IBM

z/OS Version 1 Release 8 Implementation

Consoles, CFRM, RMF, WLM, Health Checker, DFSMS, zAAPs

zIIPs, RSM and SRM, z/OS UNIX, zFS, z/OS XML, Unicode

JES2, JES3, LE, GRS, HCD, HCM

Paul Rogers
Fabio Becker
Olivier Daurces
Redelf Janßen
Lutz Kuehner
Gil Peleg
Zulfiye Meral Temel

Redbooks

ibm.com/redbooks

International Technical Support Organization

z/OS Version 1 Release 8 Implementation

February 2007

SG24-7265-00

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

First Edition (February 2007)

This edition applies to Version 1 Release 8 of z/OS (5694-A01), Version 1 Release 8 of z/OS.e (5655-G52), and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Notices

Trademarks

Preface

This IBM® Redbook describes the functional enhancements to z/OS® for Version 1 Release 8 (z/OS V1R8). These enhancements are intended to help installations install, tailor, migrate, and configure z/OS V1R8.

The following topics are discussed:

- z/OS Version 1 Release 8 Overview
- Installation and migration to z/OS V1R8
- Console restructure
- CFRM performance enhancements
- RMF™ enhancements
- WLM enhancements
- zSeries® Integrated Information Processors (zIIPs)
- RSM™ and SRM
- Miscellaneous BCP changes
- z/OS UNIX® System Services
- zFS enhancements
- JES2 z/OS V1R8 enhancements
- JES3 z/OS V1R8 enhancements
- DFSMS enhancements
- Program management enhancements
- z/OS XL C/C++ enhancements
- Language Environment® enhancements
- GRS enhancements
- z/OS XML (eXtensible Markup Language)
- z/OS V1R8 Common Information Model enhancements
- Enterprise Workload Manager (EWLM)
- IBM Health Checker for z/OS in z/OS V1R8
- Storage clear for PL/I applications
- HCM/HCD enhancements
- Unicode services
- REXX/CLIST variable storage constraint relief

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.

Paul Rogers is a Consulting IT Specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on various aspects of z/OS, z/OS UNIX, JES3, and Infoprint® Server. Before joining the ITSO 19 years ago, Paul worked in the IBM Installation Support Center (ISC) in Greenford, England for seven years, providing OS/390® and JES support for IBM EMEA, and also in the Washington Systems Center for three years. He has worked for IBM for 39 years.

Fabio Becker is a Support Analyst in Brazil. He has 27 years of experience in several areas of IT including hardware and zLinux. He has been working for Banrisul since 1989 in the MVS™ field. His areas of expertise include z/OS problem determination.

Olivier Daurces is an Advisory IT Specialist working for IBM Technical Support in France. He has eight years of experience with z/OS. His areas of expertise include RACF® and Parallel Sysplex®. As a sytems programmer, he is also responsible for upgrading and maintaining the current test system. His interests include cryptography and performance.

Redelf Janßen is an IT Architect in IBM Global Technology Services in IBM Bremen, Germany. He holds a degree in Computer Science from the University of Bremen and joined IBM in 1988. He is responsible for supporting zSeries customers in Germany, and teaches the z/OS DFP DASD and subsystems class in Germany. His areas of expertise include IBM System z9™, IBM zSeries, z/OS, storage management, and availability management. He has written Redbooks™ on OS/390 Releases 3, 4, and 10, and was one of the authors of *ABCs of z/OS System Programming* (Volumes 3 and 5).

Lutz Kuehner is a z/OS IT specialist working in IBM Global Services Germany. He has 20 years of experience in the mainframe z/OS field.

Gil Peleg is a z/OS system programmer working for Tangram-Soft in Israel. He has nine years of experience in mainframe systems and holds a degree in Computer Science. He is responsible for supporting zSeries customers in Israel and teaching zSeries-related courses. He specializes in performance and availability management.

Zulfiye Meral Temel has been working as an MVS system programmer in Garanti Technology, Turkey for 10 years. Her areas of experience include System z™ hardware, Parallel Sysplex, z/OS performance, WLM/SRM, and I/O management. She wrote the CFRM, RMF, zIIP, RSM, consoles, and WLM chapters.

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  ibm.com/redbooks

- Send your comments in an email to:

  redbook@us.ibm.com

- Mail your comments to:

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

1

z/OS Version 1 Release 8 overview

This chapter describes the enhancements affecting installation and any positioning activities including the new required migration tasks that are introduced in z/OS V1R8 for the following selected elements:

- BCP
- CIM
- Communications Server
- DFSMS
- DFS™
- Infoprint Server
- JES2
- JES3
- Language Environment
- NFS
- RMF
- SDSF
- Security Server
- XL C/C++
- z/OS UNIX
- HyperPAV support

1.1 z/OS Version 1 Release 8

z/OS V1R8 offers enhancements in many areas of the operating system. These enhancements are described briefly in this section and in greater detail in subsequent chapters of this redbook.

1.1.1 z/OS UNIX System Services

The z/OS UNIX System Services base is enhanced for greater adherence to both industry-approved open standards and industry standards. Enablement of these standard features enhances porting capabilities onto the z/OS platform and provides installations with the tools they need to fully operate in a UNIX environment. Adherence to such standards provides direction to continue to support the open environment and the z/OS UNIX platform.

z/OS UNIX System Services (USS) has enhancements in z/OS Version 1 Release 8 as follows:

- BPXBATCH now allows STDOUT and STDERR DD statements to be represented as SYSOUT or as traditional MVS data sets such as PDS/Es, PDSs, and sequential data sets and not just as files residing in the hierarchical file system.
- The STDENV DD statement can be represented by PDS/Es, not just SYSIN, sequential, or partitioned data sets.
- Parameter strings now support up to 65,536 bytes from both a batch and TSO command invocation environment. This support was formerly 100 and 500 bytes, respectively.
- The file format flag for a file can be set from the shell environment via the extattr utility.
- A **cp/mv** enhancement provides a change to the default behavior of the **cp** (copy) and **mv** (move) commands when copying or moving a file to an NFS-mounted file system with a different tag than source. This change enhances an alignment with the latest in industry UNIX standards.
- The flockfile() family of functions consists of common UNIX functions found within the single UNIX specification V3 (SUSV3) standards. In z/OS V1R8, the following functions of flockfile() are implemented:
  - ftrylockfile()
  - funlockfile()
  - getc_unlocked()
  - getchar_unlocked()
  - putc_unlocked()
  - putchar_unlocked()
- The pax utility is enhanced to implement the pax interchange archive format for archives as described in SUSV3. This format allows this utility to create extended USTAR archives, which are able to store file attributes, which cannot be stored in original USTAR archives due to limitations of the format.
- Support for the /etc/inittab file that is used on other UNIX systems to identify which system processes to be started during system initialization, as well as support for daemon restart. This allows you to identify processes that should receive additional system management by z/OS UNIX System Services.
- USS also provides a method to prevent file systems from being automounted during a file system shutdown. This function provides the ability to handle unexpected mounts that occur via the automount facility when a system is shut down for file system ownership.

Linux for zSeries messages

z/OS UNIX System Services provides a function to integrate Linux® for zSeries messages into the z/OS operations log (OPERLOG) via the syslog daemon by providing a fast path alternative for Linux on zSeries messages.

This is done by having Linux syslog() messages sent to a remote z/OS syslog daemon and providing an alternative target (to a hardcopy log) called OPERLOG. Messages targeted for this new support are written to /dev/operlog and are picked up by the new z/OS function, which will build the message data blocks for the message and have them written directly to the operlog system logger stream.

Note: This path does not support automation and results in a much shorter path length than the WTO that is currently issued on behalf of the syslog daemon's console target. This support gives you the ability to integrate logging from important Linux applications on z/OS, rather than having to manage multiple log streams.

1.1.2 zFS enhancements

The performance of sharing a zFS file system across a sysplex is an inhibitor to many customers. The current design, in which USS forwards every file request when the file system is owned on a different member than the requester, adds significant overhead. zFS improves performance by avoiding the forwarding of requests in many cases. Local memory caching is used to avoid forwarding of requests for data that is already contained in the cache. A token management mechanism is used to maintain cache consistency between file system clients and the owner of the file system.

This support benefits Parallel Sysplex environments that use shared HFS support. It is of particular benefit to customers that access read-write file systems from multiple sysplex members. Application administrators can configure applications with less concern for which sysplex member actually owns USS data.

zFS improves performance for shared file systems by becoming sysplex-aware for read-write file systems. zFS and HFS are already sysplex-aware for read-only file systems. Since zFS is sysplex-aware, USS sends file requests directly to the zFS PFS without forwarding them to the USS file system owner. zFS uses XCF communications to forward requests to the owning zFS PFS when necessary.

Physical file system (PFS) termination for zFS is now simplified by being done with a MODIFY operator command so that the logical file system (LFS) is notified first, resulting in a more orderly termination, especially in a shared file system environment.

zFS improves the performance of zFS mounts by recording additional information on-disk so that mounts can proceed more quickly. zFS file systems are converted to this new format automatically on the first z/OS V1R8 mount.

1.1.3 JES2 enhancements

SAPI interface

In JES2 V1R8, an option is added to permit READ authority to SSS2PUGE functions within the SAPI interface that do not make any updates to the spool. Any subsequent requests in the same conversation that require UPDATE authority result in termination of the thread and add a count of the job's SYSOUT data sets that match the criteria specified on a SSS2COUN request. This gives you the capability to access JES spool files through the FTP server, but this function cannot currently be implemented if users are not allowed,

1.1.4 SDSF enhancements

SDSF has to display zAAP utilization on the DA screen. SDSF obtains all data for the DA screen via RMF Monitor II data retrieval services ERBSMFI and ERB2XDGS. Both APIs already return the system CP utilization and the MVS view of the CP utilization. The APIs are extended to return the system zAAP utilization.

1.1.5 Console enhancements

The console enhancements work continues to reduce sysplex outages related to message processing, as follows:

- Infrastructure simplification
- Master console elimination and console switch elimination
- 1-byte console ID elimination
- Queuing attributes for the master console with the 1-byte console ID elimination
- RAS Enhancement—ORE backward linkage
- Console metadata volume reduction and entry-level serialization of the console metadata restructure
- Console ID-to-console name mapping to allow removal of the 99 console limit

VARY online processing

VARY online processing is enhanced to eliminate excessive wait time when varying a range of devices online. Allocation will assume responsibility for issuing the console messages that may occur when bringing a device online.

1.1.6 IBM Health Checker for z/OS

The IBM Health Checker for z/OS provides a foundation to help simplify and automate verification of best practices for z/OS and sysplex configuration values. It compares active values and settings to those generally recommended or to those you specify for your installation. IBM Health Checker for z/OS consists of a Framework, which is designed to let anyone write a check for it to perform. Checks are provided separately and are independent of the Framework. Checks can be added to the system dynamically. You can provide overrides to check defaults, display the status of checks, as well as dynamically change their status and parameters.

Health Checker provides a means for reducing system outages or performance problems due to configuration or setup errors. Since Health Checker runs on a live system, it provides the ability to perform real-time checking of specific settings and current status and, most importantly, alerting an installation to things directly pertaining to that installation. Potential problems can be corrected prior to those problems impacting the customer’s business.

NLS enhancement

In z/OS V1R8, the Health Checker framework is enabled for NLS. For messages from the Health Checker and individual checks that are externalized to users via WTO, the ability to translate them is supported. Exploiters using the Health Checker messaging services can include their exception messages in MMS. Report messages, such as those displayed by SDSF, are not translated.

Other enhancements

Health Checker framework also provides the following:

- **Remote checks:** Ability to have checks (authorized or unauthorized) run outside of Health Checker. This includes checks that are not REXX, REXX checks that are part of SYSREXX, and REXX checks that are not part of SYSREXX.
- **Parmlib check definition:** Ability to define or add a check via a parmlib definition instead of needing a program that does the definition.
- **Enhanced POLICY support:** Support multiple policies (for example, “first shift”, “weekends”) so that you can easily switch your check updates to match your working environment.
- **Verbose command output:** A customer way of asking the check output to be as verbose as possible.
- **Parameter parser:** Provide a service usable by check routines to parse parameters so that we can move towards consistency in our parameter definition (for example, Key(value) instead of positional).

New checks in z/OS V1R8

Additional checks are provided in this release, as follows:

- **z/OS Communications Server:** Provides a check to alert you to non-recommended TCP/IP configuration values.
- **ASM:** Provides paging data sets on separate volumes that issue a warning message if any paging data sets are on the same volume and the device is not a PAV-capable device, as follows:
  - Three or more local paging sets that will issue a warning message if there are fewer than three local paging sets active
  - A check for unused PARTE entries that will issue a warning message if all PARTEs are in use, which would limit the ability to dynamically add paging data sets. These checks will be run once a day. They will also provide 30% Slot Usage on Local Paging Data Set, which will issue a warning message if slot usage on a local paging data set exceeds 30%.
  - A check for the current total aux slot usage that will issue a message that reports the current aux slot usage amount. These checks are run once an hour. Check the common page data sets and ensure that they are big enough.
- **VSM**: Adds a check for CSA tracker.
- **GRS**: Provides GRS_GRSQ_SETTING, which examines the current GRSQ setting. This check is only applicable in STAR mode. IBM recommends having a GRSQ setting of CONTENTION, which may significantly reduce the amount of time required for the dump. GRS will also provide GRS_RNL_IGNORED_CONV to make it possible for an RNL entry to exist on the RESERVE Conversion RNL and have a “matching” entry on the SYSTEMS Exclusion RNL. The entry on the RESERVE Conversion RNL is ignored or superseded by the entry on the SYSTEMS exclusion RNL.
- **RRS**: Provides a check to highlight when a configuration varies from the IBM recommendation. The check produces an exception message when the archive log’s CF structure can be shared by two or more log streams.

1.1.7 WLM enhancements

Performance management is among the most important system management disciplines provided by WLM. WLM provides autonomic, policy-based z/OS performance management. This self-optimization is more than merely software-based. For example, WLM interacts with the PR/SM™ Hipervisor to move resources from LPARs that are overachieving to LPARs that need more in real time.

zAAP workloads

Today, zAAP workloads are managed based on CP utilization. In z/OS V1R8, workload delays imposed by zAAPs are handled the same way they are for CPs. As zAAP utilizations climb and the ratio between an application's zAAP-resident work and its CP-resident work changes to use zAAPs more and CPs less, this improvement can be needed. The enhancements provide the following:

- The processor projection and assessment logic is extended to project the change of dispatch priorities for zAAPs in the same way as for regular processors. This includes that a service class will be helped because zAAPs state samples indicate that a change for this resource is beneficial for it.
- With z/OS V1R6, when zAAPs were introduced, work can run on zAAPs in three ways:
  - Purely on zAAPs without having the possibility to run on regular processors even if the regular processors are idle.
  - They can run on regular processors if these are idle and there is only zAAP-eligible work to run. This mode is called CROSSOVER.
  - They can run on regular processors based on their priority. This mode is called HONORPRIORITY.
- With z/OS V1R8, the HONORPRIORITY mode is changed such that HONORPRIORITY becomes a function for a zAAP that needs help. The zAAP can turn on a *need help* indication so that a regular processor can help it. This is primarily a function of the z/OS supervisor, which only needs assistance from WLM/SRM to identify the ranges of work for which help is required.
- WLM's routing algorithms return recommendations for routing products based on the capacity of regular processors. This must be extended to include the available capacity of zAAPs in case the workload, which is distributed by the routing services, uses zAAPs.

Load balancing support

In z/OS V1R8, WLM is to begin addressing a key customer issue referred as the "storm drain" problem. Storm drain is used to describe a set of scenarios where load balancing is being performed (such as using the Sysplex Distributor or the new Load Balancing Advisor and an external load balancer) to z/OS subsystems/applications in a sysplex environment. Most of this function was shipped for WLM in z/OS V1R7; however, z/OS V1R8 now includes abnormal terminations in the enhanced recommendations that we introduced in z/OS V1R7, and allows applications to set a health status, which is also added. This helps you significantly improve availability for key environments in a sysplex environment.

1.1.8 CFRM performance enhancements

The objective of the CFRM performance enhancements is to improve sysplex availability by eliminating the contention on the CFRM couple data set as much as possible, resulting in faster recovery due to CF-related failures and sysplex partitioning actions.

This enhancement is designed for all zSeries environments that currently run a Parallel Sysplex. Before this change, such environments required Coupling Facility resource management to manage structure and connector events that can occur in the sysplex using the CFRM couple data sets as a means of coordinating those events. Installations with larger production Parallel Sysplex configurations—with a large number of structures and connectors—should see a major improvement in sysplex availability, for the following occurrences:

- Sysplex partitioning and recover
- Structure rebuilds
- Duplexing failover
- CF structure rebuilds
- Connect/disconnect processing

XES events using a message-based protocol

Delivery of XES events and collection of responses to XES events now use a message-based protocol to greatly reduce serialized access of the CFRM active policy on the CFRM couple data set.

The message-based protocol defines a single system as manager and all other systems as participants. A system is a participant when structures are being used by subsystems running on a system. The manager system is responsible for coordinating events and confirmations with the participating systems. The manager system is responsible for updating (as needed) the CFRM active policy, which reduces CFRM CDS I/O to a central control point. Communication between the manager system and participant systems uses XCF signaling to send messages between the systems. Recovery for the failure of a system that is a manager and/or participant, is also provided. There are also associated serviceability enhancements, including dumping and IPCS updates.

1.1.9 Security enhancements

Improvements in security throughout the z/OS platform are a main initiative in z/OS V1R8. Enhancements to RACF, PKI, SAF, and LDAP are in place to further enhance the world-class security that the platform currently has.

