not the name of an actual log stream but a constant, something like a token, that RRS uses to identify a particular set of log streams. A resource manager can also create and maintain a resource manager log name to identify the set of logs that it uses.

If RRS fails, a resource manager might receive a return code of ATR_NOT_AVAILABLE from a call to an RRS service. If the resource manager has set a NOTIFICATION exit routine for RRS, the system invokes the routine to notify the resource manager that RRS has restarted. The NOTIFICATION exit routine should call Set_Exit_Information to reset the RRS exit routines and perform restart processing.

When RRS restarts after a failure, its records are in the RRS logs. From these logs, RRS recreates the state of all incomplete URs and interests in them.

13.5.2 RRS Logging

RRS uses five log streams that are shared by the systems in a sysplex. Every MVS image with RRS running needs access to the coupling facility and the DASD on which the system logger log streams are defined.

The RRS images on different systems in a sysplex run independently but share log streams to keep track of the work. If a system fails, RRS on a different system in the sysplex can use the shared logs to take over the failed system′s work.

The logs are:

- RRS archive log - Information about completed URs. This log is recommended but optional.
- RRS resource manager data log - Information about the resource managers using RRS services.
- RRS main UR state log - The state of active URs. RRS periodically moves this information into the RRS delayed UR state log when UR completion is delayed.
- RRS delayed UR state log - The state of active URs, when completion is delayed.
- RRS restart log - Information about incomplete URs needed during restart. This information enables a functioning RRS instance to take over incomplete work left over from an RRS instance that failed.

To minimize any risk of losing data, you can use the unconditional duplexing system logger option for the RRS logs. This option ensures that all the data in the log is constantly backed up to DASD. If an error occurs in the coupling facility′s data, the DASD backup is a reliable copy of valid data that is available for restart.

Note, however, that duplexing the logs can significantly slow performance, and RRS will run effectively without duplexing. But, if RRS logs are damaged, RRS might be unable to maintain integrity for the work it coordinates, resulting in inconsistent resources or RRS failure.

While your installation must decide on the risk it can afford to take, IBM strongly recommends that you use unconditional duplexing for the resource manager data log. This log is small and infrequently updated, so its impact on performance is minimal, but any loss of data, unresolved gap, or any permanent error will force an RRS cold start.

13.6 Running RRS

Before you can start RRS, you must:

- Set up your log streams and ensure that the system logger is active. (RRS requires the system logger; if it is not active when RRS starts, RRS waits until the system logger is active.)
- Establish the priority of the RRS address space. The best way to control RRS priority is through the workload manager (WLM). IBM recommends that you use the system default for started tasks as the goal of the RRS address space.
- Define RRS as a subsystem. Place the following statement in the IEFSSNxx parmlib member:

  ```
  SUBSYS SUBNAME(RRS)
  ```

  Place the statement after the statement that defines the primary subsystem.

  You can replace RRS with a subsystem name of your choice but do not supply any other parameters. In particular, do not supply an initialization routine.

13.6.1 Starting and Stopping RRS

Once you have set address space priority, provided the statement in IEFSSNxx, and know that system logger is active, you can start RRS with the following operator command:

```
START ATRRRS
```

Note: ATRRRS is the name of the cataloged procedure that IBM supplies.

The start can be a warm start or a cold start.

You can stop RRS with the following operator command:

```
SETRRS CANCEL
```

If SETRRS CANCEL does not stop RRS, you can use the FORCE RRS,ARM command. In this command, RRS is the subsystem name your installation assigned to RRS in parmlib member IEFSSNxx.

RRS should be active on an MVS system that has programs involved in resource recovery. In a sysplex, RRS should be active on every MVS system image that might take part in distributed resource recovery. Only one instance of RRS can be active on each system image.

Usually RRS should be active all the time. Use a START ATRRRS command in the COMMNDxx parmlib member to start RRS during system initialization. If RRS fails, it can restart.

Note: Do not try to start RRS from the IEACMD00 parmlib member; programs RRS depends on have not been started.

13.6.2 Automatic Restart

If RRS fails, it can use automatic restart management to restart itself in a different address space on the same system. However, it will not restart itself following a SETRRS CANCEL command. To stop RRS and cause it to restart automatically, use the FORCE command with ARM and ARMRESTART.

To make automatic restart possible, your installation must:

- Provide an ARM couple data set that contains, either explicitly or through defaults, an automatic restart management policy for RRS.
- Activate the ARM couple data set through a COUPLExx parmlib member or a SETXCF operator command. The data set must be available when RRS starts and when it restarts.
- Make sure that no element-restart denies the restart of an RRS element or changes its restart. An exception is an exit routine that vetoes RRS restart but then itself starts the RRS address space.

  This technique, however, might delay other elements in the restart group that have to wait for RRS services to become available.

As with other automatic restart management elements, an ENF signal for event 38 occurs when RRS registers with automatic restart management or is automatically restarted.

13.6.3 Warm Start

In a warm start, RRS can complete work that was in progress when a previous RRS instance failed or was intentionally stopped. A warm start occurs when all of the RRS logs are intact and available to the restarting RRS instance.

To enable a warm start, enter the SETRRS CANCEL command, or the FORCE command with ARMRESTART, to stop RRS. Then enter a START command, specifying the name of the RRS cataloged procedure in the SYS1.PROCLIB system library.

A warm start also occurs when RRS is started on any system in a sysplex after the first. In effect, any attempt to start RRS when its logs are not empty is a warm start.

Note: While you can warm start RRS as long as the resource manager data log is intact, a warm start after damage to other logs generally causes loss of data about incomplete transactions.

13.6.4 Cold Start

In contrast, a cold start occurs when the RRS logs are empty. An RRS cold start is a sysplex-wide operation, and it wipes out any information in the logs. In a cold start, RRS cannot complete any work that was in progress; the RRS logs are not available.

To enable a cold start, do the following:

1. Route the SETRRS CANCEL command to all systems in the sysplex where RRS is active.
2. Decide whether to save the contents of the logs. To save the contents, which is the safer choice, choose a log group name that is different from the one previously in use. If there is no need to save any data in the logs, use the same log group name.
3. Specifying the log group name you need, use the IXCMIAPU utility to delete and redefine the RRS resource manager data log. (A log group is a group of systems that share an RRS workload.) The ATRCOLD member of SYS1.SAMPLIB contains sample JCL to invoke IXCMIAPU to delete and redefine the RRS resource manager data log.

   Note: Do not run IXCMIAPU when RRS is active; stop RRS before you run the utility.
4. Start RRS on one system.

In response, the first RRS that starts initiates a sysplex-wide cold start. This cold start does the following:

- Moves logged information about incomplete URs to the RRS archive log. Later, you can use ISPF panels to browse the RRS archive log to see the incomplete URs.
- Deletes the contents of the RRS main UR state log, RRS delayed UR state log, and RRS restart log.

Once RRS has been started for the first time, do not use the IXCMIAPU utility to change any logs other than the RRS resource manager data log. Such action might cause serious database inconsistencies that require manual verification and updating.

Chapter 14. Communications Server for OS/390

The communications server integrated with OS/390 provides communication and network function as a foundation for critical applications. Network access to OS/390 is not constrained by the brand of workstations, servers, or platforms in use throughout the enterprise. The communications server supports the programming interfaces most common in the current information technology industry: APPC, SOCKETS/RPC, SNA(3270). It also supports wide-area networking protocols such as SNA, TCP/IP, and ATM. Because it integrates the VTAM AnyNet feature, which implements the multiprotocol transport networking (MPTN) architecture, the OS/390 communications server gives you a choice of which application to use over which network. OS/390 and its communications server can play the role of Internet host.

The communications server includes VTAM (including the AnyNet function), TCP/IP, TIOC, and FFST. It provides SNA (3270), APPC, High Performance Routing, ATM support, Sockets, and RPC.

14.1 Performance Enhancements

The communications server provides improved performance, reliability, availability, and serviceability (RAS) for users who migrate from TCP/IP V2R1 for MVS to TCP/IP Version 3 Release 2 for MVS. The bulk of the RAS enhancements were provided in TCP/IP Version 3 Release 1 for MVS. Among the new TCP/IP functions, extensions to functions that were introduced in previous releases, are usability enhancements, and some restructuring of TCP/IP code to enhance its reliability, availability, and serviceability. TCP/IP Version 3 Release 1 for MVS enhancements includes new and updated APIs:

- Sockets Extended - TCP/IP Version 3 Release 1 for MVS provides a generalized API to conduct peer-to-peer conversations with other hosts in the TCP/IP network by creating socket applications in COBOL, PL/I, and assembler. Sockets Extended support includes both the macro API and the CALL instruction API. The socket applications can be developed for the TSO, batch, IMS, and CICS environments. They can be designed to be re-entrant and multithreaded.
- IMS Feature - Starting with TCP/IP Version 3 Release 1 for MVS, the IMS TCP/IP feature supports the development of peer-to-peer applications in which IMS and a TCP/IP-connected peer form a client/server relationship. Using this support, IMS can be either client or server.
- CICS Sockets Extended Support - TCP/IP Version 3 Release 1 for MVS provides the ability to use the generalized API and create socket applications in COBOL, PL/I, and assembler.
- REXX Socket API - Starting with TCP/IP Version 3 Release 1 for MVS, you can write a REXX socket program to access the TCP/IP socket interface. Calls are provided to initialize sockets, exchange data, perform management activities, and close sockets. This allows you to use REXX to implement and test TCP/IP applications. Using REXX also eliminates the need for CLISTs, and the REXX programming tools can help you create interfaces such as linking NETSTAT to the NetView program.
- X/Open Transport Interface (XTI) Enhancement - Starting with TCP/IP Version 3 Release 1 for MVS, you can write applications that can access the open

transport interface. The transport services conform to RFC 1006, which defines a protocol mapping component enabling applications to be mapped to the TCP/IP socket interface.

- OpenEdition MVS Applications Support - TCP/IP Version 3 Release 1 for MVS supports the OpenEdition MVS socket API in the TCP/IP environment.

14.2 TCP/IP Version 3 Release 2 Enhancements

TCP/IP Version 3 Release 2 for MVS includes further enhancements:

- Crossbook-Linking Enhancement - Enhancements were made to the TCP/IP Version 3 Release 1 for MVS softcopy books. These enhancements take advantage of BookManager's linking functions to improve the usability and retrievability of the TCP/IP softcopy library. These functions allow you to move between related pieces of information. Links can go from one point to another within the same softcopy book, or from one softcopy book to another.
- Problem Determination and Diagnosis - TCP/IP Version 3 Release 2 for MVS provides numerous aids for diagnosis, for example, System Error Return Codes for Socket Calls. Several socket call error numbers (ERRNOs) are new or changed for Version 3 Release 2.
- Numerous enhancements to FTP Server, including support of the checkpoint restart functions and data compression as defined in RFC 959, and SQL.

14.3 High Performance Routing

This section describes the characteristics of high performance routing, and contains several acronyms. The meanings of these acronyms are found in the footnotes which follow.

High performance routing (HPR) has been extended to serve a broader range of network configurations, maximizing its value to the enterprise. In VTAM V4R3, a composite network node (CNN)[3] can provide only automatic network routing (ANR)-level HPR support through an NCP of the CNN, and can thus only be an intermediate node on an HPR route. A CNN cannot provide rapid transport protocol (RTP)[4]-level HPR support as the endpoint of an HPR route. HPR support is not available for connections to or from VTAM in a CNN.

In VTAM Version 4 Release 4, HPR support is available through VTAM in a CNN. A CNN now provides automatic network routing (ANR)[5]-level support, through NCP or VTAM, as an intermediate node on an HPR route. VTAM in a CNN can also provide RTP-level HPR support as the endpoint of an HPR route.

---

[3] Composite network node (CNN) is a type 5 node and its subordinate type 4 nodes that support APPN network node protocols and which appear to an attached APPN or low-entry networking (LEN) node as a single network node.

[4] Rapid transport protocol (RTP) is a connection-oriented, full-duplex transport protocol for carrying session traffic over high performance routing (HPR) routes. This connection is between two high performance routing (HPR) nodes that may traverse one or more intermediate HPR nodes and links. The connection endpoints provide error recovery and adaptive rate-based flow control for the connection traffic, and nondisruptive switching of the underlying physical path in the case of route outage. The intermediate HPR nodes minimize their routing overhead using automatic network routing (ANR) protocols, which rely on header information to permit efficient source routing and prioritized transmission along the RTP connection.

[5] In high performance routing (HPR), automatic network routing (ANR) is a highly efficient routing protocol that minimizes cycles and storage requirements for routing network layer packets through intermediate nodes on the route.

14.4 Availability Enhancements

Dramatic new availability support for the S/390 Parallel Sysplex insulates end-users from applications and processor failures.

In pre-VTAM Version 4 Release 4 releases, if an application program fails or is brought down, VTAM can retain active sessions, allowing the same or another application program to reconnect to the sessions, and avoiding the need to re-establish the sessions. If an application program fails, it can reconnect to the retained sessions when it recovers. Also, another application program can take over the sessions.

In VTAM Version 4 Release 4, multinode persistent session (MNPS) support is extended to VTAM, the operating system, and hardware failures. A persistent session is an LU-LU session that is retained after the failure of VTAM, the operating system, or the hardware. Since VTAM maintains session and network information in the MVS coupling facility, MNPSs enable you to re-establish an application's sessions after any of the following failures:

- Application
- VTAM
- Operating system

14.5 TSO/E Enhancements

Single systems image capability is extended to TSO/E SNA users of S/390 Parallel Sysplex for improved usability. In previous releases, TSO/E was identified by a unique application name. To start a session with TSO/E, the TSO/E user specified this unique name together with the TSO/E user ID and other logon parameters. This, however, requires that this particular TSO/E system be up and running. If the system is down, a session cannot be started with it until the system comes up. In addition, there is no mechanism to evenly distribute sessions across a group of TSO/Es.

Starting with OS/390 Release 3, TSO/E can also be identified by a generic name. TSO/E on one MVS system can be known by the same generic name as TSO/E on any other MVS system in the sysplex and all of the TSO/Es sharing the same generic name can be concurrently active. This provides both increased TSO/E availability and workload balancing. MVS workload manager makes efficient use of system resources by selecting a session partner based on balanced load distribution from among TSO/E instances with the same generic name within the Parallel Sysplex.

TSO/E availability increases because all TSO/Es can now be accessed by the same generic name. If a particular TSO/E system is down, a session request using a generic name can still be satisfied by another TSO/E system with the same generic name. The MVS workload manager makes efficient use of system resources by selecting the TSO/E system based on balanced load distribution from among all TSO/E systems with the same generic name.

When a TSO/E user loses connectivity to the TSO/E system, the user address space is not terminated if the installation specified a reconnect interval. If the user logs back on to TSO/E using the generic name before the reconnect interval has expired, the user is reconnected to the original user address space and can

resume previous activity with minimal loss. If the reconnect interval expires before the user logs back on, the original user address space is terminated.

14.6 VTAM APPN Networks

S/390 applications can take advantage of high speed, Asynchronous Transfer Mode (ATM) connections in the network without application change. VTAM supports ATM in two important ways:

- By enabling subarea and advanced peer-to-peer networking (APPN) data to be routed across ATM networks--both public wide area networks (WANs) and private campus networks through local area network (LAN) emulation access.
- By enabling high performance routing (HPR) Advanced Peer-to-Peer Networking (APPN)[6] across ATM networks--both public and private--through native access.

In pre-VTAM Version 4 Release 4 releases, VTAM routes subarea and APPN data across ATM networks only through LAN emulation access. ATM networks accessed through LAN emulation are attached to VTAM through the IBM S/390 Open Systems Adapter. However, they are defined to VTAM as though they are Ethernets or Ethernet-type LANs or Token-Ring (full duplex) networks. Although sessions occurring over ATM LAN emulation connections enjoy the benefits of fast transfer of data, they cannot exploit the full functions of guaranteed bandwidth and quality of service (QoS) offered by ATM networks. Generally, ATM networks accessed through LAN emulation appear to VTAM to be, and to offer the capabilities of, Ethernet or Ethernet-type LANs or Token-Ring networks.

In VTAM Version 4 Release 4, VTAM routes APPN data across ATM networks through LAN emulation access and native access. VTAM can route only HPR APPN data across ATM native connections. Routing of subarea and non-HPR APPN data across ATM networks requires ATM LAN emulation connections. ATM networks accessed through native ATM are also attached to VTAM through the IBM S/390 Open Systems Adapter. They are defined to VTAM as ATM networks. HPR APPN sessions occurring over native ATM connections can exploit the full services of ATM networks, such as guaranteed bandwidth and quality of service.

14.7 High Performance Data Transfer

New high performance data transfer (HPDT) facilities offer improved throughput for applications. High performance data transfer (HPDT) optimizes the performance of large message transfers for VTAM LU 6.2 applications. The performance enhancements become more substantial as the size of PIUs and amount of data transferred on the APPCCMD API increase.

HPDT includes a service that reduces the number of times data is moved for an APPCCMD send or receive request. HPDT services are provided for sessions

[6] An extension to SNA featuring: (a) greater distributed network control that avoids critical hierarchical dependencies, thereby isolating the effects of single points of failure; (b) dynamic exchange of network topology information to foster ease of connection, reconfiguration, and adaptive route selection; (c) dynamic definition of network resources; and (d) automated resource registration and directory lookup. APPN extends the LU 6.2 peer orientation for end-user services to network control and supports multiple LU types, including LU 2, LU 3, and LU 6.2.

that traverse certain high bandwidth network attachments (IBM S/390 Open Systems Adapter connected to a native ATM network, APPN node-to-node connections (ANNC) that use HPR, XCF links between processors in a sysplex), or when sessions connect two intrahost applications.

HPDT services are available to all APPCCMD applications. No application change is required to receive performance benefits. However, an HPDT interface is also provided to enable authorized applications to obtain performance benefits even greater than that provided by HPDT services.

HPDT services are available when the session is not using VTAM cryptography or VTAM compression. Both of these functions require VTAM to move the data before the transfer. HPDT services also rely on certain high-bandwidth, network-attached facilities, which include:

- Asynchronous transfer mode (ATM) connections
- APPN node-to-node connections that use HPR
- Sysplex (XCF)
- Intrahost application-to-application

An array of additional enhancements improves security, serviceability, and usability.

Appendix A. Sample SYSOUT Application Program (SAPI)

The following SAPI sample program may be used to “play” with SYSOUT application program interfaces. The SAPI program must be link-edited with authorization code 1 (one). If it will be used in TSO (in foreground or background), the AUTHCMD and AUTHPGM sections in the IKJTSOxx parmlib member must be also updated to include the SAPI program name.

Internet access to code

Internet access to source code found in SG24-2067 can be accessed via the Internet.

Sample code/procedures are available in softcopy on the internet from the redbooks webserver.

Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG24xxxx

Alternatively you can go to:

http://www.redbooks.ibm.com

and select SEARCH and then Additional Redbook Materials. (or follow the instructions given since the web pages change frequently!)

```
*PROCESS USING(WARN(2))
*  This will keep the caps and line numbers off!
MAIN     TITLE 'SAPI - Main external writer processing loop'
SAPI     AMODE 31                  keep every thing below the line
SAPI     RMODE 24                  ditto
         ENTRY LDASTRT
SAPI     START 0
* ------------------------- Reference Info --------------------------
*
*  The SAPI program can be run as an authorized batch program,
* started task, called TSO program, and TSO command processor.
* When the SAPI program runs in TSO foreground, the user/program
* interactions are through TGET/TPUT services. The TSO user can
* enter new selection criteria by hitting the ATTN key and
* then typing in the new options. In TSO foreground the HELP
* option will display all the available selection criteria.
*
*  The SAPI application works in the same manner as the IBM
* external writer. When initially started, the default SAPI
* selection criteria used for GET/PUT processing is show below:
*
*                        - DEST=SAPI
*                        - CLASS=O1ZA
*                        - Job Name=SAPI*
*                        - Job range=5 to 150
*                        - AGE = 0
*                        - MIN LINE = 0
*                        - MAX LINE = 1000
*                        - MIN PAGE = 0
*                        - MAX PAGE = 100
*                        - DS Disp  = delete
*                        - Selection via all queues
*
*     (SSS2SEL1 = x'FD', SSS2SEL2/3/5 = x'00', SSS2SEL4 = x'38')
*
*  Once started, the selection criteria is changed as is done with the
* IBM external writer. The applicable keywords for the SAPI application
* are listed in the below table under the heading KEYWORD. The table
* also shows the SSS2 flag and field set by the application per the
* keyword included with the 'F xxx' command when running the program
* outside TSO foreground or are entered as the response to the ATTN
* prompt in TSO foreground:
*
*                              KEYWORD        FLAG        FIELD
*                              -------        ----        -----
*  Keywords applicable to SAPI program execution control:
*
*     -> Stop processing       END/STOP       n/a          n/a
*     -> Help (TSO only)       HELP           n/a          n/a
*     -> Process one JOB       ONEJOB         n/a          n/a
*     -> Process one DS        ONEDS          n/a          n/a
*
*  Keywords applicable to SAPI processing are as follows:
*
*     -> Get/Put               GET            n/a         SSS2TYPE
*     -> Count request         COUNT          n/a         SSS2TYPE
*     -> Bulk Modify req       BULK           n/a         SSS2TYPE
*
*
*                                          SSS2SEL1:
*     -> default selection     DEFAULT         n/a          n/a
*        - defined above
*
*     -> TSO Hold              SELTSO      (SSS2SHLD)       n/a
*     -> EXTWTR Hold           SELXWTR     (SSS2SXWH)       n/a
*     -> TSO/EXTWTR Hold       SELHOLD     (SSS2SHOL)       n/a
*     -> WTR                   SELWTR      (SSS2SWTR)       n/a
*     -> TSO/EXTWTR/WTR        SELALL      (SSS2SAWT)       n/a
*     -> CLASS                 CL=         (SSS2SCLS)     SSS2CLSL
*     -> DEST                  D=          (SSS2SDST)     SSS2DEST
*     -> JOBNAME               JBN=        (SSS2SJBN)     SSS2JOBN
*     -> JOBID                 JBI=        (SSS2SJBI)     SSS2JBIL
*     -> JOBID                 HJBI=       (SSS2SJBI)     SSS2JBIH
*
*                                          SSS2SEL2:
*     -> PROGRAM NAME          W=          (SSS2SPGM)     SSS2PGMN
*     -> FORM                  F=          (SSS2SFRM)     SSS2FORM
*     -> CREATOR USERID        U=          (SSS2SCRE)     SSS2CREA
*     -> PROCESS MODE          PM=         (SSS2SPRM)     SSS2PRMO
*     -> w/IP address          IP          (SSS2SIPA)       n/a
```

```
*     -> w/o IP address       NIP         (SSS2SIPN)       n/a
*     -> FCB                  C=          (SSS2SFCB)     SSS2FCB
*     -> UCS                  U=          (SSS2SUCS)     SSS2UCS
*     -> SWB                  SWB         (SSS2FSWB)       n/a
*
*                                         SSS2SEL3:
*     -> Started Task         SELSTC      (SSS2SSTC)       n/a
*     -> Time Sharing User    SELTSU      (SSS2STSU)       n/a
*     -> Batch job            SELJOB      (SSS2SJOB)       n/a
*     -> APPC                 SELAPC      (SSS2SAPC)       n/a
*
*                                         SSS2SEL4:
*     -> MOD                  M=          (SSS2SMOD)     SSS2MOD
*     -> FLASH                FL=         (SSS2SFLS)     SSS2FLSH
*     -> AGE                  A=          (SSS2SAGE)     SSS2AGE
*     -> LINE LIMIT           LL=         (SSS2SLIN)     SSS2LMIN
*                             MLL=        (SSS2SLIN)     SSS2LMAX
*     -> PAGE LIMIT           PL=         (SSS2SPAG)     SSS2PMIN
*                             MPL=        (SSS2SPAG)     SSS2PMAX
*     -> CHARS                CH=         (SSS2SCHR)     SSS2CHAR
*
*                                         SSS2SEL5:
*     -> No CPDS              NCPDS       (SSS2SCPN)       n/a
*
*
*  Keywords applicable to dataset disposition for GET/PUT processing
*  are as follows:
*                                         SSS2DSP1:
*     -> Delete ds            DDEL        (  n/a   )       n/a
*     -> Keep ds              DKEEP       (SSS2DKPE)       n/a
*     -> OP Hold              DRHLD       (SSS2RHLD)       n/a
*     -> Don't process        DRNPR       (SSS2RNPR)       n/a
*          again
*     -> Hold ds              DHOLD       (SSS2DHLD)       n/a
*     -> Release ds           DREL        (SSS2DRLS)       n/a
*     -> Checkpoint RBA       DRBA        (SSS2CHKP)     SSS2RBA
*     -> Null writer name     DNWR        (SSS2DNWR)       n/a
*     -> New class            DCL=           n/a         SSS2DCLS
*     -> New forms            DF=            n/a         SSS2DFOR
*     -> New wtr name         DW=            n/a         SSS2DPGM
*     -> New dest             DD=            n/a         SSS2DDES
*     -> New copy count       DC=            n/a         SSS2CLFT
*
*
*  Keywords applicable to Bulk Modify processing are as follows:
*                                         SSS2UFLG:
*     -> New class            NCL=        (SSS2SETC)     SSS2CLAS
*     -> Delete               DEL         (SSS2DELC)       n/a
*     -> New destination      ND=         (SSS2ROUT)     SSS2DES2
*     -> Release              REL         (SSS2RLSE)       n/a
*
*
*           For error processing scenarios, the following
*           keywords can be used (the application code will
*           set an incorrect value).
*
*      -> wrong type         WTYPE                    SSS2TYPE
*      -> wrong length       WLEN                     SSS2LEN
*      -> duplicate jobs     DUPJ
*
*
*  As an example, suppose you want to select work with a minimum page
* count of 10, maximum page count of 80, class of A and forms of 2PRT.
* The command to enter is: F OO2,PL=10,MPL=80,CL=A,F=2PRT.
*
*  When entering a command, all prior selection criteria no longer
* apply. For example, if the default selection criteria is being used
* and the above example cmd is entered, none of the default selection
* criteria is used (eg, no job name, etc).
*
*  To use default selection criteria with additional keywords, the
* modify command should look like: F OO2,DEFAULT,DNWR (see below
* selection note).
*
*  BULK MODIFY/COUNT Processing
*  ----------------------------
*
*  When changing from Get/Put processing to BULK or COUNT processing,
* specify the keyword and selection criteria for your intended
* processing. For example, if you wanted to delete all class A work in
* the system, the following cmd should be entered: F OO2,CL=A,BULK,DEL.
*
*  Do keep in mind that you need to enter another command after a COUNT
* or BULK request is processed since there are no ECBs involved with
* either of these processes. The appl message displayed when either of
* these cmds are processed is 'SAPI900I Waiting for operator command'.
* This differs from the 'normal' ''SAPI900I Waiting for work' message.
*
*  Once the type is changed to COUNT or BULK, to get back to Get/PUT,
* the GET keyword should be used. The DEFAULT keyword can be used
* to set the selection criteria to revert back to the original default
* selection criteria. Just issue the following cmd: F OO2,GET,DEFAULT.
*
*  Forcing Error Scenarios
*  -----------------------
*
*  For error processing scenarios, the following keywords can be
* used (the application code will set an incorrect value).
*
*        -> wrong type         WTYPE                    SSS2TYPE
*        -> wrong length       WLEN                     SSS2LEN
*        -> duplicate jobs     DUPJ
*
*  Selection Note
*  --------------
*
*  When changing the selection criteria, understand what you are
* doing. For example, say class selection is via the default O1ZA
* and class A work is returned to the appl. When the next request for
* a dataset is made, the class selection list is still O1ZA. This
* will cause JES3 IATOSSO to get to the OSE for the dataset to
* dispose of. Disposition will occur. When the next dataset is
* obtained, the class list of O1ZA is used. Say the next dataset
* in the OSE is class A and no class O, 1 or Z exists. The OSE
* will be written to spool since there is no class O datasets in
* it. We will eventually have to read in the same OSE buffer when
* we get to searching for class A datasets. No problems occur but
* I just want to point out the overhead in the processing that
* occurs.
*
*  Say the above scenario is performed with one difference. Say the
* SSS2DNWR flag is set to clear out the writer name. A loop condition
* will arise, in JES3 anyway. Since writer name is not part of the
* selection criteria and the request is made to set the writer name
* to null, IATOSSO does that on the second IEFSSREQ request. We
* loop through searching for class O, 1 and then Z work as described
* above. When we get to class A we get to the dataset whose writer
* name was changed to a null value. Since the dataset was not
* deleted and it's characteristics match the selection criteria,
* it is scheduled and returned. This repeats for quite some time.
* Be careful for what you ask.
*
* The SAPI program has not been any formal IBM test process.
*
* E&OE
* ---------------------------------------------------------------------
       MACRO
       EXTYEQU
       GBLB  &YEQU
       AIF   (&YEQU).MEND
&YEQU  SETB 1
       SPACE 1
*--------------------------------------------------------------*
*   Equate statements needed by all the modules in this        *
*   library for proper assembly.                               *
*--------------------------------------------------------------*
       SPACE 2
*--------------------------------------------------------------*
*   General Register equates                                   *
*--------------------------------------------------------------*
R0     EQU   0
R1     EQU   1
R2     EQU   2
R3     EQU   3
R4     EQU   4
R5     EQU   5
R6     EQU   6
R7     EQU   7
R8     EQU   8
R9     EQU   9
R10    EQU   10
R11    EQU   11
R12    EQU   12
R13    EQU   13
R14    EQU   14
R15    EQU   15
*--------------------------------------------------------------*
*   Access  Register equates                                   *
*--------------------------------------------------------------*
AR0    EQU   0
AR1    EQU   1
AR2    EQU   2
AR3    EQU   3
AR4    EQU   4
AR5    EQU   5
AR6    EQU   6
AR7    EQU   7
AR8    EQU   8
AR9    EQU   9
AR10   EQU   10
AR11   EQU   11
AR12   EQU   12
AR13   EQU   13
AR14   EQU   14
AR15   EQU   15
*--------------------------------------------------------------*
*   General equates                                            *
*--------------------------------------------------------------*
NOP      EQU   0                 NO OPERATION
FF       EQU   X'FF'             ALL BITS ON
*--------------------------------------------------------------*
*   Compare Instruction equates                                *
*--------------------------------------------------------------*
GT       EQU   2                 A HIGH
LT       EQU   4                 A LOW
NE       EQU   7                 A NOT EQUAL B
```

```
EQ       EQU   8                 A EQUAL B
GE       EQU   11                A NOT LOW
LE       EQU   13                A NOT HIGH
*-------------------------------------------------------------*
*   Arithmetic Instruction equates                            *
*-------------------------------------------------------------*
OV       EQU   1                 OVERFLOW
PLUS     EQU   2                 PLUS
MINUS    EQU   4                 MINUS
NZERO    EQU   7                 NOT ZERO
ZERO     EQU   8                 ZERO
NMINUS   EQU   11                NOT MINUS
NPLUS    EQU   13                NOT PLUS
*-------------------------------------------------------------*
*   Test under mask instruction equates                       *
*-------------------------------------------------------------*
ALLON    EQU   1                 ALL ON
MIXED    EQU   4                 MIXED
NALLOFF  EQU   5                 ALLON+MIXED
ALLOFF   EQU   8                 ALL OFF
NALLON   EQU   12                ALLOFF+MIXED
*-------------------------------------------------------------*
*   Test and set instruction equates                          *
*-------------------------------------------------------------*
LOCKED   EQU   4                 ONE  I.E. LOCKED
UNLOCKED EQU   8                 ZERO I.E. UNLOCKED
*-------------------------------------------------------------*
*   Load real address instruction equates                     *
*-------------------------------------------------------------*
INREAL   EQU   8                 PAGE IS IN REAL STORAGE
NOTIREAL EQU   7                 PAGE NOT IN REAL STORAGE
SEGTBINV EQU   4                 SEGMENT TABLE ENTRY INVALID
PAGTBINV EQU   2                 PAGE TABLE ENTRY INVALID
LENTHINV EQU   1                 LENGTH INVALID
*-------------------------------------------------------------*
*   Access register mode switch equates                       *
*-------------------------------------------------------------*
ARMODON  EQU   512               TURN ACCESS REGISTER MODE
*                                ON
ARMODOFF EQU   0                 TURN ACCESS REGISTER MODE
*                                OFF
*-------------------------------------------------------------*
*   symbols used to set or clear a high order bit             *
*-------------------------------------------------------------*
EQUHOBON EQU   X'80000000'       HIGH ORDER BIT ON
EQUHOBOF EQU   X'7FFFFFFF'       HIGH ORDER BIT OFF
*
         SPACE 2
         MEXIT
ND    ANOP
***  The EXTYEQU has been previously generated. ***
XIT   ANOP
         MEND
*
         TITLE 'EXTYEQU   - Standard Equates'
         EXTYEQU
         TITLE 'CVT       - Communication Vector Table'
         CVT   DSECT=YES,PREFIX=YES
         TITLE 'DCBD      -                                       '
         DCBD  DSORG=PS
         TITLE 'IATYCKSB  - CHECKSWB Parameter List'
         IATYCKSB
         TITLE 'IATYSTBL  - SWB String Table'
         IATYSTBL
         TITLE 'IEFJESCT  - JES Communication Table'
         IEFJESCT
         TITLE 'IEFJSBVT  -                                      '
         IEFJSBVT
         TITLE 'IEFJSCVT  - Subsystem Communications Vector Table'
         IEFJSCVT
         TITLE 'IEFJSIPL  -                                      '
         IEFJSIPL
         TITLE  'IEZCIB   -                          '
CIBSTRT  DSECT
         IEZCIB
         TITLE  'IEZCOM   -                          '
         DSECT
         IEZCOM
         TITLE  'IEZJSCB  -                          '
         IEZJSCB
         TITLE  'IEFJSSIB - Subsystem Information Block'
         IEFJSSIB
         TITLE  'IEFJSSOB - Subsystem Option Block'
         IEFSSOBH
         TITLE  'IAZSSS2  - SAPI Extension '
SSOBGN   EQU   *
         IAZSSS2
         TITLE 'IAZBTOKP  - Token'                                @RFA
         IAZBTOKP                                                 @RFA
         TITLE 'IEFDOKEY  - Dynamic OUTPUT Key Mapping'
         IEFDOKEY
         TITLE 'IEFSJREP  - SJF Retrieve Parm List'
         IEFSJREP
         TITLE 'IEFSJRC - SJF Reason Codes'
         IEFSJRC
         TITLE 'IEFSJTRC - SWBTUREQ Return and Reason Codes'
         IEFSJTRC
         TITLE 'IEFSJTRP - SWBTUREQ Retrieve Parameter List'
         IEFSJTRP
         TITLE 'IEFZB4DO  - Dynamic allocation block mappings'
         IEFZB4DO
         TITLE 'IEFZB4D2  - Dynamci allocation text units'
         IEFZB4D2
         TITLE 'IHASDWA   - System Diagnostic Work Area'
         IHASDWA
         TITLE 'IHAPSA    - Prefixed System Area'
         IHAPSA
         TITLE 'IHAASCB   - Address Space Control Block'
         IHAASCB
         TITLE 'IKJTCB    - Task Control Block  '
         IKJTCB
         TITLE 'IGDMCSMG  - SVC 99 Message '
         IGDMCSMG
SAPIDSEC DSECT
SAPISJFA DS    F                 SJF parm list address
         DS    0F
SAPISJFT DS    XL(SJTRSLEN)      SJF pointer list entry
         DS    0F
SAPISJFK DS    XL(SAPIKYL)       SJF key list
         DS    0F
SAPISJFP DS    XL(SJTRLGTH)      SJF parameter list
SAPISJFS DS    18F               SJF 18 fullword savearea
SAPISJFO DS    XL(SAPISJOS)      SJF output area
SAPISIZ  EQU   *-SAPIDSEC        Size not including SJF
*                                  working storage area since
*                                  that is subject to change
SAPISJFW DS    0F                SJF working storage area
         TITLE 'SAPIXWTR  - External Writer for Testing'
*-------------------------------------------------------------*
*        KWDEFINE - Internal macro to define an entry in     *
*                   the keyword table                        *
*                                                            *
*        The format of the table is:                         *
*          4 bytes - length of keyword                       *
*          8 bytes - keyword                                 *
*          4 bytes - address of keyword processing routine   *
*                                                            *
*                                                            *
*-------------------------------------------------------------*
         MACRO
&LABEL   KWDEFINE &NAME=,&ROUTINE=NOOP
&NULLBL  SETC  'KW&SYSNDX'
&NULLBL  DC    OC'&NAME'
&LABEL   DC    A(L'&NULLBL)      Length of keyword
         DC    CL8'&NAME'        Keyword
         DC    A(&ROUTINE)       Processing routine
         MEND
         TITLE  'SAPI  - MONITOR CONTROL ROUTINE'
*-------------------------------------------------------------*
*        There is really very little to do in this routine:   *
*          - Initialize the LDA                               *
*          - Set the ESTAE                                    *
*       =SapiSapiSapi=                                        *
*-------------------------------------------------------------*
SAPI     CSECT O
         SAVE  (14,12),,SAPIDRVR_&SYSDATE._&SYSTIME
         LR    R10,R15           Set R10 and R9 as base regs
         LA    R9,2048(,R10)
         LA    R9,2048(,R9)
         USING SAPI,R10,R9       Tell the assembler
         LR    R6,R1             Save R1 for the time being
*-------------------------------------------------------------*
*        Getmain the workarea below the 16M line...           *
*-------------------------------------------------------------*
         LA    R5,LDASIZE        Get the work area size
         GETMAIN RU,                                                  +
               LV=(5),           ... length                           +
               BNDRY=PAGE,                                            +
               LOC=BELOW
         LR    R4,R1             Save the address for the time
         L     R14,@LDASTRT      Get local data constant start
         LA    R15,LDASIZE          and the length
         MVCL  R4,R14            Copy into the storage obtained
         ST    R13,4(R1)         Chain things together
         ST    R1,8(R13)            so we can get out later.
         LR    R13,R1            Set R13
         USING LDASTRT,R13          and tell the assembler
*-------------------------------------------------------------*
*        Check for TSO mode                                   *
*-------------------------------------------------------------*
         L     R1,PSAAOLD-PSA    Get ASCB address
         ICM   R0,15,ASCBTSB-ASCB(R1) Get TSB address
         BC    NZERO,FOREGR      Running in foreground
         OI    LDAFLG2,FLGBACKG  Running in background
FOREGR   DS    0H
*-------------------------------------------------------------*
*        Finish initializing and set the ESTAE                *
*          - Save all the landmarks needed for recovery       *
*            and then issue the ESTAE.                        *
*-------------------------------------------------------------*
         ST    R6,@@PARM         Save the parmlist pointer
         ST    R10,BASEREG       Save the base register address
         ST    R9,BASEREG2       Save the base register address
         ST    R13,@LDA          Save the LDA address
         LA    R15,CLEANUPR      Get addr of ESTAE retry
         ST    R15,@RETRY           and save it.
```

```
         LA    R15,CLEANUP         Get addr of cleanup routine
         ST    R15,@RETRYP            and save it.
*
         L     RO,=A(ESTAEXIT)     Get ESTAE rtn address
         LA    R15,ESTAPARM        Get ESTAE parm address
         ESTAE (0),                Establish ESTAE                   +
               PARAM=(15),                                         +
               MF=(E,ESTAE)
         LTR   R15,R15             Was it successful?
         BC    NZERO,NONET         Nope - too bad, so sad
         OI    LDAFLG1,FLGESTAE    Show ESTAE established
         B     NET_OK
*-------------------------------------------------------------*
*        ESTAE failure                                        *
*-------------------------------------------------------------*
NONET    DS    OH
         BAL   R14,MSG2CLR
         MVC   MSG2ID,=CL8'SAPI901I'
         MVC   MSG2TEXT(28),=c128'*ESTAE failed - Terminating'
         BAL   R14,MSG2OUT
         B     CLEANUP             Go cleanup
*-------------------------------------------------------------*
*        ESTAE set OK....continue processing                  *
*        Get the name of the JES to ask work of.              *
*-------------------------------------------------------------*
NET_OK   DS    OH
         L     R1,PSATOLD-PSA      Get our TCB address
         L     R1,TCBJSCB-TCB(,R1) Get JSCB from TCB
         L     R1,JSCBSSIB-IEZJSCB(,R1) get address of ssib
         ST    R1,@SSIB            Save address of SSIB
         MVC   LDAJES,SSIBSSNM-SSIB(R1) Set the sub-system name
*-------------------------------------------------------------*
*        Set up the operator communication stuff              *
*-------------------------------------------------------------*
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLOFF,SETSTAX      No - Issue STAX
         LA    R2,COMMADDR         Address answer area for extract
         EXTRACT (R2),FIELDS=COMM,MF=(E,EXTRACT)  Get CSCB address
         L     R2,COMMADDR         Get the COMM ptr
         MVC   @COMECB,COMECBPT-COMLIST(R2) ECB address in list
         B     NOSTAX
*-------------------------------------------------------------*
*        Set up STAX for foreground                           *
*-------------------------------------------------------------*
SETSTAX  DS    OH
         STAX  AE,USADDR=R13,REPLACE=NO,DEFER=NO,OBUF=(OB,6),          X
               TOPLEVL=YES,MF=(E,STAXL)
         OI    LDAFLG2,FLGSTAX
         LA    RO,COMECB           Point at communication ECB
         ST    RO,@COMECB          Se t address into ECB list
NOSTAX   DS    OH
         OI    @COMECB,X'80'       Set end of list marker
         LA    R3,ZEROES           Get dummy ECB
         ST    R3,@WTRECB             and set this ECB
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLOFF,COMON        No - Forget the CIB
         ICM   R3,B'1111',COMCIBPT-COMLIST(R2) Get 1ST CIB if any
         BC    ZERO,NSTCIB         Bin, skip
         SR    R1,R1               Clear R1
         ICM   R1,B'0011',CIBDATLN-CIBNEXT(R3) Get data length if any
         BC    ZERO,STNOPRM        NO parameter given
         STH   R1,COMMLEN          Save the length
         B     GOTONE
*-------------------------------------------------------------*
*        Prompt TSO user for initial selection criteria       *
*-------------------------------------------------------------*
COMON    DS    OH
         MVI   COMMBUFF,X'40'      Clear the buffer
         MVC   COMMBUFF+1(L'COMMBUFF-1),COMMBUFF
         LA    R1,=CL6'SAPI?'
         LA    RO,6
         TPUT  (1),(0)
         LA    R1,COMMBUFF         Get initial data
         LA    RO,L'COMMBUFF
         TGET  (1),(0)
         LTR   R15,R15             OK?
         BNZ   COMON               Nope - again
         CLC   COMMBUFF(16),BLANKS Any input?
         BC    NE,SOMEIP           Yes - use it
         SLR   R1,R1               Set no input
SOMEIP   DS    OH
         STH   R1,COMMLEN          Save the length
         LTR   R1,R1               Any input?
         BZ    SSOBINI             None - Skip KW processing
GOTONE   DS    OH
         OI    LDAFLG2,FLGINIIP    Set initial input
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLOFF,SSOBINI      No - Forget the CIB
         BCTR  R1,0                1 less for EX
         EX    R1,*+4              do the move
         MVC   COMMBUFF(*-*),CIBDATA-CIBNEXT(R3) move the initial data
STNOPRM  DS    OH                  CONTINUE
         QEDIT ORIGIN=COMCIBPT-COMLIST(R2),                          +
               BLOCK=(R3)          Delete START CIB
NSTCIB   DS    OH                  CONTINUE
         QEDIT ORIGIN=COMCIBPT-COMLIST(R2),                          +
               CIBCTR=1            Allow more CIBs to follow
*-------------------------------------------------------------*
*        Initialize the SSOB.                                 *
*-------------------------------------------------------------*
SSOBINI  DS    OH                  CONTINUE
         LA    R2,LDASSOB          Get address of SSOB
         O     R2,=X'80000000'     Set hob
         ST    R2,@LDASSOB         Save the address of ssob
         USING SSOBEGIN,R2         Tell the assembler
         MVC   SSOBID,=CL4'SSOB'   Set the eyecather
         LA    R1,SSOBHSIZ         Get the length
         STH   R1,SSOBLEN             and set it.
         LA    R1,SSOBSOU2         Get the function code (SSI 79)
         STH   R1,SSOBFUNC            and set it.
         LA    R1,LDASSS2          Get address of SSS2
         ST    R1,SSOBINDV            and set it.
         DROP  R2                  SSOBEGIN
*-------------------------------------------------------------*
*        Initialize the SSS2.                                 *
*                                                             *
*          Initial SSS2 setting:                              *
*            - SSS2 version number                            *
*            - SSS2 eyecatcher                                *
*            - GET/PUT                                        *
*-------------------------------------------------------------*
         LA    R2,LDASSS2          Get address of SSS2
         USING SSS2,R2             Tell the assembler
         LA    R1,SSS2SIZE         Get extension length
         STH   R1,SSS2LEN             and set it
         LA    R1,SAPIECB          Get SAPI Task Wait For Work
*                                   ECB
         ST    R1,SSS2ECBP             and set it
         MVI   SSS2VER,SSS2CVER    Set current version number
         MVC   SSS2EYE,=CL4'SSS2'  Set SSS2 eyecatcher
         MVI   SSS2TYPE,SSS2PUGE   Set as PUT/GET type request
         DROP  R2                  SSS2
*-------------------------------------------------------------*
*        Initialize the S99RB                                 *
*-------------------------------------------------------------*
         LA    R2,LDAS99RB         Get address of S99RB
         O     R2,=X'80000000'     Set hob
         ST    R2,@LDAS99          Save address
         USING S99RB,R2
         LA    R1,S99RBEND-S99RB   Get the length
         STC   R1,S99RBLN             and set it.
         LA    R1,LDAS99RX         Set S99RBX..
         ST    R1,S99S99X          .address
         DROP  R2                  S99RB
*-------------------------------------------------------------*
*        Chain the text unit pointers together.               *
*-------------------------------------------------------------*
         LA    R1,LDADSNAM
         ST    R1,LDATXTA1
         LA    R1,LDASSNAM
         ST    R1,LDATXTA2
         LA    R1,LDADDNAL
         ST    R1,LDATXTA3
*-------------------------------------------------------------*
*        Now the unallocation units
*-------------------------------------------------------------*
         LA    R1,LDADDNUN
         O     R1,=X'80000000'
         ST    R1,LDATXTU1
         BAL   R14,MSG2CLR
         MVC   MSG2ID,=CL8'SAPI004I'
         MVC   MSG2TEXT(28),=CL28' Finished initialization...'
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        Set the default SSS2 selection information.          *
*                                                             *
*        The idea is to issue a general SAPI GET request      *
*        for a dataset with characteristics in the            *
*        following categories:                                *
*                                                             *
*            - DEST=SAPI                                      *
*            - CLASS=A                                        *
*            - Job Name=SAPI*                                 *
*            - Job range=5 to 100                             *
*            - AGE = 0                                        *
*            - MIN LINE = 0                                   *
*            - MAX LINE = 1000                                *
*            - MIN PAGE = 0                                   *
*            - MAX PAGE = 100                                 *
*                                                             *
*        The above is the default selection criteria. Via     *
*        operator commands, this selection criteria can be    *
*        modified. If processing is a result of a POST        *
*        from an operator command, the operator command       *
*        selection criteria will be used.                     *
*-------------------------------------------------------------*
         SLR   R8,R8
         ICM   R8,3,COMMLEN        Any initial input?
         BC    NZERO,GETJOB        Yes - Continue
         BAL   R8,SETDFALT         Go set default selection info
*-------------------------------------------------------------*
*                                                             *
*        This is the main processing loop...                  *
*                                                             *
*-------------------------------------------------------------*
*                                                             *
```

```
*        Routine OPERCHK is given control to determine       *
*        the type of SAPI request to be made. The default    *
*        is GET/PUT. Via an operator command, a GET/PUT,     *
*        a COUNT or a BULK request can be made with all      *
*        the respective trimmings.                           *
*                                                            *
*------------------------------------------------------------*
GETJOB   DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2ID,=CL8'SAPIO05I'
         MVC   MSG2TEXT(24),=CL24' Main look for work loop'
         BAL   R14,MSG2OUT
         BAL   R8,OPERCHK          Check to see if operator
         B     CLEANUP          +0 is quit
*                                 +4 is continue
         LA    R2,LDASSS2          Get address of SSS2
         USING SSS2,R2             Tell the assembler
         CLI   SSS2TYPE,SSS2COUN   Is COUNT request made?
         BC    EQ,GETCNT           Yep, go procee it
         CLI   SSS2TYPE,SSS2BULK   Is BULK request made?
         BC    EQ,BLKMOD           Yep, go procee it
         LA    R15,LDATKOUT        Get token output area
         ST    R15,SSS2SECT        Save in SSS2
         DROP  R2                  SSS2
************************************************************
*                                                          *
*              GET/PUT Processing                          *
*                                                          *
*----------------------------------------------------------*
*                                                          *
*        If nothing is available go wait for work or for   *
*        the operator to shut us down.                     *
*                                                          *
*        If work is available, repeatedly ask for all the  *
*        datasets belonging to the job until none remain.  *
*        After all datasets for the job have been processed,*
*        delete the job, go look for more work to do.      *
*----------------------------------------------------------*
*----------------------------------------------------------*
*        Ask mom for work.                                 *
*----------------------------------------------------------*
         BAL   R8,ASKJES           Ask mom for a job
         B     CLEANUP          +0 quit
*                                 +4 OK, continue
         LA    R2,LDASSOB          Get the SSOB
         USING SSOBEGIN,R2         Tell the assembler
*----------------------------------------------------------*
*        Print out SSOB return code....                    *
*----------------------------------------------------------*
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(28),=CL28' GETJOB SSOBRETN return: '
         L     R1,SSOBRETN         Get the subsytem return code
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+25(8),0(R1)
         BAL   R14,MSG2OUT
*----------------------------------------------------------*
*        Print out SSS2 reason code....                    *
*----------------------------------------------------------*
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(28),=CL28' GETJOB SSS2REAS return: '
         LA    R14,LDASSS2         Get the SSS2 address
         USING SSS2,R14            Tell the assembler
         SR    R1,R1               Set to zeroes
         IC    R1,SSS2REAS         Get the reason code
         DROP  R14                 SSS2
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+25(8),0(R1)
         BAL   R14,MSG2OUT
*----------------------------------------------------------*
*        Check SSOB return code for type of processing to  *
*        occur next....                                    *
*----------------------------------------------------------*
         L     R1,SSOBRETN         Get the subsytem return code
         C     R1,=A(SSS2RTOK)     Anything to do?
         BC    EQ,GOTJOB           BIY, Just do it
         C     R1,=A(SSS2EODS)     End of datasets?
         BC    EQ,WAITWORK         Nothing to do, so wait
         C     R1,=A(SSS2LERR)     Was is a logical error?
         BC    EQ,LOGICERR         Nothing to do, so wait
         C     R1,=A(SSS2BDIS)     Was is a disposition error?
         BC    EQ,DISPERR          Nothing to do, so wait
         DROP  R2                  SSOBEGIN
*----------------------------------------------------------*
*        Unexpected SAPI return code....wait               *
*----------------------------------------------------------*
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(36),=CL36'*Unexpected General SAPI request rc:'
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+38(8),0(R1)
         MVC   MSG2TEXT+42(12),=CL12' Waiting...'
         BAL   R14,MSG2OUT
         B     WAITWORK            Wait for a operator command
*----------------------------------------------------------*
*        Logical error processing                          *
*----------------------------------------------------------*
LOGICERR DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(34),=CL34'*Logical Error....reason code: '
         LA    R14,LDASSS2         Get the SSS2 address
         USING SSS2,R14            Tell the assembler
         SR    R1,R1               Clear to zeroes
         IC    R1,SSS2REAS         Get reason code
         DROP  R14                 SSS2
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+31(8),0(R1)
         BAL   R14,MSG2OUT
         B     ERRCONT             Continue processing
*----------------------------------------------------------*
*        Disposition error processing                      *
*----------------------------------------------------------*
DISPERR  DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(32),=CL32'*Disp Error....reason code: '
         LA    R14,LDASSS2         Get the SSS2 address
         USING SSS2,R14            Tell the assembler
         SR    R1,R1               Clear to zeroes
         IC    R1,SSS2REAS         Get reason code
         DROP  R14                 SSS2
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+28(8),0(R1)
         BAL   R14,MSG2OUT
*----------------------------------------------------------*
*        If the error was due to a bad type, the type will *
*        be 'reset' on the next IEFSSREQ request. So there *
*        is nothing to do here.                            *
*        If the error was due to a bad length, the length  *
*        needs to be reset since it is set only once, during*
*        initialization.                                   *
*----------------------------------------------------------*
ERRCONT  DS    0H
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         SR    R1,R1
         IC    R1,SSS2REAS
         C     R1,=A(SSS2RLEN)     Was it bad SSS2 length?
         BC    NE,WAITWORK         No, wait for a operator cmd
         LA    R1,SSS2SIZE         Get SSS2 length
         STH   R1,SSS2LEN          Get SSS2 length
         B     WAITWORK            Wait for a operator command
         DROP  R2                  SSS2
*----------------------------------------------------------*
*        Job returned from JES3......                      *
*                                                          *
*        Now remember the jobid for later, enter the       *
*        allocation/unallocation loop, when finished with  *
*        this job, dispose of it and get another.          *
*                                                          *
*----------------------------------------------------------*
GOTJOB   DS    0H
         TM    SAPIFLG1,SAPIABND   Are we to force an abend?
         BC    ALLON,FRCABEND      Go do it
         LA    R2,LDASSS2          Get the SSS2
         USING SSS2,R2             Tell the assembler
         MVC   LDACLSR,SSS2CLAR    Save the ds class
         MVC   LDADESTR,SSS2DESR   Save the ds dest
         MVC   LDAJOBIR,SSS2JBIR   Save the jobid
         MVC   LDAJOBNR,SSS2JOBR   Save the jobname
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(16),=CL16' Selected job: '
         MVC   MSG2TEXT+15(8),LDAJOBNR
         MVI   MSG2TEXT+23,C'('
         MVC   MSG2TEXT+24(8),LDAJOBIR
         MVI   MSG2TEXT+32,C')'
         BAL   R14,MSG2OUT
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(16),=CL16' Prgmr Name: '
         MVC   MSG2TEXT+13(20),SSS2PNAM