RACF enhancements

In z/OS R8, profile extension focuses on improving or making more flexible and relevant the security rules as implemented by RACF today. *Password phrase* provides an alternative to traditional passwords. This alternative allows a much longer value than a password, as well as a larger character set. A password phrase is a character string, made up of mixed case letters, numbers and special characters, including blanks.

The intention of the password phrase is to have something that is long enough to provide security, but easy enough to remember so it won’t be written down. It is unlikely that a random string of characters longer than 8 can be memorized easily; therefore, a password phrase is more likely to be made up of words. Adding numbers and special characters makes the phrase more secure. RACF now supports password phrases from 14 to 100 characters in length, and will enforce a basic set of rules to increase the strength of the password phrase. Since a user ID can have both a password and a password phrase, the same user ID can be used for both traditional applications that accept a password, and for new applications that take advantage of the password phrase infrastructure.

RACF has availability improvements that help to identify and correct problems that have resulted in past system outages. This item addresses circumstances such as the following:

- Clients needing to create a consistent backup copy of the primary RACF data set and activate the backup without loss of synchronization between the two data sets. In the past, IRRUT200 did not allow this; the primary RACF database could be updated between the copy process and the activation of the backup database, thus resulting in an error. IRRUT400 with the LOCKINPUT option could create a consistent RACF backup database. However, it could cause applications attempting to access the locked database to fail.

- Simple errors in the DD statements for IRRUT200 or IRRUT400 could result in an active RACF data set being an output data set for these utilities. The utilities should detect this and prevent the overwriting of an active RACF data set.

  To solve these problems, RACF has made the following availability improvements:

  – A new parameter on the IRRUT200 utility tells the utility to activate the backup data set pointed to as output. This is accomplished by the utility internally issuing an RVARY ACTIVE for the backup data set after the copy is complete.

  – IRRUT200 and IRRUT400 utilities now check whether their output data sets are active as primary or backup RACF data sets on this system. If so, the utility fails with an error message.

PKI enhancements

In z/OS R8, PKI is being enhanced with the following:

- **Virtual key rings:** In the past, SSL-enabled applications required creating a key ring for each unique user ID, along with any CA certificates. This situation propagated itself, when in most cases the same set of CA certificates would be used for each user ID. Thus these key rings would all be replicas.

  To solve this problem, RACF now treats all the certificates installed under a given user ID as a virtual key ring. This key ring is created when the user ID is added and destroyed when the user ID is deleted.

- **Enabling multiple CA support for PKI Services:** This function lifts the restriction that prevented more than one instance of the PKI Services daemon from being started simultaneously on a single MVS image. This allows PKI Services customers to establish multiple certificate authorities on a single MVS image.
- **Add SCEP support to PKI Services:** Simple certificate enrollment protocol (SCEP) allows SCEP-enabled clients to request certificates by sending messages to a certificate authority using the http protocol. Adding SCEP support allows PKI Services to accept, decrypt, and respond to the various SCEP messages and supports both the manual and automatic enrollment modes as defined in the standard. Manual enrollment requires the CA administrator to manually approve requests. With automatic enrollment, certificate requests are auto-approved and fulfilled synchronously, based on the requestor’s knowledge of a predetermined secret: the challenge password.

SAF enhancements

The SAF identity token provides increased user accountability and audit resources by providing end-to-end auditing that tracks the identity initially used for authentication as well as the identity on the current platform. This support is especially valuable to customers maintaining heterogeneous environments, where requests and entry points to network resources come from a variety of platforms.

LDAP server enhancements

With z/OS Version 1 Release 8, the LDAP server is rewritten, restructured, and enhanced to address performance, stronger affinity with the z/OS environment, and functional deficiencies. Enhancements in this rewrite include the following:

- Better performance

  The z/OS LDAP V1R8 server is rewritten from C++ to C, which includes a restructure of front end and back end responsibilities. The SDBM back end is modified to use the enhanced SAF R_Admin interface. A file-based back end, LDBM, provides better performance because all entries are cached in storage for quick retrieval.

- Better availability

The z/OS LDAP R8 server now provides ARM, TCP/IP restart, Dynamic network interface management, and sysplex support for LDBM.

- Change log (GDBM) support

  DB/2 provides sysplex support for the database with a TDBM cache and a server-wide schema. Providing this sysplex support allows LDAP servers to also participate in replication with other LDAP servers that are not part of the same sysplex group.

- Scalability and constraint relief

  The z/OS LDAP V1R8 server is 64-bit capable for all back ends except for TDBM because DB/2 does not support 64-bit in the V1R8 time frame.

- Auditing support

  z/OS LDAP V1R8 provides new SMF 83 audit records for the LDAP server.

- Additional RAS support

  z/OS LDAP V1R8 provides two Health Checks. In addition, the ability to trace and invoke dumps in z/OS LDAP V1R8 is provided.

- Ease of use support

  The z/OS V1R8 LDAP server provides a file-based back end, LDBM, eliminating the need for DB/2 when that back end is used. Also, the change log (GDBM) no longer uses DB/2. Instead, it is file based. Both of these new supports make it easier for you to set up and deploy z/OS LDAP.

- Cross-platform consistency

  z/OS R8 LDAP Client and Server provide additional functions that the current IBM Tivoli® Directory Server already has.

- Security enhancements

  z/OS LDAP V1R8 eliminates the requirement for OCSF for its cryptographic support. The functions provided by OCSF are now contained in z/OS LDAP R8.

- Elimination of internal interfaces

  With the current z/OS LDAP server, internal LDAP interfaces are being used by HCD. With the rewrite of the server, this interface is eliminated.

1.1.10 RRS enhancements

In z/OS V1R8, RRS expands the length of the Resource Manager Log Name that is supported for ATRIRLN (Retrieve_Log_Name) and ATRISLN (Set_Log_Name) from 32/64 bytes of printable data to 32k of binary data (or whatever the maximum size is that is available in an RM log data record). This would require a change to the service to support more than printable and national characters and to support a much larger length specification.

These services are used by an RRS-enabled resource manager to log up to 64 bytes of information in the RRS Resource Manager data log stream. The intent is that the RM now logs a unique string of data to identify the Resource Manager's own log streams. This allows the Resource Manager to determine upon restart whether the RRS log streams contain the expected data. This allows for better synchronization of RRS and XA partner logs and facilitates the low-latency takeover of transaction recovery responsibilities from one server to another.

z/OS V1R8 RRS provides enhancements to the MVS DISPLAY commands to include Recoverable Resource Services status information, which makes it easier to implement automatic alerts and to capture information into SYSLOG. A new SHUTDOWN command to end RRS instead of doing a CANCEL is now available. This avoids unnecessary abends and makes shutdown messages clearer to the operations staff.

1.1.11 Communications Server enhancements

z/OS Communications Server V1R8 provides z/OS applications with secure network access in next-generation IPv6 networks. Enhancements for sysplex environments offer improved availability and usability for workloads distributed across multiple processors. Technologies such as Intrusion Detection Services, Enterprise Extender, and TN3270 will benefit from improved manageability and ease of configuration.

Enhancements to callable APIs

z/OS Communication Server provides a rich set of callable APIs for use by network management applications. In z/OS V1R8, a number of additions and improvements have been made to this set of APIs, as follows:

- A new API is made available to allow for the retrieval of TN3270-related performance data.
- An API is provided for dropping multiple TCP connections or UDP endpoints.
- The existing Enterprise Extender network management interface is enhanced to allow the specification of wildcard characters in the CP name on HPR connection requests.

The ability to manage TN3270 connections is improved by:

- The provision of a new API to allow for the retrieval of TN3270-related performance data
- The addition of the TCP connection ID to the TN3270 SMF records

RAS enhancements

A number of RAS items are provided, particularly in the areas of serviceability and general problem determination. Focus areas include OMPRoute, FTP, SNA messages and traces.

SYSTCPSM Network Management Interface

The SYSTCPSM Network Management Interface (introduced in z/OS V1R5) enabled application programs to monitor FTP and TN3270 activity in real time. This interface is enhanced in V1R8 for FTP in the following ways:

- A new FTP type 119 SMF record for client and server is provided.
- A new username section for the FTP Client Transfer Completion record is included.
- New and existing FTP SMF type FTP records include the TCP connection ID.

SNA and Enterprise Extender

The following changes are introduced in SNA and Enterprise Extender (EE):

- AnyNet® is removed from z/OS CS. AnyNet has not been enhanced in many years, and is inferior to Enterprise Extender both in function and performance.
- The ability to define parallel EE TGs using multiple SAPs is removed in V1R8. Parallel TGs defined in such a manner provide no benefit over single EE logical links.
- A number of miscellaneous usability, serviceability, and problem determination improvements are made to SNA, and in particular EE.

Sysplex networking

z/OS Communications Server support for sysplex networking receives a number of enhancements:

- An option is introduced that allows Sysplex Distributor to favor local system target servers where possible, while still avoiding servers that are no longer active or overloaded. If Sysplex Distributor chooses a target on the same system as the client, the connection is optimized in the following manner:
  - Traffic for the target connection is no longer routed to the Sysplex Distributor routing stack.
  - The connections are eligible for the “fast local sockets” optimized path.
  - Sysplex sockets report these sockets as being on the same system.
  - A mechanism is provided that allows the user to indicate what source IP address is selected by TCP/IP based on the destination IP address or destination network.
- z/OS Communications Server now allows the specification of a more granular scope for networking sysplex functions. This ability is provided to subdivide the sysplex into multiple “subplex” scopes from a sysplex networking function perspective. For example, some VTAM® and TCP/IP instances in a sysplex may belong to one subplex, while other VTAM or TCP/IP instances in the same sysplex belong to different subplexes.
- The Sysplex Autonomics function is enhanced to provide monitoring of the external network so that Sysplex Autonomics recovery can be triggered should failures in critical components occur.

z/OS Communications Server policy

In V1R8, Communications Server provides a flat file equivalent to the IDS policy that has been stored in LDAP in previous releases. The policy continues to be read and processed by the policy agent and gives z/OS Communications Server policy a consistent flat file support for all policy disciplines.

TCP/IP stack

A number of miscellaneous changes are made in V1R8 to TCP/IP stack functionality, as follows:

- Support is removed for the ASSORTEDPARMS and KEEPALIVEOPTIONS statements in the TCP/IP profile. The options that were part of these statements were moved to the TCPCONFIG, IPCONFIG/IPCONFIG6, and UDPCONFIG statements in prior releases.
- TCP/IP reduces its ECSA consumption via changes to the inbound data path, and to the dispatching infrastructure.
- The interface between z/OS Communications Server and the z/OS UNIX System Services PreRouter are enhanced to provide more predictable routing results in a multiple-stack environment.

AES algorithm

z/OS V1R8 Communication Server introduces support for the Advanced Encryption Standard (AES) algorithm for IP Security with 128-bit key length.

z/OS Load Balancing Advisor

In V1R8, z/OS Communications Server provides an application that communicates with a global workload manager (GWM) such as the z/OS Load Balancing Advisor, and based on information from the GWM, manages DNS records for host, host groups, servers, and server groups in name servers and their associated zones.

IP filtering

In V1R8, z/OS Communications Server provides the necessary function to support IP filtering, IPSec, and Internet Key Exchange (IKE) for IPv6. This includes the following enhancements:

- The Policy Agent is enhanced to configure IP filters, manual tunnels, and dynamic tunnels for IPv6.
- The TCP/IP profile is enhanced to allow configuration of default IP filters for IPv6 when the policy-based IP filters are not active.
- The z/OS Communications Server IKE daemon is enhanced to negotiate dynamic tunnels for IPv6.
- The z/OS Network Security Configuration Assistant GUI is enhanced to configure IPSec for IPv6 in the Policy Agent and the IKE daemon.
- The `ipsec` command is enhanced to display and modify installed IP filter information, manual tunnel information and dynamic tunnel information for IPv6.
- The traffic regulation management daemon (TRMD) is enhanced to support logging of IP Security events for IPv6 such as IP filter permits and denies.

Note: This item does not include support for sysplex-wide security associations (SWSA) or NAT traversal.

1.1.12 DFSMS enhancements

Data Facility Storage Management Subsystem (DFSMS) is a component of zSeries software that automatically manages data from creation to expiration. DFSMS provides allocation control for availability and performance, backup/recovery and disaster recovery services, space management, tape management, and reporting and simulation for performance and configuration tuning. The enhancements in DFSMS Version 1 Release 8 are designed to meet the zSeries software to provide added value for automated and centralized storage management, virtualization of tape and disk volumes, policy-based storage management, improved management of tape and disk resources, and improved storage administrator productivity.

Fast replication - data set recovery

An individual data set can be recovered from a copy pool backup version using the RRECOV command. The FRRECOV command syntax will be enhanced to enable one or more fully or partially qualified data set names to be specified. Each data set must be cataloged and allocated on the same volumes that it resided on at the time the backup copy was created. The backup version being recovered may reside on either disk or tape. If the version resides on both disk and tape, the default is to recover from disk. If the backup version is recovered from disk, then the recovery can be performed using either fast replication or traditional copy methods. Up to 64 concurrent data set recoveries are supported.

Copy pool recovery

The following functions are enhanced:

- HOLD and RELEASE are updated to enable users to hold data set level copy pool recovery requests.
- CANCEL is updated to enable users to cancel queued data set level copy pool recovery requests.

- The DEFINE DUMPCLASS and SETSYS commands are updated to enable users to customize the data set level copy pool recovery environment.
- LIST and QUERY are updated to provide users with useful information for this new function.
- The RACF facility class profiles introduced with the fast replication support provided in z/OS V1R5 is being modified to support the entire copy pool name.

Fast replication for tape support

Fast replication for tape support functions is enhanced as follows:

- FRBACKUP**:** Allows this command to initiate the dump of any existing backup copies or to create a new copy and then dump it. If in a VERSIONS=0 (NOCOPY) environment, withdraw the relationships when the dump is finished. Allow resumption of an incomplete dump version if any of the required dump classes failed to complete, only dumping the volumes not successfully completed by the previous attempt.
- FRRECOV**:** Allows the recovery of a dump copy to a single volume.
- FRDELETE**:** Deletes one or more unneeded copy pool dump versions.

DFSMShsm dump conditioning

DFSMShsm™ records now indicate that the source DASD volume (not the target DASD) was dumped. This is enabled with DFSMSdss™ dump conditioning when the source DASD volume is copied to the target DASD volume. The time stamp for the dump records reflects the time the DASD version (not the dump version) was created.

Fast replication dump copies

The DFSMShsm LIST, REPORT, and QUERY commands and the ARCXTRCT macro are modified to aid you in the use and monitoring of the fast replication dump copies. Auto dump also is modified to create dump copies of copy pools. Auto dump is only done if the copy pool has AUTODUMP=Y specified and only generation 0 is considered. Audit is now able to check the consistency of the FRBACKUP control data set records.

DFSMSdss logical data set copy

Currently, IMS™ invokes the DFSMSdss cross memory application interface (XMAPI) to logically dump IMS databases that are in use at the time of processing using concurrent copy. IMS uses a similar functionality for logical data set copy so that it may make instantaneous backups of data sets that exist on FlashCopy® Version 2 devices using Data Set FlashCopy. This support enables IMS users to instantaneously back up data sets that are in use at the time of processing using Data Set FlashCopy.

In addition to making instantaneous backups, IMS users may also use this support to immediately recover these backup copies that exist on FlashCopy V2 devices with minimal impact to application programs. IMS invokes DFSMSdss logical data set copy with FASTREPLICATION(REQUIRED) specified for backup copy operations and FASTREPLICATION(PREFERRED) specified for recover copy operations. This support is designed so that BWO functionality for IMS data sets during logical data set copy is similar to that of logical data set dump and restore. Existing interfaces that were originally implemented for IMS BWO for logical data set dump and restore are used to implement the IMS BWO for logical data set copy support.

DFSMSdss logical data set copy processing is modified when invoked via an application programming interface (API) and the EI22IMS and EI22BWOE bits are set ON by a UIM in the exit identification block.

DADSM/CVAF rapid index rebuild

This enhancement increases the speed and efficiency of DADSM's conversion from index to OS VTOC. This rebuild processing is moved into the device manager address space (DEVMAN) and both ICKDSF and DADSM are changed to use this new DEVMAN interface. This also speeds up the BuildIX and Reformat VTOC operations in ICKDSF.

OAM DB2 binary large object support

This enhancement enables objects larger than 32K to be stored using DB2®'s large object (LOB) support and the binary large object (BLOB) data type. The LOB environment is intended to be used for objects larger than 32 KB continuing the need for our 4 KB and 32 KB tables. Today, these same large objects would be stored in multiple rows in the 32 KB table with a 256 MB object, for example, taking 8000 plus rows to store. With this support, large objects can be stored directly in DB2 (in a LOB column), with each BLOB being a varying length string up to 256 MB in size.

For each object storage group, today there is an object directory table and an object storage table hierarchy consisting of a 4 KB and a 32 KB table. New with this support is an addition to this object storage table hierarchy, an LOB storage database structure. This new structure consists of an LOB base table and an LOB auxiliary table. The LOB base table resembles a 32 KB table with the addition of a ROWID column and changing the OTOBJ column datatype from *long varchar* to BLOB. The LOB auxiliary table contains the actual BLOB object represented by the OTOBJ column in the LOB base table. The LOB auxiliary table is managed exclusively by DB2 and is transparent to OAM. A new column ODLOBFL is also needed in the object directory table that will track whether the object resides in a LOB storage structure. As with other DB2 tables, sample jobs are provided for the addition of any new columns and tables.