         BAL   R14,MSG2OUT
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(16),=CL16' Notify Userid: '
         MVC   MSG2TEXT+16(8),SSS2NOTU
         BAL   R14,MSG2OUT
         ICM   R15,B'1111',SSS2ACCT   Get accounting info
         BC    ZERO,SWBRETRV
         SR    R1,R1               Set to zeroes
         IC    R1,0(,R15)          Get number of acct fields
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(36),=CL36' Number of accounting fields is: '
         BAL   R15,CVTHEX          Pass count in R1
         MVC   MSG2TEXT+33(8),0(R1)
         BAL   R14,MSG2OUT
         DROP  R2                  SSS2
*----------------------------------------------------------*
*        Use SJF RETRIEVE to return keyword data (in text  *
*        unit form) for the current output SWB chain.      *
*                                                          *
*        The output parameter area contains the following  *
*        data.                                             *
*                                                          *
*        (1) ORIGINAL KEYWORD LIST - FOR EACH KEYWORD THAT *
*            APPEARS IN THE OUTPUT SWB, THE ADDRESS FIELD  *
*            POINTS TO A LIST OF TEXT UNIT POINTERS.       *
*                                                          *
*        (2) TEXT UNIT POINTERS - THERE IS A TEXT UNIT     *
*            POINTER FOR EACH POSITIONAL PARAMETER.  EACH  *
*            ONE POINTS TO THE TEXT UNIT DESCRIBING THE    *
```

```
*           PARAMETER.  THE LAST TEXT UNIT FOR A KEYWORD    *
*           HAS THE HIGH ORDER BIT ON TO INDICATE THE END   *
*           OF THE LIST.                                    *
*                                                           *
*       (3) TEXT UNIT - EACH TEXT UNIT DESCRIBES A          *
*           POSITIONAL PARAMETER.  THE DATA IS IN           *
*           KEY/COUNT/LENGTH/DATA FORMAT.  THE COUNT FIELD  *
*           CONTAINS THE NUMBER OF LENGTH/DATA PAIRS WHICH  *
*           FOLLOW.  THERE IS A LENGTH/DATA PAIR FOR        *
*           EACH PARAMETER FIELD.                           *
*                                                           *
*                                                           *
*       FOR EXAMPLE:                                        *
*                                                           *
*           MODIFY=(MODNAME,TRC)                            *
*                                                           *
*           THE MODIFY KEYWORD CONTAINS TWO POSITIONAL      *
*           PARAMETERS AND IS REPRESENTED BY TWO TEXT UNIT  *
*           POINTERS.                                       *
*                                                           *
*           CHARS=(TBLNAME,TBLNAME,...)                     *
*                                                           *
*           THE CHARS KEYWORD CONTAINS A SINGLE POSITIONAL  *
*           PARAMETER AND IS REPRESENTED BY ONE TEXT UNIT   *
*           POINTER.  THE TEXT UNIT CONTAINS A LENGTH/DATA  *
*           PAIR FOR EACH OCCURRENCE OF "TBLNAME".          *
*                                                           *
*                                                           *
*            SJF                  OUTPUT                  *
*          RETRIEVE               PARAMETER               *
*          PARMLIST                 AREA                  *
*         ----------            -----------               *
*         |SJREKWDL|----------->| KEYWORD |               *
*         |        |            |LIST PTR.|-----          *
*         |SJREAREA|---         | KEYWORD |    |          *
*         |        |  |         |LIST PTR.|--- |          *
*         |        |  |         |   .     |  | |          *
*         |        |  |         |   .     |  | |          *
*         |        |  |         |   .     |  | |          *
*         |        |  |         |---------|  | |          *
*         |        |  ---------->|TEXT UNIT|<----          *
*         ----------       -----| POINTERS|               *
*                          |    |---------|               *
*                          ---->|  TEXT   |               *
*                               |  UNITS  |               *
*                               -----------               *
*                                                         *
*                                                         *
*          TEXT UNIT POINTERS (FOR ONE KEYWORD)           *
*         ---------------------------------               *
*         |TEXT UNIT|TEXT UNIT| ...|TEXT UNIT|<--HIGH ORDER  *
*         | POINTER | POINTER |    | POINTER |   BIT ON MARKS*
*         ---------------------------------   LAST PTR.   *
*              |                                          *
*         ------                                          *
*         |                                               *
*         V       TEXT UNIT                               *
*         ---------------------------------------         *
*         |KEY|COUNT|LENGTH|DATA|...|LENGTH|DATA|        *
*         ---------------------------------------         *
*                                                         *
*      NOTE: This code is from IATIIOS, which does more   *
*            that retrieve just one text unit.            *
*-------------------------------------------------------------*
SWBRETRV DS    OH
         LA    R2,LDASSS2          Get the SSS2
         USING SSS2,R2             Tell the assembler
         TM    SSS2MSC1,SSS2FSWB   Did we ask for SWBs?
         BC    ALLOFF,GOTDS        No, skip SWB processing
         DROP  R2                  SSS2
         LA    R15,MYPLLST         Addr the SJF Retrieve Block
         USING SJREP,R15           Addressability
         XC    SJREP(SJRELGTH),SJREP  Clear the block
         MVC   SJREID,=C'SJRE'     Eyecatcher movement
         MVI   SJREVERS,SJRECVER   Version ID movement
         LA    R1,SJRELGTH         Put length of block...
         STH   R1,SJRELEN          ..into the parm list
         LA    R2,LDASSS2          Get the SSS2
         USING SSS2,R2             Tell the assembler
         MVC   SJRETOKN,SSS2SWBT   Token from SAPI movement
         DROP  R2                  SSS2
         LA    R1,MYAREA           Point to the answer area and...
         ST    R1,SJREAREA         ..put into plist
         LA    R1,L'MYAREA         Obtain the size of answer area...
         STH   R1,SJRESIZE         ..and put into plist
         LA    R1,3                Three keywords for time being...
         STH   R1,SJRENKWD         ...and store in plist
         LA    R1,MYKEYLST         Point to my keylist and...
         ST    R1,SJREKWDL         ...save in plist
         ST    R15,MYPLADR         Save addr of plist in pointer
         DROP  R15                 SJREP
         L     R14,=A(MSKZMS)      Goto Key 0 and supervisor state
         BALR  R14,R14
         OI    LDAFLG1,FLGSUP      Indicate our state
         LA    R1,MYPLADR          Point R1 to pointer of plist
         SJFREQ REQUEST=RETRIEVE   Let 'er rip!
         LR    R5,R15              Save return code
         L     R14,=A(MSKNMP)      Back to problems state....
         BALR  R14,R14
         NI    LDAFLG1,FF-FLGSUP   Indicate our state
         LTR   R5,R5               Request successful?
         BC    ZERO,SJFROK         Yep, issue SWB message
*-------------------------------------------------------------*
*        Print out SJFREQ reason code.                        *
*-------------------------------------------------------------*
         LA    R2,MYPLLST          Point to my parmlist
         USING SJREP,R2            Addressability
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(21),=CL21'*SJFREQ return code: '
         LR    R1,R5               Get the SJFREQ reason code
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+21(8),0(R1)
         BAL   R14,MSG2OUT
         MVC   MSG2TEXT(21),=CL21' SJFREQ reason code: '
         L     R1,SJREREAS         Get the SJFREQ reason code
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+21(8),0(R1)
         BAL   R14,MSG2OUT
         DROP  R2                  SJREP
         B     SWBTURET            Try SWBTUREQ
*-------------------------------------------------------------*
*        SJFREQ OK!!                                          *
*-------------------------------------------------------------*
SJFROK   DS    OH
         LA    R3,MYKEYLST         Point to my keylist
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(17),=CL17' 1st SJFREQ SWB: '
         MVC   MSG2TEXT+17(8),0(R3)
         MVC   MSG2TEXT+25(5),=CL5' is: '
         ICM   R3,B'1111',8(R3)    Get ptr to ptr list (SJRELIST DSECT)
         BC    ZERO,GOTDS          Branch if none there
         ICM   R3,B'1111',0(R3)    Get ptr to FORMS SWB
         BC    ZERO,GOTDS          Branch if none there
         MVC   MSG2TEXT+30(8),6(R3)  Move is SWB FORM data (S99TUNIT)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*                                                             *
*      This routine will retrieve the SWB FCB.                *
*                                                             *
*      The SWBTU retrieve function is excersised to ensure    *
*      the basics of this interface is functional. This is    *
*      the reason for obtaining only the FCB. If the FCB      *
*      retrieve works, retrieval of all other keywords        *
*      will also work.                                        *
*                                                             *
*      SWBTUREQ RETRIEVE returns a set of SWBTUs in the       *
*      format:                                                *
*                                                             *
*      key,#parameters,length of parameter(i),parameter value *
*                                                             *
*      Hence for either of these OUTPUT JCL statements        *
*                                                             *
*      //OUT1 OUTPUT TITLE='CHESAPIB',NAME='T.J. WATSON'      *
*      //OUT2 OUTPUT NAME='T.J. WATSON',TITLE='CHESAPIB'      *
*                                                             *
*      the following SWBTU set is returned:                   *
*                                                             *
*   002A 0001 0008 C3C8C5..E6C2 002D 0001 000B E34BD1...D6D5  *
*    A                           A                            *
*   title key                   name key                      *
*                                                             *
* NOTE: The order in which the SWBTUs are returned depends    *
*       upon the order in which they are specified in the     *
*       keylist passed to SWBTUREQ (see SAPIKEY).             *
*                                                             *
*       The SWB table storage obtained by CHESAPIB is freed   *
*       by the caller during end-of-job cleanup.              *
*                                                             *
*       NOTE: This code is from IATNTSR, which does more      *
*             that retrieve just one text unit.               *
*-------------------------------------------------------------*
*-------------------------------------------------------------*
*        Get size of working storage needed.  SAPISIZ         *
*        includes all storage needed except what is needed    *
*        for the SJF working storage area, since this value   *
*        is subject to change.  The working storage will be   *
*        mapped by SAPIDSEC and will look like this:          *
*                                                             *
*          Ptr to SJF                                         *
*            parm lst-> _____________________                 *
*                      |     SAPISJFA          |              *
*                      |_______________________|              *
*          SJTRSBTL --> |                       |              *
*            dsect      |     SAPISJFT          |              *
*                       |_______________________|              *
*          SAPIKEY ---> |                       |              *
*            dsect      |     SAPISJFK          |              *
*                       |_______________________|              *
*          SJTRP -----> |                       |              *
*            dsect      |     SAPISJFP          |              *
*                       |_______________________|              *
*          18-fullword  |                       |              *
*            savearea-> |     SAPISJFS          |              *
*                       |_______________________|              *
*          SJF Output   |                       |              *
*            area ----> |     SAPISJFO          |              *
```

```
*                     |                       |          *
*                     |                       |          *
*                     |                       |          *
*                     |                       |          *
*                     |_______________________|          *
*          SJF working  |                       |          *
*            storage -> |     SAPISJFW          |          *
*                       .                       .          *
*                     |        .              |          *
*                       .                       .          *
*                     |_______________________|          *
*                                                        *
*                                                        *
*--------------------------------------------------------*
SWBTURET DS    OH
         LA    R5,SAPISIZ          Get size of storage needed
         A     R5,SJFWKSZ          Add SJF working storage area
         GETMAIN R,                Get storage for SJF               +
               LV=(5),             ... length                        +
               LOC=BELOW
*--------------------------------------------------------*
*        Clear the storage obtained and build the SWBTU  *
*        pointer list.                                   *
*--------------------------------------------------------*
         LR    R3,R1               Address of ptr to parm list
         LR    R0,R1               Address tp clear
         LR    R1,R5               Size to clear
         XR    R15,R15             Pad with zero
         MVCL  R0,R14              Initialize the storage
         USING SAPIDSEC,R3         Establish addressability
*                                    to working storage
*--------------------------------------------------------*
*        Set SWBTUREQ parm list pointer.                 *
*--------------------------------------------------------*
         LA    R1,SAPISJFP         Get addr of SJF parm list
         ST    R1,SAPISJFA         Store address for SWBTUREQ
*--------------------------------------------------------*
*        Set up SWB TU pointer.                          *
*--------------------------------------------------------*
         LA    R1,SAPISJFT         Ptr. to SWBTU address list
         USING SJTRSBTL,R1         Establish addressability
         LA    R4,LDASSS2          Get the SSS2
         USING SSS2,R4             Tell the assembler
         MVC   SJTRSTUP,SSS2SWTU   Set SWBTU pointer
         DROP  R4                  SSS2
         DROP  R1                  SJTRSBTL
*--------------------------------------------------------*
*        Set up retrieve key list for SJF SWBTUREQ       *
*--------------------------------------------------------*
         LA    R1,SAPISJFK         Start of key list
         MVC   0(SAPIKYL,R1),SAPIKEY Init key list for SJF
*--------------------------------------------------------*
*        Set up SWBTUREQ parameter list                  *
*--------------------------------------------------------*
         LA    R4,SAPISJFP         Get addr for SJF parm list
         USING SJTRP,R4
         ST    R1,SJTRKIDL         Key list address
         LA    R0,SAPISJFT         Point to SWBTU ptr list
         ST    R0,SJTRSWBA         SWBTU ptr list address
         LA    R0,1                Get number of SWB TUs
         STH   R0,SJTRSWBN         Number of SWB TU addresses
         LA    R0,SAPISJKN         Get num keys to retrieve
         STH   R0,SJTRKIDN         Save it in the parm list
         LA    R0,SJTRCVER         Get version number
         STC   R0,SJTRVERS         Save version in parm list
         MVC   SJTRID,SAPISJID     Control block identifier
         LA    R0,SJTRLGTH         Get parameter list length
         STH   R0,SJTRLEN          Save length in parm list
         LA    R1,SAPISJFW         Address of working storage
         ST    R1,SJTRSTOR         Save working storage addr
         L     R1,SJFWKSZ          Get SJF working storage size
         STH   R1,SJTRSTSZ         Size of working storage
         LA    R1,SAPISJFO         Address of output area
         ST    R1,SJTRAREA         Set addr in parm list
         LA    R0,SAPISJOS         Size of output area
         STH   R0,SJTRSIZE         Set size in parm list
         DROP  R4                  SJF PARAMETER LIST
*--------------------------------------------------------*
*        R1  = address of the SJF parameter list.        *
*        R13 = address of save area for SJF (REQUIRED).  *
*                                                        *
*        Invoke the SWBTUREQ service. R1 must point to a *
*        fullword containing the address of the parm list.*
*                                                        *
*--------------------------------------------------------*
SAPITREQ DS    OH
         L     R14,=A(MSKZMS)
         BALR  R14,R14
         OI    LDAFLG1,FLGSUP      Indicate our state
         LR    R4,R13              Save module work area addr
         LA    R13,SAPISJFS        Set save area for SJF
         LA    R1,SAPISJFA         Set up register 1
         SWBTUREQ REQUEST=RETRIEVE Retrieve the keywords
         LR    R5,R15              Save return code
         LR    R13,R4              Restore module work area addr
         L     R14,=A(MSKNMP)
         BALR  R14,R14
         NI    LDAFLG1,FF-FLGSUP   Indicate our state
*--------------------------------------------------------*
*        If the return code from the SWBTUREQ indicates that *
*        a storage error exists, and the reason code         *
*        indicates that the working storage size was not     *
*        sufficient, additional working storage is obtained. *
*        The amount of storage required is found in the SJF  *
*        parameter list, field SJTRWKSZ.                     *
*                                                            *
*        The new address and size are saved in the SJF       *
*        parameter list and the working storage size is      *
*        updated for follow on SJF processing. The SWBTUREQ  *
*        is issued again with the proper amount of storage.  *
*------------------------------------------------------------*
         LTR   R5,R5               Request successful?
         BC    ZERO,SAPIOTB        Yes, continue
         LA    R4,SAPISJFP         SJF parm list address
         USING SJTRP,R4            SJF parm list addressability
         LA    R0,SJTRCSIZ         Storage size error or no
*                                    requested items found ret code
         CR    R5,R0               Is that what we received?
         BC    NE,SAPIFREE         NO, keywords not available
         LA    R0,SJTRWSZR         Working storage insufficient
         C     R0,SJTRREAS         More working stg needed?
         BC    NE,SAPIFREE         NO, item not found
         LH    R0,SJTRWKSZ         Working stg size required
         STH   R0,SJTRSTSZ         Set size for SJF
         GETMAIN R,                Get additional working stg        +
               LV=(0),             ... length                        +
               LOC=ANY
         ST    R1,SJTRSTOR         Set new working stg address
         B     SAPITREQ            Try SWBTUREQ again
         DROP  R4                  SJF parameter list
*------------------------------------------------------------*
*      FCB                                                   *
*                                                            *
*      Find the FCB text unit pointer address in the         *
*      keyword list entry corresponding to FCB. The          *
*      FCB text unit, if found, will be the only one         *
*      returned.                                             *
*                                                            *
*      If it is not found or zero, there is no FCB value.    *
*                                                            *
*      Save the OUTBIN ID in the CHESAPIB parameter list     *
*      (IATYCKSB) and set a flag to indicate an ID has       *
*      been found.                                           *
*------------------------------------------------------------*
SAPIOTB  DS    OH
         LA    R5,SAPISJFP       Set up addressability...
         USING SJTRP,R5            ... to the parm list
         L     R1,SJTRKIDL       Set up addressability...
         USING SJTRKEYL,R1           ... to the keylist
         ICM   R4,B'1111',SJTRTPAD Get the first tu pointer
         BC    ZERO,SAPINOTU     None, skip FCB
         DROP  R1                SJTRKEYL
         USING S99TUNIT,R4       Text unit addressability
         CLC   S99TUNUM,HEXZERO  Is there any count
         BC    EQ,SAPINOCT       No, skip FCB
         CLC   S99TULNG,HEXZERO  Does parm have a length?
         BC    EQ,SAPINOLN       No, skip FCB
         LA    R7,S99TUPAR       Get address of FCB text unit
         DROP  R5                SJTRP
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(18),=CL18' SWBTUREQ FCB is: '
         MVC   MSG2TEXT+18(8),0(R7)  Move in SWB FCB data
         BAL   R14,MSG2OUT
*------------------------------------------------------------*
*        Free storage used for SWBTUREQ.                     *
*                                                            *
*        If additional working storage was required by the   *
*        SWBTUREQ services, that storage must be freed.      *
*        The address and size of the additional storage is   *
*        obtained from the SJF parameter area. If no         *
*        additional working storage was obtained then only   *
*        the block of storage used for the SWBTUREQ must be  *
*        freed.                                              *
*------------------------------------------------------------*
SAPIFREE DS    OH
         LA    R15,SAPISJFW      Get initial working stg addr
         LA    R4,SAPISJFP       Get SJF parm list address
         USING SJTRP,R4          SJF parameter list
         L     R1,SJTRSTOR       Working storage address
*                                  last used by SJF
         CR    R1,R15            Additional stg getmained?
         BC    EQ,SAPINWKS       NO, free SJF storage
         LH    R0,SJTRSTSZ       Size of working storage
         DROP  R4                SJF parameter list
         FREEMAIN R,             Free additional working stg       +
               LV=(R0),          ... length                        +
               A=(R1)
SAPINWKS DS    OH
         LA    R5,SAPISIZ        Get size of storage needed
         A     R5,SJFWKSZ        Add SJF working storage area
         FREEMAIN R,             Free SJF storage                  +
               LV=(R5),          ... length                        +
               A=(R3)
         B     GOTDS
         DROP  R3                SAPIDSEC
*------------------------------------------------------------*
*        SWBTUREQ returned zero pointer.                     *
```

```
*-------------------------------------------------------------*
SAPINOTU DS    OH
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(26),=CL26'*SWBTUREQ returned 0 ptr! '
         BAL   R14,MSG2OUT
         B     SAPIFREE            Return storage
*-------------------------------------------------------------*
*        SWBTUREQ returned zero count.                        *
*-------------------------------------------------------------*
SAPINOCT DS    OH
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(28),=CL28'*SWBTUREQ returned 0 count! '
         BAL   R14,MSG2OUT
         B     SAPIFREE            Return storage
*-------------------------------------------------------------*
*        SWBTUREQ returned zero parm length.                  *
*-------------------------------------------------------------*
SAPINOLN DS    OH
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(30),=CL30'*SWBTUREQ returned 0 parm len! '
         BAL   R14,MSG2OUT
         B     SAPIFREE            Return storage
*-------------------------------------------------------------*
*        SWBTUREQ DATA CONSTANTS.                             *
*                                                             *
*        The size of the output area must be large enough to *
*        hold a 2-byte length field for each keyword and the *
*        largest possible keyword value plus a 4 byte prefix *
*        area to contain a text unit pointer.                 *
*                                                             *
*        The calculation would be as follows for the          *
*        keywords being retrieved:                            *
*                                                             *
*        (SAPINUMK) * (60 + L'S99TULNG + L'S99TUPAR)          *
*            :        :   :            :                     *
*            V        :   :            :                     *
*        Number of keys to retrieve:   :                      *
*                      :   :            :                     *
*                      V   :            :                     *
*        All keys retrieved have same max size of 60          *
*            (except OUTBIN)                                  *
*                          :            :                     *
*                          V            :                     *
*                  Length value for each keyword             *
*                                       :                     *
*                                       V                     *
*                                        Text unit pointer    *
*-------------------------------------------------------------*
CKSMAXKW EQU   (60+L'S99TUPTR+L'S99TULNG)  Maximum SJF kwd size
SAPISJOS EQU   (SAPINUMK)*(CKSMAXKW)  SJF output area size
*                                     Output area size for SJF
SAPISJKN EQU   SAPIKYL/SJTRKLEN       Number of actual keys for
*                                       SJF SWBTUREQ service
SAPINUMK EQU   SAPIKYL/SJTRKLEN       Number of keys including 4
*                                       subparameters of address
*                                       keyword used to compute
*                                       size of output area
SJFWKSZ  DC    F'500'                 A guess of what it will be
SAPISJID DC    A(SJTRCID)             SJF SWBTUREQ parm list id
SAPIBTID DC    CL4'SWBT'              SWB table identifier
       SPACE 1
*-------------------------------------------------------------*
*        KEY LIST DEFINITION FOR SWBTUREQ RETRIEVE.           *
*-------------------------------------------------------------*
SAPIKEY  DS    OH
         DC    AL2(DOFCB)             FCB Key
         DC    F'0'
SAPIKYL  EQU   *-SAPIKEY              END OF KEY LIST FOR SJF
*-------------------------------------------------------------*
*        Show dataset to be allocated.                        *
*-------------------------------------------------------------*
GOTDS    DS    OH
         LA    R2,LDASSS2             Get the SSS2
         USING SSS2,R2                Tell the assembler
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(15),=CL15' SSS2DSN info: '
         MVC   MSG2TEXT+15(44),SSS2DSN
         BAL   R14,MSG2OUT
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(15),=CL15' Dataset name: '
         MVC   MSG2TEXT+15(8),SSS2PRCD
         MVI   MSG2TEXT+23,C'.'
         MVC   MSG2TEXT+24(8),SSS2STPD
         MVI   MSG2TEXT+32,C'.'
         MVC   MSG2TEXT+33(8),SSS2DDND
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        Update allocation text units with dsname and         *
*        browse token.                                        *
*-------------------------------------------------------------*
         MVC   LDADSN,SSS2DSN       Save the Data Set name for alloc
         L     R1,SSS2BTOK          Get ds token address            @RFA
         O     R1,=X'80000000'      Set as last alloc text unit     @RFA
         ST    R1,LDATXTA4          Set address                     @RFA
         SR    R6,R6                Clear reg 6
         ICM   R6,B'0011',SSS2MLRL  Get max record length
*                                    .. assumed not to exceed 150 bytes
         LA    R6,5(,R6)            Add some padding
         STH   R6,RECLEN            Save max record length
         DROP  R2                   SSS2
*-------------------------------------------------------------*
*        Time to allocate the returned dataset.               *
*-------------------------------------------------------------*
ALLOCDS  DS    OH
         LA    R2,LDAS99RB          Get SVC99 parm block address
         USING S99RB,R2             Tell the assembler
         MVI   S99VERB,S99VRBAL     Set the allocation verb
         LA    R1,LDATXTAL          Get allocation text units
         ST    R1,S99TXTPP          Save them away
         DROP  R2                   S99RB
         BAL   R8,SVC99             Go do the allocation
         B     CLEANEX          +0 error return
         B     OPENDS           +4 OK, continue
*
CLEANEX  DS    OH
         TM    LDAFLG1,FLGSAPI
         BC    ALLOFF,WAITWORK
         LA    R2,LDASSS2           Get SSS2 address
         USING SSS2,R2
         LA    R0,SSS2INPT          Point to input start field
         LA    R1,L'SSS2INPC        Get length of input field
         SR    R15,R15              Zero length and pad field
         MVCL  R0,R14               Clear input SSS2 section
         MVI   SSS2DSP1,SSS2DKPE    Show keep dataset
         MVI   SSS2MSC1,SSS2CTRL    Indicate this is a control
         DROP  R2                   SSS2
         NI    LDAFLG1,FF-FLGSAPI   Loop control
         BAL   R8,ASKJES            Go ask jes to dispose of job
         B     WAITWORK
         B     WAITWORK
*-------------------------------------------------------------*
*        Do the open, get loop and close to process...        *
*-------------------------------------------------------------*
OPENDS   DS    OH
         OI    LDAFLG1,FLGALLOC     Indicated dataset allocated.
         LA    R2,INDCB             Get the DCB
         USING IHADCB,R2            Tell the assembler
         MVC   DCBDDNAM,LDADDNMA    Get address of DDNAME
         SR    R3,R3                Set to zero
         OPEN  ((R2),(INPUT))       Open the dataset
         OI    LDAFLG1,FLGOPEN      Indicate we have open DS
         TM    DCBOFLGS,DCBOFOPN    Did it work?
         BC    ALLOFF,NOOPEN        No, issue WTO
*-------------------------------------------------------------*
*        Get the SYSOUT records.                              *
*-------------------------------------------------------------*
         LA    R15,CLOSEIT          Get addr of cleanup routine
         ST    R15,@RETRYP          Set retry continuation point
GETDATA  DS    OH                   Loop for reading/displaying
         GET   INDCB                R1==> record after GET
*-------------------------------------------------------------*
*        Process the returned data ....                       *
*-------------------------------------------------------------*
         LA    R3,1(,R3)            Up record count by 1
         CH    R3,=H'10'            Show max..
         BC    GT,GETDATA           .ten records
         BC    NE,GETPUTR           Show data
         MVI   RECTEXT,C'='         Mark out record...
         MVC   RECTEXT+1(L'RECTEXT-1),RECTEXT  ..as 'end of show'
         MVC   RECTEXT+9(13),=CL13' End of Show '
         LA    R0,31
         B     GETPUTD
GETPUTR  DS    OH                   Loop for reading/displaying
         MVI   RECTEXT,C' '         Clear out record...
         MVC   RECTEXT+1(L'RECTEXT-1),RECTEXT  ..for the next one
         LH    R0,DCBLRECL
         LA    R6,150               Max show length 150
         CR    R6,R0                Record length > 150?
         BC    GT,*+4+2             No - show all
         LR    R0,R6
         LR    R6,R0
         BCTR  R6,0
         EX    R6,MOVEIT            Move up to 150 bytes of rec
         L     R1,=A(TRPR)
         EX    R6,TRANSIT
GETPUTD  DS    OH                   Loop for reading/displaying
         STH   R0,RECLEN
         TM    LDAFLG2,FLGBACKG     Running in background?
         BC    ALLON,GETDWTO        Yes - Issue WTO
         TPUT  RECTEXT,(R0)
         B     GETDATA              Go get the next record
GETDWTO  DS    OH
         LA    R6,125               Max WTO length 125
         CR    R6,R0                Record length > 125?
         BC    GT,*+4+4             No - show everything
         STH   R6,RECLEN
         LA    R11,RECLEN           Point to record for output
         WTO   TEXT=(11),ROUTCDE=2,MF=(E,MSG1) Display
         B     GETDATA              Go get the next record
         DROP  R2                   IHADCB
MOVEIT   MVC   RECTEXT(*-*),0(R1)  OBJ OF AN EXECUTE
TRANSIT  TR    RECTEXT(*-*),0(R1)  OBJ OF AN EXECUTE
*-------------------------------------------------------------*
*        OPEN failed....issue WTO.                            *
*-------------------------------------------------------------*
NOOPEN   DS    OH
```

```
         BAL   R14,MSG2CLR
         MVC   MSG2ID,=CL8'SAPI905I'
         MVC   MSG2TEXT(24),=CL24'*Dataset failed to open'
         BAL   R14,MSG2OUT
         NI    LDAFLG1,FF-FLGOPEN  Indicate DS not opened
         B     SKIPCLS
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        Close the dataset.                                   *
*-------------------------------------------------------------*
CLOSEIT  DS    0H
         LA    R15,CLEANUP         Get addr of cleanup routine
         ST    R15,@RETRYP         Set retry continuation point
         LR    R1,R3               Get number of records
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(32),=CL32' Number of records obtained is: '
         BAL   R15,CVTHEX          Pass count in R1
         MVC   MSG2TEXT+32(8),0(R1)
         BAL   R14,MSG2OUT
         LA    R2,INDCB            Get the DCB
         CLOSE ((R2))              Close the dataset
         NI    LDAFLG1,FF-FLGOPEN  Indicate closed DS
*-------------------------------------------------------------*
*        Unallocate the dataset.                              *
*-------------------------------------------------------------*
SKIPCLS  DS    0H
         MVC   LDADDNMU,LDADDNMA   Set the DD name to deallocate
         LA    R2,LDAS99RB         Get SVC99 parm block address
         USING S99RB,R2            Tell the assembler
         MVI   S99VERB,S99VRBUN    Set the deallocation verb
         LA    R1,LDATXTUN         Get deallocation text units
         ST    R1,S99TXTPP         Save them away
         DROP  R2                  S99RB
         BAL   R8,SVC99            do the deallocation
         B     CLEANUP          +0 error return
*                               +4 OK
         NI    LDAFLG1,FF-FLGALLOC indicated nothing allocated
*-------------------------------------------------------------*
*        If the request is to process one dataset, we're      *
*        done.                                                *
*-------------------------------------------------------------*
         TM    SAPIFLG1,SAPIONED   Process only one dataset?
         BC    ALLON,FINLCALL      Yep, issue final call
*-------------------------------------------------------------*
*        Time to get the next dataset.                        *
*-------------------------------------------------------------*
GETDS    DS    0H
         BAL   R8,OPERCHK          Check to see if operator
         B     CLEANUP          +0 Quit
*                               +4 OK, continue
*-------------------------------------------------------------*
*        Ask mom for work.                                    *
*-------------------------------------------------------------*
         BAL   R8,ASKJES           Ask mom for a job
         B     CLEANUP          +0 Quit
*                               +4 OK, continue
         LA    R2,LDASSOB          Get the SSOB
         USING SSOBEGIN,R2         Tell the assembler
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(30),=CL30' Specific SAPI request retcd: '
         L     R1,SSOBRETN         Get the subsytem return code
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+30(8),0(R1)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        Print out SSS2 reason code....                       *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(30),=CL30' Specific SAPI request reacd: '
         LA    R14,LDASSS2         Get the SSS2 address
         USING SSS2,R14            Tell the assembler
         SR    R1,R1               Set to zeroes
         IC    R1,SSS2REAS         Get the reason code
         DROP  R14                 SSS2
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+30(8),0(R1)
         BAL   R14,MSG2OUT
         L     R1,SSOBRETN         Get the subsytem return code
         C     R1,=A(SSS2RTOK)     Anything to do?
         BC    EQ,CKFIRST          BIY, Just do it
         C     R1,=A(SSS2EODS)     End of datasets?
         BC    EQ,JOBEND           BIY, Just do it
*-------------------------------------------------------------*
*        Handle the unexpected return code.                   *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(36),=CL36'*Unexpected specific SAPI req rc:'
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+38(2),6(R1)
         MVC   MSG2TEXT+42(12),=CL12'*Terminating...'
         BAL   R14,MSG2OUT
         B     CLEANUP
         DROP  R2                  SSOBEGIN
*-------------------------------------------------------------*
*        Check to see if this is a new job. If so,            *
*        display the appropriate information. If not a new    *
*        job, determine if the dataset is part of a new       *
*        group. If so, display the appropriate information.   *
*-------------------------------------------------------------*
CKFIRST  DS    0H
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         CLC   LDAJOBIR,SSS2JBIR   Is it same job?
         BC    NE,GOTJOB           No, show job info
         TM    SSS2RET1,SSS2DSF    Is it 1st dataset?
         BC    ALLON,SWBRETRV      Yep, display SWB info
         DROP  R2                  SSS2
         B     GOTDS               Process subsequent dataset
*-------------------------------------------------------------*
*        End of job reached...issue message and go ask        *
*        for another job.                                     *
*-------------------------------------------------------------*
JOBEND   DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2ID,=CL8'SAPIO06I'
         MVC   MSG2TEXT(16),=CL16' Finished job:  '
         MVC   MSG2TEXT+15(8),LDAJOBNR
         MVI   MSG2TEXT+23,C'('
         MVC   MSG2TEXT+24(8),LDAJOBIR
         MVI   MSG2TEXT+32,C')'
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        If the request is to process one job, we're          *
*        done.                                                *
*-------------------------------------------------------------*
         TM    SAPIFLG1,SAPIONEJ   Process only one job?
         BC    ALLON,FINLCALL      Yep, issue final call
         B     WAITWORK            Go wait for op cmd
***
***    Below code is dead code but I carenot to remove it
***    at this time.
***
*-------------------------------------------------------------*
*        If the processing just completed included the        *
*        releasing of datasets and selection includes         *
*        work on the WTR queue, we're done. One possible      *
*        side affect that you don't want to run into (as      *
*        I have) is to select work on the HOLD queue,         *
*        release it to the WTR queue and have your selection  *
*        criteria general enough to select the same group     *
*        of datasets. And now thta they are on the WTR        *
*        queue, you are in loop city. Instead of getting      *
*        fancy with this process, I'll quit here since        *
*        this isn't a product we are going to sell to         *
*        customers.                                           *
*-------------------------------------------------------------*
         LA    R2,LDASSS2          Get the SSS2
         USING SSS2,R2             Tell the assembler
         TM    SSS2DSP1,SSS2DRLS   Request to release ds?
         BC    ALLOFF,JOBEND10     No, OK for generic request
         TM    SSS2SEL1,SSS2SWTR   Request to select WTR queue?
         BC    ALLON,WAITWORK      Yep, go wait for op cmd
         DROP  R2                  SSS2
*-------------------------------------------------------------*
*        Set the default SSS2 selection information.          *
*-------------------------------------------------------------*
JOBEND10 DS    0H
         BAL   R8,SETDFALT         Go set default selection info
         B     GETJOB              Go get another job
************************************************************
*                                                          *
*              COUNT Processing                            *
*                                                          *
*-------------------------------------------------------------*
*        Set up and request JES to return a count of datasets *
*        matching the selection criteria.                     *
*-------------------------------------------------------------*
GETCNT   DS    0H
*-------------------------------------------------------------*
*        Ask mom for work.                                    *
*-------------------------------------------------------------*
         BAL   R8,ASKJES           Ask mom for a job
         B     CLEANUP          +0 quit
*                               +4 OK, continue
         LA    R2,LDASSOB          Get the SSS2
         USING SSOBEGIN,R2         Tell the assembler
*-------------------------------------------------------------*
*        Print out SSOB return code....                       *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(32),=CL32' COUNT request SSOBRETN return: '
         L     R1,SSOBRETN         Get the subsytem return code
         BAL   R15,CVTHEX
         DROP  R2                  SSOBEGIN
         MVC   MSG2TEXT+32(8),0(R1)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        Print out SSS2 reason code....                       *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
```

```
         MVC   MSG2TEXT(32),=CL32' COUNT request SSS2REAS reason: '
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         SR    R1,R1               Set to zeroes
         IC    R1,SSS2REAS         Get the reason code
         BAL   R15,CVTHEX
         DROP  R2                  SSS2
         MVC   MSG2TEXT+32(8),0(R1)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        Check SSOB return code for type of processing to     *
*        occur next....                                       *
*-------------------------------------------------------------*
         LA    R2,LDASSOB          Get the SSS2
         USING SSOBEGIN,R2         Tell the assembler
         L     R1,SSOBRETN         Get the subsytem return code
         C     R1,=A(SSS2RTOK)     Anything to do?
         BC    EQ,GOTCNT           BIY, Just do it
         C     R1,=A(SSS2LERR)     Was is a logical error?
         BC    EQ,LOGCNERR         Yep, show reason
*-------------------------------------------------------------*
*        Unexpected SAPI return code....wait                  *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(36),=CL34'*Unexpected SAPI COUNT request rc:'
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+36(8),0(R1)
         MVC   MSG2TEXT+40(12),=CL12'*Waiting...'
         BAL   R14,MSG2OUT
         B     WAITWORK            Wait for a operator command
         DROP  R2                  SSOBEGIN
*-------------------------------------------------------------*
*        Logical error processing                             *
*-------------------------------------------------------------*
LOGCNERR DS    OH
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(36),=CL31'*Logical Error....reason code: '
         SR    R1,R1
         IC    R1,SSS2REAS
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+31(8),0(R1)
         BAL   R14,MSG2OUT
         B     WAITWORK            Wait for a operator command
         DROP  R2                  SSS2
*-------------------------------------------------------------*
*        Display the returned counts..they are:               *
*             - dataset count                                 *
*             - line count                                    *
*             - page count                                    *
*             - record count                                  *
*             - byte count                                    *
*-------------------------------------------------------------*
GOTCNT   DS    OH
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
*-------------------------------------------------------------*
*              dataset count                                  *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(25),=CL25' COUNT request SSS2CDS: '
         L     R1,SSS2CDS          Get the dataset count
         BAL   R15,CVTDEC
         MVC   MSG2TEXT+25(8),0(R1)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*              line count                                     *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(25),=CL25' COUNT request SSS2LNCT: '
         L     R1,SSS2LNCT         Get the line count
         BAL   R15,CVTDEC
         MVC   MSG2TEXT+25(8),0(R1)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*              page count                                     *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(25),=CL25' COUNT request SSS2PGCT: '
         L     R1,SSS2PGCT         Get the page count
         BAL   R15,CVTDEC
         MVC   MSG2TEXT+25(8),0(R1)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*              record count                                   *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(25),=CL25' COUNT request SSS2RCCT: '
         L     R1,SSS2RCCT         Get the record count
         BAL   R15,CVTDEC
         MVC   MSG2TEXT+25(8),0(R1)
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*               byte count                                    *
*-------------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(25),=CL25' COUNT request SSS2BYCT: '
         L     R1,SSS2BYCT+4       Get the byte count (some anyway)
         BAL   R15,CVTDEC
         MVC   MSG2TEXT+25(8),0(R1)
         BAL   R14,MSG2OUT
         B     WAITWORK            All done, so wait
         DROP  R2                  SSS2
************************************************************
*                                                          *
*              BULK Processing                             *
*                                                          *
*----------------------------------------------------------*
*        Set up and request JES to dispose of all datasets *
*        matching the selection criteria per the disposition *
*        flag in field SSS2UFLG.                           *
*----------------------------------------------------------*
BLKMOD   DS    OH
*----------------------------------------------------------*
*        Ask mom for work.                                 *
*----------------------------------------------------------*
         BAL   R8,ASKJES           Ask mom for a job
         B     CLEANUP          +0 quit
*                               +4 OK, continue
         LA    R2,LDASSOB          Get the SSS2
         USING SSOBEGIN,R2         Tell the assembler
*----------------------------------------------------------*
*        Print out SSOB return code....                    *
*----------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(23),=CL23' BULK SSOBRETN return: '
         L     R1,SSOBRETN         Get the subsytem return code
         BAL   R15,CVTHEX
         DROP  R2                  SSOBEGIN
         MVC   MSG2TEXT+23(8),0(R1)
         BAL   R14,MSG2OUT
*----------------------------------------------------------*
*        Print out SSS2 reason code....                    *
*----------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(23),=CL23' BULK SSS2REAS reason: '
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         SR    R1,R1               Set to zeroes
         IC    R1,SSS2REAS         Get the reason code
         BAL   R15,CVTHEX
         DROP  R2                  SSS2
         MVC   MSG2TEXT+23(8),0(R1)
         BAL   R14,MSG2OUT
*----------------------------------------------------------*
*        Check SSOB return code to see if logical error    *
*        to be displayed.                                  *
*----------------------------------------------------------*
         LA    R2,LDASSOB          Get the SSS2
         USING SSOBEGIN,R2         Tell the assembler
         L     R1,SSOBRETN         Get the subsytem return code
         C     R1,=A(SSS2RTOK)     All done?
         BC    EQ,WAITWORK         Yep, go wiat for op post
         C     R1,=A(SSS2LERR)     Was is a logical error?
         BC    EQ,LOGBMERR         Yep, show reason
*----------------------------------------------------------*
*        Unexpected SAPI return code....wait               *
*----------------------------------------------------------*
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(36),=CL34'*Unexpected SAPI COUNT request rc:'
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+36(8),0(R1)
         MVC   MSG2TEXT+40(12),=CL12'*Waiting...'
         BAL   R14,MSG2OUT
         B     WAITWORK            Wait for a operator command
         DROP  R2                  SSOBEGIN
*----------------------------------------------------------*
*        Logical error processing                          *
*----------------------------------------------------------*
LOGBMERR DS    OH
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(31),=CL31'*Logical Error....reason code: '
         SR    R1,R1
         IC    R1,SSS2REAS
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+31(8),0(R1)
         BAL   R14,MSG2OUT
         B     WAITWORK            All done, so wait
         DROP  R2                  SSS2
*----------------------------------------------------------*
*        Issue MSG2TEXT                                    *
```

```
*-------------------------------------------------------------*
MSG2OUT  DS    0H
         ST    R14,TRCREG
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLON,MSG2WTO       Yes - Issue WTO
         TPUT  MSG2TEXT,L'MSG2TEXT
         L     R14,TRCREG
         BR    R14
MSG2WTO  DS    0H
         WTO   MF=(E,MSG2),ROUTCDE=2
         L     R14,TRCREG
         BR    R14
MSG2CLR  DS    0H
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2ID,=CL8'SAPIO01I'  Message id
         BR    R14
*-------------------------------------------------------------*
*        FORCE an ABEND after a dataset has been returned     *
*        from GET/PUT processing.                             *
*-------------------------------------------------------------*
FRCABEND DS    0H
         DC    X'FFFF'             Hope it isn't a new op code
***************************************************************
*                                                             *
*              SELECTION UPDATE                               *
*                                                             *
*-------------------------------------------------------------*
*        This routine sets SSS2 default selection             *
*        information.                                         *
*                                                             *
*            - TYPE=GET/PUT                                   *
*            - DEST=SAPI                                      *
*            - CLASS=O1ZAW                                    *
*            - Job Name=SAPI*                                 *
*            - Job range=5 to 500                             *
*            - AGE = 0                                        *
*            - MIN LINE = 0                                   *
*            - MAX LINE = 1000                                *
*            - MIN PAGE = 0                                   *
*            - MAX PAGE = 100                                 *
*            - return SWB token and buffer                    *
*                                                             *
*-------------------------------------------------------------*
SETDFALT DS    0H
         XC    LDACLSR,LDACLSR     Clear returned class
         XC    LDADESTR,LDADESTR   Clear returned destination
         XC    LDAJOBIR,LDAJOBIR   Clear returned job number
         XC    LDAJOBNR,LDAJOBNR   Clear returned job name
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         MVI   SSS2TYPE,SSS2PUGE   Indicate GET request
         MVI   SSS2UFLG,0          Clear disposition flag to
*                                    GET request
         MVI   SSS2SEL1,(SSS2SAWT+SSS2SDST)
*                                  Set selection by class, dest
*                                    job name and job range on any Q
         MVC   SSS2DEST,DEFDEST    Set the destination
         MVC   SSS2CLSL,DEFCLASS   Set the classes
         MVC   SSS2JOBN,DEFJOBNM   Set the job name
         MVC   SSS2JBIL,DEFJOBIL   Set the lower job number limit
         MVC   SSS2JBIH,DEFJOBIH   Set the upper job number limit
         MVI   SSS2SEL2,0          Clear selection flag two
         MVI   SSS2SEL3,0             and selection flag three
         MVI   SSS2SEL4,(SSS2SAGE+SSS2SLIN+SSS2SPAG)  Set selection
*                                    via age, line and page counts
         MVC   SSS2AGE,=F'0'       Set the minimum age
         MVC   SSS2LMIN,=F'0'      Set the minimum line count
         MVC   SSS2LMAX,=F'1000'   Set the maximum line count
         MVC   SSS2PMIN,=F'0'      Set the minimum page count
         MVC   SSS2PMAX,=F'100'    Set the maximum page count
         MVI   SSS2MSC1,SSS2FSWB   Return SWB token and buffer
*-------------------------------------------------------------*
*        Return to caller.                                    *
*-------------------------------------------------------------*
         BR    R8                  Return to caller
         DROP  R2                  SSS2
***************************************************************
*                                                             *
*              SSS2 RESET                                     *
*                                                             *
*-------------------------------------------------------------*
*        This routine sets the SSS2 selection fields          *
*        to zero.                                             *
*-------------------------------------------------------------*
CLRSSS2  DS    0H
         LA    R1,LDASSS2          Get the SSS2 address
         USING SSS2,R1             Establish addressability
         LA    R2,SSS2RBA          Get clear start point
         LA    R3,SSS2JEST-SSS2RBA  Get clear length
         SR    R15,R15             Set zero length and pad
         MVCL  R2,R14              Clear selection and disp
*                                    sections
         DROP  R1                  SSS2
*-------------------------------------------------------------*
*        Return to caller.                                    *
*-------------------------------------------------------------*
         BR    R8                  Return to caller
***************************************************************
*                                                             *
*              FINAL CALL Processing                          *
*                                                             *
*-------------------------------------------------------------*
FINLCALL DS    0H
         XC    SAPIFLG1,SAPIFLG1   Reset flag setting
         LA    R2,LDASSS2          Get SSS2 address
         USING SSS2,R2
         LA    R0,SSS2INPT         Point to input start field
         LA    R1,L'SSS2INPC       Get length of input field
         SR    R15,R15             Zero length and pad field
         MVCL  R0,R14              Clear input SSS2 section
         MVI   SSS2DSP1,SSS2DKPE   Show keep dataset
         MVI   SSS2MSC1,SSS2CTRL   Indicate this is a control
         DROP  R2                  SSS2
         NI    LDAFLG1,FF-FLGSAPI  Loop control
         BAL   R8,ASKJES           Go ask jes to dispose of job
***************************************************************
*                                                             *
*              WAIT FOR JES POST/COMMAND                      *
*                                                             *
*-------------------------------------------------------------*
*        Wait for post indicating newly created SYSOUT        *
*        exists to process.                                   *
*-------------------------------------------------------------*
WAITWORK DS    0H
         LA    R2,LDASSS2          Get the SSS2 address
         USING SSS2,R2             Tell the assembler
         CLI   SSS2TYPE,SSS2PUGE   Is it set for GET/PUT?
         BC    EQ,WTWRK10          Yep, use appropriate msg text
         DROP  R2                  SSS2
         BAL   R14,MSG2CLR
         MVC  MSG2TEXT(32),=CL32' Waiting for operator command!'
         BAL   R14,MSG2OUT
         B     WTWRK20
WTWRK10  DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(18),=CL18'*Waiting for work'
         BAL   R14,MSG2OUT
WTWRK20  DS    0H
         LA    R2,SAPIECB          Get addres of writer ECB
         ST    R2,@WTRECB          Save it away
         L     R14,=A(MSKZMS)       Off to super state
         BALR  R14,R14
         OI    LDAFLG1,FLGSUP
         WAIT  1,ECBLIST=LDAECBL   Wait on our ECB list
         L     R14,=A(MSKNMP)        Back to problem state
         BALR  R14,R14
         NI    LDAFLG1,FF-FLGSUP
         MVI   SAPIECB,0           Turn off POSTed bit  (C J Kinn)
*-------------------------------------------------------------*
*        Clear the SSS2 selection information to undo what    *
*        was previously set so that residual information      *
*        will not conflict with the operator specification.   *
*-------------------------------------------------------------*
         L     R1,@COMECB          Get CIB ECB address
         TM    0(R1),X'40'         Operator posted?
         BC    ALLOFF,GETJOB       No, leave SSS2 for JES post
         BAL   R8,CLRSSS2          Go clear residual info
         B     GETJOB              Go see if there is something to do
***************************************************************
*                                                             *
*        This routine is the cleanup routine in preparation   *
*        for exit.                                            *
*                                                             *
***************************************************************
CLEANUPR DS    0H
         LR    R2,R1
         L     R10,BASEREG-ESTAPARM(,R2)
         L     R9,BASEREG2-ESTAPARM(,R2)
         L     R13,@LDA-ESTAPARM(,R2)
         L     R14,@RETRYP
         BR    R14
CLEANUP  DS    0H
         B     PRINTIT
         TM    LDAFLG2,FLGESCAN    Keyword scan error?
         BC    ALLON,WAITWORK      Yes, wait for another cmd
         TM    LDAFLG2,FLGESTXT    Are we here due to error
         BC    ALLOFF,NOTESTXT     BIN, continue on with exit
*-------------------------------------------------------------*
*        Estae exit caught me, print the info for the post    *
*        mortem.                                              *
*                                                             *
*          -  Print the PSW, abend code, registers and        *
*             the trace table.                                *
*-------------------------------------------------------------*
PRINTIT  DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(6),=CL6' PSW: '
         L     R1,ABENDPSW
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+6(8),0(R1)
         L     R1,ABENDPSW+4
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+15(8),0(R1)
         MVC   MSG2TEXT+23(11),=CL11' COMPCODE:'
```

```
         L     R1,COMPCODE
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+34(8),0(R1)
         XR    R0,R0
         BAL   R14,MSG2OUT
*-------------------------------------------------------------*
*        Now print the registers
*-------------------------------------------------------------*
         LA    R8,4
         LA    R7,ABENDREG
REGLOOP  DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(8),=CL8' Regs:'
         L     R1,0(R7)
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+10(8),0(R1)
         LA    R7,4(,R7)
         L     R1,0(R7)
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+20(8),0(R1)
         LA    R7,4(,R7)
         L     R1,0(R7)
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+30(8),0(R1)
         LA    R7,4(,R7)
         L     R1,0(R7)
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+40(8),0(R1)
         LA    R7,4(,R7)
         XR    R0,R0
         BAL   R14,MSG2OUT
         BCT   R8,REGLOOP
*************************************************************
*        This is where the normal cleanup will be done,       *
*        if possible.                                         *
*************************************************************
*-------------------------------------------------------------*
*        If the SQA storage has not been freed, get into      *
*        supervisor state and return the SQA storage.         *
*-------------------------------------------------------------*
NOTESTXT DS    0H
         TM    LDAFLG1,FLGSQA
         BC    ALLOFF,FREEDOM
         TM    LDAFLG1,FLGSUP
         BC    ALLON,FREESQA
         L     R14,=A(MSKZMS)
         BALR  R14,R14
         OI    LDAFLG1,FLGSUP
*-------------------------------------------------------------*
*        Get rid of storage in SQA.                           *
*-------------------------------------------------------------*
FREESQA  DS    0H
         L     R2,@LDASQA          Get address of SQA storage
         XC    @LDASQA,@LDASQA     Get rid of storage address
         NI    LDAFLG1,FF-FLGSQA   Reset the SQA flag
         FREEMAIN R,                                          +
               LV=8,               ... length                 +
               A=(R2),                                        +
               SP=245
*-------------------------------------------------------------*
*        Get back to problem state...                         *
*-------------------------------------------------------------*
FREEDOM  DS    0H
         TM    LDAFLG1,FLGSUP
         BC    ALLOFF,FREEOPEN
         L     R14,=A(MSKNMP)      Back to problems state....
         BALR  R14,R14
         NI    LDAFLG1,FF-FLGSUP
*-------------------------------------------------------------*
*        Close the opened dataset...                          *
*-------------------------------------------------------------*
FREEOPEN DS    0H
         TM    LDAFLG1,FLGOPEN
         BC    ALLOFF,FREEDYN
         NI    LDAFLG1,FF-FLGOPEN  Indicate closed DS
         LA    R2,INDCB            Get the DCB
         CLOSE ((R2))              Close the dataset
*-------------------------------------------------------------*
*        Unallocate the dataset.....                          *
*-------------------------------------------------------------*
FREEDYN  DS    0H
         TM    LDAFLG1,FLGALLOC
         BC    ALLOFF,FREESAPI
         LA    R2,LDAS99RB         Get SVC99 parm block address
         USING S99RB,R2            Tell the assembler
         MVI   S99VERB,S99VRBUN    Set the deallocation verb
         LA    R1,LDATXTUN         Get deallocation text units
         ST    R1,S99TXTPP         Save them away
         MVC   LDADDNMU,LDADDNMA   Set the DD name to deallocate
         DROP  R2                  S99RB
         NI    LDAFLG1,FF-FLGALLOC Reset flag for loop control
         BAL   R8,SVC99            Get address of
         B     FREESAPI         +0 error return
*                               +4 don't matter
*-------------------------------------------------------------*
*        Issue final call....                                 *
*-------------------------------------------------------------*
FREESAPI DS    0H
         TM    LDAFLG1,FLGSAPI
         BC    ALLOFF,FREEEST
         LA    R2,LDASSS2          Get SSS2 address
         USING SSS2,R2
         LA    R0,SSS2INPT         Point to input start field
         LA    R1,L'SSS2INPC       Get length of input field
         SR    R15,R15             Zero length and pad field
         MVCL  R0,R14              Clear input SSS2 section
         MVI   SSS2DSP1,SSS2DKPE   Show keep dataset
         MVI   SSS2MSC1,SSS2CTRL   Indicate this is a control
         DROP  R2                  SSS2
         NI    LDAFLG1,FF-FLGSAPI  Loop control
         BAL   R8,ASKJES           Go ask jes to dispose of job
         B     FREEEST         +0 error return
*                               +4 don't matter
*-------------------------------------------------------------*
*        Get rid of the estae                                 *
*-------------------------------------------------------------*
FREEEST  DS    0H
         TM    LDAFLG1,FLGESTAE
         BC    ALLOFF,FINISH
         ESTAE 0
         NI    LDAFLG1,FF-FLGESTAE
*-------------------------------------------------------------*
*        End of show.                                         *
*-------------------------------------------------------------*
FINISH   DS    0H
         TM    LDAFLG2,FLGSTAX
         BC    ALLOFF,FINI
         STAX
FINI     DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(24),=CL24'*Terminating...DA END!'
         BAL   R14,MSG2OUT
         LR    R1,R13
         L     R13,4(R1)
         FREEMAIN RU,A=(1),LV=LDASIZE
         L     R14,12(R13)
         LM    R14,R12,12(R13)
         XR    R15,R15
         BR    R14
*-------------------------------------------------------------*
*   CONVERT A NUMBER TO PRINTABLE HEXADECIMAL                 *
*   R1 CONTAINS NUMBER ON ENTRY AND ADDRESS OF EBCDIC ON EXIT *
*-------------------------------------------------------------*
CVTHEX   DS    0H
         STCM  R1,B'1111',DWORK2+3
         UNPK  DWORK1(9),DWORK2+3(5)
         TR    DWORK1,TRANSLAT
         LA    R1,DWORK1
         BR    R15
*-------------------------------------------------------------*
*   CONVERT A HEX NUMBER TO PRINTABLE CHARACTERS.             *
*   R1 CONTAINS NUMBER ON ENTRY AND ADDRESS OF EBCDIC ON EXIT *
*-------------------------------------------------------------*
CVTDEC   DS    0H
         CVD   R1,DWORK2
         OI    DWORK2+7,X'0F'
         LA    R1,DWORK1
         UNPK  0(L'DWORK1,R1),DWORK2
         TR    DWORK1,TRANSLAT
         LA    R1,DWORK1
         BR    R15
*-------------------------------------------------------------*
*   Subroutine to analyze the operator command, all that is   *
*   accepted is a stop command or a modify command.  Return   *
*   is 0(R8) to exit or 4(R8) to continue processing.         *
*                                                             *
*   Linkage: BAL   R8,OPERCHK                                 *
*                                                             *
*   Return:  +0: error return       +4: keyword(s) accepted   *
*                                                             *
*-------------------------------------------------------------*
OPERCHK  DS    0H
         MVI   OPVERB,CIBMODFY     Set modify command
         NI    LDAFLG2,FF-FLGESCAN Indicate no scan error
         TM    LDAFLG2,FLGINIIP    Initial input?
         BC    ALLON,OPEROK        Yes - Process it
         L     R1,@COMECB          Get CIB ECB address
         TM    0(R1),X'40'         Posted?
         BC    ALLON,OPERPST       BIY, continue
         B     4(,R8)              Take the continue on return
OPERPST  DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(24),=CL24' Oper input routine'
         BAL   R14,MSG2OUT
         STM   R14,R3,COMREGS      Save the Registers
         LH    R1,COMMLEN          Get the length
OPERNOK  DS    0H
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLON,OPERBKGR      Yes - Get data from CIB
*-------------------------------------------------------------*
*        Prompt TSO user for new input                        *
*-------------------------------------------------------------*
         L     R1,@COMECB          Get CIB ECB address
         NI    0(R1),FF-X'40'      Reset posted flag
         MVI   COMMBUFF,X'40'      Clear the buffer
         MVC   COMMBUFF+1(L'COMMBUFF-1),COMMBUFF
```

```
         LA    R1,COMMBUFF         Get initial data
         LA    R0,L'COMMBUFF
         TGET  (1),(0)
         LTR   R15,R15
         STH   R1,COMMLEN          Save the length
         BC    ZERO,OPEROK
         LA    R1,=CL6'WHAT?'
         LA    R0,6
         TPUT  (1),(0)
         B     OPERNOK
*-------------------------------------------------------------*
*        Extract input from the CIB                           *
*-------------------------------------------------------------*
OPERBKGR DS    0H
         MVI   COMMBUFF,X'40'      Clear the buffer
         MVC   COMMBUFF+1(L'COMMBUFF-1),COMMBUFF
         L     R2,COMMADDR         Get COMM pointers
         USING COMLIST,R2          Tell the assembler
         ICM   R3,15,COMCIBPT      Address of CIB
         BZ    GETOUT              No CIB - Nothing do
         USING CIBNEXT,R3          Tell the assembler
         LH    R1,CIBDATLN-CIBNEXT(R3) Get data length if any
         STH   R1,COMMLEN          Save the length
         LTR   R1,R1               Any data?
         BC    NPLUS,OPERCIBF      Bin, skip
         BCTR  R1,0                1 less for EX
         EX    R1,*+4              do the move
         MVC   COMMBUFF(*-*),CIBDATA-CIBNEXT(R3) move the initial data
OPERCIBF DS    0H
         MVC   OPVERB,CIBVERB      Save the verb
         QEDIT ORIGIN=COMCIBPT,BLOCK=(R3)   Free the CIB
         DROP  R3,R2
OPEROK   DS    0H
*-------------------------------------------------------------*
*        This is a HELP command for TSO.                      *
*-------------------------------------------------------------*
         NI    LDAFLG2,FF-FLGINIIP Reset initial input
         OC    COMMBUFF,BLANKS     Case - Upper
         CLC   =CL4'HELP',COMMBUFF Help?
         BC    NE,OPERNH           No
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLON,OPERNOK       Yes - No HELP
         L     R15,=A(SAPIHELP)
         BALR  R14,R15
         B     OPERNOK
*-------------------------------------------------------------*
*        If this is a stop command, return to exit.           *
*-------------------------------------------------------------*
OPERNH   DS    0H
         LH    R1,COMMLEN          Get the command length
         CLC   =CL4'END',COMMBUFF  Stop?
         BNE   *+4+4               No
         MVI   OPVERB,CIBSTOP      This a stop command
         CLC   =CL4'STOP',COMMBUFF Stop?
         BNE   *+4+4               No
         MVI   OPVERB,CIBSTOP      This a stop command
         CLI   OPVERB,CIBSTOP      Is this a stop command
         BC    NE,CHKMOD           BIN, check modify
         LM    R14,R3,COMREGS
         B     0(,R8)              Quit return
*-------------------------------------------------------------*
*        If this is a modify command, echo the parameters     *
*        and process them.                                    *
*                                                             *
*          CIBDATLN = length of command data                  *
*          CIBDATA  = pointer to command data                 *
*                                                             *
*-------------------------------------------------------------*
CHKMOD   DS    0H
         CLI   OPVERB,CIBMODFY     Is this a modify command
         BC    NE,GETOUT           BIN, just return
         CH    R1,=H'1'            Any acceptable input?
         BC    LE,CHKMODSD         No - Set defaults
         CLC   COMMBUFF,BLANKS     Any input?
         BC    NE,CHKMODIP         Yes - Use it
CHKMODSD DS    0H
         MVC   COMMBUFF(54),=CL54'SELALL,JBN=SAPI*,D=SAPI,A=0,MLL=1000,C
               LL=0,PL=0,MPL=100'  Set default
         LA    R15,54              Length of command data
         STH   R15,COMMLEN         Set length of command data
CHKMODIP DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2ID,=CL8'SAPIO02I'  Message id
         LH    R15,COMMLEN         Length of command data
         LTR   R15,R15             ANY DATA?
         BNP   CHKMODA             Nope
         BCTR  R15,0               Decrement for execute
         EX    R15,*+4             Move data into text
         MVC   MSG2TEXT+1(0),COMMBUFF Move command data
CHKMODA  DS    0H
         BAL   R14,MSG2OUT
         NI    LDAFLG2,FF-FLGESCAN Indicate no scan error
         LA    R14,OPTIONS         Get options area address
         LR    R0,R14              Get options area address
         LA    R1,OPTLEN           Get options area length
         SR    R15,R15             Zero length and pad
         MVCL  R0,R14              Clear options area
         LH    R15,COMMLEN         Get length of operands
         ST    R15,REMLEN          Initial length
         LA    R15,COMMBUFF        Get address of data
         ST    R15,NEXTKWD         Initial pointer
SCANIT   DS    0H
         BAL   R7,OPERSCAN         Scan next keyword
         B     GETOUT              +0: no more input
*                                  +4: keyword found
         L     R15,=A(KEYPROC)     Gey keyword process rtn addr
         BALR  R7,R15              Process keyword
         B     SCANIT              Go do it again
*-------------------------------------------------------------*
*        All done with the input modify command....free the   *
*        CIB and take normal return.                          *
*-------------------------------------------------------------*
GETOUT   DS    0H
         LM    R14,R3,COMREGS      Restore caller's regs
         TM    LDAFLG2,FLGESCAN    Keyword scan error?
         BC    ALLON,CHKMORE       Error - check more
         B     4(,R8)              Normal return
CHKMORE  DS    0H
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLOFF,OPERNOK      No - Prompt TSO user
         BR    R8                  Return
*-------------------------------------------------------------*
*   Subroutine to scan input                                  *
*                                                             *
*   Input:  REMLEN   = Length of remaining input              *
*           NEXTKWD  = Address to start scanning              *
*                                                             *
*   Output: REMLEN   = Updated length of remaining input      *
*           NEXTKWD  = Address to continue scan next time     *
*           CURRKWD  = Address of current keyword             *
*           CURROPER = Address of current operand             *
*           KWDLEN   = Length of current keyword              *
*           OPERLEN  = Length of current operand              *
*                                                             *
*   Linkage: BAL   R7,OPERSCAN                                *
*                                                             *
*   Return:  +0: no more keywords   +4: keyword returned      *
*                                                             *
*-------------------------------------------------------------*
OPERSCAN DS    0H
         L     R1,NEXTKWD          Address to start scan
         L     R15,REMLEN          Length of remaining input
         LTR   R0,R15              Remember original length
         BCR   ZERO,R7             Return if nothing
         ST    R1,CURRKWD          Keyword starts at beginning
         XC    CURROPER,CURROPER   Assume no operand
SCAN0100 DS    0H
         CLI   0(R1),C','          Is this a comma?
         BC    EQ,SCAN0700         Yes, done with this guy
         CLI   0(R1),C'='          Is this an equals?
         BC    EQ,SCAN0200         Yes, done with keyword
         LA    R1,1(,R1)           Bump past this character
         BCT   R15,SCAN0100        Go check the next one
         ST    R0,KWDLEN           Save keyword length
         B     SCAN0800            Go set up for next time
SCAN0200 DS    0H
         SR    R0,R15              Calculate keyword length
         ST    R0,KWDLEN           And save it
         LA    R1,1(,R1)           Bump past equals
         ST    R1,CURROPER         Save operand address
         BCT   R15,SCAN0300        Decrement remaining count
*                                    and continue
         ST    R15,OPERLEN         Operand length is zero
         B     SCAN0800            Go set up for next time
SCAN0300 DS    0H
         LR    R0,R15              Remember remaining length
SCAN0400 DS    0H
         CLI   0(R1),C','          Is this a comma?
         BC    EQ,SCAN0500         Yes, done with operand
         LA    R1,1(,R1)           Bump past this character
         BCT   R15,SCAN0400        Go check the next one
         ST    R0,OPERLEN          Save operand length
         B     SCAN0800            Go set up for next time
SCAN0500 DS    0H
         SR    R0,R15              Calculate operand length
         ST    R0,OPERLEN          And save it
         LA    R1,1(,R1)           Bump past comma
         BCTR  R15,0               Account for comma
         B     SCAN0800            Go set up for next time
SCAN0700 DS    0H
         XC    OPERLEN,OPERLEN     Operand length zero
         SR    R0,R15              Calculate keyword length
         ST    R0,KWDLEN           And save it
         LA    R1,1(,R1)           Bump past comma
         BCTR  R15,0               Account for comma
SCAN0800 DS    0H
         ST    R15,REMLEN          Save remaining length
         ST    R1,NEXTKWD          Save starting point for
*                                    next time
         B     4(,R7)              Return to caller
*-------------------------------------------------------------*
*        Subroutine to go ask JES for something to do.        *
*          Bad return 0(R8)                                   *
*          Good return 4(R8)                                  *
*-------------------------------------------------------------*
ASKJES   DS    0H
```

```
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(24),=CL24' IEFSSREQ routine'
         BAL   R14,MSG2OUT
*--------------------------------------------------------------*
*        Get to Key 0 and supervisor state.                    *
*--------------------------------------------------------------*
         L     R14,=A(MSKZMS)