To enable this support, a new keyword is supported on the OAM1 statement of the IEFSSNxx parmlib member to indicate that DB2's large object (LOB) support should be used. Once enabled, all objects larger than 32 KB (one row in the 32K table) are stored in the new LOB storage structure. When running in an OAMplex and sharing data across systems, prior to enabling this support, it is recommended that all systems be capable of the new LOB support. Coexistence support is also provided at the earlier release levels to detect and fail the retrieve of an object that is stored using DB2's LOB support.

There are also existing mechanisms within OAM that could be used to move object data to the new LOB storage structure using our single object recovery facility. It is also expected that any object transitioned to disk or recalled to disk using immediate recall support, is stored in the new LOB storage structure, as appropriate for the object size.

Object tape enhancements

This enhancement exploits VTS and tape with two functional enhancements. It adds the following:

- Automatic selection of RECYCLE with eligible tape volumes to the existing MOVEVOL with RECYCLE function

  Tape recycle support expands on the newly added RECYCLE option of the MOVEVOL utility to allow automatic selection of tape volumes for MOVEVOL with RECYCLE processing, based on installation-specified thresholds and limits.

- Support for an immediate backup copy

  Immediate backup copy currently in OAM supports a primary copy of the object and one or two backup copies with the backup copies typically being made in the evening or on weekends when our object storage management cycle (OSMC) is run.

3592 WORM tape support

With the 3995 optical library being withdrawn from marketing, tape and the virtual tape server (VTS) will play an increasing role in our support. With the recently available 3592 WORM tape support, there is a requirement that a backup copy be available on WORM media soon after the primary copy is stored, plus there is added insurance in knowing that the backup copy was made sooner rather than later.

Immediate backup copy support

To enable the immediate backup copy support, the backup frequency value for the assigned management class is checked for a value of 0 (existing management class parameter that OAM now uses that currently has a default of 1). If this new parameter is specified and auto backup is also specified, after the primary copy is stored, an immediate backup copy is scheduled. This immediate backup copy is made based on existing OAM parameter specifications for the first backup copy. If a second backup copy is also indicated, the second backup copy continues to be made when the object storage management cycle (OSMC) is run.

64-bit PDSE address space support

The PDSE restartable address space support shipped in earlier releases solved the ECSA/CSA problem. With that support, the PDSE control blocks were moved from common addressable storage into the new PDSE address space private area of virtual storage. Since the internal design of PDSE uses 31-bit virtual storage, there is an upper limit of about a million concurrently opened members of all PDSEs that can be supported. The multiple (two actually—one for linklist PDSEs and the other one for user PDSEs) PDSE Address Space support shipped in z/OS DFSMS 1.6 relieves this constraint to a small extent. However, there continues to be a limitation on the number of concurrently opened PDSE members that can exist on a system.

z/OS V1R8 solves this problem proactively by exploiting 64-bit addressable virtual storage. The area of PDSE that is the best candidate to be moved into the PDSE server's private area 64-bit addressable storage is cached directory pages for BMF and IMF, which currently reside in the SYSBMFDS dataspace.

This move to 64-bit virtual storage should be transparent to the end user of PDSE. There is one external change that provides installation control to limit of the amount of 64-bit virtual address space used for directory cache by the SMSPDSE and SMSPDSE1 address spaces. The 64-bit virtual storage limitation value is definable at PDSE startup processing via a new SMS initialization parameter. SMS initialization processing will be modified to provide support for the 64-bit limiting value for directory storage. PDSE components BMF and IMF will be modified to use 64-bit virtual storage. PDSE continues to use SSF services, and modifications to SSF is required.

DFSMSrmm support

DFSMSrmm™ extends its support for managing removable media across your enterprise. The DFSMSrmm CIM agent is updated to support creation, change and deletion of volumes and data sets in addition to the query and display capability provided in z/OSV1R7. The CIM agent now uses the OpenPegasus CIMOM and can run either on z/OS or on any other OpenPegasus supported Server. To better support systems across your enterprise, DFSMSrmm now exploits use of common time and provides support for displaying and setting dates and times in any chosen time zone.

Securing tape data sets with SAF

DFSMS introduces new options for securing tape data sets using SAF. The new options enable you to avoid the use of the RACF TAPEDSN option and the TAPEVOL class. Instead, you can use the DATASET class, enabling you to have common authorization for data sets regardless of the type of volume on which they are stored. DFSMS also provides options for ensuring that all data sets on a tape volume have common authorization, and that a user is authorized to overwrite an existing first file on a volume.

1.1.13 ISPF enhancements

In z/OS R8, ISPF SuperC Compare and Search-For are added to the list of functions supported on the Data Set List Actions panel. This means that you can invoke these functions directly on the selected data sets. When a data set or member is opened in ISPF Edit or Browse as a result of a Search-For operation, the Find function is initialized with the first SRCHFOR string.

ISPF Edit is enhanced so that the HEX command only displays data lines in hexadecimal format. Nondata lines such as profile lines and message lines will remain as text. This means that more data can be displayed on each panel.

A new member list primary command enables you to filter or subset the list of members to display only those members matching the supplied comparison argument. In the previous release it was possible to list the subset of members whose names matched a given pattern. It is now possible to subset using all member attributes, such as Size, Lib, Created, and Alias. The FILTER command can be applied repeatedly to drill down to members matching a particular set of attributes. A new option on the SRCHFOR command enables member list filtering to display only those members containing the search string. These enhancements reduce the effort needed to locate desired members.

The ISPF workstation connection program WSCON will be enhanced to automatically discover the IP address of the workstation. You won't need to remember or find its IP address to use the ISPF GUI or transfer files between host and workstation. This makes administration easier, especially in environments where you don't always connect to ISPF from the same workstation (IEL).

The ISPF client/server code is converted to use IBM's own C/C++ libraries. The client/server (ISPF GUI) code previously used an older version of the SAS/C runtime libraries. The benefits of this are mostly internal, and relate to cost and serviceability.

ISPF ensures that when a member that has aliases is renamed, the aliases are not "orphaned". In other words, as well as renaming the member, the `rename` command will also update all of the member's aliases to point to the new name.

The SCLM component of ISPF includes the following enhancements:

- You can specify a language description when defining an SCLM language. This helps you to select the correct language when using the SPROF (SCLM Edit Profile) command and SCLMINFO service.
- Error messages generated by the COBOL parser FLMLPCBL are improved to provide more information about return codes.

In z/OS V1R8, ISPF edit information messages are externalized. In interactive mode, the message identifier, short message text, and long message text are displayed on the screen. This change places these messages in dialog variables so that application developers can test for the outcome when a command is issued in an edit macro. Previously, a developer would not receive the same level of information from an EDIT command when issued within a macro as they would when issuing the command interactively within an EDIT session. This change will enable you to automate editing functions using more complex and robust edit macros.

File tailoring skeletons

The control statement syntax for file tailoring skeletons is greatly enhanced. This change provides the following capabilities:

- Better support for handling strings, especially strings containing embedded blanks
- Better arithmetic functionality
- The ability to invoke REXX to process dialog variables in ways that are not available in the native skeleton syntax

Previously, the control statement syntax available to users of skeletons was very limited. File tailoring did not handle strings containing blank characters and could not perform substring functions. Plus (+) and minus (-) were the only arithmetic operators supported by the)SET control statement.

HILITE language parser

The HILITE language parser in ISPF Edit is enhanced to support alternate margins for source statements. This is to allow syntax highlighting to work with languages like PL/I that support setting of left and right boundaries. Previously it was assumed that source statements conformed to the default margins. There was no way to specify alternate margins. For sites that followed different standards, highlighting could produce unpredictable results.

Fault analyzer language definitions

In z/OS V1R8, sample fault analyzer language definitions are shipped with the SCLM component of ISPF. Language definitions specify to SCLM the languages that are used for a project. They provide language-specific control information such as the language name and the definition of the language translators. Previously, only Pascal and assembler definitions were shipped with SCLM.

SCLM project managers have the option to enable member-level locking for a project. Member-level locking prevents all users (except the one which originally drew down the member into the development area) from changing the member, unless they are registered as an administrative user. In situations where a development level is shared among users, previously there was nothing to prevent multiple users from editing the same member.

Support is added to the SCLM COBOL parser (FLMLPCBL) so that when it encounters SQL includes, a unique SQL include set is assigned. One of the functions of the SCLM parsers is to determine all of a module’s dependencies that will be copied into the source. This enhancement enables FLMLPCBL to handle COBOL DB2 programs in the same manner as COBOL copybooks.

1.1.14 Binder support

Program management binder has added the following functions in this release:

- Imod support for relative immediate instructions with references between compile units, a capability other platforms already have and one that z/OS needs for competitiveness.
- New binder options are being provided that allow you to remove unused parts of executable modules, thus reducing virtual storage usage.
- An enhancement to include providing an AMBLIST JCL Interface control statement syntax more consistent with that used for traditional MVS data sets, thus preventing the creation of non-executable load modules. This allows the Binder to work more like UNIX linkers.
- Binder diagnostics are included for files in the hierarchical file system to simplify development of z/OS UNIX applications that use the binder API.

- An auto call from archive libraries allows simplification of binding in the UNIX shell.

1.1.15 Language Environment enhancements

In order to improve the execution environment between XPLINK and non-XPLINK compiled programs, Language Environment (LE) now allows a function pointer to be passed from a non-XPLINK compiled program to an XPLINK compiled program without any restrictions.

Note: The non-XPLINK compiled and XPLINK compiled programs must be in separate DLLs.

Language Environment is providing a new interface that allows re-initialization of the writeable static area (WSA) for a DLL without the overhead of an additional load/delete. This interface is used by the z/OS XL C/C++ compiler to improve compile performance.

PL/I run-time migration aid

In order to help customers migrating from the pre-Language Environment PL/I run-time product to Language Environment, a new migration aid is provided. The capability of the STORAGE run-time option is enhanced to include CLEAR as an acceptable value for the third sub-option. This will cause the unused portion of the initial stack segment to be cleared just prior to invoking the main procedure. Many old PL/I applications relied on this behavior.

IPA link component

The IPA link component of the C/C++ feature of z/OS V1R8 exploits 64-bit addressing, allowing for increased storage limits and the capability to handle very large applications. CDA is enhanced to provide assembler debug support, 64-bit go number support, and improved debugging information for advanced C++ constructs.

New LE callable services

Several new callable services in LE have been created to provide various new functions, as follows:

- A parameter string greater than 80 in length is returned to the caller, enhancing the string's dynamics.
- You can specify your own reason codes, rather than having a service do it for you, as well as allowing you to specify various cleanup values.
- You are provided with current system information about the enclave, for example the subsystem it's running on, environment information, and member languages.
- A new interface has been created that provides a language-neutral Language Environment feature that fully supports accessing environment variables (get, set, and clear).

XPLINK tracing

A new capability for tracing XPLINK and non-XPLINK transactions is designed to benefit porting of UNIX applications to z/OS. Currently there is no way to isolate the performance degradation due to the XPLINK or non-XPLINK transitions in a mixed XPLINK/non-XPLINK application. Language Environment now provides the tracing capability that can be customized to trace the transitions between upward (non-XPLINK) and downward (XPLINK) growing stacks and more easily diagnose the performance bottlenecks in the mixed XPLINK/non-XPLINK applications.

CONVLIT compiler option

The C/C++ CONVLIT compiler option, used to control the conversion of string literals in code, is extended to handle Unicode data or constant literals in programs. For example, the option facilitates C/C++ application programs to use ICU. The support for the keyword, UNICODE, allows for easier porting of applications from ASCII platforms to z/OS.

1.1.16 REXX enhancements

z/OS V1R8 has removed the limit for the REXX variable pool storage. This allows applications to continue to function as the amount of data generated by existing applications continues to grow. Prior to this support, output trapped from authorized commands or programs invoked by REXX were limited by the amount of available below-the-line storage. In addition, the amount of storage available for CLIST variables was limited also by the amount of available below-the-line storage necessary for the system name table (SNT) and the system variable table (SVT).

1.1.17 z/OS XML System Services

z/OS XML System Services (z/OS XML) is a new component that is designed to provide an optimized set of basic services for parsing XML documents. It is expected to be of use to IBM, ISV, and customer middleware and applications having high performance or unique environmental XML parsing requirements. These services provide the ability to run in cross-memory and service request block (SRB) modes. Support is provided in the assembler language interface. IBM plans to add C/C++ high-level language support in a future release.

1.1.18 Unicode enhancements

Unicode is a universal encoding scheme, allowing applications to store data regardless of code pages and character sets (such as ASCII and EBCDIC). This is vital to IBM's globalization imperatives. The Unicode services element of z/OS provides general purpose programming interfaces (APIs) that applications like DB2 can use to convert data to and from Unicode. On zServers, there are specific hardware instructions to support Unicode conversion and translation.

Enhancements to Unicode will benefit general customer environments, given that Unicode conversion services are used by middleware applications such as the current exploiters including DB2 and the COBOL compiler. Future exploiters include CICS®, EWLM, and other parts of the operating system.

Provide StringPrep support

StringPrep prepares Unicode strings for use in network protocols based on profiles. Profiles of StringPrep are sets of rules and data that describe how Unicode strings are prepared. Each profile contains tables that describe how a code point should be treated. These profiles allow you to enter internationalized text strings in applications and have the highest chance of getting the content of the strings correct. In this case, "correct" means that if two different people enter what they think is the same string into two different input mechanisms, the strings should match on a character-by-character basis. This new support of StringPrep involves the creation of new callable services, in both Assembler and C, and the shipment of these profiles.

Using Unicode in SRB mode

Software products that use Unicode in SRB mode, such as DB2, cannot take advantage of dynamic loading, as when DB2 tries to make a query. By replacing the current use of

WAIT/POST with PAUSE/RELEASE, this allows for dynamic loading of tables in SRB mode. Currently, WAIT/POST cannot be used in SRB mode, as when executing DB2 queries. In addition, the current behavior of page-fixing the conversion tables is eliminated. This allows for better storage management and will not affect the existing Unicode services functionality.

Bidi and character shaping support

Bidirectional languages are languages such as Arabic and Hebrew, which are written and read mainly from right to left; but some portions of the text, such as numbers and embedded Latin languages (for example, English) are written and read left to right. Additional characteristics of bidirectional languages include the following:

- Visual order versus logical order
- Symmetric swapping
- Number formats
- Cursive (shaping) versus noncursive

To be able to handle these languages, a layout transformation must be created. To implement this, this release provides new callable services, in both Assembler and C.

Unicode normalization

Unicode normalization service works with the Unicode data base version 3.0.1, so in this release, the upgrade of the normalization service to the latest Unicode version 4.0.1 allows Unicode to handle the latest character suite of surrogates and supplementary code points. In addition, code is added to support 4-byte characters called surrogates.

Collation sequences for languages

New collation tables, specifically needed by Siebel®, PeopleSoft® applications, and DB2 V9 are required. This release provides support for collator names (all based on the UCA (Unicode collation algorithm) that conforms to Unicode standard v4.0. New keywords allow collation sequences for languages.

Euro and locale support

New Euro support in locales is provided in alignment to new countries that have joined the ECC (European Economic Community). Euro support is now provided for the Czech Republic, Estonia, Hungary, Latvia, Lithuania, Poland, Slovakia, and Slovenia. Additionally, new locales available from the Globalization Center of Competency are added in compliance with the Globalization White Paper.

1.2 SMP/E V3R4

SMP/E V3R4 is available under its own product number and also remains a base element of z/OS. This allows customers who are licensed for a currently supported release of z/OS to order and install the latest release of SMP/E without having to upgrade their entire operating system. The advantage is that products other than z/OS can exploit the packaging and installation enhancements in SMP/E without having to install the prerequisites for a new level of the operating system.

1.2.1 Internet software delivery

In addition, since SMP/E plays a key role in Internet delivery of software, it allows IBM to exploit the Internet delivery and installation technologies in SMP/E sooner without having to wait for customers to migrate to new levels of the operating system. SMP/E V3R4 is available at no additional charge. It is intended for installations who have a license for z/OS Version 1 (5694-A01). SMP/E V3R4 is incorporated into z/OS R7 and R8 and provides the following enhancements for this release:

1.2.2 Internet service delivery

SMP/E Internet service retrieval is the latest enhancement in IBM's service delivery evolution for the z/OS platform. SMP/E V3R4 addresses the common inhibitors to Internet delivery of service, and can simplify and automate the entire z/OS service acquisition process. SMP/E Internet service retrieval is designed to order and retrieve z/OS platform service from a dedicated IBM server via the Internet in one simple step right from z/OS. With SMP/E V3R4 you are now able to order and retrieve service on demand to obtain exactly what you need when you need it, or you can automate the service delivery process. By scheduling an SMP/E job to run once a week, or even every night, you can order and download the latest HOLDDATA and PTFs and have these service updates available exactly when you want them. You can request corrective service, three forms of preventive service, and even HOLDDATA.

1.3 Infoprint Server enhancements

Improvements delivered in z/OS V1R8 benefit z/OS customers with more robust communication between IP PrintWay™ and PCL and PostScript printers that support the PJL Back-Channel protocol. Customers can reduce their costs for distributed printing by having jobs that are interrupted while printing resume at or near the point at which they were interrupted, rather than starting over from the first page of the job. Secure print from Infoprint Server using the industry standard IPSec security protocol gives customers the ability to implement distributed printing solutions over TCP/IP in industries that require the assurance that their information is secure from the server to the printer—government agencies, Defense Department, clinics and hospitals and financial institutions. Printer and job management is enhanced with usability improvements to the Web-based Infoprint Central management user interface.