         BALR  R14,R14
         OI    LDAFLG1,FLGSUP
*--------------------------------------------------------------*
*        Issue IEFSSREQ.....                                   *
*--------------------------------------------------------------*
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(24),=CL24' Issue IEFSSREQ request'
         BAL   R14,MSG2OUT
         LA    R1,@LDASSOB         Get address of SSOB PTR
         IEFSSREQ ,                HIT THE SUBSYSTEM
         OI    LDAFLG1,FLGSAPI     Show SAPI request made
         LTR   R15,R15             Was request processed OK?
         BC    ZERO,ASKOK
         LR    R3,R15              Save return code
         NI    LDAFLG1,FF-FLGSAPI  It didn't work, so there...
*--------------------------------------------------------------*
*        Get back to where we belong.....                      *
*--------------------------------------------------------------*
         L     R14,=A(MSKNMP)          Back to problem state
         BALR  R14,R14
         NI    LDAFLG1,FF-FLGSUP   prevent a looping situation.
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(8),=CL8' SSI RC: '
         LR    R1,R3
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+16(8),0(R1)
         BAL   R14,MSG2OUT
         B     0(R8)               Take the error return
*--------------------------------------------------------------*
*        OK IEFSSREQ return.                                   *
*--------------------------------------------------------------*
ASKOK    DS    0H
         OI    LDAFLG1,FLGSUP
*--------------------------------------------------------------*
*        Get back to where we belong.....                      *
*--------------------------------------------------------------*
         L     R14,=A(MSKNMP)        Back to problem state
         BALR  R14,R14
         NI    LDAFLG1,FF-FLGSUP
         B     4(R8)               Take the good return
*--------------------------------------------------------------*
*        Subroutine to go do the svc99 request....             *
*   Bad return 0(R8) good return 4(r8)                         *
*--------------------------------------------------------------*
SVC99    DS    0H
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(18),=CL18' DYNALLOC routine'
         BAL   R14,MSG2OUT
         LA    R2,LDAS99RB
         USING S99RB,R2
         L     R1,S99S99X          POINT AT S99RB EXTENSION
         USING S99RBX,R1
         MVI   S99ENMSG,0          Zero message blocks count
         XC    S99EMSGP,S99EMSGP   Zero first message block pointer
         DROP  R2,R1
         L     R14,=A(MSKZMS)
         BALR  R14,R14
         LA    R1,@LDAS99          Get address of SSOB PTR
         DYNALLOC ,                Hit allocation
         LTR   R15,R15             Did it work?
         BC    ZERO,SVC99OK        BIY
         NI    LDAFLG1,FF-FLGALLOC Indicate nothing allocated
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(12),=CL12'*DYNAMIC -'
         LA    R2,LDAS99RB
         USING S99RB,R2
         MVC   MSG2TEXT+12(12),=CL12'ALLOCATION'
         CLI   S99VERB,S99VRBAL
         BC    EQ,SVC9901
         MVC   MSG2TEXT+12(12),=CL12'DEALLOCATION'
SVC9901  DS    0H
         MVC   MSG2TEXT+24(12),=CL12'Failed RC:'
         LR    R1,R15
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+32(8),0(R1)
         BAL   R14,MSG2OUT
*--------------------------------------------------------------*
*        Set flags1, flags 2 error code and infocode.          *
*--------------------------------------------------------------*
         MVI   MSG2+12,X'40'
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2TEXT(7),=CL7' flg1: '
         LH    R1,S99FLAG1
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+7(4),4(R1)
         MVC   MSG2TEXT+11(7),=CL7' flg2: '
         LH    R1,S99FLAG2
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+18(4),0(R1)
         MVC   MSG2TEXT+22(7),=CL7' ErCd: '
         LH    R1,S99ERROR
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+29(4),4(R1)
         MVC   MSG2TEXT+33(7),=CL7' info: '
         LH    R1,S99INFO
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+40(4),4(R1)
         BAL   R14,MSG2OUT
         STM   R14,R5,12(R13)      Save temporarily
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLON,SAPIRTCA      Yes - Split
         L     R3,S99S99X          POINT AT S99RB EXTENSION
         USING S99RBX,R3
         CLI   S99ENMSG,0          Any message blocks from DYNALLOC?
         BZ    SAPIRTCA            No - Return to caller
         SR    R5,R5
         IC    R5,S99ENMSG         Get message block count
         L     R4,S99EMSGP         Point at first message block
         USING MCSM,R4
*        - TPUT message blocks from DYNALLOC
SAPIWTML DS    0H
         LH    R0,MCSMLNG          Pick up message length
         LA    R1,MCSMTXT          Point at message text
         LTR   R15,R0
         BNP   SAPIRTNM
         BCTR  R15,0
         MVC   RECTEXT(*-*),0(R1)
         EX    R15,*-6
         TPUT  RECTEXT,(0)
SAPIRTNM DS    0H
         L     R4,MCSMPTRN         Point at next message in chain
         BCT   R5,SAPIWTML         Loop for all messages
SAPIRTCA DS    0H
         LM    R14,R5,12(R13)      Restore temporaries
         L     R14,=A(MSKNMP)
         BALR   R14,R14
         B     0(R8)               Take the error return
         DROP  R3,R4
         DROP  R2                  S99RB
*--------------------------------------------------------------*
*        OK SVC 99 return.                                     *
*--------------------------------------------------------------*
SVC99OK  DS    0H
         L     R14,=A(MSKNMP)
         BALR  R14,R14
         LA    R2,LDAS99RB
         USING S99RB,R2
         MVI   MSG2+12,X'40'
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(12),=CL12' DYNAMIC -'
         MVC   MSG2TEXT+12(12),=CL12'ALLOCATION'
         CLI   S99VERB,S99VRBAL
         BC    EQ,SVC9901A
         MVC   MSG2TEXT+12(12),=CL12'DEALLOCATION'
SVC9901A DS    0H
         MVC   MSG2TEXT+24(12),=CL12' return code:'
         LR    R1,R15
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+32(8),0(R1)
         BAL   R14,MSG2OUT
*--------------------------------------------------------------*
*        Set flags1, flags 2 error code and infocode.          *
*--------------------------------------------------------------*
         MVI   MSG2+12,X'40'
         BAL   R14,MSG2CLR
         MVC   MSG2TEXT(7),=CL7' flg1: '
         LH    R1,S99FLAG1
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+7(4),4(R1)
         MVC   MSG2TEXT+11(7),=CL7' flg2: '
         LH    R1,S99FLAG2
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+18(4),0(R1)
         MVC   MSG2TEXT+22(7),=CL7' ErCd: '
         LH    R1,S99ERROR
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+29(4),4(R1)
         MVC   MSG2TEXT+33(7),=CL7' info: '
         LH    R1,S99INFO
         BAL   R15,CVTHEX
         MVC   MSG2TEXT+40(4),4(R1)
         DROP  R2                  S99RB
         BAL   R14,MSG2OUT
         OI    LDAFLG1,FLGALLOC    It didn't work, so there...
         B     4(R8)               Take the good return
*--------------------------------------------------------------*
*        Constants and data area                               *
*--------------------------------------------------------------*
@LDASTRT DC    V(LDASTRT)
*--------------------------------------------------------------*
*        Literals                                              *
*--------------------------------------------------------------*
         LTORG
         DROP  R10,R9
*--------------------------------------------------------------*
*        Subroutine to handle keywords.                        *
*                                                              *
*   Linkage: BALR  R7,R15                                      *
```

```
*                                                               *
*   Return:  +0: only return                                    *
*     - flag FLGESCAN off: no scan error                        *
*     - flag FLGESCAN on : error detected during keyword scan   *
*                                                               *
*---------------------------------------------------------------*
KEYPROC  DS    0H
         LR    R10,R15             Get routine address
         USING KEYPROC,R10         Establish addressability
         LA    R4,LDASSS2          Get address of SSS2
         USING SSS2,R4             Tell the assembler
         OI    LDAFLG2,FLGCMDCH    Show command changed sel criteria
         LA    R14,TBLSTART        Point to keyword table
         LA    R0,TBLCOUNT         Number of entries in table
CKENTRY  DS    0H
         L     R1,0(,R14)          Load keyword size
         C     R1,KWDLEN           Size match?
         BC    NE,NEXTNTRY         No, try next one
         L     R15,CURRKWD         Point to keyword
         BCTR  R1,0                Decrement for execute
         EX    R1,COMPKWD          Keyword match?
         BC    NE,NEXTNTRY         No, try next one
         L     R15,12(,R14)        Get routine address
         BR    R15                 Go to it
NEXTNTRY DS    0H
         LA    R14,ENTRYSIZ(,R14)  Point to next entry
         BCT   R0,CKENTRY          Try next one
*---------------------------------------------------------------*
*        Keyword not found in table.                            *
*---------------------------------------------------------------*
         OI    LDAFLG2,FLGESCAN    Indicate scan error
         BAL   R14,KWM2CLR
         MVC   MSG2ID,=CL8'SAPIO03I'  Message id
         MVC   MSG2TEXT+1(L'LDAXWNAM),LDAXWNAM
         MVC   MSG2TEXT+9(22),=CL22'*Unrecognized keyword '
         L     R1,KWDLEN           Get length of bad keyword
         BCTR  R1,0                Decrement for execute
         EX    R1,MSG9910X         Move keyword in
         BAL   R14,KWM2OUT
         B     KEYRTN              Return to caller
MSG9910X MVC   MSG2TEXT+31(0),0(R15)  Move keyword in
*---------------------------------------------------------------*
*        Process the CL= keyword.                               *
*---------------------------------------------------------------*
KWCLASS  DS    0H
         MVC   SSS2CLSL,BLANKS     Clear to blanks
         NI    SSS2SEL1,FF-SSS2SCLS   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL1,SSS2SCLS   Show select by class
         LA    R15,SSS2CLSL        Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the D= keyword.                                *
*---------------------------------------------------------------*
KWDEST   DS    0H
         MVC   SSS2DEST,BLANKS     Clear to blanks
         NI    SSS2SEL1,FF-SSS2SDST   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL1,SSS2SDST   Show select by dest
         LA    R15,SSS2DEST        Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the JBN= keyword.                              *
*---------------------------------------------------------------*
KWJOBN   DS    0H
         MVC   SSS2JOBN,BLANKS     Clear to blanks
         NI    SSS2SEL1,FF-SSS2SJBN   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL1,SSS2SJBN   Show select by job name
         LA    R15,SSS2JOBN        Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the JBI= keyword.                              *
*---------------------------------------------------------------*
KWJOBI   DS    0H
         MVC   SSS2JBIL,CHARZERO   Zero out job number
         NI    SSS2SEL1,FF-SSS2SJBI   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL1,SSS2SJBI   Show select by job number
         LA    R15,SSS2JBIL        Point to jobnumbr field
         LA    R15,L'SSS2JBIL(,R15)   point beyond jobnumbr field
         L     R14,CURROPER        Get pointer to input string
         L     R1,OPERLEN          Length of input string
         SR    R15,R1              Point to target of move
         BCTR  R1,0                Decrement 1 for execute
         EX    R1,OPERMOVE         Move job number
         B     KEYRTN              Return to caller
*---------------------------------------------------------------*
*        Process the HJBI= keyword.                             *
*---------------------------------------------------------------*
KWHIJBI  DS    0H
         MVC   SSS2JBIH,CHARZERO   Zero out job number
         NI    SSS2SEL1,FF-SSS2SJBI   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL1,SSS2SJBI   Show select by job number
         LA    R15,SSS2JBIH        Point to jobnumbr field
         LA    R15,L'SSS2JBIH(,R15)  point beyond jobnumbr field
         L     R14,CURROPER        Get pointer to input string
         L     R1,OPERLEN          Length of input string
         SR    R15,R1              Point to target of move
         BCTR  R1,0                Decrement 1 for execute
         EX    R1,OPERMOVE         Move job number
         B     KEYRTN              Return to caller
*---------------------------------------------------------------*
*        Process the W= keyword.                                *
*---------------------------------------------------------------*
KWPGM    DS    0H
         MVC   SSS2PGMN,BLANKS     Blank out WRITER NAME
         NI    SSS2SEL2,FF-SSS2SPGM   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL2,SSS2SPGM   Show select by program name
         LA    R15,SSS2PGMN        Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the F= keyword.                                *
*---------------------------------------------------------------*
KWFORM   DS    0H
         MVC   SSS2FORM(L'SSS2FORC),BLANKS  Blank out FORM
         NI    SSS2SEL2,FF-SSS2SFRM   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL2,SSS2SFRM   Show select by forms
         LA    R15,SSS2FORM        Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the U= keyword.                                *
*---------------------------------------------------------------*
KWCRUID  DS    0H
         MVC   SSS2CREA,BLANKS     Blank out creator userid
         NI    SSS2SEL2,FF-SSS2SCRE   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL2,SSS2SCRE   Show select by creator userid
         LA    R15,SSS2CREA        Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the PM= keyword.                               *
*---------------------------------------------------------------*
KWPMODE  DS    0H
         MVC   SSS2PRMO(L'SSS2PRMC),BLANKS Blank out process mode
         NI    SSS2SEL2,FF-SSS2SPRM   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL2,SSS2SPRM   Show select by process mode
         LA    R15,SSS2PRMO        Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the C= keyword.                                *
*---------------------------------------------------------------*
KWFCB    DS    0H
         MVC   SSS2FCB,BLANKS      Blank out FCB
         NI    SSS2SEL2,FF-SSS2SFCB   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL2,SSS2SFCB   Show select by FCB
         LA    R15,SSS2FCB         Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the U= keyword.                                *
*---------------------------------------------------------------*
KWUCS    DS    0H
         MVC   SSS2UCS,BLANKS      Blank out UCS
         NI    SSS2SEL2,FF-SSS2SUCS   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL2,SSS2SUCS   Show select by UCS
         LA    R15,SSS2UCS         Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the M= keyword.                                *
*---------------------------------------------------------------*
KWMOD    DS    0H
         MVC   SSS2MOD,BLANKS      Blank out MOD
         NI    SSS2SEL4,FF-SSS2SMOD   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL4,SSS2SMOD   Show select by MOD
         LA    R15,SSS2MOD         Point to target of move
         B     MOVEOPER            Move operand to work area
*---------------------------------------------------------------*
*        Process the FL= keyword.                               *
*---------------------------------------------------------------*
KWFLASH  DS    0H
         MVC   SSS2FLSH,BLANKS     Blank out FLASH
         NI    SSS2SEL4,FF-SSS2SFLS   Reset select flag
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         OI    SSS2SEL4,SSS2SFLS   Show select by FLASH
         LA    R15,SSS2FLSH        Point to target of move
```

```
        B     MOVEOPER            Move operand to work area
*-------------------------------------------------------------*
*       Process the CH= keyword.                              *
*-------------------------------------------------------------*
KWCHARS DS    OH
        MVC   SSS2CHAR(L'SSS2CHAC),BLANKS  Blank out CHARS
        NI    SSS2SEL4,FF-SSS2SCHR   Reset select flag
        ICM   R15,B'1111',OPERLEN  Get operand length
        BC    ZERO,KEYRTN         Return if reset is OK
        OI    SSS2SEL4,SSS2SCHR   Show select by chars
        LA    R15,SSS2CHAR        Point to target of move
        B     MOVEOPER            Move operand to work area
*-------------------------------------------------------------*
*       Process the A= keyword.                               *
*-------------------------------------------------------------*
KWAGE   DS    OH
        MVC   SSS2AGE,HEXZERO     Zero out AGE
        NI    SSS2SEL4,FF-SSS2SAGE   Reset select flag
        ICM   R15,B'1111',OPERLEN  Get operand length
        BC    ZERO,KEYRTN         Return if reset is OK
        OI    SSS2SEL4,SSS2SAGE   Show select by AGE
        L     R14,CURROPER        Get pointer to input string
        L     R1,OPERLEN          Length of input string
        BCTR  R1,0                Decrement 1 for execute
        EX    R1,PACKIT           Pack the line limit
        CVB   R1,PACKWK           Convert to binary
        ST    R1,SSS2AGE          Set in target area
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the LL= keyword.                              *
*-------------------------------------------------------------*
KWLLIM  DS    OH
        MVC   SSS2LMIN,HEXZERO    Zero out min line limit
        NI    SSS2SEL4,FF-SSS2SLIN   Reset select flag
        ICM   R15,B'1111',OPERLEN  Get operand length
        BC    ZERO,KEYRTN         Return if reset is OK
        OI    SSS2SEL4,SSS2SLIN   Show select by line limit
        L     R14,CURROPER        Get pointer to input string
        L     R1,OPERLEN          Length of input string
        BCTR  R1,0                Decrement 1 for execute
        EX    R1,PACKIT           Pack the line limit
        CVB   R1,PACKWK           Convert to binary
        ST    R1,SSS2LMIN         Set in target area
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the MLL= keyword.                             *
*-------------------------------------------------------------*
KWMLLIM DS    OH
        MVC   SSS2LMAX,HEXZERO    Zero out max line limit
        ICM   R15,B'1111',OPERLEN  Get operand length
        BC    ZERO,KEYRTN         Return if reset is OK
        L     R14,CURROPER        Get pointer to input string
        L     R1,OPERLEN          Length of input string
        BCTR  R1,0                Decrement 1 for execute
        EX    R1,PACKIT           Pack the max line limit
        CVB   R1,PACKWK           Convert to binary
        ST    R1,SSS2LMAX         Set in target area
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the PL= keyword.                              *
*-------------------------------------------------------------*
KWPLIM  DS    OH
        MVC   SSS2PMIN,HEXZERO    Zero out min page limit
        NI    SSS2SEL4,FF-SSS2SPAG   Reset select flag
        ICM   R15,B'1111',OPERLEN  Get operand length
        BC    ZERO,KEYRTN         Return if reset is OK
        OI    SSS2SEL4,SSS2SPAG   Show select by page limit
        L     R14,CURROPER        Get pointer to input string
        L     R1,OPERLEN          Length of input string
        BCTR  R1,0                Decrement 1 for execute
        EX    R1,PACKIT           Pack the page limit
        CVB   R1,PACKWK           Convert to binary
        ST    R1,SSS2PMIN         Set in target area
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the MPL= keyword.                             *
*-------------------------------------------------------------*
KWMPLIM DS    OH
        MVC   SSS2PMIN,HEXZERO    Zero out max page limit
        ICM   R15,B'1111',OPERLEN  Get operand length
        BC    ZERO,KEYRTN         Return if reset is OK
        L     R14,CURROPER        Get pointer to input string
        L     R1,OPERLEN          Length of input string
        BCTR  R1,0                Decrement 1 for execute
        EX    R1,PACKIT           Pack the max page limit
        CVB   R1,PACKWK           Convert to binary
        ST    R1,SSS2PMAX         Set in target area
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELHOLD keyword.                          *
*-------------------------------------------------------------*
KWSLHLD DS    OH
        OI    SSS2SEL1,SSS2SHOL   Set proper selection flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELWTR keyword.                           *
*-------------------------------------------------------------*
KWSLWTR DS    OH
        OI    SSS2SEL1,SSS2SWTR   Set proper selection flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELXWTR keyword.                          *
*-------------------------------------------------------------*
KWSLXWR DS    OH
        OI    SSS2SEL1,SSS2SXWH   Set proper selection flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELTSO keyword.                           *
*-------------------------------------------------------------*
KWSLTSO DS    OH
        OI    SSS2SEL1,SSS2SHLD   Set proper selection flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELALL keyword.                           *
*-------------------------------------------------------------*
KWSLALL DS    OH
        OI    SSS2SEL1,SSS2SAWT   Set proper selection flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELSTC keyword                            *
*-------------------------------------------------------------*
KWSELSTC DS   OH
        OI    SSS2SEL3,SSS2SSTC   Show select by STC
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELTSU keyword                            *
*-------------------------------------------------------------*
KWSELTSU DS   OH
        OI    SSS2SEL3,SSS2STSU   Show select by STC
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELJOB keyword                            *
*-------------------------------------------------------------*
KWSELJOB DS   OH
        OI    SSS2SEL3,SSS2SJOB   Show select by JOB
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SELAPC keyword                            *
*-------------------------------------------------------------*
KWSELAPC DS   OH
        OI    SSS2SEL3,SSS2SAPC   Show select by JOB
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the NCPDS keyword                             *
*-------------------------------------------------------------*
KWSELNCP DS   OH
        OI    SSS2SEL5,SSS2SCPN   Show select not to include CPDS
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the IP keyword.                               *
*-------------------------------------------------------------*
KWIP    DS    OH
        OI    SSS2SEL2,SSS2SIPA   Show select by IP
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the NIP keyword.                              *
*-------------------------------------------------------------*
KWNIP   DS    OH
        OI    SSS2SEL2,SSS2SIPN   Show select by NIP
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the SWB keyword.                              *
*-------------------------------------------------------------*
KWSWB   DS    OH
        OI    SSS2MSC1,SSS2FSWB   Return SWB token and buffer
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the DDEL keyword.                             *
*-------------------------------------------------------------*
KWDDEL  DS    OH
        XC    SSS2DSP1,SSS2DSP1   Indicate delete request
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the DKEEP keyword.                            *
*-------------------------------------------------------------*
KWDKEEP DS    OH
        OI    SSS2DSP1,SSS2DKPE   Set dataset keep flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the DRHLD keyword.                            *
*-------------------------------------------------------------*
KWDRHLD DS    OH
        OI    SSS2DSP1,SSS2DKPE   Show keep dataset
        OI    SSS2DSP1,SSS2RHLD   Set ds op hold flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the DRNPR keyword.                            *
*-------------------------------------------------------------*
KWDRNPR DS    OH
        OI    SSS2DSP1,SSS2DKPE   Show keep dataset
        OI    SSS2DSP1,SSS2RNPR   Set ds op hold flag
        B     KEYRTN              Return to caller
*-------------------------------------------------------------*
*       Process the DHOLD keyword.                            *
*-------------------------------------------------------------*
KWDHOLD DS    OH
```

```
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         OI    SSS2DSP1,SSS2DHLD   Set HOLD Q flag
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the DREL keyword.                             *
*--------------------------------------------------------------*
KWDREL   DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         OI    SSS2DSP1,SSS2DRLS   Set WTR Q flag
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the DCKPT keyword.                            *
*--------------------------------------------------------------*
KWDCKPT  DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         OI    SSS2DSP1,SSS2CHKP   Set ckpt RBA flag
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the DNWR keyword.                             *
*--------------------------------------------------------------*
KWDNWR   DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         OI    SSS2DSP1,SSS2DNWR   Set null writer name flag
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the DCL= keyword.                             *
*--------------------------------------------------------------*
KWDCLS   DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         MVC   SSS2DCLS,BLANKS     Blank out new class
         LA    R15,SSS2DCLS        Point to target of move
         B     MOVEOPER            Move operand to work area
*--------------------------------------------------------------*
*        Process the DD= keyword.                              *
*--------------------------------------------------------------*
KWDDST   DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         MVC   SSS2DDES,BLANKS     Blank out new dest
         LA    R15,SSS2DDES        Point to target of move
         B     MOVEOPER            Move operand to work area
*--------------------------------------------------------------*
*        Process the DF= keyword.                              *
*--------------------------------------------------------------*
KWDFORM  DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         MVC   SSS2DFOR,BLANKS     Blank out new class
         LA    R15,SSS2DFOR        Point to target of move
         B     MOVEOPER            Move operand to work area
*--------------------------------------------------------------*
*        Process the DW= keyword.                              *
*--------------------------------------------------------------*
KWDWTR   DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         MVC   SSS2DPGM,BLANKS     Blank out new dest
         LA    R15,SSS2DPGM        Point to target of move
         B     MOVEOPER            Move operand to work area
*--------------------------------------------------------------*
*        Process the DC= keyword.                              *
*--------------------------------------------------------------*
KWDCOPY  DS    OH
         OI    SSS2DSP1,SSS2DKPE   Show keep dataset
         MVC   SSS2CLFT,HEXZERO    Zero out copy count
         ICM   R15,B'1111',OPERLEN  Get operand length
         BC    ZERO,KEYRTN         Return if reset is OK
         L     R14,CURROPER        Get pointer to input string
         L     R1,OPERLEN          Length of input string
         BCTR  R1,0                Decrement 1 for execute
         EX    R1,PACKIT           Pack the max page limit
         CVB   R1,PACKWK           Convert to binary
         STH   R1,SSS2CLFT         Set in target area
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the COUNT keyword.                            *
*--------------------------------------------------------------*
KWGET    DS    OH
         MVI   SSS2TYPE,SSS2PUGE   Indicate GET request
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the COUNT keyword.                            *
*--------------------------------------------------------------*
KWCOUNT  DS    OH
         MVI   SSS2TYPE,SSS2COUN   Indicate COUNT request
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the BULK keyword.                             *
*--------------------------------------------------------------*
KWBULK   DS    OH
         MVI   SSS2TYPE,SSS2BULK   Indicate COUNT request
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the NCL= keyword.                             *
*--------------------------------------------------------------*
KWNCLS   DS    OH
         MVC   SSS2CLAS,BLANKS     Blank out new class
         OI    SSS2UFLG,SSS2SETC   Change the dataset(s) class
         LA    R15,SSS2CLAS        Point to target of move
         B     MOVEOPER            Move operand to work area
*--------------------------------------------------------------*
*        Process the ND= keyword.                              *
*--------------------------------------------------------------*
KWNDST   DS    OH
         MVC   SSS2DES2,BLANKS     Blank out new dest
         OI    SSS2UFLG,SSS2ROUT   Change the dataset(s) dest
         LA    R15,SSS2DES2        Point to target of move
         B     MOVEOPER            Move operand to work area
*--------------------------------------------------------------*
*        Process the REL keyword.                              *
*--------------------------------------------------------------*
KWREL    DS    OH
         OI    SSS2UFLG,SSS2RLSE   Release the dataset(s)
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the DEL keyword.                              *
*--------------------------------------------------------------*
KWDEL    DS    OH
         OI    SSS2UFLG,SSS2DELC   Delete the dataset(s)
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the DEFAULT keyword.                          *
*--------------------------------------------------------------*
KWDFLT   DS    OH
         MVI   SSS2UFLG,0          Clear disposition flag to
*                                    GET request
         MVI   SSS2SEL2,0          Clear selection flag two
         MVI   SSS2SEL3,0            and selection flag three
         MVI   SSS2SEL1,(SSS2SAWT+SSS2SCLS+SSS2SDST+SSS2SJBN+SSS2SJBI)
*                                  Set selection by class, dest
*                                    job name and job id on any Q
         MVC   SSS2DEST,DEFDEST    Set the destination
         MVC   SSS2CLSL,DEFCLASS   Set the classes
         MVC   SSS2JOBN,DEFJOBNM   Set the job name
         MVC   SSS2JBIL,DEFJOBIL   Set the lower job number limit
         MVC   SSS2JBIH,DEFJOBIH   Set the upper job number limit
         MVI   SSS2SEL4,(SSS2SAGE+SSS2SLIN+SSS2SPAG)  Set selection
*                                    via age, line and page counts
         MVC   SSS2AGE,=F'0'       Set the minimum age
         MVC   SSS2LMIN,=F'0'      Set the minimum line count
         MVC   SSS2PMIN,=F'0'      Set the minimum page count
         MVC   SSS2LMAX,=F'1000'   Set the maximum line count
         MVC   SSS2PMAX,=F'100'    Set the maximum page count
         MVI   SSS2MSC1,SSS2FSWB   Return SWB token and buffer
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the WTYPE keyword.                            *
*--------------------------------------------------------------*
KWWTYPE  DS    OH
         MVI   SSS2TYPE,4          Set wrong type
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the WLEN keyword.                             *
*--------------------------------------------------------------*
KWWLEN   DS    OH
         MVC   SSS2LEN,=H'255'     Set wrong length
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the DUPJ keyword.                             *
*--------------------------------------------------------------*
KWDUPJ   DS    OH
         OI    SSS2SEL1,SSS2SDUP   Show select via duplicate job
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the FABEND keyword.                           *
*--------------------------------------------------------------*
KWFABEND DS    OH
         OI    SAPIFLG1,SAPIABND   Force an abend after ds returned
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the ONEDS keyword.                            *
*--------------------------------------------------------------*
KWONEDS  DS    OH
         OI    SAPIFLG1,SAPIONED   Show process one dataset
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Process the ONEJOB keyword.                           *
*--------------------------------------------------------------*
KWONEJOB DS    OH
         OI    SAPIFLG1,SAPIONEJ   Show process one dataset
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Move input operand into respective work area loc.     *
*              R15 set to target of move.                      *
*--------------------------------------------------------------*
MOVEOPER DS    OH
         L     R14,CURROPER        Get pointer to input string
         L     R1,OPERLEN          Length of input string
         BCTR  R1,0                Decrement 1 for execute
         EX    R1,OPERMOVE         Move input operand
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        NOP.                                                  *
*--------------------------------------------------------------*
NOOP     DS    OH
         B     KEYRTN              Return to caller
*--------------------------------------------------------------*
*        Return to caller                                      *
```

```
*-------------------------------------------------------------*
KEYRTN   DS    OH
         L     R10,BASEREG         Restore mainline 1st base reg
         BR    R7                  Return to caller
OPERMOVE MVC   0(0,R15),0(R14)     Move value
COMPKWD  CLC   4(0,R14),0(R15)     Compare keyword
PACKIT   PACK  PACKWK,0(,R14)      Pack input numeric
PACKWK   DC    D'0'                Pack work area
EQUALS   DC    C'..=..'            Equals for WTO text
*-------------------------------------------------------------*
*        Issue KEYWORD messages                               *
*-------------------------------------------------------------*
KWM2OUT  DS    OH
         ST    R14,TRCREG
         TM    LDAFLG2,FLGBACKG    Running in background?
         BC    ALLON,KWM2WTO       Yes - Issue WTO
         TPUT  MSG2TEXT,L'MSG2TEXT
         L     R14,TRCREG
         BR    R14
KWM2WTO  DS    OH
         WTO   MF=(E,MSG2),ROUTCDE=2
         L     R14,TRCREG
         BR    R14
KWM2CLR  DS    OH
         MVI   MSG2TEXT,X'40'
         MVC   MSG2TEXT+1(L'MSG2TEXT-1),MSG2TEXT
         MVC   MSG2ID,=CL8'SAPI0011'  Message id
         BR    R14
*-------------------------------------------------------------*
*        Set/Reset processing mode                            *
*-------------------------------------------------------------*
MSKZMS   DS    OH
         STM   R14,R3,12(R13)      SAVE A FEW REGS
         BALR  R3,0
         USING *,R3
         MODESET KEY=ZERO,MODE=SUP
         LM    R14,R3,12(R13)      ....RESTRE A FEW REGS
         DROP  R3
         BR    R14                 .....BACK TO CALLER
*------- Reset APF authorization
MSKNMP   DS    OH
         STM   R14,R3,12(R13)      SAVE A FEW REGS
         BALR  R3,0
         USING *,R3
         MODESET KEY=NZERO,MODE=PROB
         LM    R14,R3,12(R13)      ....RESTRE A FEW REGS
         DROP  R3
         BR    R14                .....BACK TO CALLER
*-------------------------------------------------------------*
*        Keyword table.                                       *
*                                                             *
*        The format of the table is:                          *
*          4 bytes - length of keyword                        *
*          8 bytes - keyword                                  *
*          4 bytes - address of keyword processing routine    *
*                                                             *
*        The SSOB allows for the following selection          *
*        criteria. The associated SSOB field and keyword      *
*        are listed.                                          *
*                                                             *
*                          KEYWORD       FLAG        FIELD    *
*                          -------       ----        -----    *
*                                                             *
*   -> TSO Hold            SELTSO     (SSS2SHLD)       n/a    *
*   -> EXTWTR Hold         SELXWTR    (SSS2SXWH)       n/a    *
*   -> TSO/EXTWTR Hold     SELHOLD    (SSS2SHOL)       n/a    *
*   -> WTR                 SELWTR     (SSS2SWTR)       n/a    *
*   -> TSO/EXTWTR/WTR      SELALL     (SSS2SAWT)       n/a    *
*   -> CLASS               CL=        (SSS2SCLS)     SSS2CLSL *
*   -> DEST                D=         (SSS2SDST)     SSS2DEST *
*   -> JOBNAME             JBN=       (SSS2SJBN)     SSS2JOBN *
*   -> JOBID               JBI=       (SSS2SJBI)     SSS2JBIL *
*   -> JOBID               HJBI=      (SSS2SJBI)     SSS2JBIH *
*                                                             *
*   -> PROGRAM NAME        W=         (SSS2SPGM)     SSS2PGMN *
*   -> FORM                F=         (SSS2SFRM)     SSS2FORM *
*   -> CREATOR USERID      U=         (SSS2SCRE)     SSS2CREA *
*   -> PROCESS MODE        PM=        (SSS2SPRM)     SSS2PRMO *
*   -> w/IP addrss         IP         (SSS2SIPA)       n/a    *
*   -> w/o IP address      NIP        (SSS2SIPN)       n/a    *
*   -> FCB                 C=         (SSS2SFCB)     SSS2FCB  *
*   -> UCS                 U=         (SSS2SUCS)     SSS2UCS  *
*   -> SWB                 SWB        (SSS2FSWB)       n/a    *
*                                                             *
*   -> Started Task        SELSTC     (SSS2SSTC)       n/a    *
*   -> Time Sharing User   SELTSU     (SSS2STSU)       n/a    *
*   -> Batch job           SELJOB     (SSS2SJOB)       n/a    *
*   -> APPC                SELAPC     (SSS2SAPC)       n/a    *
*   -> No CPDS             NCPDS      (SSS2SCPN)       n/a    *
*                                                             *
*   -> MOD                 M=         (SSS2SMOD)     SSS2MOD  *
*   -> FLASH               FL=        (SSS2SFLS)     SSS2FLSH *
*   -> AGE                 A=         (SSS2SAGE)     SSS2AGE  *
*   -> LINE LIMIT          LL=        (SSS2SLIN)     SSS2LMIN *
*                          MLL=       (SSS2SLIN)     SSS2LMAX *
*   -> PAGE LIMIT          PL=        (SSS2SPAG)     SSS2PMIN *
*                          MPL=       (SSS2SPAG)     SSS2PMAX *
*   -> CHARS               CH=        (SSS2SCHR)     SSS2CHAR *
*                                                             *
*   -> Get/Put             GET           n/a         SSS2TYPE *
*                                                             *
*   Keywords applicable to dataset disposition for GET/PUT    *
*   processing are as follows:                                *
*   -> Delete ds           DDEL       (  n/a   )        n/a   *
*   -> Keep ds             DKEEP      (SSS2DKPE)        n/a   *
*   -> OP Hold             DRHLD      (SSS2RHLD)        n/a   *
*   -> Don't process       DRNPR      (SSS2RNPR)        n/a   *
*        again                                                *
*   -> Hold ds             DHOLD      (SSS2DHLD)        n/a   *
*   -> Release ds          DREL       (SSS2DRLS)        n/a   *
*   -> Null writer name    DNWR       (SSS2DNWR)        n/a   *
*   -> New class           DCL=          n/a          SSS2DCLS *
*   -> New dest            DD=           n/a          SSS2DDES *
*   -> New forms           DF=           n/a          SSS2DFOR *
*   -> New wtr name        DW=           n/a          SSS2DPGM *
*   -> New copy count      DC=           n/a          SSS2CLFT *
*                                                             *
*   -> Count request       COUNT         n/a          SSS2TYPE *
*   -> Bulk Modify req     BULK          n/a          SSS2TYPE *
*                                                             *
*   Keywords applicable to Bulk Modify processing are as      *
*   follows:                                                  *
*   -> New class           NCL=       (SSS2SETC)     SSS2CLAS *
*   -> New destination     ND=        (SSS2ROUT)     SSS2DES2 *
*   -> Release             REL        (SSS2RLSE)        n/a   *
*   -> Delete              DEL        (SSS2DELC)        n/a   *
*                                                             *
*        For error processing scenarios, the following        *
*        keywords can be used (the application code will      *
*        set an incorrect value).                             *
*                                                             *
*   -> wrong type          WTYPE                      SSS2TYPE *
*   -> wrong length        WLEN                       SSS2LEN  *
*   -> duplicate jobs      DUPJ                               *
*   -> forced abend        ABEND                              *
*                                                             *
*-------------------------------------------------------------*
*-------------------------------------------------------------*
*        CLASS keyword.                                       *
*-------------------------------------------------------------*
TBLSTART DS    OF
         KWDEFINE NAME=CL,         Define the CL=                  +
               ROUTINE=KWCLASS       keyword
ENTRYSIZ EQU   *-TBLSTART          Size of one entry
*-------------------------------------------------------------*
*        DEST keyword.                                        *
*-------------------------------------------------------------*
         KWDEFINE NAME=D,          Define the D=                   +
               ROUTINE=KWDEST        keyword
*-------------------------------------------------------------*
*        JOBNAME keyword.                                     *
*-------------------------------------------------------------*
         KWDEFINE NAME=JBN,        Define the JBN=                 +
               ROUTINE=KWJOBN        keyword
*-------------------------------------------------------------*
*        JOBNUMBER keyword.                                   *
*-------------------------------------------------------------*
         KWDEFINE NAME=JBI,        Define the JBI=                 +
               ROUTINE=KWJOBI        keyword
*-------------------------------------------------------------*
*        HIGH JOBNUMBER keyword.                              *
*-------------------------------------------------------------*
         KWDEFINE NAME=HJBI,       Define the HJBI=                +
               ROUTINE=KWHIJBI       keyword
*-------------------------------------------------------------*
*        WRITER keyword.                                      *
*-------------------------------------------------------------*
         KWDEFINE NAME=W,          Define the W=                   +
               ROUTINE=KWPGM         keyword
*-------------------------------------------------------------*
*        FORM keyword.                                        *
*-------------------------------------------------------------*
         KWDEFINE NAME=F,          Define the F=                   +
               ROUTINE=KWFORM        keyword
*-------------------------------------------------------------*
*        CREATOR USERID keyword.                              *
*-------------------------------------------------------------*
         KWDEFINE NAME=U,          Define the U=                   +
               ROUTINE=KWCRUID       keyword
*-------------------------------------------------------------*
*        PROCESS MODE keyword.                                *
*-------------------------------------------------------------*
         KWDEFINE NAME=PM,         Define the PM=                  +
               ROUTINE=KWPMODE       keyword
*-------------------------------------------------------------*
*        FCB keyword.                                         *
*-------------------------------------------------------------*
         KWDEFINE NAME=C,          Define the C=                   +
               ROUTINE=KWFCB         keyword
*-------------------------------------------------------------*
*        UCS keyword.                                         *
*-------------------------------------------------------------*
         KWDEFINE NAME=U,          Define the U=                   +
               ROUTINE=KWUCS         keyword
*-------------------------------------------------------------*
```

```
*        MOD keyword.                                          *
*--------------------------------------------------------------*
         KWDEFINE NAME=M,          Define the M=                 +
               ROUTINE=KWMOD         keyword
*--------------------------------------------------------------*
*        FLASH keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=FL,         Define the FL=                +
               ROUTINE=KWFLASH       keyword
*--------------------------------------------------------------*
*        AGE keyword.                                          *
*--------------------------------------------------------------*
         KWDEFINE NAME=A,          Define the A=                 +
               ROUTINE=KWAGE         keyword
*--------------------------------------------------------------*
*        Line Limit keyword.                                   *
*--------------------------------------------------------------*
         KWDEFINE NAME=LL,         Define the LL=                +
               ROUTINE=KWLLIM        keyword
*--------------------------------------------------------------*
*        Max Line limit keyword.                               *
*--------------------------------------------------------------*
         KWDEFINE NAME=MLL,        Define the MLL=               +
               ROUTINE=KWMLLIM       keyword
*--------------------------------------------------------------*
*        Page Limit keyword.                                   *
*--------------------------------------------------------------*
         KWDEFINE NAME=PL,         Define the PL=                +
               ROUTINE=KWPLIM        keyword
*--------------------------------------------------------------*
*        Max Page limit keyword.                               *
*--------------------------------------------------------------*
         KWDEFINE NAME=MPL,        Define the MPL=               +
               ROUTINE=KWMPLIM       keyword
*--------------------------------------------------------------*
*        CHARS keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=CH,         Define the CH=                +
               ROUTINE=KWCHARS       keyword
*--------------------------------------------------------------*
*        SELSTC keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELSTC,     Define the SELSTC             +
               ROUTINE=KWSELSTC      keyword
*--------------------------------------------------------------*
*        SELTSU keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELTSU,     Define the SELTSU             +
               ROUTINE=KWSELTSU      keyword
*--------------------------------------------------------------*
*        SELJOB keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELJOB,     Define the SELJOB             +
               ROUTINE=KWSELJOB      keyword
*--------------------------------------------------------------*
*        SELAPPC keyword.                                      *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELAPPC,    Define the SELAPPC            +
               ROUTINE=KWSELAPC      keyword
*--------------------------------------------------------------*
*        NCPDS keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=NCPDS,      Define the NCPDS              +
               ROUTINE=KWSELNCP      keyword
*--------------------------------------------------------------*
*        IP keyword.                                           *
*--------------------------------------------------------------*
         KWDEFINE NAME=IP,         Define the IP                 +
               ROUTINE=KWIP          keyword
*--------------------------------------------------------------*
*        NIP keyword.                                          *
*--------------------------------------------------------------*
         KWDEFINE NAME=NIP,        Define the NIP                +
               ROUTINE=KWNIP         keyword
*--------------------------------------------------------------*
*        SWB keyword.                                          *
*--------------------------------------------------------------*
         KWDEFINE NAME=SWB,        Define the SWB                +
               ROUTINE=KWSWB         keyword
*--------------------------------------------------------------*
*        WTYPE keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=WTYPE,      Define the WTYPE              +
               ROUTINE=KWWTYPE       keyword
*--------------------------------------------------------------*
*        WLEN keyword.                                         *
*--------------------------------------------------------------*
         KWDEFINE NAME=WLEN,       Define the WLEN               +
               ROUTINE=KWWLEN        keyword
*--------------------------------------------------------------*
*        DUPJ keyword.                                         *
*--------------------------------------------------------------*
         KWDEFINE NAME=DUPJ,       Define the DUPJ               +
               ROUTINE=KWDUPJ        keyword
*--------------------------------------------------------------*
*        FABEND keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=FABEND,     Define the FABEND             +
               ROUTINE=KWFABEND      keyword
*--------------------------------------------------------------*
*        SELHOLD keyword.                                      *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELHOLD,    Define the SELHOLD            +
               ROUTINE=KWSLHLD       keyword
*--------------------------------------------------------------*
*        SELWTR keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELWTR,     Define the SELWTR             +
               ROUTINE=KWSLWTR       keyword
*--------------------------------------------------------------*
*        SELXWTR keyword.                                      *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELXWTR,    Define the SELXWTR            +
               ROUTINE=KWSLXWR       keyword
*--------------------------------------------------------------*
*        SELTSO keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELTSO,     Define the SELTSO             +
               ROUTINE=KWSLTSO       keyword
*--------------------------------------------------------------*
*        SELALL keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=SELALL,     Define the SELALL             +
               ROUTINE=KWSLALL       keyword
*--------------------------------------------------------------*
*        DDEL keyword.                                         *
*--------------------------------------------------------------*
         KWDEFINE NAME=DDEL,       Define the DDEL               +
               ROUTINE=KWDDEL        keyword
*--------------------------------------------------------------*
*        DKEEP keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=DKEEP,      Define the DKEEP              +
               ROUTINE=KWDKEEP       keyword
*--------------------------------------------------------------*
*        DRHLD keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=DRHLD,      Define the DRHLD              +
               ROUTINE=KWDRHLD       keyword
*--------------------------------------------------------------*
*        DRNPR keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=DRNPR,      Define the DRNPR              +
               ROUTINE=KWDRNPR       keyword
*--------------------------------------------------------------*
*        DHOLD keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=DHOLD,      Define the DHOLD              +
               ROUTINE=KWDHOLD       keyword
*--------------------------------------------------------------*
*        DREL keyword.                                         *
*--------------------------------------------------------------*
         KWDEFINE NAME=DREL,       Define the DREL               +
               ROUTINE=KWDREL        keyword
*--------------------------------------------------------------*
*        DCKPT keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=DCKPT,      Define the DCKPT              +
               ROUTINE=KWDCKPT       keyword
*--------------------------------------------------------------*
*        DNWR keyword.                                         *
*--------------------------------------------------------------*
         KWDEFINE NAME=DNWR,       Define the DNWR               +
               ROUTINE=KWDNWR        keyword
*--------------------------------------------------------------*
*        DCLASS keyword.                                       *
*--------------------------------------------------------------*
         KWDEFINE NAME=DCL,        Define the DCL=               +
               ROUTINE=KWDCLS        keyword
*--------------------------------------------------------------*
*        DDEST keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=DD,         Define the DD=                +
               ROUTINE=KWDDST        keyword
*--------------------------------------------------------------*
*        DFORM keyword.                                        *
*--------------------------------------------------------------*
         KWDEFINE NAME=DF,         Define the DF=                +
               ROUTINE=KWDFORM       keyword
*--------------------------------------------------------------*
*        DCOPY COUNT keyword.                                  *
*--------------------------------------------------------------*
         KWDEFINE NAME=DC,         Define the DC=                +
               ROUTINE=KWDCOPY       keyword
*--------------------------------------------------------------*
*        DWTR keyword.                                         *
*--------------------------------------------------------------*
         KWDEFINE NAME=DW,         Define the DW=                +
               ROUTINE=KWDWTR        keyword
*--------------------------------------------------------------*
*        GET keyword.                                          *
*--------------------------------------------------------------*
         KWDEFINE NAME=GET,        Define the GET T              +
               ROUTINE=KWGET         keyword
*--------------------------------------------------------------*
```

```
*       COUNT keyword.                                  *
*-------------------------------------------------------*
        KWDEFINE NAME=COUNT,      Define the COUNT              +
              ROUTINE=KWCOUNT       keyword
*-------------------------------------------------------*
*       BULK keyword.                                   *
*-------------------------------------------------------*
        KWDEFINE NAME=BULK,       Define the BULK               +
              ROUTINE=KWBULK        keyword
*-------------------------------------------------------*
*       NCLASS keyword.                                 *
*-------------------------------------------------------*
        KWDEFINE NAME=NCL,        Define the NCL=               +
              ROUTINE=KWNCLS        keyword
*-------------------------------------------------------*
*       NDEST keyword.                                  *
*-------------------------------------------------------*
        KWDEFINE NAME=ND,         Define the ND=                +
              ROUTINE=KWNDST        keyword
*-------------------------------------------------------*
*       REL keyword.                                    *
*-------------------------------------------------------*
        KWDEFINE NAME=REL,        Define the REL                +
              ROUTINE=KWREL         keyword
*-------------------------------------------------------*
*       DEL keyword.                                    *
*-------------------------------------------------------*
        KWDEFINE NAME=DEL,        Define the DEL                +
              ROUTINE=KWDEL         keyword
*-------------------------------------------------------*
*       ONEDS keyword.                                  *
*-------------------------------------------------------*
        KWDEFINE NAME=ONEDS,      Define the ONEDS              +
              ROUTINE=KWONEDS       keyword
*-------------------------------------------------------*
*       ONEJOB keyword.                                 *
*-------------------------------------------------------*
        KWDEFINE NAME=ONEJOB,     Define the ONEJOB             +
              ROUTINE=KWONEJOB      keyword
*-------------------------------------------------------*
*       DEFAULT keyword.                                *
*-------------------------------------------------------*
        KWDEFINE NAME=DEFAULT,    Define the DEFAULT            +
              ROUTINE=KWDFLT        keyword
TBLEND  EQU   *
TBLCOUNT EQU  (TBLEND-TBLSTART)/ENTRYSIZ  Number of entries
*-------------------------------------------------------*
*       Literals                                        *
*-------------------------------------------------------*
        LTORG
        DROP  R10                 KEYPROC
        PUSH  USING
        DROP  ,
*-------------------------------------------------------*
*   ESTAE exit                                          *
*-------------------------------------------------------*
ESTAEXIT DS   OH                  ESTAE EXIT
        USING SDWA,R1             SDWA ADDRESSABILITY
        LR    R6,R15              SET BASE REG
        USING ESTAEXIT,R6
        LR    R7,R14              SAVE RETURN ADDRESS
        C     R0,TWELVE           SDWA PRESENT?
        BC    NE,GOTSDWA          Yep, use it
        L     R10,BASEREG-ESTAPARM(,R2)
        L     R9,BASEREG2-ESTAPARM(,R2)
        L     R13,@LDA-ESTAPARM(,R2)
        USING LDASTRT,R13           and tell the assembler
        OI    LDAFLG2,FLGESTXT    SET THE ABEND BIT
        OI    LDAFLG2,FLGERROR    SET THE ABEND BIT
        ST    R1,COMPCODE         SAVE THE ABEND CODE
        BAL   R4,ESTAEOUT
        L     R0,@RETRY-ESTAPARM(,R2) RETRY ROUTINE ADDRESS
        LA    R15,4               SPECIFY RETRY
        BR    R7                  RETURN
*-------------------------------------------------------*
*       Processing with SDWA.                           *
*-------------------------------------------------------*
GOTSDWA  DS   OH
        L     R10,SDWAPARM        SET R10 TO PARM LIST
        L     R13,@LDA-ESTAPARM(,R10)
        L     R2,@RETRY-ESTAPARM(,R10)
        L     R10,BASEREG-ESTAPARM(,R10)
        L     R9,BASEREG2-ESTAPARM(,R10)
        OI    LDAFLG2,FLGERROR    SET THE ABEND BIT
        OI    LDAFLG2,FLGESTXT    SET THE ABEND BIT
        MVC   COMPCODE+1(3),SDWACMPC SAVE COMPLETION CODE
        MVC   ABENDREG,SDWAGRSV           AND REGS
        MVC   ABENDPSW,SDWAEC1            AND PSW
        LR    R3,R1                Save R1 over message
        BAL   R4,ESTAEOUT
        LR    R1,R3                Restore R1 after message
        SETRP WKAREA=(R1),RC=4,FRESDWA=YES,RETADDR=(R2),DUMP=NO
        BR    R7
        DROP  R1                  SDWA
TWELVE   DC   F'12'
*-------------------------------------------------------*
*       Issue ESTAE message                             *
*-------------------------------------------------------*
ESTAEOUT DS   OH
        TM    LDAFLG2,FLGBACKG    Running in background?
        BC    ALLON,ESTAEWTO      Yes - Issue WTO
        TPUT  =CL30'*ESTAE exit in control',30
        BR    R4
ESTAEWTO DS   OH
        WTO   'SAPI900I *ESTAE exit in control',ROUTCDE=2
        BR    R4
        DROP  R6                  ESTAEXIT
        POP   USING
*-------------------------------------------------------*
*       Literals                                        *
*-------------------------------------------------------*
        LTORG
TRPR     DS   0XL256              TRANSLATE and test table
        DC    256CL1' '
        ORG   TRPR+X'4A'
        DC    X'4A4B4C4D4E4F50'
        ORG   TRPR+X'5A'
        DC    X'5A5B5C5D5E5F6061'
        ORG   TRPR+X'6A'
        DC    X'6A6B6C6D6E6F'
        ORG   TRPR+X'79'
        DC    X'797A7B7C7D7E7F'
        ORG   TRPR+C'a'
        DC    C'abcdefghi'
        ORG   TRPR+C'j'
        DC    C'jklmnopqr'
        ORG   TRPR+X'A1'
        DC    X'A1',C'stuvwxyz'
        ORG   TRPR+X'C0'
        DC    X'C0',C'ABCDEFGHI'
        ORG   TRPR+X'D0'
        DC    X'D0',C'JKLMNOPQR'
        ORG   TRPR+X'E0'
        DC    X'E040',C'STUVWXYZ'
        ORG   TRPR+C'0'
        DC    C'0123456789'
        ORG
        TITLE 'LDA - Local Data Area'
*-------------------------------------------------------*
*       This is the local data area, pointed to by R13. *
*-------------------------------------------------------*
        ORG   SAPI+3*4096
LDASTRT  DS   0D
LDASAVE  DC   18F'0'              18 word save area for OS
LDAID    DC   CL4'LDA'            Eyecatcher
*-------------------------------------------------------*
*       Translate and Test table.                       *
*-------------------------------------------------------*
TRANTEST DS   0XL256              TRANSLATE and test table
        DC    256XL1'0'
        ORG   *-240
TRANSAVE DC   5F'0'
TRANSLAT DS   0XL256              Table for translation to EBCDIC
        ORG   TRANSLAT+C'0'
        DC    C'0123456789ABCDEF'
        ORG
*-------------------------------------------------------*
*       Miscellaneous constants                         *
*-------------------------------------------------------*
CHARZERO DC   CL8'00000000'       Initial value for workarea
HEXZERO  DC   XL8'00000000'       Initial value for workarea
*
*-------------------------------------------------------*
*       Flag bytes                                      *
*-------------------------------------------------------*
LDAFLG1  DC   X'0'                Flag byte number 1
FLGESTAE EQU  X'80'               ESTAE is active
FLGSQA   EQU  X'40'               Got SQA
FLGSUP   EQU  X'20'               MODE=SUP state
FLGSAPI  EQU  X'08'               SAPI CNTL request is owed
FLGALLOC EQU  X'04'               SYSOUT dataset is allocated.
FLGOPEN  EQU  X'02'               SYSOUT dataset is open
FLGDOM   EQU  X'01'               Action message outstanding.
*
LDAFLG2  DC   X'0'                Flag byte number 2
FLGERROR EQU  X'80'               Error detected
FLGESTXT EQU  X'40'               ESTAE detected error
FLGCLNUP EQU  X'20'               Cleanup in progress
FLGESCAN EQU  X'10'               Keyword scan error
FLGCMDCH EQU  X'08'               Selection changed via cmd
FLGBACKG EQU  X'04'               Running in background
FLGINIIP EQU  X'02'               Initial input
FLGSTAX  EQU  X'01'               STAX active
*
*-------------------------------------------------------*
*       Trace work area                                 *
*-------------------------------------------------------*
TRCREG   DC   F'0'                Save area for a reg
*
DWORK1   DC   D'0'                Double word for working
        DC    X'00'               Pad byte
FWORK1   DC   F'0'                Single work for working
DWORK2   DC   D'0'                Ditto
        DC    X'00'               Ditto
FWORK2   DC   F'0'                Ditto
```

```
*
OPTIONS  DS    OF                  Start of options
*-------------------------------------------------------------*
*        SAPIFLG1 - SAPI processing flag                      *
*-------------------------------------------------------------*
SAPIFLG1 DS    XL1                 Process flags
SAPIONED EQU   X'80'               Process one ds and stop
SAPIONEJ EQU   X'40'               Process one job and stop
SAPIABND EQU   X'20'               Force an abend
OPTLEN   EQU   *-OPTIONS
*-------------------------------------------------------------*
*        SSOB                                                 *
*-------------------------------------------------------------*
@LDASSOB DC    A(0)                pointer to ssob
LDASSOB  DC    XL(SSOBHSIZ)'00'    area for SSOB
LDASSS2  DC    XL256'00'              and SSS2
         DC    XL256'00'                 ....still SSS2
         DC    XL256'00'                 ....still SSS2
         DC    XL256'00'                 ....still SSS2
         DC    XL100'00'                 ....still SSS2
LDATKOUT DS    OCL80               Output token field
         DC    AL1(80,1)           length, version
         DS    XL78
*-------------------------------------------------------------*
*        SWB retrieve fields                                  *
*-------------------------------------------------------------*
MYPLADR  DC    F'0'                Pointer to SJFPLLST
MYPLLST  DS    XL(SJRELGTH)        SJF Retrieve Parm List
MYAREA   DS    XL512               Ought to cover a couple of keys!
MYKEYLST DC    CL8'FORMS'          Keylist - Retrieve FORMS
         DS    F                   Pointer
         DC    CL8'DEST'           Keylist - Retrieve DEST
         DS    F                   Pointer
         DC    CL8'FLASH'          Keylist - Retrieve FLASH
         DS    F                   Pointer
*-------------------------------------------------------------*
*        Dynamic allocation work area                         *
*-------------------------------------------------------------*
@LDAS99  DC    A(0)                pointer to S99RB
S99RBLEN EQU   S99RBEND-S99RB      EQUATE for length of S99RB
LDAS99RB DC    XL(S99RBLEN)'00'    area for the RB
LDAS99RX DS    OF                  area for the RBX
         DC    XL(L'S99ERSN+S99ERSN-S99EID)'00'
         ORG   LDAS99RX
         DC     CL6'S99RBX'
         DC     AL1(S99RBXVR)          VERSION NUMBER
         DC     AL1(S99ERMSG)          PROCESSING OPTIONS
         ORG
LDATXTAL DS    OH                  Allocation text unit ptrs
LDATXTA1 DC    A(0)                First pointer
LDATXTA2 DC    A(0)                Second pointer
LDATXTA3 DC    A(0)                Third pointer                  @RFC
LDATXTA4 DC    A(0)                Fourth and last pointer        @RFA
LDATXTUN DS    OH                  Deallocation text unit ptrs
LDATXTU1 DC    A(0)                First/last pointer
*-------------------------------------------------------------*
*        Allocation text units                                *
*-------------------------------------------------------------*
LDADSNAM DC    Y(DALDSNAM)         DSN to be allocated   1st
         DC    Y(1)                  One entry
         DC    Y(L'LDADSN)           Length of entry
LDADSN   DC    XL44'00'              Data set name
LDASSNAM DC    Y(DALSSREQ)         Subsystem request     2nd
         DC    Y(1)                  One entry
         DC    Y(L'LDAJES)           Length of entry
LDAJES   DC    XL4'00'               Subsystem name
LDADDNAL DC    Y(DALRTDDN)         DDname returned       3rd
         DC    Y(1)                  One entry
         DC    Y(L'LDADDNMA)         Length of entry
LDADDNMA DC    XL8'00'               DDname
*-------------------------------------------------------------*
*        Unallocation text units                              *
*-------------------------------------------------------------*
LDADDNUN DC    Y(DUNDDNAM)         Unallocate DDname     1st
         DC    Y(1)                  One entry
         DC    Y(L'LDADDNMU)         Length of entry
LDADDNMU DC    XL8'00'               DDname to unallocate
*
CPPLADDR DS    A                   Address of CPPL
LDARTNCD DS    H                   Return code for messages
HOBON    EQU   X'80'               High order bit on for OI
INDCB    DCB   DSORG=PS,           DCB for open/close                +
               MACRF=GL,                                           +
               EODAD=CLOSEIT
*-------------------------------------------------------------*
*        This portion of the LDA is specific to the external  *
*        writer SAPIXWTR.                                     *
*-------------------------------------------------------------*
@@PARM   DC    A(0)                R1 on entry to routine
@LDASQA  DC    A(0)                Address of SQA 8-bytes
@SSIB    DC    A(0)                Address of SSIB for this job
KWDLEN   DC    F'00'               Keyword length for scan
OPERLEN  DC    F'00'               Operand length for scan
NEXTKWD  DC    F'00'               Starting point for next scan
REMLEN   DC    F'00'               Remaining length for scan
CURRKWD  DC    F'00'               Address of current keyword
CURROPER DC    F'00'               Address of current operand
LDAXWNAM DC    CL8' '              Step name of external wtr
*-------------------------------------------------------------*
*        ESTAE exit parms etc.                                *
*-------------------------------------------------------------*
ESTAPARM DS    OD
@LDA     DC    F'0'                Address of LDA
@RETRYP  DC    F'0'                Address if retry point
@RETRY   DC    F'0'                Address if retry routine
BASEREG  DC    F'0'                Main base register contents
BASEREG2 DC    F'0'                Main base register #2 contents
ESTAE    ESTAE ESTAEXIT,PARAM=0,CT,TERM=YES,MF=L
COMPCODE DC    F'0'                Completion code for abend
ABENDPSW DC    D'0'                PSW at abend
ABENDREG DC    XL64'00'            registers 0:15 at abend
*-------------------------------------------------------------*
*        Messages                                             *
*-------------------------------------------------------------*
MSG1     WTO   TEXT=0,ROUTCDE=2,MF=L
MSG2     WTO   'SAPI000I                                              +
                                                        ',          +
               DESC=(5),ROUTCDE=(2),MF=L
MSG2END  EQU   *
MSG2LEN  EQU   MSG2END-MSG2
MSG2ID   EQU   MSG2+4,8
MSG2TEXT EQU   MSG2+13,MSG2LEN-20
*-------------------------------------------------------------*
*        ECB list for the writer an communication area.       *
*-------------------------------------------------------------*
LDAECBL  DS    OF                  ECB list to wait on
@WTRECB  DC    A(0)                SAPI ECB address
@COMECB  DC    A(0)                COMM ECB address
COMECB   DC    A(0)                COMM ECB
ZEROES   DC    F'0'                Dummy ECB
SAPIECB  DC    F'0'                SAPI ECB
COMMADDR DC    A(0)                Address of stuff from EXTRACT
COMMLEN  DC    H'0'                Length of command
COMMBUFF DC    CL128' '            text of command
COMREGS  DC    6F'0'               Reg SA for COMCHECK subroutine
OPVERB   DC    XL1'00'             place to keep CIBVERB
*
LDACLSR  DC    CL1' '              Class of work returned by jes
LDADESTR DC    CL18'SAPI    '      Default destination of desired work
LDAJOBIR DC    XL8'00'             Job id of work returned by jes
LDAJOBNR DC    CL8' '              Jobname of work returned by jes
*
DEFCLASS DC    CL36'01ZAW'         Default class to search for work
DEFDEST  DC    CL18'SAPI'          Default destination of desired work
DEFJOBNM DC    CL8'SAPI*'          Default jobname used to select work
DEFJOBIL DC    CL8'JOB00005'       Lower job number range
DEFJOBIH DC    CL8'JOB00500'       Upper job number range
*-------------------------------------------------------*
AE       DS    OH
         STM   R14,R12,12(R13)
         USING AE,R12
         LR    R12,R15
         LA    R2,AE
         S     R2,COMECBO
         POST  (R2),123
         LM    R14,R12,12(R13)
         BR    R14
         DROP  R12
*-------------------------------------------------------*
COMECBO  DC    A(AE-COMECB)
STAXL    STAX  AE,USADDR=0,REPLACE=NO,DEFER=NO,MF=L
OB       DC    CL6'SAPI!'
BLANKS   DC    CL128' '
*-------------------------------------------------------------*
RECLEN   DS    H                   LENGTH OF OUTPUT RECORD
RECTEXT  DS    CL150               UP TO 150 BYTES OF SYSOUT
EXTRACT  EXTRACT FIELDS=COMM,MF=L
LDAEND   DS    OD
LDASIZE  EQU   LDAEND-LDASTRT
LDAMAX   EQU   4096-LDASIZE        Generate assembly error
*-------------------------------------------------------------*
SAPIHELP CSECT
         SAVE  (14,12)
         USING SAPIHELP,R12
         LR    R12,R15
         LA    R2,HTB
         LA    R3,L'HTB
SAPIHTLP DS    OH
         CLI   0(R2),X'FF'
         BC    EQ,SAPIHTRE
         TPUT  (R2),(R3)
         AR    R2,R3
         B     SAPIHTLP
SAPIHTRE DS    OH
         RETURN (14,12),RC=0
*-------------------------------------------------------------*
*        Keyword table.                                       *
*-------------------------------------------------------------*
HTB DS OCL60
*              1         2         3         4         5         6
*      '12345678901234567890123456789012345678901234567890123456789'
    DC C'* ------------------------------------------------------- *'
    DC C'*  SAPI Help for Keywords                                  *'
    DC C'*                        KEYWORD       FLAG        FIELD   *'
    DC C'* ------------------------------------------------------- *'
    DC C'* -> Stop processing    END/STOP       n/a          n/a    *'
```

```
DC C'* -> Help (TSO only)    HELP          n/a          n/a    */
DC C'* -> Process one JOB    ONEJOB        n/a          n/a    */
DC C'* -> Process one DS     ONEDS         n/a          n/a    */
DC C'*                                                         */
DC C'* -> TSO Hold           SELTSO     (SSS2SHLD)      n/a    */
DC C'* -> EXTWTR Hold        SELXWTR    (SSS2SXWH)      n/a    */
DC C'* -> TSO/EXTWTR Hold    SELHOLD    (SSS2SHOL)      n/a    */
DC C'* -> WTR                SELWTR     (SSS2SWTR)      n/a    */
DC C'* -> TSO/EXTWTR/WTR     SELALL     (SSS2SAWT)      n/a    */
DC C'* -> CLASS              CL=        (SSS2SCLS)    SSS2CLSL */
DC C'* -> DEST               D=         (SSS2SDST)    SSS2DEST */
DC C'* -> JOBNAME            JBN=       (SSS2SJBN)    SSS2JOBN */
DC C'* -> JOBID              JBI=       (SSS2SJBI)    SSS2JBIL */
DC C'* -> JOBID              HJBI=      (SSS2SJBI)    SSS2JBIH */
DC C'*                                                         */
DC C'* -> PROGRAM NAME       W=         (SSS2SPGM)    SSS2PGMN */
DC C'* -> FORM               F=         (SSS2SFRM)    SSS2FORM */
DC C'* -> CREATOR USERID     U=         (SSS2SCRE)    SSS2CREA */
DC C'* -> PROCESS MODE       PM=        (SSS2SPRM)    SSS2PRMO */
DC C'* -> w/IP addrss        IP         (SSS2SIPA)      n/a    */
DC C'* -> w/o IP address     NIP        (SSS2SIPN)      n/a    */
DC C'* -> FCB                C=         (SSS2SFCB)    SSS2FCB  */
DC C'* -> UCS                U=         (SSS2SUCS)    SSS2UCS  */
DC C'* -> SWB                SWB        (SSS2FSWB)      n/a    */
DC C'*                                                         */
DC C'* -> Started Task       SELSTC     (SSS2SSTC)      n/a    */
DC C'* -> Time Sharing User  SELTSU     (SSS2STSU)      n/a    */
DC C'* -> Batch job          SELJOB     (SSS2SJOB)      n/a    */
DC C'* -> APPC               SELAPC     (SSS2SAPC)      n/a    */
DC C'* -> No CPDS            NCPDS      (SSS2SCPN)      n/a    */
DC C'*                                                         */
DC C'* -> MOD                M=         (SSS2SMOD)    SSS2MOD  */
DC C'* -> FLASH              FL=        (SSS2SFLS)    SSS2FLSH */
DC C'* -> AGE                A=         (SSS2SAGE)    SSS2AGE  */
DC C'* -> LINE LIMIT         LL=        (SSS2SLIN)    SSS2LMIN */
DC C'*                       MLL=       (SSS2SLIN)    SSS2LMAX */
DC C'* -> PAGE LIMIT         PL=        (SSS2SPAG)    SSS2PMIN */
DC C'*                       MPL=       (SSS2SPAG)    SSS2PMAX */
DC C'* -> CHARS              CH=        (SSS2SCHR)    SSS2CHAR */
DC C'*                                                         */
DC C'* -> Get/Put            GET           n/a        SSS2TYPE */
DC C'*                                                         */
DC C'* Keywords applicable to dataset disposition for GET/PUT  */
DC C'* processing are as follows:                              */
DC C'* -> Delete ds          DDEL       (  n/a   )       n/a    */
DC C'* -> Keep ds            DKEEP      (SSS2DKPE)       n/a    */
DC C'* -> OP Hold            DRHLD      (SSS2RHLD)       n/a    */
DC C'* -> Do not process     DRNPR      (SSS2RNPR)       n/a    */
DC C'*      again                                              */
DC C'* -> Hold ds            DHOLD      (SSS2DHLD)       n/a    */
DC C'* -> Release ds         DREL       (SSS2DRLS)       n/a    */
DC C'* -> Null writer name   DNWR       (SSS2DNWR)       n/a    */
DC C'* -> New class          DCL=          n/a         SSS2DCLS */
DC C'* -> New dest           DD=           n/a         SSS2DDES */
DC C'* -> New forms          DF=           n/a         SSS2DFOR */
DC C'* -> New wtr name       DW=           n/a         SSS2DPGM */
DC C'* -> New copy count     DC=           n/a         SSS2CLFT */
DC C'*                                                         */
DC C'* -> Count request      COUNT         n/a         SSS2TYPE */
DC C'* -> Bulk Modify req    BULK          n/a         SSS2TYPE */
DC C'*                                                         */
DC C'* Keywords applicable to Bulk Modify processing are as    */
DC C'* follows:                                                */
DC C'* -> New class          NCL=       (SSS2SETC)     SSS2CLAS */
DC C'* -> New destination    ND=        (SSS2ROUT)     SSS2DES2 */
DC C'* -> Release            REL        (SSS2RLSE)       n/a    */
DC C'* -> Delete             DEL        (SSS2DELC)       n/a    */
DC C'*                                                         */
DC C'*      For error processing scenarios, the following      */
DC C'*      keywords can be used (the application code will    */
DC C'*      set an incorrect value).                           */
DC C'*                                                         */
DC C'* -> wrong type         WTYPE                    SSS2TYPE */
DC C'* -> wrong length       WLEN                     SSS2LEN  */
DC C'* -> duplicate jobs     DUPJ                             */
DC C'* -> forced abend       ABEND                            */
DC C'* ------------------------------------------------------- */
DC X'FFFF'
*---------------------------------------------------------------*
*        THE END                                                *
*---------------------------------------------------------------*
         END   SAPI
```