Note: The improved management of PCL and PostScript printers using the PJL protocol is the most important new function in z/OS V1R8. It has been a long-standing customer requirement.

1.3.1 IP PrintWay error recovery

In z/OS V1R8, Infoprint Server provides improved management of error recovery in IP PrintWay for PCL and PostScript printers that use the Printer Job Language (PJL) to send and receive information from a printer that supports PJL. The commands allow IP PrintWay to record information from the printer on how far a job has successfully printed. The PJL information allows IP PrintWay to determine:

- That the job fully printed on the printer.
- On what page the job failed if it did not complete printing.
- On retries, IP PrintWay can send a command to the printer to tell the printer on what page to start printing the data again. IP PrintWay has to send all the data again but the printer can determine on what page to actually start printing the data.

1.3.2 IPSec security protocol

Infoprint Server is using Communication Server's built-in IPSec security protocol, and configuration with Infoprint Server is documented, with pointers to the complete Communication Server IPSec documentation and configuration tool. This enables you to more easily take advantage of IPSec to encrypt print files sent from Infoprint Server to distributed printers over a TCP/IP network. IPSec rules allow you to select, from among multiple encryption algorithms, the one most appropriate for your needs. The system administrator can also define rules governing who can use TCP/IP networking programs, who can connect to systems, and what security each connection is to have. The ability to secure print data during transmission to a remote location has become increasingly important to customers in many industries who must comply with regulatory requirements that demand that client data be kept secure and private. This function is supported by any distributed printer that supports the IPSec protocol.

1.3.3 Infoprint Central

Infoprint Server's Infoprint Central Web-based print management GUI was introduced in z/OS V1R5. In z/OS V1R8, it is enhanced with several new functions and usability features, as follows:

- Ability to stop an IP PrintWay print job but hold (not cancel) the job.
- Allow selection of messages in a log by hours in addition to days to improve search granularity.
- Provide an audit trail for GUI actions into job and printer logs.
- Force error messages to open when an action is done on the Summary table that causes an error condition.
- Expand the ping action to include the TRACEROUTE command to provide more extensive diagnostic information.

1.3.4 JES extended status support

Infoprint Central, which displays job information and attributes in a Web-based management GUI, now exploits enhancements to the JES extended status subsystem interface to obtain data set level information about spool data sets. This eliminates duplicate calls from Infoprint Server to obtain a single piece of status information. It also eliminates "lock-out" conditions on the print data set. This enhancement makes Infoprint Central faster and more efficient in obtaining and presenting job status.

1.3.5 Internet delivery of PTFs

Serviceability improvements have been made to restructure the Infoprint Server program so that you can use the preferred and timelier electronic delivery method for PTFs.

1.3.6 JCL support

In z/OS R8, a new JCL output descriptor is added, PRTATTRS. This descriptor allows the user to pass job attributes to Infoprint Server. The PRTATTRS parameter is used as a general-purpose interface to pass information to Infoprint Server in cases where a specific JCL keyword does not exist. This gives you the ability to pass more job attributes to Infoprint Server and increase flexibility for handling print requests on z/OS.

1.4 HyperPAV support

HyperPAV is the name of a new feature of Parallel Access Volumes (PAV) that reduces the number of PAV-aliases needed per logical subsystem (LSS) by an order of magnitude and still maintain optimal response times.

This is accomplished by no longer statically binding PAV-aliases to PAV-bases and adjusting the bindings via the Workload Manager. Instead, in HyperPAV mode, PAV-aliases are bound to PAV-bases only for the duration of a single I/O operation. This reduces the number of aliases required per LSS

1.4.1 PAV support today

Parallel Access Volume (PAV) technology has been around for quite some time now. With PAVs, each base volume within a logical subsystem (LSS) can be configured with one or more aliases. Each exposure (base or alias) is capable of having an I/O active simultaneously. For example, a base volume with 2 aliases could have 3 I/Os started at the same time. Additional I/Os initiated to the base volume remain queued in z/OS until an exposure to it becomes available. Applications direct I/O to the base volume. If the base volume is busy during I/O initiation, z/OS will automatically select one of the alias exposures assigned to that base volume and start the I/O using that exposure.

As shown in Figure 1-1 on page 24, the UCB is the z/OS representation of a device. Base PAV devices are used by applications. Alias PAV devices are used only by z/OS.

Alias exposures are relatively static today. During initialization, z/OS queries the storage server to determine which alias exposures belong to which base volumes. The ESS is used to provide the initial alias/base configuration. Over the life of an IPL of the storage server, this alias/base configuration can change due to I/O loads within the logical subsystem. WLM is used to determine if alias exposures should be reassigned to different bases, and will facilitate those alias moves to meet overall system workload goals.

Each image within a sysplex shares the same view of the LSS, and, therefore, the same alias/base relationships. If an alias is moved, all systems in the sysplex must reassociate the alias with its new base volume.

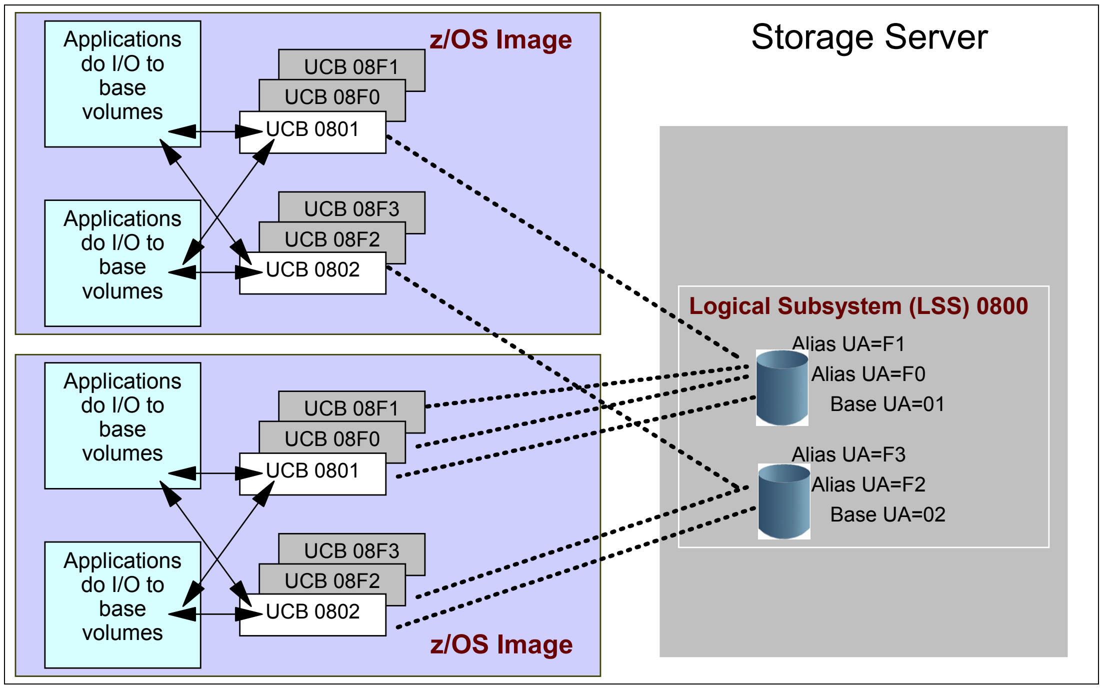


Figure 1-1 Base PAV devices used by applications

1.4.2 HyperPAV with z/OS V1R8

With HyperPAV technology, z/OS uses pools of aliases. As each application I/O is requested, if the base volume is busy with another I/O, z/OS selects a free alias from the pool, quickly binds the alias device to the base device and starts the I/O. When the I/O completes, the alias device is used for another I/O on the LSS or is returned to the free alias pool.

If too many I/Os are started simultaneously, z/OS queues the I/Os at the LSS level. When an exposure frees up that can be used for queued I/Os, they are started. Queued I/O is done FIFO within assigned I/O priority.

As shown in Figure 1-2 on page 25, for each z/OS image within the sysplex, aliases are used independently. WLM is not involved in alias movement so it does not need to collect information to manage HyperPAV aliases.

Benefits of HyperPAV

There is an increases in I/O parallelism. As CPU speeds increase and I/O distances increase, the need for on demand I/O parallelism increases. HyperPAV allows you to define aliases that are applicable to more devices, since they are not statically used.

As a result of this support, it is possible to reduce the number of required aliases, allowing more addressable device numbers to be available. WLM is not involved in measuring and moving aliases.

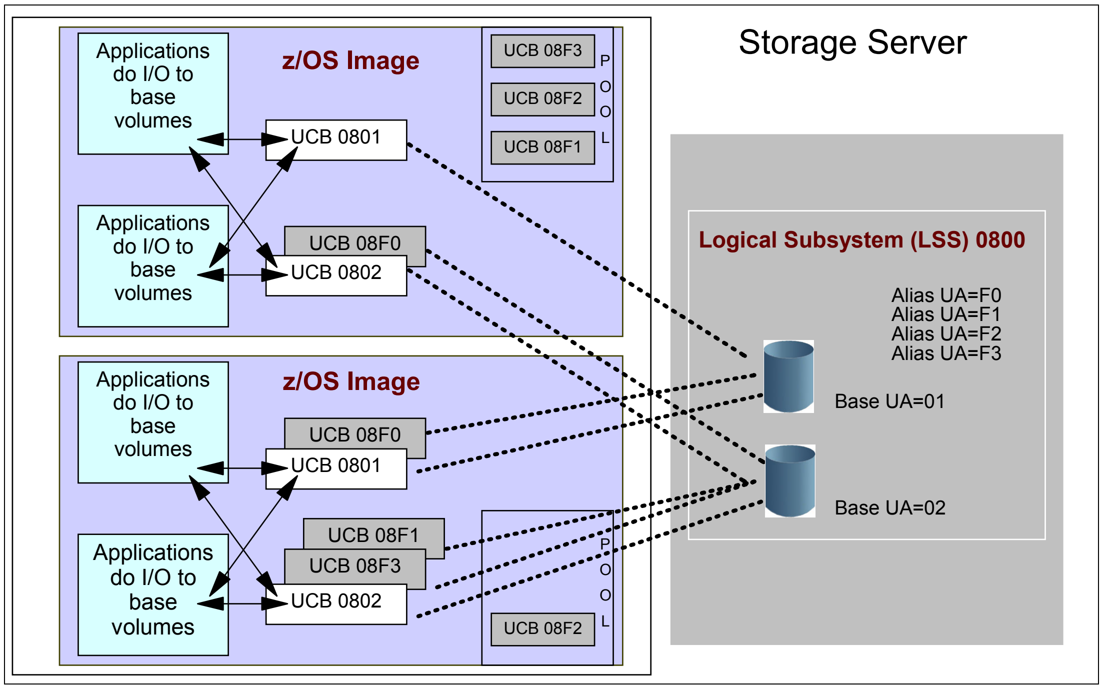


Figure 1-2 Base HyperPAV devices used by applications

1.4.3 Migration considerations

There are several migration considerations to implement HyperPAV.

Hardware requirements

The hardware needed to support HyperPAV:

- DS/8000 version 2.4
- FICON® connectivity

Software requirements

The software requirements are:

- z/OS V1R6, z/OS V1R7, and z/OS v1R8 with the following APARs:
  - OA13915 – IOS support
  - OA13928, OA13929, OA14002, OA14005 – DFSMS support
  - OA12699 – WLM support
  - OA14556 – GRS support
  - OA14248 – ASM support
- RMF support is available with the following APAR:
  - APAR OA12865

  For information on the RMF enhancements, see 5.5, “HyperPAV support” on page 95.

SYS1.PARMLIB specification

The SYS1.PARMLIB(IECIOSxx) member has the following options in support of HyperPAV:

HYPERPAV=YES|NO|BASEONLY

YES	Attempt to initialize LSSes in HyperPAV mode
NO	Do not attempt to initialize LSSes in HyperPAV mode
BASEONLY	Attempt to initialize LSSes in HyperPAV mode, but only start I/Os on base volumes

Operator commands

The following operator commands relate to the HyperPAV support:

- SETIOS HYPERPAV=YES|NO|BASEONLY can be used to dynamically implement the parmlib specification.
- SET IOS=xx
- D M=DEV, as shown in Figure 1-3

```
d m=dev(0710)
IEE174I 23.35.49 DISPLAY M 835
DEVICE 0710   STATUS=ONLINE
CHP                   10   20   30   40
DEST LINK ADDRESS     10   20   30   40
PATH ONLINE           Y    Y    Y    Y
CHP PHYSICALLY ONLINE Y    Y    Y    Y
PATH OPERATIONAL      Y    Y    Y    Y
MANAGED               N    N    N    N
CU NUMBER             0700 0700 0700 0700
MAXIMUM MANAGED CHPID(S) ALLOWED:  0
DESTINATION CU LOGICAL ADDRESS = 07
SCP CU ND         = 002107.000.IBM.TC.03069A000007.00FF
SCP TOKEN NED     = 002107.900.IBM.TC.03069A000007.0700
SCP DEVICE NED    = 002107.900.IBM.TC.03069A000007.0710
HYPERPAV ALIASES IN POOL  4
```

Figure 1-3 Command to display HyperPAV aliases

- D IOS,HYPERPAV
- DEVSERV QPAV,dddd

Installation and migration to z/OS V1R8

This chapter contains the following information regarding insalling and migrating to z/OS V1R8:

- Ordering z/OS V1R8
- z/OS base elements
  - New and changed base elements
- Base elements removed
- z/O V1R8 optional features

2.1 Ordering z/OS V1R8

The program number for z/OS Version 1 Release 8 is 5694-A01. When ordering this program number, remember to order all the optional features that you were licensed for in previous releases of z/OS.

Attention: In z/OS V1R8 there are only two export controlled unpriced features:

- z/OS Security Level 3 and Communications Server Security Level 3.
- z/OS Security Level 3 contains the sub-elements:
  - IBM Tivoli Directory Server for z/OS Security Level 3
  - Network Authentication Service Level 3
  - OCSF Security Level 3
  - System Secure Sockets Layer (SSL) Security Level 3

Typically, when one new z/OS release becomes orderable in ServerPac and CBPDO, the previous release is orderable for only a month. Due to this short overlap, it is very important that you order the z/OS release you need for migration and coexistence while it's still available for ordering.

Note: z/OS V1R4 users should plan to order z/OS V1R7 before the availability of z/OS V1R8.

Ordering methods

z/OS V1R8 can be ordered from one of the following methods:

- ServerPac electronic delivery which has been available since January 2005
- SMP/E Internet Service Retrieval which has been available since September 2005
- SystemPac® electronic delivery which has been available since October 2005
- ShopzSeries enhancements made in January 2006

2.1.1 Electronic delivery

z/OS V1R8 can be ordered electronically. In many countries you may order z/OS electronically through ShopzSeries. ShopzSeries provides customers a self-service capability for planning and ordering S/390® software (and service) upgrades over the Web. It is the strategic worldwide self-service ordering system for zSeries software. You can order products through ShopzSeries and have them delivered electronically in some countries. To access ShopzSeries, use the web site at:

```
http://www.ibm.com/software/shopzseries
```

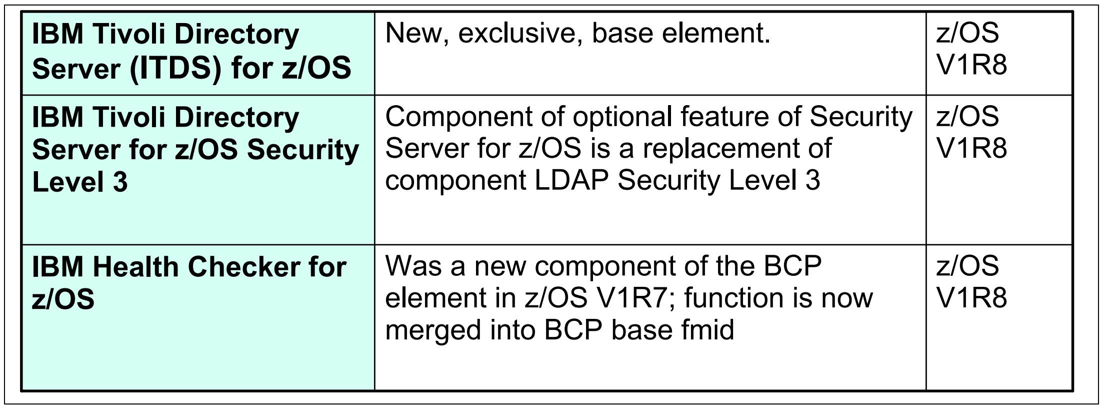
The z/OS product on ServerPac can be ordered from ShopzSeries electronically. This electronic ability was made generally available on January 10, 2005.

2.2 Base elements

z/OS and z/OS.e consist of base elements and optional features. The base elements (or simply elements) deliver essential operating system functions. When you order z/OS or z/OS.e, you receive all of the base elements. However, with z/OS.e, some base elements are not functional or not licensed for use, or both.

2.2.1 New and changed base elements and features

IBM Tivoli Directory Server (ITDS) for z/OS ITDS for z/OS is a new base element in z/OS V1R8.