Appendix B. Sample ANFUXMSG Exit

Internet access to code

Internet access to source code found in SG24-2067 can be accessed via the Internet.

Sample code/procedures are available in softcopy on the internet from the redbooks webserver.

Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG24xxxx

Alternatively you can go to:

http://www.redbooks.ibm.com

and select SEARCH and then Additional Redbook Materials. (or follow the instructions given since the web pages change frequently!)

The IP PrintWay message exit ANFUXMSG allows you to either suppress or modify the ID or text of any message that IP PrintWay issues to the IP PrintWay message-log data set, including messages created by another IP PrintWay exit.

The sample exit included in this section is divided into two pieces:

1. *IP PrintWay ANFUXMSG Dynamic Exit Stub* controls the invocation of the actual exit code (called UXMSFANF) through the OS/390 dynamic exit facility.
2. *IP PrintWay ANFUXMSG Exit - Issue Messages to SYSLOG* issues all messages passed to the exit to SYSLOG.

The following section shows the dynamic exit stub for the actual IP Printway ANFUXMSG exit. (This exit must be called ANFUXMSG.)

```
ANFUXMSG TITLE 'IP PrintWay ANFUXMSG DYNAMIC EXIT STUB'
ANFUXMSG CSECT
ANFUXMSG AMODE 31
ANFUXMSG RMODE ANY
         YREGS
*
*   This ANFUXMSG dynamic exit stub program uses the exit point name
*   ANF_ANFUXMSG for the IP Printway ANFUXMSG exit.  If the exit point
*   is not defined, it defines it. It does not, however, associate
*   any programs with the exit.
*
*   The actual exit program - for example UXMSGANF -  should be added
*   through parmlib PROGxx member or be defined through SETPROG the
*   operator command:
*
*   PROGxx parmlib definition:
*
*        EXIT ADD
*             EXITNAME(ANF_ANFUXMSG)
*             MODNAME(UXMSGANF)          /* <== Change */
*             STATE(ACTIVE|INACTIVE)
*             JOBNAME=PRINTWAY           /* <== Change */
*
*   Operator command:
*
*        SETPROG EXIT,ADD,EXITNAME=ANF_ANFUXMSG,MODNAME=UXMSGANF, ...
*        ... STATE=ACTIVE|INACTIVE,JOBNAME=PRINTWAY
*
*        The SETPROG EXIT,MODIFY commands allows you to dynamically
*        change the status of the exit from ACTIVE to INACTIVE or
*        vise versa:
*
*        SETPROG EXIT,MODIFY,EXITNAME=ANF_ANFUXMSG,MODNAME=UXMSGANF, ..
*        ... STATE=ACTIVE|INACTIVE,JOBNAME=PRINTWAY
*
*
*  You can use the message exit (ANFUXMSG) to either suppress or
*  modify the ID or text of any message that IP PrintWay issues to the
*  IP PrintWay message-log data set.
*
*  NOTE: This exit stub request to write all messages unchanged
*  to the IP PrintWay message-log data set by setting in ANFUEXTP
*  the flag XTPM2MDS=1.
*
*  At entry to the Message exit, the registers contain the following:
*
*  Register 1   Pointer to the address of the ANFUEXTP control block.
*  Register 13  Address of an 18-word save area for saving the caller's
*               registers.
*  Register 14  Return address.
*  Register 15  Entry-point address.
*
         SAVE  (14,12),,ANFUXMSG_&SYSDATE._&SYSTIME
         LR    R12,R15              Set base reg
         USING ANFUXMSG,R12
         LR    R3,R1                Save XTP pointer pointer
         L     R2,0(,R1)            Get address of the ANFUEXTP
         USING ANFUEXTP,R2
         OI    XTPMSGFL,XTPM2MDS    Issue message to message data set
         LA    R0,WSLENGTH             GET LENGTH OF WORKING STORAGE
         GETMAIN R,LV=(0)              GETMAIN WORKING STORAGE
         LR    R4,R1                   ADDRESS OF WS TO R4
         LR    R6,R1                   ADDRESS OF WS TO R6
         LA    R5,WSLENGTH             GET LENGTH OF WORKING STORAGE
         SLR   R7,R7                   CLEAR PATTERN REGISTER
         MVCL  R4,R6                   CLEAR WORKING STORAGE
         ST    R1,8(,R13)              SAVE ADDR OF NEW SAVE IN OLD
         ST    R13,4(,R1)              SAVE ADDR OF OLD SAVE IN NEW
         LR    R13,R1                  SETUP NEW SAVE/WS
         USING WS,R13                  TELL ASSEMBLER
* ------ QUERY EXIT
         MVC   ERRMSGF,=CL6'QUERY'
         XC    LRSNCODE,LRSNCODE
         XC    LRETCODE,LRETCODE
         CSVDYNEX  REQUEST=QUERY,EXITNAME=LEX,QTYPE=CALL,              C
               RETCODE=LRETCODE,RSNCODE=LRSNCODE,WORKAREA=WSDYNQRY,    C
               MF=(E,WSDYNEX)
         NC    LRSNCODE,=AL4(CSVDYNEXRSNCODEMASK) AND OFF EXTRA BITS
         CLC   LRETCODE,=AL4(CSVDYNEXRC_OK)
         BE    CALLROUT                EXIT AND MODULE AVAIABLE
         CLC   LRETCODE,=AL4(CSVDYNEXRC_WARN)
         BNE   ERRWTO
         CLC   LRSNCODE,=AL4(CSVDYNEXRSNNOMODULES)
         BE    RETURNOO                  NO MODULES IN EXIT ... END
         CLC   LRSNCODE,=AL4(CSVDYNEXRSNQUERYNOTFOUND)
         BE    DEFROUT             EXIT NOT DEFINED ..... GO DEFINE IT
         CLC   LRSNCODE,=AL4(CSVDYNEXRSNIMPLICITLYDEFINED)
         BE    DEFROUT             EXIT ONLY IMPLICIT ... GO DEFINE IT
         B     ERRWTO
* ------ DEFINE EXIT
DEFROUT  DS    OH
         MVC   ERRMSGF,=CL6'DEFINE'
```

```
         XC    LRSNCODE,LRSNCODE
         XC    LRETCODE,LRETCODE
         CSVDYNEX  REQUEST=DEFINE,EXITNAME=LEX,AMODE=31,PERSIST=IPL,   C
               FASTPATH=NO,RETCODE=LRETCODE,RSNCODE=LRSNCODE,          C
               MF=(E,WSDYNEX)
         CLC   LRETCODE,=AL4(CSVDYNEXRC_OK)
         BE    CALLROUT                    ONLY RC=00 ACCEPTED
         B     ERRWTO
* ------ CALL ACTUAL EXIT MODULE
CALLROUT DS    0H
         LR    R1,R3                  Restore original parm reg
         MVC   ERRMSGF,=CL6'CALL'
         XC    LRSNCODE,LRSNCODE
         XC    LRETCODE,LRETCODE
         MVC   LRUBBITS,=X'40040000'   PASS R1 AND R13 TO EXIT MOD
         L     R10,4(,R13)
         ST    R13,LRUBR13             STORE OUR SAVE AREA FOR EXIT MOD
         LA    R10,20(,R10)
         MVC   LRUBR1(L'LRUBR1),4(R10) STORE OLD R1 FOR EXIT
         XC    LNXTTKN,LNXTTKN          INITIALIZE NEXT TOKEN
         CSVDYNEX  REQUEST=CALL,EXITNAME=LEX,FASTPATH=NO,              C
               NEXTTOKEN=LNXTTKN,RUB=LRUB,RETINFO=LAST,                C
               RETAREA=LRETAREA,RETLEN==AL4(RETALEN),                  C
               RETCODE=LRETCODE,RSNCODE=LRSNCODE,                      C
               MF=(E,WSDYNEX)
         NC    LRSNCODE,=AL4(CSVDYNEXRSNCODEMASK) AND OFF EXTRA BITS
         CLC   LRETCODE,=AL4(CSVDYNEXRC_OK)
         BE    SETRC                     THE END, GET R15 FROM MODULE
         CLC   LRETCODE,=AL4(CSVDYNEXRC_WARN)
         BNE   ERRWTO
         CLC   LRSNCODE,=AL4(CSVDYNEXRSNNOMODULES)
         BE    RETURN00
         B     ERRWTO
* ------ ISSUE AN ERROR MESSAGE
ERRWTO   DS    0H
         MVC   ERRMSGX,LEX
         MVC   ERRMSGA,=C' - '
         LA    R2,LRETAREA
         USING EXRET,R2
         TM    EXRETFLAGS,EXRETABEND      DID MODULE ABEND ?
         BZ    ERRWTX
         MVC   ERRMSGB(ERRMSGE-ERRMSGB),=CL20' ABEND      '
         UNPK  ERRMSGB+7(9),EXRETABENDCODE(5)
         TR    ERRMSGB+7(8),X2C-C'0'
         MVI   ERRMSGB+7+8,C' '
         B     ERRWTY
         DROP  R2
ERRWTX   DS    0H
         MVC   ERRMSGB,=C' RC = '
         UNPK  ERRMSGR(L'ERRMSGR+1),LRETCODE+3(2)
         TR    ERRMSGR(L'ERRMSGR),X2C-C'0'
         MVC   ERRMSGC,=C' RSN = '
         UNPK  ERRMSGS(L'ERRMSGS+1),LRSNCODE+2(3)
         TR    ERRMSGS(L'ERRMSGS),X2C-C'0'
ERRWTY   DS    0H
         MVC   ERRMSGT,=AL2(ERRMSGE-ERRMSGX)
         MVC   ERRWTOL(ERRWTOLE-ERRWTOL),EWTOL
         LA    R2,ERRMSGT
         WTO   TEXT=(2),MF=(E,ERRWTOL)
         B     RETURN00
* ------ BACK TO CALLER
SETRC    DS    0H
         LA    R2,LRETAREA
         USING EXRET,R2
         TM    EXRETFLAGS,EXRETABEND      DID MODULE ABEND ?
         BNZ   ERRWTO
         L     R3,EXRETCODE       INSERT RETURN CODE FROM EXIT MODULE
         B     RETURN             RETURN TO FREEMAIN, R3 CONTAINS RC
RETURN00 DS    0H
         SLR   R3,R3              RESET R3
RETURN   DS    0H
         LA    R0,WSLENGTH              GET LENGTH OF WS
         LR    R1,R13                   ADDRESS OF WS TO R1
         L     R13,4(,R13)              GET ADDRESS OF PREVIOUS SAVE
         DROP  R13                      TELL ASSEMBLER
         FREEMAIN R,LV=(0),A=(1)        FREE WORKING STORAGE
         LR    R15,R3                   SET RETURN CODE
         RETURN (14,12)                 RETURN TO CALLER - NO RC
* ------ DATA AREAS
LEX      DC    CL16'ANF_ANFUXMSG'
X2C      DC    C'0123456789ABCDEF'
EWTOL    WTO   TEXT=,MF=L
EWTOLE   DS    F
         LTORG
* ------ DSECTS
WS       DSECT                          WORKING STORAGE DSECT
DSSA     DS    18F                      FIRST WORD OF SAVEAREA
DYNMODEL CSVDYNEX MF=(L,WSDYNEX)
WSDYNQRY DS    CL512
RETALEN  EQU   L'EXRETHDR+L'EXRETINFO
LRETAREA DS    CL(RETALEN)
LNXTTKN  DS    D
LRETCODE DS    F
LRSNCODE DS    F
LRUB     DS    0XL12
LRUBBITS DS    BL.32
LRUBR1   DS    A
LRUBR13  DS    A
ERRMSGT  DC    AL2(ERRMSGE-ERRMSGX)
ERRMSGX  DC    CL16' '
ERRMSGA  DC    C' - '
ERRMSGF  DC    CL6'      '
ERRMSGB  DC    C' RC = '
ERRMSGR  DC    CL2' '
ERRMSGC  DC    C' RSN = '
ERRMSGS  DC    CL4' '
ERRMSGE  DC    C' '
ERRWTOL  WTO   TEXT=,MF=L
ERRWTOLE DS    F
         DS    0D
WSLENGTH EQU   *-WS
         DROP  R12
         PRINT NOGEN
         ANFUEXTP
         CSVEXRET
         END ANFUXMSG
```

The following shows the actual IP Printway ANFUXMSG exit (called UXMSGANF):

```
UXMSGANF TITLE 'IP PrintWay ANFUXMSG Exit - Issue Messages to SYSLOG'
UXMSGANF CSECT
UXMSGANF AMODE 31
UXMSGANF RMODE ANY
         YREGS
*
*  This IP PrintWay message exit re-routes all messages to SYSLOG.
*  It does not change the actual message ids or the message text.
*  Nor does it suppress any messages.
*
*  If a message is less than 125 characters long, it will be issued
*  as a single line WTO message with the MCSFLAG=HRDCPY set.
*  If the message length exceeds 125 characters, the message is split
*  into roughly 70 character pieces at word boundary and issued as
*  a multiline message with the MCSFLAG=HRDCPY set.
*
*  At entry to the Message exit, the registers contain the following:
*
*  Register 1   Pointer to the address of the ANFUEXTP control block.
*  Register 13  Address of an 18-word save area for saving the caller's
*               registers.
*  Register 14  Return address.
*  Register 15  Entry-point address.
*
*  Note: The Dynamic Exit Stub ANFUXMSG has turned XTPMSGFL.XTPM2MDS
*  bit on (i.e. write messages to message data set)
*
         SAVE  (14,12),,UXMSGANF_&SYSDATE._&SYSTIME
         LR    R12,R15              Set base reg
         USING UXMSGANF,R12
         L     R2,0(,R1)            Get address of the ANFUEXTP
         USING ANFUEXTP,R2
         OI    XTPMSGFL,XTPM2MDS    Issue message to message data set
         LA    R0,PRIVATE-PRIVAT
         GETMAIN R,LV=(0)
         LR    R3,R1
         USING PRIVAT,R3
         XR    R4,R4
         ICM   R4,3,XTPMSGLN        Length of the message
         BZ    RETURN
         L     R5,XTPMSGP           Address of the message
         CH    R4,=AL2(L'PRIVATC)   More than SL-WTO can handle
         BH    SPLITIT              Yes - Do ML-WTO
* ------ Issue Single line WTO
         MVC   PRIVATS,WTOSL
         STH   R4,PRIVATB
         BCTR  R4,0
         MVC   PRIVATC(*-*),0(R5)   Copy message
         EX    R4,*-6
         WTO   TEXT=PRIVATB,MF=(E,PRIVATS)
         B     DONEHERE
* ------ Issue Multiline WTO(s)
SPLITIT  DS    0H
         MVC   PRIVATM,WTOML
         SLR   R10,R10              Init connect id
         LR    R7,R5                Copy message address
         SLR   R6,R6                Set R6 to..
         BCTR  R6,0                 .-1
         LA    R9,70(,R7)           Point at E.O. max split size
SPLITMOR DS    0H
         CLI   0(R9),C' '           Good point to split message?
         BE    SPLITWTO
         BXH   R9,R6,SPLITMOR
SPLITWTO DS    0H
         LR    R1,R9                Compute split..
         SR    R1,R7                .text length
         BP    SPLITMVC
         LA    R9,70(,R7)           Point at E.O. max split
         B     SPLITWTO
SPLITMVC DS    0H
```

```
         STH   R1,PRIVATB
         BCTR  R1,0
         MVC   PRIVATC(*-*),0(R7)   Copy message (up to 70 bytes)
         EX    R1,*-6
         SH    R4,PRIVATB           Calculate remaining message length
         BP    SPLITNOE             Some left - Not MLWTO end line
         OI    PRIVATF,X'10'        Set "E" on top of "D"
SPLITNOE DS    0H
         MVC   PRIVATD,PRIVATB      Save original WTO length
         CLI   PRIVATC,C' '         Strip first blank from the WTO?
         BNE   SPLITNOB             No - WTO
         LH    R1,PRIVATB           Adjust WTO..
         BCTR  R1,R0                .text..
         STH   R1,PRIVATB           ..length
         BCTR  R1,R0
         MVC   PRIVATC(*-*),PRIVATC+1 Strip
         EX    R1,*-6
SPLITNOB DS    0H
         WTO   TEXT=PRIVATB,CONNECT=(R10),MF=(E,PRIVATM)
         LR    R10,R1               Save connect id
         MVC   PRIVATB,PRIVATD      Restore original WTO length
         LTR   R4,R4
         BNP   DONEHERE
         LH    R1,PRIVATB           Get split length
         AR    R7,R1                Point at next message split
         LA    R9,70(,R7)           Point at E.O. max split
         CH    R4,=H'70'            Split size > max MLWTO line?
         BH    SPLITMOR             Yes - Keep splitting
         LA    R9,0(R4,R7)          Issue WTO for the..
         B     SPLITWTO             .last part of the split
         DROP  R2
DONEHERE DS    0H
         LR    R1,R3
         LA    R0,PRIVATE-PRIVAT
         FREEMAIN R,LV=(0),A=(1)
RETURN   DS    0H
         RETURN (14,12)
*
         DROP  R3
         LTORG
WTOSL    WTO   TEXT=,ROUTCDE=(7),MCSFLAG=(HRDCPY),DESC=(6),MF=L
WTOSLE   DS    0H
WTOML    WTO   TEXT=((0,D)),ROUTCDE=(7),MCSFLAG=(HRDCPY),DESC=(6),MF=L
WTOMLE   DS    0H
*
PRIVAT   DSECT
PRIVATS  DS    XL(WTOSLE-WTOSL)
         ORG   PRIVATS
PRIVATM  DS    XL(WTOMLE-WTOML)
PRIVATF  EQU   *-4,2,C'X'
         ORG
PRIVATD  DS    H
PRIVATB  DS    H
PRIVATC  DS    CL125
PRIVATE  DS    0D
         ANFUEXTP
         IKJTCB
         END UXMSGANF
```

Appendix C. Special Notices

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter 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 the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

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

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

The following document contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples contain 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.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

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

ACF/VTAM
Advanced Function Printing
AnyNet
CBIPO
CICS
CICS/MVS
DFSMS
DFSMSdfp
IBM
IMS/ESA
Advanced Function Presentation
AFP
APPN
CBPDO
CICS/ESA
DB2
DFSMS/MVS
FFST
IMS
Intelligent Printer Data Stream

IP PrintWay
MVS (logo)
NetSpool
Parallel Sysplex
PrintWay
RACF
ServicePac
SystemPac
IPDS
MVS/ESA
OS/390
Print Services Facility
PSF
S/390
System/390
VTAM

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Inc.

Microsoft, Windows, Windows NT, and the Windows 95 logo
are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is
used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or
registered trademarks of Intel Corporation in the U.S. and other
countries.

UNIX is a registered trademark in the United States and other
countries licensed exclusively through X/Open Company Limited.

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

Appendix D. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

D.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 233.

Short Title	Title	Order Number
OS/390 Release 3 Implementation	*OS/390 Release 3 Implementation*	SG24-2067
OS/390 Release 2 Implementation MVS, SMP/E, SDSF, and RMF	*OS/390 Release 2 Implementation*	SG24-4834
Version 5 Implementation Guide	*MVS/ESA Version 5 Implementation Guide*	SG24-4584
MVS 5.1 Presentation Guide	*MVS/ESA SP 5.1.0 Technical Presentation Guide*	GG24-4137
Parallel Sysplex Perf.	*S/390 Parallel Sysplex Performance*	SG24-4356
HCD and Dynamic I/O Reconfiguration Primer	*MVS/ESA HCD and Dynamic I/O Reconfiguration Primer*	SG24-4037
Sysplex Migration Guide	*MVS/ESA Version 5 Sysplex Migration Guide*	SG24-4581
S/390 G3 Enterprise Server: Complex Systems Availability and Recovery Presentation Guide	*S/390 G3 Enterprise Server: CSAR Presentation Guide*	SG24-4911

D.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RISC System/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RISC System/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection	SBOF-7250	SK2T-8042

D.3 Other Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook. A publication whose order number begins with the prefix **LY** is available to IBM-licensed customers only.

Note: A few publications in the following list are no longer available in hardcopy format and cannot be separately ordered. These publications are only available on CD-ROMs.

These publications are also relevant as further information sources:

- OS/390 OpenEdition

Short Title	Title	Order Number
OS/390 OpenEdition MVS User's Guide	*OS/390 OpenEdition MVS User's Guide*	SC28-1891
OS/390 OpenEdition MVS Command Reference	*OS/390 OpenEdition MVS Command Reference*	SC28-1892
OS/390 OpenEdition MVS Programming Tools	*OS/390 OpenEdition MVS Programming Tools*	SC28-1904
OS/390 OpenEdition MVS Messages and Codes	*OS/390 OpenEdition MVS Messages and Codes*	SC28-1908
OS/390 OpenEdition MVS Programming: Assembler Callable Services Reference	*OS/390 OpenEdition MVS Programming: Assembler Callable Services Reference*	SC28-1899
OS/390 OpenEdition MVS Planning	*OS/390 OpenEdition MVS Planning*	SC28-1890
OS/390 OpenEdition MVS File System Interface Reference	*OS/390 OpenEdition MVS File System Interface Reference*	SC28-1909
OS/390 OpenEdition MVS Using REXX and OpenEdition MVS	*OS/390 OpenEdition MVS Using REXX and OpenEdition MVS*	SC28-1905
OS/390 OpenEdition MVS Communications Server Guide	*OS/390 OpenEdition MVS Communications Server Guide*	SC28-1906

- I/O Configuration Management

Short Title	Title	Order Number
OS/390 HCD Planning	*OS/390 Hardware Configuration Definition Planning*	GC28-1750
OS/390 HCD User's Guide	*OS/390 HCD User's Guide*	SC28-1848
HCD Messages	*OS/390 HCD Messages*	GC28-1849
HCD Scenarios	*OS/390 HCD Scenarios*	SC28-1850

- RMF

Short Title	Title	Order Number
RMF Messages and Codes	*OS/390 RMF Messages and Codes*	GC28-1948
RMF Performance Management Guide	*OS/390 RMF Performance Management Guide*	SC28-1951
RMF User's Guide	*OS/390 RMF User's Guide*	SC28-1949
RMF Report Analysis	*OS/390 RMF Report Analysis*	SC28-1950
RMF Programmer's Guide	*OS/390 RMF Programmer's Guide*	SC28-1952

- Multi-System Configuration Management

Short Title	Title	Order Number
OS/390 Parallel Sysplex Systems Management	*OS/390 Parallel Sysplex Systems Management*	GC28-1861
OS/390 Parallel Sysplex Hardware and Software Migration	*OS/390 Parallel Sysplex Hardware and Software Migration*	GC28-1862

Short Title	Title	Order Number
OS/390 Parallel Sysplex Application Migration	*OS/390 Parallel Sysplex Application Migration*	GC28-1863
OS/390 V1R3.0 MVS Setting Up a Sysplex	*OS/390 MVS Setting Up a Sysplex*	GC28-1779
OS/390 V1R3.0 MVS Sysplex Services Guide	*OS/390 MVS Programming: Sysplex Services Guide*	GC28-1771
OS/390 V1R3.0 MVS Sysplex Services Reference	*OS/390 MVS Programming: Sysplex Services Reference*	GC28-1772

- OS/390 Operating System

Short Title	Title	Order Number
OS/390 V1R3.0 MVS Auth Assembler Services Reference ALE-DYN	*OS/390 MVS Programming: Authorized Assembler Services Reference, Volume 1, ALE-DYN*	GC28-1764
OS/390 V1R3.0 MVS Auth Assembler Services Reference ENF-ITT	*OS/390 MVS Programming: Authorized Assembler Services Reference, Volume 2, ENF-ITT*	GC28-1765
OS/390 V1R3.0 MVS Auth Assembler Services Reference LLA-SDU	*OS/390 MVS Programming: Authorized Assembler Services Reference, Volume 3, LLA-SDU*	GC28-1766
OS/390 V1R3.0 MVS Auth Assembler Services Reference SET-WTO	*OS/390 MVS Programming: Authorized Assembler Services Reference, Volume 4, SET-WTO*	GC28-1767
OS/390 V1R3.0 MVS Extended Addressability Guide	*OS/390 MVS Programming: Extended Addressability Guide*	GC28-1769
OS/390 V1R3.0 MVS Assembler Services Guide	*OS/390 MVS Programming: Assembler Services Guide*	GC28-1762
OS/390 V1R3.0 MVS Assembler Services Reference	*OS/390 MVS Programming: Assembler Services Reference*	GC28-1910
OS/390 V1R3.0 MVS Auth Assembler Services Guide	*OS/390 MVS Programming: Authorized Assembler Services Guide*	GC28-1763
Introducing OS/390	*OS/390 Introduction and Release Guide*	GC28-1725
OS/390 V1R2.0 MVS JCL User's Guide	*OS/390 MVS JCL User's Guide*	GC28-1758
OS/390 V1R3.0 MVS JCL Reference	*OS/390 MVS JCL Reference*	GC28-1757
OS/390 V1R3.0 MVS Callable Services for HLL	*OS/390 MVS Programming: Callable Services for High-Level Languages*	GC28-1768
OS/390 V1R3.0 MVS Writing TPs for APPC/MVS	*OS/390 MVS: Writing Transaction Programs for APPC/MVS*	GC28-1775
OS/390 V1R3.0 MVS Planning: APPC/MVS Management	*OS/390 MVS Planning: APPC/MVS Management*	GC28-1807
OS/390 V1R3.0 MVS IPCS Commands	*OS/390 MVS Interactive Problem Control System (IPCS) Commands*	GC28-1754
OS/390 V1R3.0 MVS IPCS User's Guide	*OS/390 MVS Interactive Problem Control System (IPCS) User's Guide*	GC28-1756
OS/390 V1R3.0 MVS IPCS Customization	*OS/390 MVS Interactive Problem Control System (IPCS) Customization*	GC28-1755

Short Title	Title	Order Number
OS/390 V1R3.0 MVS Initialization and Tuning Guide	*OS/390 MVS Initialization and Tuning Guide*	SC28-1751
OS/390 V1R3.0 MVS Initialization and Tuning Reference	*OS/390 MVS Initialization and Tuning Reference*	SC28-1752
OS/390 V1R3.0 MVS Installation Exits	*OS/390 MVS Installation Exits*	SC28-1753
OS/390 V1R3.0 MVS Using the Functional Subsystem Interface	*OS/390 MVS Using the Functional Subsystem Interface*	SC28-1911
OS/390 V1R3.0 MVS Conversion Notebook	*OS/390 MVS Conversion Notebook*	GC28-1747
OS/390 V1R3.0 MVS Using the Subsystem Interface	*OS/390 MVS Using the Subsystem Interface*	SC28-1789
OS/390 V1R3.0 MVS System Commands Summary	*OS/390 MVS System Commands Summary*	GX22-0040
OS/390 V1R3.0 Planning for Installation	*OS/390 Planning for Installation Release 3*	GC28-1726
OS/390 V1R3.0 MVS System Commands	*OS/390 MVS System Commands*	GC28-1781
OS/390 V1R3.0 MVS System Management Facilities (SMF)	*OS/390 MVS System Management Facilities (SMF)*	GC28-1783
OS/390 V1R3.0 MVS Planning: Operations	*OS/390 MVS Planning: Operations*	GC28-1760
OS/390 V1R3.0 MVS Planning: Global Resource Serialization	*OS/390 MVS Planning: Global Resource Serialization*	GC28-1759
OS/390 V1R3.0 MVS System Data Set Definition	*OS/390 MVS System Data Set Definition*	GC28-1782
OS/390 V1R3.0 MVS Device Validation Support	*OS/390 MVS Device Validation Support*	GC28-1748
OS/390 V1R3.0 MVS System Messages, Vol 1 (ABA-ASA)	*OS/390 MVS System Messages, Volume 1 (ABA-ASA)*	GC28-1784
OS/390 V1R3.0 MVS System Messages, Vol 2 (ASB-ERB)	*OS/390 MVS System Messages, Volume 2 (ASB-ERB)*	GC28-1785
OS/390 V1R3.0 MVS System Messages, Vol 3 (GDE-IEB)	*OS/390 MVS System Messages, Volume 3 (GDE-IEB)*	GC28-1786
OS/390 V1R3.0 MVS System Messages, Vol 4 (IEC-IFD)	*OS/390 MVS System Messages, Volume 4 (IEC-IFD)*	GC28-1787
OS/390 V1R3.0 MVS System Messages, Vol 5 (IGD-IZP)	*OS/390 MVS System Messages, Volume 5 (IGD-IZP)*	GC28-1788
OS/390 V1R3.0 MVS Dump Output Messages	*OS/390 MVS Dump Output Messages*	GC28-1749
OS/390 V1R3.0 MVS System Codes	*OS/390 MVS System Codes*	GC28-1780
OS/390 V1R3.0 MVS Routing and Descriptor Codes	*OS/390 MVS Routing and Descriptor Codes*	GC28-1778
OS/390 V1R3.0 MVS Recovery and Reconfiguration Guide	*OS/390 MVS Recovery and Reconfiguration Guide*	GC28-1777
OS/390 V1R3.0 MVS JES Common Coupling Services	*OS/390 MVS Programming: JES Common Coupling Services*	GC28-1770
OS/390 V1R3.0 MVS: Writing Servers for APPC/MVS	*OS/390 MVS: Writing Servers for APPC/MVS*	GC28-1774
OS/390 V1R3.0: MVS Writing Transaction Schedulers for APPC/MVS	*OS/390 MVS: Writing Transaction Schedulers for APPC/MVS*	GC28-1776

Short Title	Title	Order Number
OS/390 MVS APPC/MVS Handbook for OS/2	*OS/390 MVS APPC/MVS Handbook for OS/2*	GC28-1746
OS/390 V1R3.0 MVS Product Registration	*OS/390 MVS Programming: Product Registration*	GC28-1729
OS/390 V1R3.0 MVS Product Management	*OS/390 MVS Product Management*	GC28-1730
OS/390 V1R3.0 MVS Diagnosis: Procedures	*OS/390 MVS Diagnosis: Procedures*	LY28-1082
OS/390 V1R3.0 MVS Diagnosis: Tools and Service Aids	*OS/390 MVS Diagnosis: Tools and Service Aids*	LY28-1085
OS/390 V1R3.0 MVS Diagnosis: Reference	*OS/390 MVS Diagnosis: Reference*	LY28-1084
OS/390 V1R3.0 MVS Planning: Workload Management	*OS/390 MVS Planning: Workload Management*	GC28-1761
OS/390 V1R3.0 MVS Workload Management Services	*OS/390 MVS Programming: Workload Management Services*	GC28-1773
OS/390 Information Roadmap	*OS/390 Information Roadmap*	GC28-1727
OS/390 V1R3.0 MVS Diagnosis: Tools and Service Aids	*OS/390 MVS Diagnosis: Tools and Service Aids*	LY28-1845
OS/390 Introduction and Release Guide	*OS/390 Introduction and Release Guide*	GC28-1725
SystemView for MVS Up and Running!	*SystemView for MVS Up and Running!*	GC28-1241
OS/390 Planning for Installation	*OS/390 V1R3.0 Planning for Installation*	GC28-1726
OS/390 Information Roadmap	*OS/390 V1R3.0 Information Roadmap*	GC28-1727

- SMP/E and Installation Manuals

Short Title	Title	Order Number
ServerPac Guide and Worksheet	*ServerPac Guide and Worksheet*	SC28-1244
MVS Packaging Rules	*Standard Packaging Rules for MVS-Based Products*	SC23-3695
OS/390 SMP/E Diagnosis Guide	*OS/390 System Modification Program Extended Diagnosis Guide*	SC28-1737
OS/390 SMP/E Messages and Codes	*OS/390 System Modification Program Extended Messages and Codes*	SC28-1738
OS/390 SMP/E User's Guide	*OS/390 System Modification Program Extended User's Guide*	SC28-1740
OS/390 SMP/E Command Reference	*OS/390 System Modification Program Extended Command Reference*	SC28-1805
OS/390 SMP/E Reference	*OS/390 System Modification Program Extended Reference*	SC28-1806

- JES2 Subsystem

Short Title	Title	Order Number
OS/390 JES2 Messages	*OS/390 JES2 Messages*	GC28-1796
OS/390 JES2 Commands	*OS/390 JES2 Commands*	GC28-1790

Short Title	Title	Order Number
OS/390 JES2 Initialization and Tuning Guide	*OS/390 JES2 Initialization and Tuning Guide*	SC28-1791
OS/390 JES2 Initialization and Tuning Reference	*OS/390 JES2 Initialization and Tuning Reference*	SC28-1792
OS/390 JES2 Diagnosis	*OS/390 JES2 Diagnosis*	LY28-1086
OS/390 JES2 Introduction	*OS/390 JES2 Introduction*	GC28-1794
OS/390 JES2 Installation Exits	*OS/390 JES2 Installation Exits*	SC28-1793
OS/390 JES2 Macros	*OS/390 JES2 Macros*	SC28-1795
OS/390 JES2 Migration Notebook	*OS/390 JES2 Migration Notebook*	GC28-1797

- JES3 Subsystem

Short Title	Title	Order Number
OS/390 JES3 Conversion Notebook	*OS/390 JES3 Conversion Notebook*	GC28-1799
OS/390 JES3 Diagnosis	*OS/390 JES3 Diagnosis*	LY28-1090
OS/390 JES3 Initialization and Tuning Guide	*OS/390 JES3 Initialization and Tuning Guide*	SC28-1802
OS/390 JES3 Introduction	*OS/390 JES3 Introduction*	GC28-1808
OS/390 JES3 Initialization and Tuning Reference	*OS/390 JES3 Initialization and Tuning Reference*	SC28-1803
OS/390 JES3 Customization	*OS/390 JES3 Customization*	LY28-1089
OS/390 JES3 Messages	*OS/390 JES3 Messages*	GC28-1804
OS/390 JES3 Commands	*OS/390 JES3 Commands*	GC28-1798
OS/390 JES3 Diagnosis Reference	*OS/390 JES3 Diagnosis Reference*	LY28-1092

- ICKDSF

Short Title	Title	Order Number
ICKDSF R16 Refresh User's Guide	*ICKDSF R16 Refresh User's Guide*	GC35-0033

- Storage Environment

Short Title	Title	Order Number
DFSMS/MVS V1R3 General Information	*DFSMS/MVS Version 1 Release 3 General Information*	GC26-4900
DFSMS/MVS V1R3 Licensed Program Specifications	*DFSMS/MVS Version 1 Release 3 Licensed Program Specifications*	GC26-4903
DFSMS/MVS V1R3 Planning for Installation	*DFSMS/MVS Version 1 Release 3 Planning for Installation*	SC26-4919
DFSMS/MVS V1R3 Installation Exits	*DFSMS/MVS Version 1 Release 3 Installation Exits*	SC26-4908
DFSMS/MVS V1R3 Managing Catalogs	*DFSMS/MVS Version 1 Release 3 Managing Catalogs*	SC26-4914

Short Title	Title	Order Number
DFSMS/MVS V1R1 Checkpoint/Restart	*DFSMS/MVS Version 1 Release 1 Checkpoint/Restart*	SC26-4907
DFSMS/MVS V1R3 Using Data Sets	*DFSMS/MVS Version 1 Release 3 Using Data Sets*	SC26-4922
DFSMS/MVS V1R3 Macro Instructions for Data Sets	*DFSMS/MVS Version 1 Release 3 Macro Instructions for Data Sets*	SC26-4913
DFSMS/MVS V1R3 DFSMSdfp Advanced Services	*DFSMS/MVS Version 1 Release 3 DFSMSdfp Advanced Services*	SC26-4921
DFSMS/MVS V1R3 Access Method Services for ICF	*DFSMS/MVS Version 1 Release 3 Access Method Services for the Integrated Catalog Facility*	SC26-4906
DFSMS/MVS V1R3 Access Method Services for VSAM	*DFSMS/MVS Version 1 Release 3 Access Method Services for VSAM Catalogs*	SC26-4905
DFSMS/MVS V1R3 DFSMSdfp Storage Administration Reference	*DFSMS/MVS Version 1 Release 3 DFSMSdfp Storage Administration Reference*	SC26-4920
DFSMS/MVS V1R3 Using ISMF	*DFSMS/MVS Version 1 Release 3 Using the Interactive Storage Management Facility*	SC26-4911
DFSMS/MVS V1R3 Utilities	*DFSMS/MVS Version 1 Release 3 Utilities*	SC26-4926
DFSMS/MVS V1R3 DFM/MVS Guide and Reference	*DFSMS/MVS Version 1 Release 3 Distributed FileManager/MVS Guide and Reference*	SC26-4915
DFSMS/MVS V1R3 Program Management	*DFSMS/MVS Version 1 Release 3 Program Management*	SC26-4916
DFSMS/MVS V1R3 DFSMSdfp Diagnosis Guide	*DFSMS/MVS Version 1 Release 3 DFSMSdfp Diagnosis Guide*	LY27-9605
DFSMS/MVS V1R3 DFSMSdfp Diagnosis Reference	*DFSMS/MVS Version 1 Release 3 DFSMSdfp Diagnosis Reference*	LY27-9606
DFSMS/MVS V1R3 Using Magnetic Tapes	*DFSMS/MVS Version 1 Release 2 Using Magnetic Tapes*	SC26-4923
DFSMS/MVS V1R3 Using the Volume Mount Analyzer	*DFSMS/MVS Version 1 Release 2 Using the Volume Mount Analyzer*	SC26-4925
DFSMS/MVS V1R3 Implementing System-Managed Storage	*DFSMS/MVS Version 1 Release 3 Implementing System-Managed Storage*	SC26-3123
MVS/ESA SML: Leading a Storage Administration Group	*MVS/ESA Storage Management Library: Leading a Storage Administration Group*	SC26-3126
MVS/ESA SML: Managing Storage Groups	*MVS/ESA Storage Management Library: Managing Storage Groups*	SC26-3125
MVS/ESA SML: Managing Data	*MVS/ESA Storage Management Library: Managing Data*	SC26-3124
DFSMS/MVS V1R3 OAM Application Programmer's Reference	*DFSMS/MVS Version 1 Release 2 Object Access Method Application Programmer's Reference*	SC26-4917
DFSMS/MVS V1R3 OAM Planning, Installation, and Administration Guide for Object Support	*DFSMS/MVS Version 1 Release 3 Object Access Method Planning, Installation, and Storage Administration Guide for Object Support*	SC26-4918
DFSMS/MVS V1R3 OAM Planning, Installation, and Storage Administration Guide for Tape Libraries	*DFSMS/MVS Version 1 Release 3 Object Access Method Planning, Installation, and Storage Administration for Tape Libraries*	SC26-3051

Short Title	Title	Order Number
DFSMS/MVS V1R3 DFSMSrmm Guide and Reference	*DFSMS/MVS Version 1 Release 3 DFSMSrmm Guide and Reference*	SC26-4931
DFSMS/MVS V1R3 DFSMSrmm Implementing and Customizing	*DFSMS/MVS Version 1 Release 3 DFSMSrmm Implementing and Customizing Guide*	SC26-4932
DFSMS/MVS V1R3 DFSMSrmm Diagnosis Guide	*DFSMS/MVS Version 1 Release 3 DFSMSrmm Diagnosis Guide*	LY27-9615
DFSMS/MVS V1R3 DFSMShsm Managing Your Own Data	*DFSMS/MVS Version 1 Release 3 DFSMShsm Managing Your Own Data*	SH21-1077
DFSMS/MVS V1R3 DFSMShsm Storage Administration Guide	*DFSMS/MVS Version 1 Release 3 DFSMShsm Storage Administration Guide*	SH21-1076
DFSMS/MVS V1R3 Managing Data Availability	*DFSMS/MVS Version 1 Release 3 Managing Data Availability*	SC26-4928
DFSMS/MVS V1R3 DFSMShsm Implementing and Customizing	*DFSMS/MVS Version 1 Release 3 DFSMShsm Implementing and Customizing*	SH21-1078
DFSMS/MVS V1R3 DFSMShsm Diagnosis Guide	*DFSMS/MVS Version 1 Release 3 DFSMShsm Diagnosis Guide*	LY27-9607
DFSMS/MVS V1R3 DFSMShsm Diagnosis Reference	*DFSMS/MVS Version 1 Release 3 DFSMShsm Diagnosis Reference*	LY27-9608
DFSMS/MVS V1R3 DFSMShsm Storage Administration Reference	*DFSMS/MVS Version 1 Release 3 DFSMShsm Storage Administration Reference*	SH21-1075
DFSMS/MVS V1R3 DFSMSdss Storage Administration Guide	*DFSMS/MVS Version 1 Release 3 DFSMSdss Storage Administration Guide*	SC26-4930
DFSMS/MVS V1R3 DFSMSdss Storage Administration Reference	*DFSMS/MVS Version 1 Release 3 DFSMSdss Storage Administration Reference*	SC26-4929
DFSMS/MVS V1R3 DFSMSdss Diagnosis Guide	*DFSMS/MVS Version 1 Release 3 DFSMSdss Diagnosis Guide*	LY27-9609
Stand-Alone Services Overview	*Data Facility Data Set Services Version 2 Release 5 and DFSMS/MVS Version 1 Stand-Alone Services Overview*	SC26-0185
Remote Copy Administrator's Guide	*Remote Copy Administrator's Guide*	SC35-0169
DFSMS/MVS Network File System User's Guide	*DFSMS/MVS Network File System User's Guide*	SC26-7028
DFSMS/MVS Network File System Customization and Operation	*DFSMS/MVS Network File System Customization and Operation*	SC26-7029
DFSMS/MVS V1R3 Network File System Performance Tuning Guide	*DFSMS/MVS Version 1 Release 2 Network File System Performance Tuning Guide*	SC26-7019

- Security Server

Short Title	Title	Order Number
OS/390 Security Server (RACF) General User's Guide	*OS/390 Security Server (RACF) General User's Guide*	SC28-1917
OS/390 Security Server (RACF) System Programmer's Guide	*OS/390 Security Server (RACF) System Programmer's Guide*	SC28-1913
OS/390 Security Server (RACF) Macros and Interfaces	*OS/390 Security Server (RACF) Macros and Interfaces*	SC28-1914

Short Title	Title	Order Number
OS/390 Security Server (RACF) Command Language Reference	*OS/390 Security Server (RACF) Command Language Reference*	SC28-1919
OS/390 Security Server (RACF) Introduction	*OS/390 Security Server (RACF) Introduction*	GC28-1912
OS/390 Security Server (RACF) Messages and Codes	*OS/390 Security Server (RACF) Messages and Codes*	SC28-1918
OS/390 Security Server (RACF) Security Administrator's Guide	*OS/390 Security Server (RACF) Security Administrator's Guide*	SC28-1915
OS/390 Security Server (RACF) Auditor's Guide	*OS/390 Security Server (RACF) Auditor's Guide*	SC28-1916
OS/390 Security Server External Security Interface (RACROUTE) Macro Reference	*OS/390 Security Server External Security Interface (RACROUTE) Macro Reference*	GC28-1922
OS/390 Security Server (RACF) Callable Services	*OS/390 Security Server (RACF) Callable Services*	GC28-1921
OS/390 Security Server (RACF) Planning: Installation and Migration	*OS/390 Security Server (RACF) Planning: Installation and Migration*	GC28-1920
OS/390 Security Server (RACF) Support for MVS OpenEdition DCE, SOMobjects for MVS and SystemView	*OS/390 Security Server (RACF) Support for MVS OpenEdition DCE, SOMobjects for MVS and SystemView*	GC28-1924
OS/390 Security Server (OpenEdition DCE Security Server) Overview	*OS/390 Security Server (OpenEdition DCE Security Server) Overview*	GC28-1938

- TSO/E

Short Title	Title	Order Number
TSO/E Administration	*OS/390 TSO/E Administration*	SC28-1966
TSO/E REXX Reference	*OS/390 TSO/E REXX Reference*	SC28-1975
TSO/E General Information	*OS/390 TSO/E General Information*	GC28-1964
TSO/E Customization	*OS/390 TSO/E Customization*	SC28-1965
TSO/E Programming Guide	*OS/390 TSO/E Programming Guide*	SC28-1970
TSO/E Programming Services	*OS/390 TSO/E Programming Services*	SC28-1971
TSO/E CLISTs	*OS/390 TSO/E CLISTs*	SC28-1973
TSO/E Guide to SRPI	*OS/390 TSO/E Guide to the Server-Requester Programming Interface*	SC28-1976
TSO/E User's Guide	*OS/390 TSO/E User's Guide*	SC28-1968
TSO/E REXX User's Guide	*OS/390 TSO/E REXX User's Guide*	SC28-1974
TSO/E System Programming Command Reference	*OS/390 TSO/E System Programming Command Reference*	SC28-1972
TSO/E Command Reference	*OS/390 TSO/E Command Reference*	SC28-1969
TSO/E Messages	*OS/390 TSO/E Messages*	GC28-1978
TSO/E Primer	*OS/390 TSO/E Primer*	GC28-1967

- PSF/MVS

Short Title	Title	Order Number
AFP Printer Information	*Advanced Function Presentation: Printer Information*	G544-3290
AFP Printer Summary	*Advanced Function Presentation: Printer Summary*	G544-3135
Guide to Advanced Function Presentation	*Guide to Advanced Function Presentation*	G544-3876
IBM IP PrintWay Guide	*IBM IP PrintWay Guide*	S544-5379
IBM NetSpool Guide	*IBM NetSpool Guide*	G544-5301
OS/390 V1R3.0: Printing Softcopy BOOKs	*OS/390 V1R3.0: Printing Softcopy BOOKs*	S544-5354
PSF/MVS Messages and Codes	*Print Services Facility/MVS: Messages and Codes Version 2 Release 2 Modification 0*	S544-3675

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at URL http://www.redbooks.ibm.com.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Home Page on the World Wide Web**

  http://w3.itso.ibm.com/redbooks

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** (Do not send credit card information over the Internet) — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
(+45) 48 14 2207 (long distance charge)	Outside North America

- **1-800-IBM-4FAX (United States)** or **(+1)001-408-256-5422 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com
- **On the World Wide Web**

Redbooks Home Page	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet e-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an e-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank).

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

DO NOT SEND CREDIT CARD INFORMATION OVER THE INTERNET.

Index

M

N

O

P

R

S

T

U

V

W

X

ITSO Redbook Evaluation

Redbook Title_____

Document Number_____

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redeval@vnet.ibm.com

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

Printed in U.S.A.