IBM Tivoli Directory Server (ITDS) for z/OS	New, exclusive, base element.	z/OS V1R8
IBM Tivoli Directory Server for z/OS Security Level 3	Component of optional feature of Security Server for z/OS is a replacement of component LDAP Security Level 3	z/OS V1R8
IBM Health Checker for z/OS	Was a new component of the BCP element in z/OS V1R7; function is now merged into BCP base fmid	z/OS V1R8

Figure 2-1 New and changed base elements and features

2.3 Base elements and functions removed in z/OS V1R8

This section lists base element items and features of some base elements that are withdrawn in z/OS V1R8. These items were last shipped in z/OS V1R7. You should take this into account as you plan your migration to z/OS V1R8. The removal of these functions may have migration actions which you can perform now, in preparation for z/OS V1R8. These base elements are also shown in Figure 2-2 on page 31.

2.3.1 One-byte console IDs

The remaining support for one-byte console IDs has removed from control blocks CIB, CSCB, ORE, WQE, and XSA. This completes the removal of one-byte console IDs in z/OS. Instead of using one-byte console IDs, console names should be used.

2.3.2 msys for Setup

Support for the following plug-ins fo msys for Setup are withdrawn in z/OS V1R8. You will no longer be able to use msys for Setup for function enablement, setup, or configuration of these areas of z/OS:

- TCP/IP services
- z/OS UNIX System Services
- Language Environment
- Parallel Sysplex
- ISPF
- RMF

The DB2 V8 msys for Setup plug-in is unaffected and remains available for setup and configuration of DB2. The TCP/IP plug-in will continue to be available for download via the Web and will no longer require msys for Setup. IBM intends to continue to deliver improvements to help with z/OS setup and configuration in the future.

2.3.3 Firewall Technologies

The Firewall Technologies component of the Integrated Security Services element has now removed in z/OS V1R8. Many Firewall Technologies functions have been stabilized for some time and can be replaced using comparable or better functions provided by or planned for Communications Server, notably, IPSecurity. In addition, a functionally rich downloadable tool is planned to replace the IPSecurity and IP Filtering configuration GUI support. The following functions will be removed without replacement:

- FTP Proxy services
- Socks V4 services
- Network Address Translation (NAT)
- RealAudio (TM) support

2.3.4 msys for Operations

msys for Operations element is removed in z/OS V1R8. IBM plans to transition many of the msys for Operations functions to a new user interface and infrastructure in a future release of z/OS.

2.3.5 Communications Server

The following functions have been removed from Communications Server in z/OS V1R8:

- TCP/IP configuration profile block definition statements:
  - ASSORTEDPARMS
  - ENDASSORTEDPARMS
  - KEEPALIVEOPTIONS,
  - ENDKEEPALIVEOPTIONS

  Equivalent capability is provided for the KEEPALIVEOPTIONS statements by INTERVAL and SENDGARBAGE on the TCPCONFIG statement. For information on TCP profile and configuration statements, refer to z/OS Communications Server IP Configuration Reference.
- Equivalent capability is provided for the ASSORTEDPARMS statements by the following statements:
  - GLOBALCONFIG
  - IPCONFIG
  - TCPCONFIG
  - UDPCONFIG

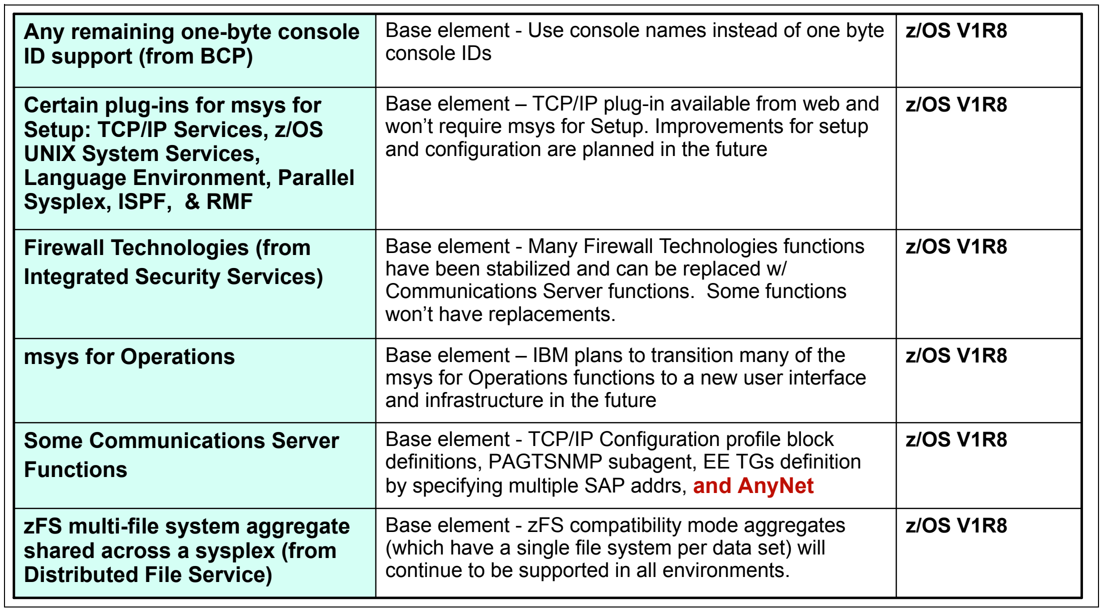
The SNMP Service Level Agreement (SLA) Version 1 MIB and the PAGTSNMP subagent. In z/OS V1R5, Communications Server provided a new SNMP SLA Version 2 MIB and NSLAPM2 subagent. IBM recommends that you migrate to the new Version 2 MIB and NSLAPM2.

The option of defining parallel Enterprise Extender TGs by specifying multiple SAP® addresses. Beginning in z/OS V1R5, parallel EE TGs may be defined by using different EE VIPAs on one (or both) of the endpoints.

AnyNet is removed, but you may implement other IBM solutions such as Enterprise Extender (EE) as a replacement for AnyNet.

2.3.6 Multi-file system aggregates in a shared environment

zFS file systems contained in multi-file system aggregates that are to be shared across systems in a sysplex support has been removed in z/OS V1R8. If you attempt to mount zFS file systems contained in multi-file system aggregates will fail in a z/OS UNIX shared file system environment. Mounting zFS compatibility mode aggregates, which have a single file system per data set, will continue to be supported in all environments.



Any remaining one-byte console ID support (from BCP)	Base element - Use console names instead of one byte console IDs	**z/OS V1R8**
Certain plug-ins for msys for Setup: TCP/IP Services, z/OS UNIX System Services, Language Environment, Parallel Sysplex, ISPF, & RMF	Base element – TCP/IP plug-in available from web and won't require msys for Setup. Improvements for setup and configuration are planned in the future	**z/OS V1R8**
Firewall Technologies (from Integrated Security Services)	Base element - Many Firewall Technologies functions have been stabilized and can be replaced w/ Communications Server functions. Some functions won't have replacements.	**z/OS V1R8**
msys for Operations	Base element – IBM plans to transition many of the msys for Operations functions to a new user interface and infrastructure in the future	**z/OS V1R8**
Some Communications Server Functions	Base element - TCP/IP Configuration profile block definitions, PAGTSNMP subagent, EE TGs definition by specifying multiple SAP addrs, **and AnyNet**	**z/OS V1R8**
zFS multi-file system aggregate shared across a sysplex (from Distributed File Service)	Base element - zFS compatibility mode aggregates (which have a single file system per data set) will continue to be supported in all environments.	**z/OS V1R8**

Figure 2-2 Base elements removed in z/OS V1R8

2.4 Optional features with z/OS V1R8

The optional features (or simply features) are orderable with z/OS or z/OS.e and provide additional operating system functions. Some optional features that are orderable with z/OS are not orderable with z/OS.e. Optional features are unpriced or priced. Unpriced features are shipped to you only if you order them. If you plan to use any unpriced features, you should order them when you order your base elements. You must not wait until the next release becomes available. Once a release's base elements are no longer orderable, neither are its unpriced features.

2.4.1 Changed optional features

IBM Tivoli Directory Server (ITDS) for z/OS Security Level 3 as a component of optional feature z/OS Security Level 3 is a replacement for the component LDAP Security Level 3, which was added in V1R6.

2.5 Priced features

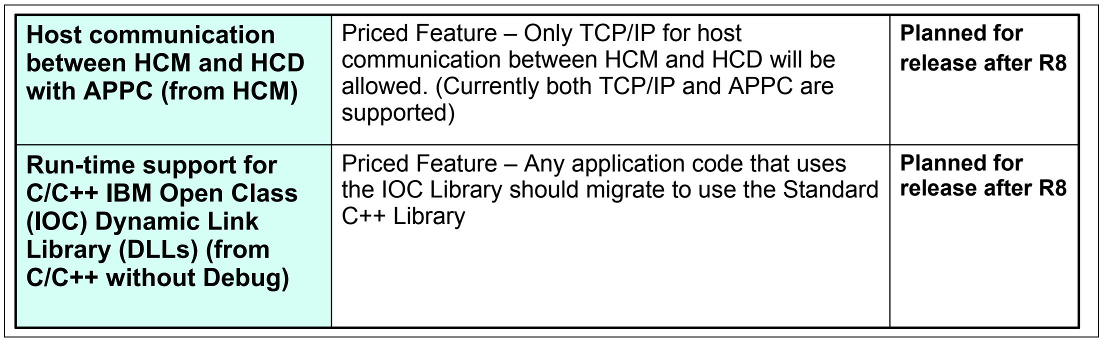
Priced features are always shipped to you. When IBM packages your order, the priced features are enabled that you ordered. These features are ready to use after you install z/OS or z/OS.e (and customize them as needed). Priced features that you did not order are disabled. Although they are installed on your system, you cannot use them. Later on, if you decide to use them, you notify IBM and you enable them dynamically (which is known as dynamic enablement). You dynamically enable by updating parmlib member IFAPRDxx and you notify IBM by contacting your IBM representative.

2.6 Functions planned to be withdrawn in a future release

IBM has announced it intends to remove in a future z/OS release the functions described in this section. You are encouraged to consider these removals when making plans for system upgrades. While these statements represent IBM's current intentions, IBM development plans are subject to change or withdrawal without further notice.

For more information, and for all previously announced statements of direction affecting z/OS V1.7 and future release, visit:

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



Host communication between HCM and HCD with APPC (from HCM)	Priced Feature – Only TCP/IP for host communication between HCM and HCD will be allowed. (Currently both TCP/IP and APPC are supported)	**Planned for release after R8**
Run-time support for C/C++ IBM Open Class (IOC) Dynamic Link Library (DLLs) (from C/C++ without Debug)	Priced Feature – Any application code that uses the IOC Library should migrate to use the Standard C++ Library	**Planned for release after R8**

Figure 2-3 Priced features removed after z/OS V1R8

BIND DSN 4.9.3

In a future release the support for BIND DNS 4.9.3 will be removed from Communications Server. Customers should implement BIND DNS 9.2.0 as a replacement. BIND DNS 9.2.0 is included in the product beginning with z/OS V1R4. Customers exploiting the Connection Optimization (DNS/WLM) feature of BIND 4.9.3 should investigate alternative solutions, such as the Sysplex Distributor function.

ISPF panels removed from DFSORT

The English and Japanese ISPF panels will be removed from DFSORT™ in a future release. This limited function interactive facility will no longer be provided, and there will be no replacement.

DFSMS VSAM functions

From DFSMS, support for the VSAM IMBED, REPLICATE, and KEYRANGE attributes will be withdrawn in a future release. No supported release of z/OS or OS/390 allows you to define new VSAM data sets with these attributes. Using them for existing data sets can waste DASD space and can often degrade performance. When this support is withdrawn, you will not be able to process data sets with these attributes. It is best to plan for this removal now, with the aid of a tool that will help you identified affected data sets.

A tool is available for download to help identify VSAM data sets that contain the obsolete attributes IMBED, REPLICATE, or KEYRANGE. It is available from the software server (ftp.software.ibm.com) in the s390/mvs/tools directory as IMBDSHIP.JCL.TRSD. This will need to be downloaded in binary format and untersed using TRSMAIN. Note that this tool is provided as is. Instructions for use of the tool are included in the downloaded JCL.

Note: While no announcement has been made of the specific date or release that data sets with these attributes will no longer be able to be opened, it would be prudent to use this tool to identify the data sets in your installation and begin converting them to versions that do NOT contain these attributes. IMBED and REPLICATE were intended as performance improvements which have been obsolete by newer cached DASD devices. Striped data sets provide much better performance than KEYRANGE, and should be viewed as a candidate for any existing KEYRANGE data sets.

zFS multi-file system aggregates

In a future release, IBM plans to withdraw support for zFS multi-file system aggregates. When this support is withdrawn, only zFS compatibility mode aggregates will be supported. A zFS compatibility mode aggregate has a single file system per data set.

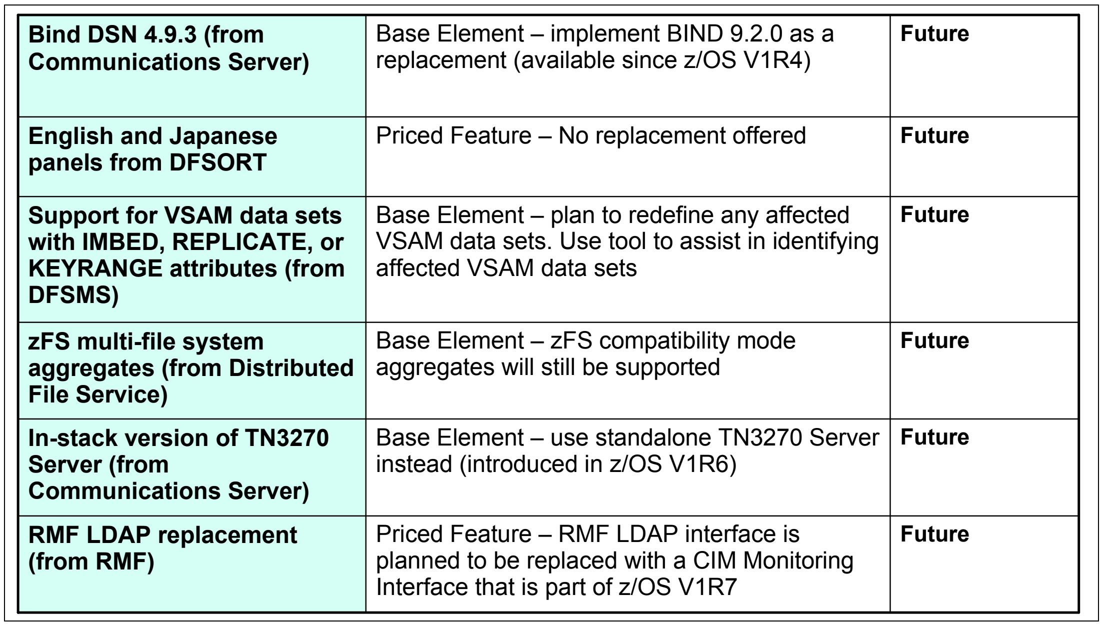
TN3270 Server

z/OS V1R6 Communications Server and subsequent releases include a standalone TN3270 Server. This standalone TN3270 server is expected to provide increased flexibility, improved reliability, and simplified problem diagnosis as compared to the in-stack version of the TN3270 Server. In a future release of z/OS Communications Server, support for the in-stack version of the TN3270 Server is planned to be discontinued. In preparation for that change, customers should consider implementing the standalone TN3270 Server. For more information, refer to:

```
http://www.ibm.com/software/network/commserver/zos/
```

RMF LDAP backend

IBM plans to replace the RMF LDAP backend in a future release of the operating system. The RMF LDAP interface currently allows access to RMF performance data from application programs. This functionality will be replaced with a Common Information Model (CIM) Monitoring interface that is now part of z/OS V1R7.



Bind DSN 4.9.3 (from Communications Server)	Base Element – implement BIND 9.2.0 as a replacement (available since z/OS V1R4)	**Future**
English and Japanese panels from DFSORT	Priced Feature – No replacement offered	**Future**
Support for VSAM data sets with IMBED, REPLICATE, or KEYRANGE attributes (from DFSMS)	Base Element – plan to redefine any affected VSAM data sets. Use tool to assist in identifying affected VSAM data sets	**Future**
zFS multi-file system aggregates (from Distributed File Service)	Base Element – zFS compatibility mode aggregates will still be supported	**Future**
In-stack version of TN3270 Server (from Communications Server)	Base Element – use standalone TN3270 Server instead (introduced in z/OS V1R6)	**Future**
RMF LDAP replacement (from RMF)	Priced Feature – RMF LDAP interface is planned to be replaced with a CIM Monitoring Interface that is part of z/OS V1R7	**Future**

Figure 2-4 Base elements and features to be removed in the future

2.7 Central storage

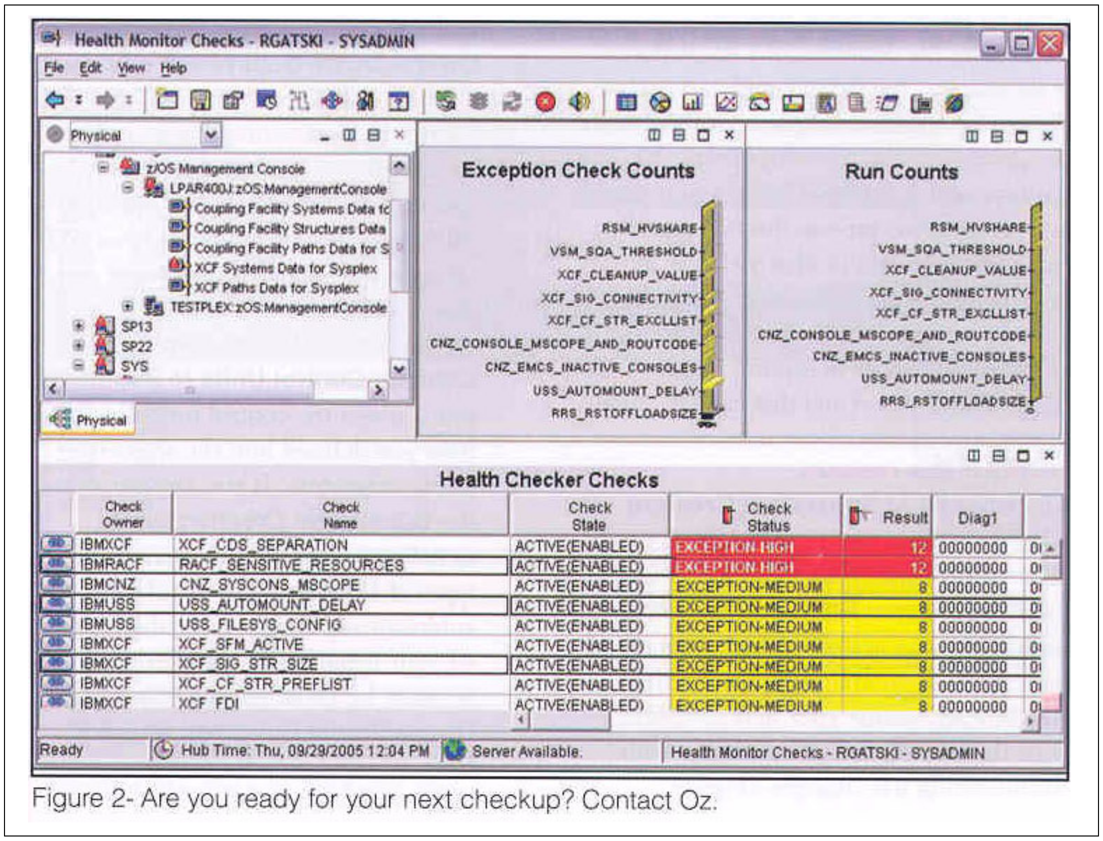
z/OS V1R8 provides the capability to support up to 4 TB (4,398,046,511,104 bytes) of central storage on a single z/OS image. This implementation is transparent to problem state applications and authorized programs. This is an increase from the current z/OS operating system limits of central storage configured to a single z/OS image of 128 GB.

The current maximum central storage supported is up to 512 GB on IBM System z9 servers and up to 256 GB on IBM zSeries z990 servers, an increase from the prior maximum of 128 GB. This allows programs that use large amounts of storage to avoid paging and swapping overheads, and to help enable workload growth.

2.8 IBM OMEGAMON z/OS Management Console V1.1

The IBM OMEGAMON® z/OS Management Console V1.1 (program number: 5698-A78) is a new no-charge availability monitoring product designed to help the new generation of IT professionals.

The console's advanced graphical user interface (GUI) delivers real-time health-check information provided by the IBM Health Checker for z/OS, as shown in Figure 2-5 on page 34, and configuration status information for z/OS systems and sysplex resources, as shown in .


Health Monitor Checks - RGATSKI - SYSADMIN
File
Edit
View
Help
Physical
z/OS Management Console
LPAR400J:zOS:ManagementConsole
Coupling Facility Systems Data fo
Coupling Facility Structures Data
Coupling Facility Paths Data for S
XCF Systems Data for Sysplex
XCF Paths Data for Sysplex
TESTPLEX:zOS:ManagementConsole
SP13
SP22
SYS
Physical
Exception Check Counts
RSM_HVSHARE
VSM_SQA_THRESHOLD
XCF_CLEANUP_VALUE
XCF_SIG_CONNECTIVITY
XCF_CF_STR_EXCLLIST
CNZ_CONSOLE_MSCOPE_AND_ROUTCODE
CNZ_EMCS_INACTIVE_CONSOLES
USS_AUTOMOUNT_DELAY
RRS_RSTOFFLOADSIZE
Run Counts
RSM_HVSHARE
VSM_SQA_THRESHOLD
XCF_CLEANUP_VALUE
XCF_SIG_CONNECTIVITY
XCF_CF_STR_EXCLLIST
CNZ_CONSOLE_MSCOPE_AND_ROUTCODE
CNZ_EMCS_INACTIVE_CONSOLES
USS_AUTOMOUNT_DELAY
RRS_RSTOFFLOADSIZE

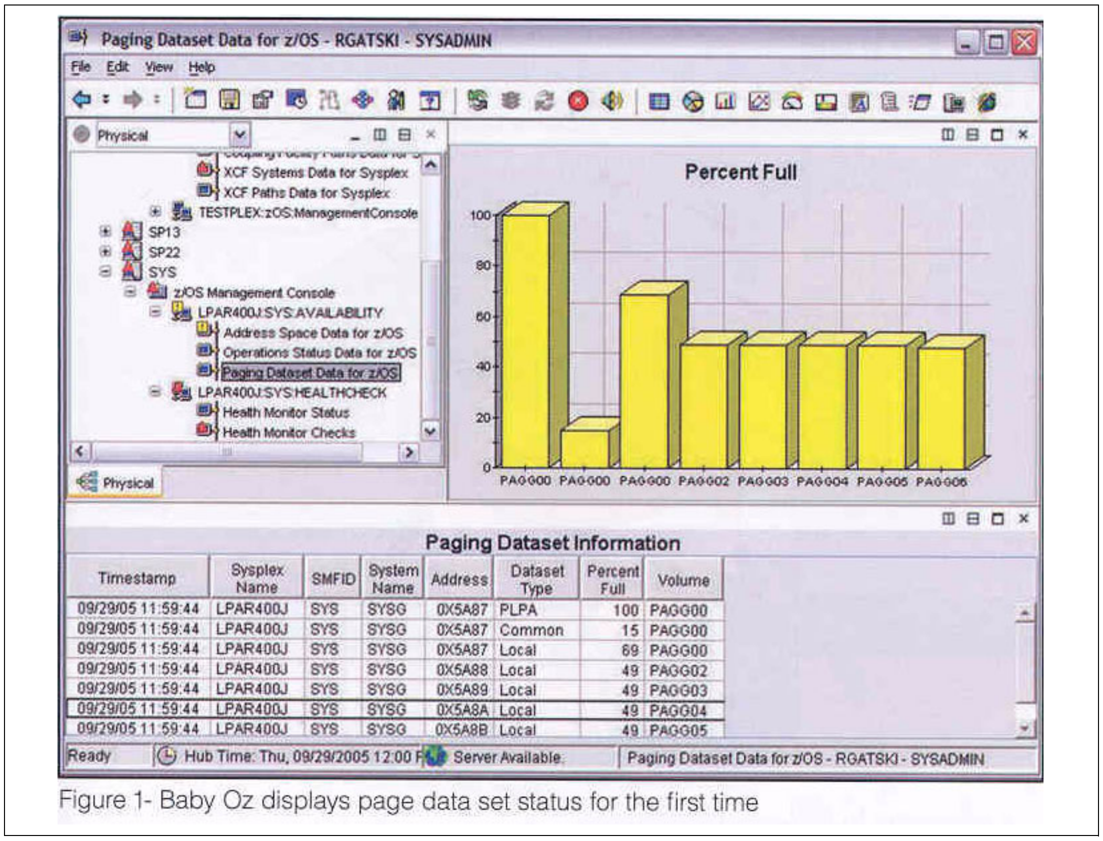
Health Checker Checks

Check Owner	Check Name	Check State	Check Status	Result	Diag1
IBMXCF	XCF_CDS_SEPARATION	ACTIVE(ENABLED)	EXCEPTION-HIGH	12	00000000
IBMRACF	RACF_SENSITIVE_RESOURCES	ACTIVE(ENABLED)	EXCEPTION-HIGH	12	00000000
IBMCNZ	CNZ_SYSCONS_MSCOPE	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000
IBMUSS	USS_AUTOMOUNT_DELAY	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000
IBMUSS	USS_FILESYS_CONFIG	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000
IBMXCF	XCF_SFM_ACTIVE	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000
IBMXCF	XCF_SIG_STR_SIZE	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000
IBMXCF	XCF_CF_STR_PREFLIST	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000
IBMXCF	XCF_FDI	ACTIVE(ENABLED)	EXCEPTION-MEDIUM	8	00000000

Ready | Hub Time: Thu, 09/29/2005 12:04 PM | Server Available. | Health Monitor Checks - RGATSKI - SYSADMIN

Figure 2- Are you ready for your next checkup? Contact Oz.

Figure 2-5 IBM Health Checker for z/OS console display


Paging Dataset Data for z/OS - RGATSKI - SYSADMIN
File Edit View Help
Physical
XCF Systems Data for Sysplex
XCF Paths Data for Sysplex
TESTPLEX:zOS:ManagementConsole
SP13
SP22
SYS
z/OS Management Console
LPAR400J:SYS:AVAILABILITY
Address Space Data for z/OS
Operations Status Data for z/OS
Paging Dataset Data for z/OS
LPAR400J:SYS:HEALTHCHECK
Health Monitor Status
Health Monitor Checks
Percent Full
100
80
60
40
20
0
PAGG00 PAGG00 PAGG00 PAGG02 PAGG03 PAGG04 PAGG05 PAGG06
Ready
Hub Time: Thu, 09/29/2005 12:00 PM
Server Available
Paging Dataset Data for z/OS - RGATSKI - SYSADMIN

Paging Dataset Information

Timestamp	Sysplex Name	SMFID	System Name	Address	Dataset Type	Percent Full	Volume
09/29/05 11:59:44	LPAR400J	SYS	SYSG	0X5A87	PLPA	100	PAGG00
09/29/05 11:59:44	LPAR400J	SYS	SYSG	0X5A87	Common	15	PAGG00
09/29/05 11:59:44	LPAR400J	SYS	SYSG	0X5A87	Local	69	PAGG00
09/29/05 11:59:44	LPAR400J	SYS	SYSG	0X5A88	Local	49	PAGG02
09/29/05 11:59:44	LPAR400J	SYS	SYSG	0X5A89	Local	49	PAGG03
09/29/05 11:59:44	LPAR400J	SYS	SYSG	0X5A8A	Local	49	PAGG04
09/29/05 11:59:44	LPAR400J	SYS	SYSG	0X5A8B	Local	49	PAGG05

Figure 1- Baby Oz displays page data set status for the first time

Figure 2-6 Page data set status display

The IBM OMEGAMON z/OS Management Console has built-in alerting and expert advice capabilities that can offer detailed contextual information on alerts and corrective actions.

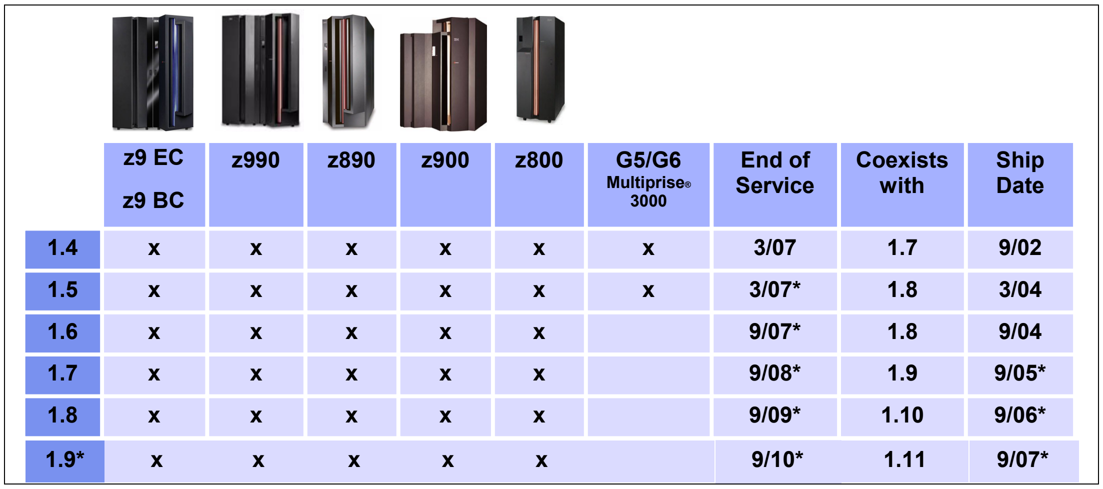
A major part of our simplification of z/OS management is built around a new user interface for the z/OS Management Console. It can be used by Information Technology workers by automating, eliminating, and simplifying many z/OS management tasks.

The first phase of the new user interface is designed to provide real-time health check information executed by the IBM Health Checker for z/OS and configuration status information for z/OS systems and sysplex resources. The new interface is intended to form the base for providing built-in automation and expert advice capabilities for z/OS management tasks. It is designed to provide detailed contextual information on alerts and corrective actions.

2.9 z/OS support summary

Figure 2-7 on page 36 summarizes much of the information about z/OS that you need to know. Server and DASD hardware support, end of service dates, coexistence, and planned availability dates are shown here for existing and planned releases through z/OS R9.

z/OS V1R8 requires an IBM System z server, as shown in Figure 2-7 on page 36.



	z9 EC z9 BC	z990	z890	z900	z800	G5/G6 Multiprise® 3000	End of Service	Coexists with	Ship Date
1.4	x	x	x	x	x	x	3/07	1.7	9/02
1.5	x	x	x	x	x	x	3/07*	1.8	3/04
1.6	x	x	x	x	x		9/07*	1.8	9/04
1.7	x	x	x	x	x		9/08*	1.9	9/05*
1.8	x	x	x	x	x		9/09*	1.10	9/06*
1.9*	x	x	x	x	x		9/10*	1.11	9/07*

Figure 2-7 z/OS support summary

2.10 Migration to z/OS V1R8

Migration is the first of two stages in upgrading to a new release of z/OS. The two stages are:

- Stage 1: Migration. During this stage you make your new system functionally compatibility with the previous system. After a successful migration, the applications and resources on the new system function the same way (or similar to the way) they did on the old system or, if that is not possible, in a way that accommodates the new system differences so that existing workloads can continue to run. Migration does not include exploitation of new functions except for new functions that are now required.
- Stage 2: Exploitation. During this stage you do whatever customizing and programming are necessary to take advantage of (exploit) the enhancements available in the new release.

z/OS Migration Version 1 Release 8, GA22-7499 describes how to migrate to z/OS Version 1 Release 8 (V1R8) from the following releases:

- z/OS V1R7
- z/OS V1R6
- z/OS V1R5

2.10.1 z/OS driving system requirements

The driver system is the system image (hardware and software) that you use to install the target system. The target system is the system software libraries and other data sets that you are installing. You log on to the driving system and run jobs there to create or update the target system. Once the target system is built, it can be IPLed on the same hardware (same LPAR or same processor) or different hardware than that used for the driving system.

If your driving system will share resources with your target system after the target system has been IPLed, be sure to install applicable coexistence service on the driving system before you IPL the target system. If you don't install the coexistence service, you will probably experience problems due to incompatible data structures (such as incompatible data sets, VTOCs, catalog records, GRS tokens, or APPC bind mappings).

Using a current z/OS system

A coexistence release can be used as a driver system with the PTFs, as follows:

z/OS V1R5 For servicing: z/OS V1R8 level of the Program Management Binder, HLASM (V1R5), and z/OS V1R8 SMP/E (SMP/E V3R4). PTFs required are:

- DFSMSdfp™: UA05520, UA14837
- z/OS UNIX: UQ79726, UQ85020

z/OS V1R6 For servicing: z/OS V1R8 level of the Program Management Binder, HLASM (V1R5) and z/OS V1R8 SMP/E (SMP/E V3R4). PTFs required are:

- DFSMSdfp: UA14838

z/OS V1R7 For servicing: z/OS V1R8 level of the Program Management Binder, HLASM (V1R5), and z/OS V1R8 SMP/E (SMP/E V3R4).

Using a Customized Offerings Driver (5665-M12)

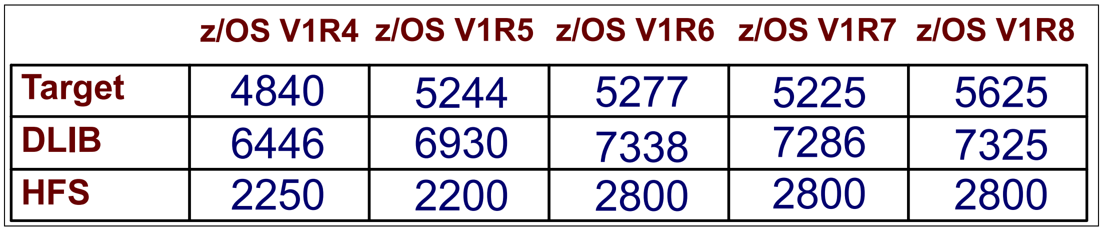
The Customized Offerings Driver V2.1 (5665-M12) is an entitled driving system you can use if:

- You don't have an existing system to use as a driving system, or
- Your existing system does not meet driving system requirements and you don't want to upgrade it to meet those requirements.

DASD space requirements

If you are migrating to z/OS V1R8 from z/OS V1R5 or you will have a different product set than your previous release, you will see increased need for DASD. How much more depends on what levels of products you are running. Keep in mind the DASD required for your z/OS system includes (per the z/OS Policy). That is, it includes ALL elements, ALL features that support dynamic enablement, regardless of your order, and ALL unpriced features that you ordered. This storage is in addition to the storage required by other products you might have installed. All sizes include 15% freespace to accommodate the installation of maintenance.

The total storage required for z/OS data sets is listed f or the last five releases in Figure 2-8 on page 37.



	z/OS V1R4	z/OS V1R5	z/OS V1R6	z/OS V1R7	z/OS V1R8
Target	4840	5244	5277	5225	5625
DLIB	6446	6930	7338	7286	7325
HFS	2250	2200	2800	2800	2800

Figure 2-8 DASD space for z/OS data sets

2.10.2 Coexistence support for z/OS V1R8

Coexistence occurs when two or more systems at different software levels share resources. The resources could be shared at the same time by different systems in a multisystem configuration, or they could be shared over a period of time by the same system in a single-system configuration.

z/OS systems can coexist with specific prior releases. This is important because it gives you flexibility to migrate systems in a multisystem configuration using rolling IPLs rather than requiring a systems-wide IPL. The way in which you make it possible for earlier-level systems to coexist with z/OS is to install coexistence service (PTFs) on the earlier-level systems. Figure 2-9 on page 38 shows the coexistence releases for z/OS V1R8 and for 2007 with z/OS V1R9.

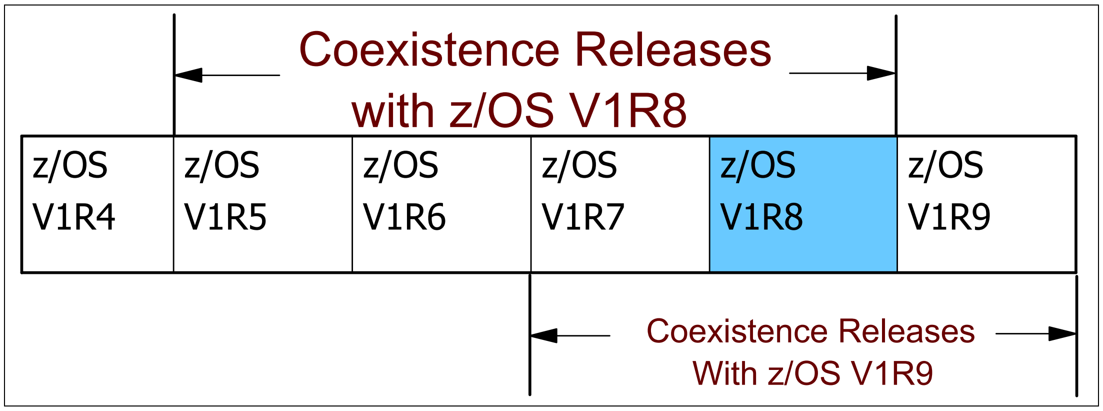

Coexistence Releases with z/OS V1R8
z/OS V1R4
z/OS V1R5
z/OS V1R6
z/OS V1R7
z/OS V1R8
z/OS V1R9
Coexistence Releases With z/OS V1R9

Figure 2-9 Coexistence releases

Coexistence PTFs

You should complete the migration of all earlier-level coexisting systems as soon as you can. Keep in mind that the objective of coexistence PTFs is to allow existing functions to continue to be used on the earlier-level systems when run in a mixed environment that contains later-level systems. Coexistence PTFs are not aimed at allowing new functions provided in later releases to work on earlier-level systems.

Coexistence releases for other products

Examples of coexistence are two different JES releases sharing a spool, two different service levels of DFSMSdfp sharing catalogs, multiple levels of SMP/E processing SYSMODs packaged to exploit the latest enhancements, or an older level of the system using the updated system control files of a newer level (even if new function has been exploited in the newer level).

2.10.3 Coexistence releases for JES

For JES2 and JES3, the way in which four consecutive releases is determined is different than for the rest of the operating system. If a JES2 or JES3 release is functionally equivalent to its predecessor (that is, its FMID is the same), then the release is considered to be the same JES release. Thus, z/OS V1R8 JES2 and JES3 are coexistence, fallback, and migration supported with the following JES releases: V1R8, V1R7, and V1R6-V1R5 (both are functionally equivalent).

As of z/OS V1R2, compliance to the coexistence, fallback, and migration policy for JES2 and JES3 is enforced. A migration to a JES2 or JES3 release level that is not supported by the policy results in the following:

- **For JES2:** If the JES2 release level for a system that is initializing is not compatible with the other active systems in the JES2 MAS, message HASP710 is issued and the JES2 address space for the initializing system is terminated.

z/OS Releases	JES2 z/OS R.2 HJE7705	JES2 z/OS R.4 HJE7707	JES2 z/OS R.5 HJE7708	JES2 z/OS R.7 HJE7720	JES2 z/OS R.8 HJE7730
z/OS R1					
z/OS R2	X				
z/OS R3	X				
z/OS R4	X	X			
z/OS R5	X	X	X		
z/OS R6	X	X	X		
z/OS R7		X	X	X	
z/OS R8			X	X	X

Figure 2-10 JES2 coexistence releases

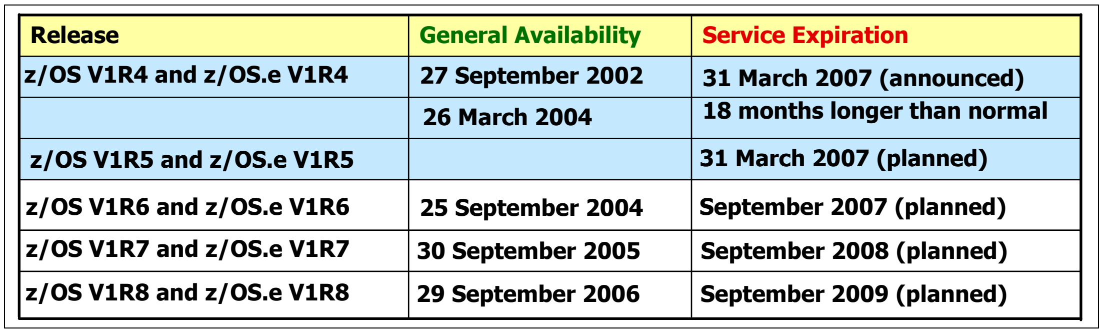
- **For JES3:** If the JES3 release level for a local is not compatible with the global in a JES3 multisystem complex, message IAT2640 is issued and the JES3 local is not allowed to connect to the global.

	JES3 z/OS V1 R4 HJS7707	JES3 z/OS V1 R5 HJS7708	JES3 z/OS V1 R6 HJS7708	JES3 z/OS V1 R7 HJS7720	JES3 z/OS V1R8 HJS7730
z/OS V1 R4	🟩				
z/OS V1 R5	🟩	🟩			
z/OS V1 R6	🟩	🟩	🟩		
z/OS V1 R7	🟩	🟩	🟩	🟩	
z/OS V1 R8		🟩	🟩	🟩	🟩

Figure 2-11 JES3 coexistence releases

Service policy

IBM's current policy is to provide maintenance (service) for each release of z/OS for three years following their general availability (GA) date. However, service on the last release of a version might be extended beyond the intended three-year period. Prior to withdrawing service for any version or release of z/OS, IBM intends to provide at least 12 months notice. The service policy for z/OS also applies to any enhancements (including but not limited to web deliverables), such as the z/OS V1R4 enhancements that were provided to support the z990 server.



Release	General Availability	Service Expiration
z/OS V1R4 and z/OS.e V1R4	27 September 2002	31 March 2007 (announced)
	26 March 2004	18 months longer than normal
z/OS V1R5 and z/OS.e V1R5		31 March 2007 (planned)
z/OS V1R6 and z/OS.e V1R6	25 September 2004	September 2007 (planned)
z/OS V1R7 and z/OS.e V1R7	30 September 2005	September 2008 (planned)
z/OS V1R8 and z/OS.e V1R8	29 September 2006	September 2009 (planned)

Figure 2-12 Service policy for releases still supported

2.10.4 Web deliverables

Sometimes enhancements are provided as Web deliverables, and not integrated in your ServerPac, CBPDO, or SystemPac deliverable. For example, some of the ICSF enhancements are available this way. z/OS Web deliverables are available from:

```
http://www.ibm.com/eserver/zseries/zos/downloads/
```

They are packaged as two files that you download:

- A readme file, which contains a sample job to uncompress the second file, transform it into a format that SMP/E can process, and invoke SMP/E to RECEIVE the file. This file must be downloaded as text.
- A pax.z file, which contains an archive (compressed copy) of the FMIDs to be installed. This file needs to be downloaded to a workstation and then uploaded to a host as a binary file.

Web downloads

For Web downloads, you must perform the following tasks:

- Allocate a R/W HFS directory on the z/OS driving system where the package will be staged (optional).
- Download both parts of the package from the download zone: http://www.ibm.com/eserver/zseries/zos/downloads/
- Run the sample job provided in the README.TXT file. The job will invoke the GIMUNZIP service routine to extract the original data from the packages.
- Obtain and install service for the target system. Service is not included in Web deliverables. You can obtain service for Web deliverables through your regular preventive service deliverables that you use for z/OS.
- Install (SMP/E APPLY) the downloaded FMIDs.

2.10.5 HCD and HCM support for z/OS V1R8

Anew function SPE with APAR OA14334, contains the HCD support for z/OS V1R8 HCM for the following HCM functions:

- Copy processor, CSS, partition, operationg system, EDT and esoteric
- Automatic activity logging
- Performance data display of RMF Monitor III data
- Export/import IODF

- Compare IODF via HCM
- OS attribute change of a group of devices

New HCD functions

Following are the new HCD profile options:

CHANGE_LOG = YES / NO	Indicates whether the IODF changes are logged in a change log data set if activity logging is enabled for the IODF. The change log data set is a VSAM data set with the name of <iodf-name>.CHLOG. The contents of the change log data set can be written in the HCD trace data set via HCD command TRACE ON,ID=CLOG, LEVEL=8
CHLOG_VOL = <volume>	Directs the change log data set to a specific non-SMS managed volume

HCD 1.7 enhancements

The following new HCD 1.7 enhancements are:

- Automatic activity logging for an IODF that has activity logging enabled and that has set the HCD profile option CHANGE_LOG = YES.
- A warning message is given if a control unit has different logical CU addresses (CUADD values) defined for different processors.

```
CBDA443I CONTROL UNIT xxxx SPECIFIES DIFFERENT LOGICAL ADDRESSES (CUADD
VALUES) FOR PROCESSORS mmmmmmmm AND nnnnnnnn.
```

- The Processor Summary Report now shows the processor support level for each processor.
- The IOCDS Report shows the same POR information as dialog option 2.11.
- An issued ACTIVATE command is written to the HCD message log and to the system log.

3

Console restructure

This chapter describes the console infrastructure enhancements in z/OS V1R8. This is a continuation of work started in the console restructure delivered in z/OS V1R4 and continued with z/OS V1R7.

We describe the following topics:

- Console restructure
- Remove of master console
- Console switch elimination
- Hardcopy switch elimination
- 1-Byte Console-ID elimination completed
- Enhanced CONVCON and new CnzConv interfaces
- MSGRT elimination
- Changes to operator commands
  - Removed commands
  - Changed commands
- New message routing attributes - INTIDS and UNKNIDS
- Migration and coexistence considerations
  - Modify programs that reference unsupported console functions
  - z/OS V1R8 console restructure migration checklist

3.1 Multisystem consoles - MVS/ESA Version 4

The introduction of a sysplex into the MVS environment provided a simpler and more flexible way to operate consoles in a multisystem environment. Many changes were introduced into multiple console support (MCS) to support the sysplex environment. These changes began with MVS/ESA™ Version 4, as shown in Figure 3-1, and have continued with each new operating system release.

Using consoles

In a sysplex, MCS, SMCS, and EMCS consoles can:

- Be attached to any system
- Receive messages from any system in the sysplex
- Route commands to any system in the sysplex

Therefore, new considerations needed to be made when defining MCS consoles in this environment, such as:

- There is no requirement that each system have consoles attached.
- The 99 console limit for the sysplex can be extended with the use of extended MCS consoles (EMCS). This added greater flexibility when attaching consoles.
- A sysplex, which can consist of up to 32 systems, can be operated from a single console.
- Multiple consoles can have master command authority.

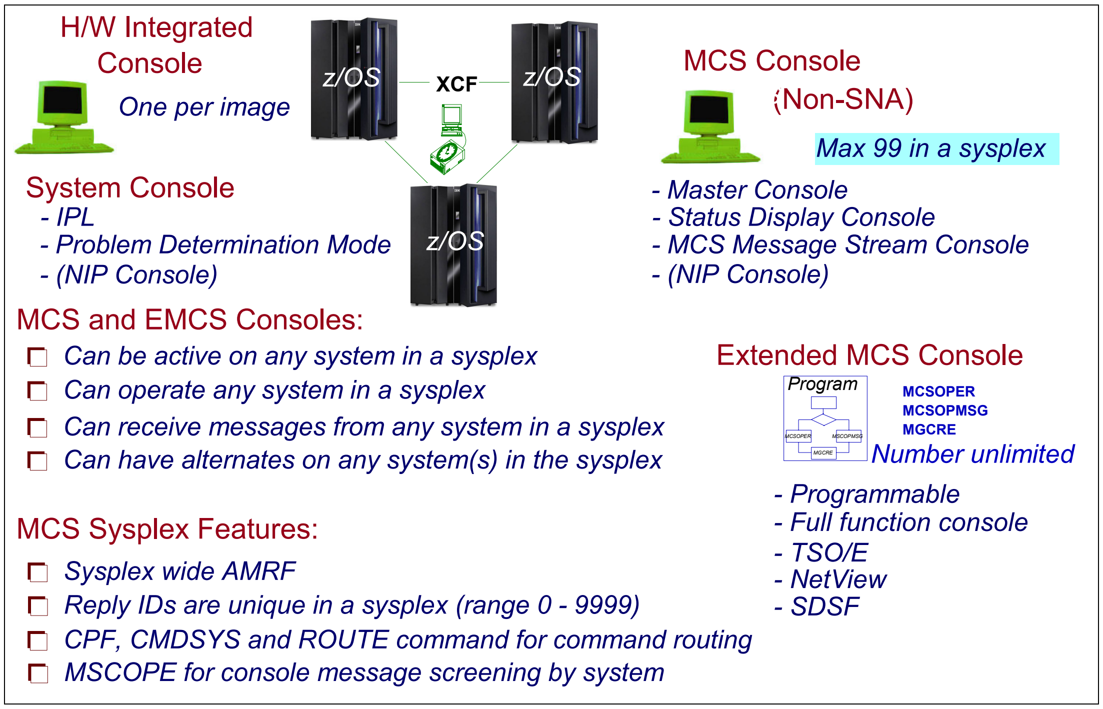

H/W Integrated Console
One per image
z/OS
XCF
z/OS
z/OS
MCS Console
(Non-SNA)
Max 99 in a sysplex
System Console
- IPL
- Problem Determination Mode
- (NIP Console)
- Master Console
- Status Display Console
- MCS Message Stream Console
- (NIP Console)
MCS and EMCS Consoles:
Can be active on any system in a sysplex
Can operate any system in a sysplex
Can receive messages from any system in a sysplex
Can have alternates on any system(s) in the sysplex
Extended MCS Console
Program
MCSOPER
MCSOPMSG
MGCRE
Number unlimited
- Programmable
- Full function console
- TSO/E
- NetView
- SDSF
MCS Sysplex Features:
Sysplex wide AMRF
Reply IDs are unique in a sysplex (range 0 - 9999)
CPF, CMDSYS and ROUTE command for command routing
MSCOPE for console message screening by system

Figure 3-1 Multisystem consoles in a sysplex began in MVS/ESA

3.1.1 Consoles in a sysplex with z/OS V1R8

z/OS V1R8 now supports MCS, SMCS, and EMCS consoles, as follows:

MCS	MCS consoles are devices that are locally attached to an MVS system and provide the basic communication between operators and MVS. (MCS consoles are attached to control unit devices that do not support systems network architecture (SNA) protocols.)
SMCS	SMCS consoles are devices that do not have to be locally attached to an MVS system and provide the basic communication between operators and MVS. SMCS consoles use z/OS Communications Server to provide communication between operators and MVS instead of using direct I/O to the console device. SMCS consoles were available as of z/OS V1R1.
EMCS	EMCS (Extended MCS) consoles are programmable consoles defined and activated through an intended programming interface.
subsystem	Subsystem allocatable consoles are defined in CONSOLxx and obtained and released using the IEAVG700 interface. Programs invoke IEAVG700 passing in the SCSR (subsystem console service routine) parmlist, which is mapped by IEZVG100. IBM highly recommends the use of extended MCS (EMCS) consoles rather than subsystem allocatable consoles.

During initialization of an MVS system, the operator uses the system console or hardware management console (HMC), which is connected to the hardware support element. From the system console, the operator initializes the system control program during the nucleus initialization program (NIP) stage.

Note: You can define up to 99 consoles including any subsystem allocatable consoles for an MVS system. In a sysplex, the limit is 99 consoles for the sysplex. You can exceed this number in a system or sysplex by using extended MCS (EMCS) consoles.

Sysplex environment grows

As sysplex environments grew in size and many products increased their message output, there became a need to restructure the console environment because some installations began to experience the following problems:

- A runaway application could take down a system or the entire sysplex.
- Large systems could overwhelm smaller systems.
- All queuing decisions were made from a single task.
- The console task was prone to backups and storage overloads.
- Traffic to a particular console was sometimes excessive due to parmlib definitions.
- Un-ended multi-line WTOs caused problems for the console task.

3.2 Console restructure

The console restructure is the result of a study to find solutions to a number of severe sysplex console support problems, such as:

- Message production and consumption problems
- Synchronization of console state information
- Sysplex limit of 99 consoles in a sysplex (MCS, SMCS, and subsystem consoles)

3.2.1 Stages of the console restructure

The console restructure began with z/OS V1R4 as a feature added on that release and was included in the base with z/OS V1R5 and introduced as the first part of the restructure.

Message production and consumption causes severe problems in a single system or a sysplex environment. Messages are sent from one system to another without regard to whether the receiving system is able to consume the messages. The receiving system has no choice except to attempt to consume the messages that it was forced to receive. If the receiving system is unable to “consume” the messages by displaying and logging them, it can suffer buffer shortages and out-of-storage conditions that can bring down the receiving system. Dissimilar systems can exacerbate the message delivery problem. A large, fast system can inundate a small, slow system with more message traffic than it can handle and still do useful work. The aim was to minimize the possibility of outages due to exhaustion of system resources used for messaging. So, the main problems were:

- Message delivery
- Buffer shortages

Enhancements in z/OS V1R4 and z/OS V1R5

With the console FMID feature to z/OS V1R4 and with this code in the base z/OS V1R5 system, the console restructure was enhanced to solve the following problems:

- Eliminate outages due to a flood of WTOs and DOMs while also introducing a new message cache dataspace, as shown in Figure 3-2 on page 47.
- During message processing, dependence on a single task was removed by moving some processing to other address spaces.
- Queuing to the SYSLOG and OPERLOG was separated from queuing to the consoles.
- Queuing to the EMCS consoles was performed in parallel to queuing to the MCS consoles.
- Multiple-line WTOs were not queued for delivery until all message lines were received.
- For assist with the 1-byte console ID removal process, the 1-byte console ID tracker facility was introduced.

All these and several other enhancements to the system provided improved message processing to avoid system outages related to buffer exhaustion and also removed possible performance impacts on the system caused by the message flow process.

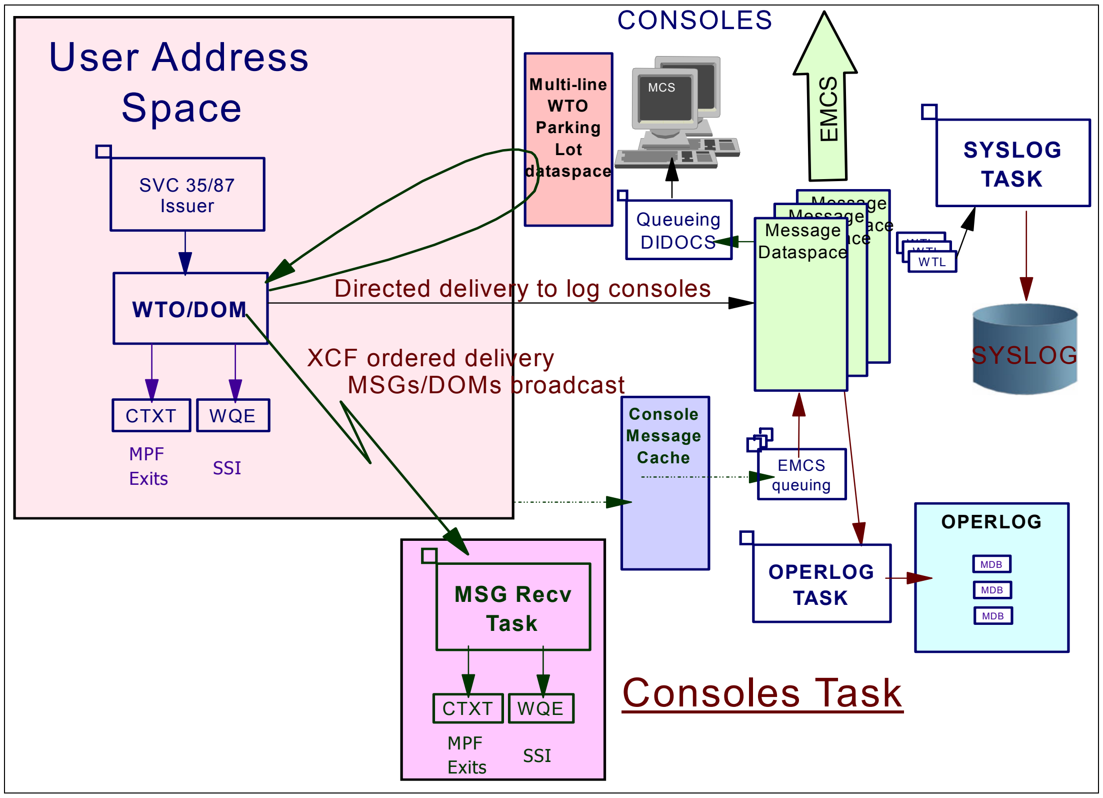


Figure 3-2 Console restructure after z/OS V1R4 FMID

Enhancements in z/OS V1R7

Console restructure enhancements continued with z/OS V1R7. When a system joins the sysplex, information about EMCS consoles needs to be exchanged. Since the number of EMCS consoles grows over time, the time to IPL a system into a sysplex keeps growing. In z/OS V1R7, the following improvements were implemented:

- You can now delete inactive EMCS consoles. This reduces the system's join time in a sysplex.
- The message monitoring process has been enhanced:
  - With the new MONITOR keyword for the SETCON command, you can monitor messages to be produced without requiring that the messages be sent to a console.
  - MONITOR message production can now be enabled independent of routing attributes.

3.3 z/OS V1R8 console restructure enhancements

Console restructuring is continued with the following enhancements:

- Removal of the sysplex master console

  New message routing attributes

- Console switch elimination
- Hardcopy switch elimination
- 1-byte console ID elimination completed
- Enhanced CONVCON and new CNZCONV interface
- Changes to operator commands

3.4 Removal of the master console

Before z/OS V1R8, in a sysplex, the first active console with master authority in the first system that joins the sysplex becomes the master console. You can define AUTH(MASTER) for other consoles in that system or for other systems that subsequently join the sysplex. These consoles have master authority, but there can be only one master console in the sysplex. Operators could assign the master authority of a console with the following command:

```
VARY CN(name),AUTH=MASTER
```

3.4.1 z/OS V1R8 master console changes

The master console is a full-capability console from which the operator has the authority to enter any MVS commands. As a result, the master console is often the focal point of MVS operations.

In z/OS V1R8, to remove a single point of failure for improved reliability, availability, and serviceability (RAS), the master console role is removed. Consoles can still have master authority but assigning a console as the master console is no longer done.

Since a master console no longer exists, z/OS V1R8 no longer detects if any MCS or SMCS consoles are active as the master console. The concept of no master console and no console conditions is eliminated.

3.4.2 New message routing attributes - INTIDS and UNKNIDS

Messages that were issued to console ID zero were sent to the master console in previous releases. Since the master console no longer exists in z/OS V1R8, a predictable destination for these messages is necessary.

In z/OS V1R8, there are two new console routing attributes. They are INTIDS for internal IDs, and UNKNIDS for unknown IDs. Using these two new console routing attributes, you can provide console destinations for messages that would previously have gone to the master console, as follows:

INTIDS You can use the INTIDS attribute on the CONSOLE statement to control whether the console is to receive messages that are directed to console ID zero. Those messages are often the responses to internally issued commands. To enable a console to receive such messages, specify INTIDS(Y) for this console in the CONSOLE statement. You can also use VARY CN command to change this console attribute. If you do not specify this attribute, the default is N.

Note: Be aware that WTORs directed to console ID zero are delivered to consoles that specify INTIDS(Y). This means that the operator at that console is able to reply to the WTOR.

UNKNIDS You can use the UNKNIDS attribute on the CONSOLE statement to control whether the console is to receive messages that are directed to unknown console ids, such as 1-byte console ids which are no longer supported. To enable a console to receive such messages, specify UNKNIDS(Y) for this console on the CONSOLE statement. You can also use VARY CN command to change this console attribute. If you do not specify this attribute, the default is N. The UNKNIDS queuing attribute allows messages issued with 1-byte console IDs to be queued to some console. Messages marked with the UNKNIDS queuing attribute are queued to any console that requests UNKNIDS messages. This may or may not be the console that would have corresponded to the 1-byte console ID.

Since the INTIDS and UNKNIDS attribute values are by default N, if no console is specified with a Y in the console attributes, messages are not sent to any console. They will be hardcopied, unless the message or an exit indicates otherwise.

Coexistence APAR OA10632

Use the new console routing attributes, INTIDS and UNKNIDS, to provide console destinations for messages that previously went to the master console. Since the master console no longer exists, a predictable destination for these messages is necessary.

In a mixed sysplex, the coexistence APAR OA10632 has partially implemented these attributes on the lower levels of z/OS. It supports sending messages to consoles on z/OS V1R8 systems, receiving INTIDS and UNKNIDS message routing from lower-level systems.

3.4.3 Console ID zero

Messages that are sent to console ID zero are usually the ones created by “internally issued” commands, which are issued by programs that have no associated console. When programs issue commands via MGCR or MGCRE, they specify a console ID as zero. In addition, command responses to instream JCL commands are also sent to console ID zero. All those messages were routed to the sysplex master console in the past. In z/OS V1R8, they are now sent to INITDS receivers.

3.4.4 Additional master console changes

The removal of the sysplex master console has caused additional changes to many parts of the console support.

Removed console commands

The VARY console,MSTCONS command—or actions such as using the external interrupt key to cause a master console switch—is eliminated. When you use the VARY xxxx,MSTCONS command, an invalid command message is issued. Also, the D CONSOLES,MCONLY command is removed. The commands and messages are shown in Figure 3-3.

```
VARY E201,MSTCONS
IEE309I VARY     UNIDENTIFIABLE KEYWORD
D CONSOLES,MCONLY
IEE535I DISPLAY  INVALID PARAMETER
```

Figure 3-3 Commands removed in z/OS V1R8

Route codes 1 and 2

In pre-z/OS V1R8 versions, the master console was always guaranteed to receive routing codes 1 and 2. Now the system does not force any console to receive routing codes 1 and 2. Since there is no master console, if you want to have a console receive routing codes 1 and 2, you should specify it in the CONSOLxx parmlib member.

Master authority

In z/OS V1R8, since there is no master console restriction, no MCS or SMCS console is forced to have master authority. Therefore, if console master authority is desired for an MCS and or SMCS console type, you should specify this in the CONSOLxx parmlib member.

External interrupt key

The external interrupt key was supported to trigger a master console switch. This is not supported in z/OS V1R8.

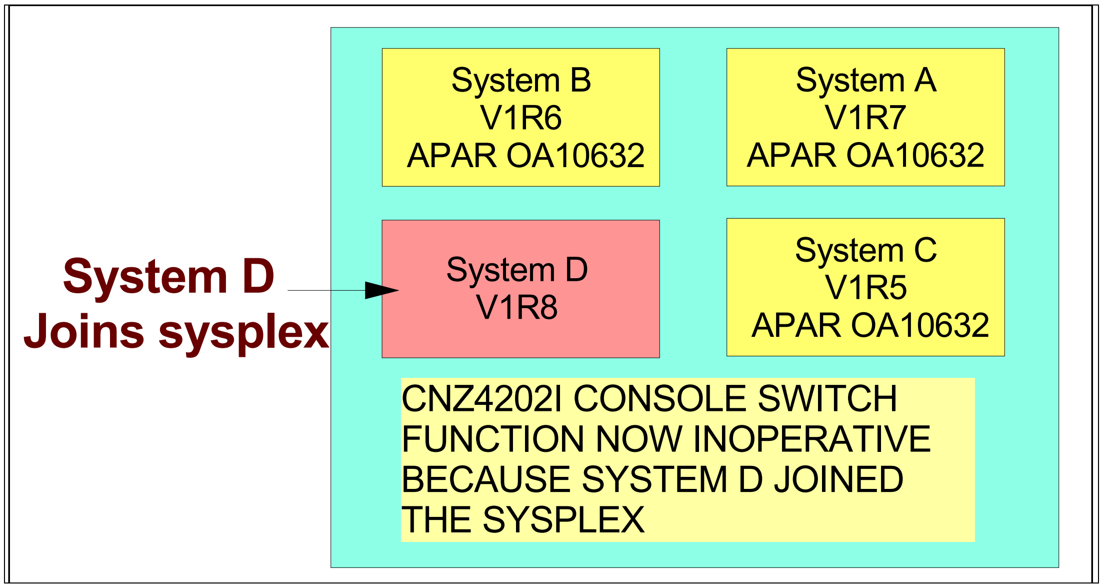
3.5 Console switch elimination

In pre-z/OS V1R8, the concept of having an alternate for a console is supported. When a console fails, its function is switched to the alternate, which was chosen from an alternate group definition in the CNGRPxx parmlib member. For most installations it was somewhat complex to set up switch definitions for all of their consoles and to choose alternates.

In z/OS V1R8, elimination of the sysplex master console reduces the importance of a console switch. Therefore, the console switch function and defining alternate console groups are eliminated.

In z/OS V1R8, if a console fails, the console is just deactivated. If it is required that a console's function should always be available, multiple consoles can be defined with the same attributes and activated to minimize the loss of a console.

When a z/OS V1R8 system joins a sysplex consisting of z/OS V1R7 (or earlier) systems, as shown in Figure 3-4, support for console switching is removed for all systems in the sysplex. The message that is issued is shown in Figure 3-4.


System B V1R6 APAR OA10632
System A V1R7 APAR OA10632
System D Joins sysplex
System D V1R8
System C V1R5 APAR OA10632

```
CNZ4202I CONSOLE SWITCH
FUNCTION NOW INOPERATIVE
BECAUSE SYSTEM D JOINED
THE SYSPLEX
```

Figure 3-4 z/OS V1R8 system joining a sysplex

When the last z/OS V1R8 system leaves the sysplex, as shown in Figure 3-5 on page 51,console switching becomes operative. The message that is issued is shown in Figure 3-5 on page 51.

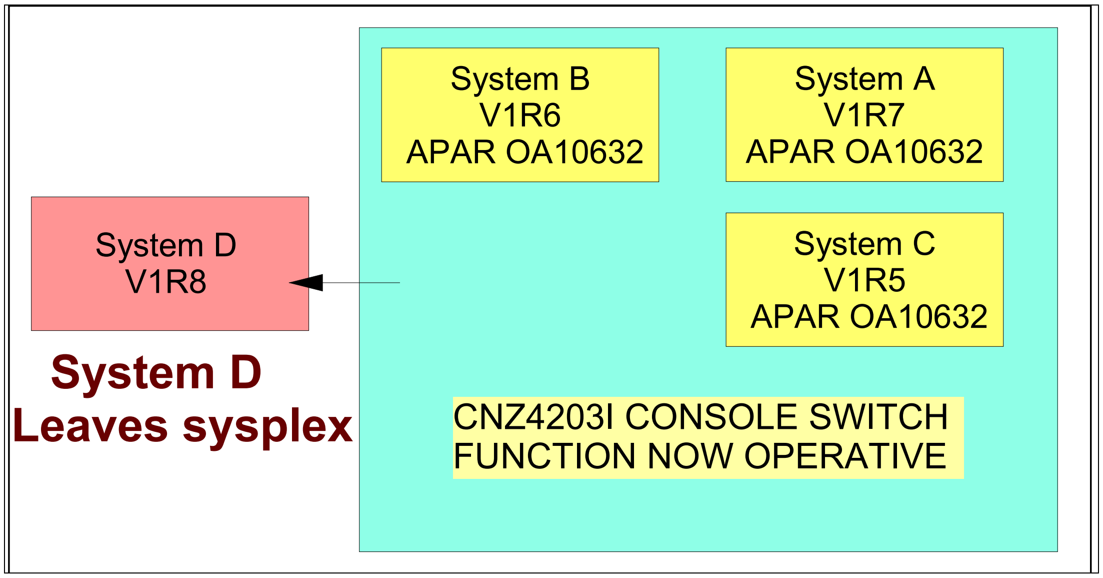


Figure 3-5 z/OS V1R8 system leaving the sysplex

3.5.1 CONSOLxx parmlib member changes

The ALTGRP keyword in the CONSOLxx parmlib member is removed. You can, however, keep the ALTGRP keyword if you plan to share the CONSOLxx parmlib member with z/OS levels earlier than V1R8. z/OS V1R8 issues warning messages, IEA196I, about unsupported keywords, but the system continues with no errors, as shown in Figure 3-6.

```
IEE252I MEMBER CONSOL00 FOUND IN SYS1.PARMLIB
IEA196I CONSOL00 CNE201: UNRECOGNIZED KEYWORD ALTGRP(MAS IGNORED.
IEA196I CONSOL00 CN9600: UNRECOGNIZED KEYWORD ALTGRP(MAS IGNORED.
IEA196I CONSOL00 CN9620: UNRECOGNIZED KEYWORD ALTGRP(MAS IGNORED.
IEA196I CONSOL00 CN9640: UNRECOGNIZED KEYWORD ALTGRP(MAS IGNORED.
IEA196I CONSOL00 CASTOR: UNRECOGNIZED KEYWORD ALTGRP(MAS IGNORED.
IEA196I CONSOL00 POLLUX: UNRECOGNIZED KEYWORD ALTGRP(MAS IGNORED.
IEA196I CONSOL00 SMCSCON1: UNRECOGNIZED KEYWORD ALTGRP(SMC IGNORED.
IEA196I CONSOL00 SMCSCON2: UNRECOGNIZED KEYWORD ALTGRP(SMC IGNORED.
IEA196I CONSOL00 SMCSCON3: UNRECOGNIZED KEYWORD ALTGRP(SMC IGNORED.
IEA196I CONSOL00 SMCSCON4: UNRECOGNIZED KEYWORD ALTGRP(SMC IGNORED.
IEE252I MEMBER CNGRP00 FOUND IN SYS1.PARMLIB
IEE712I SET CNGRP PROCESSING COMPLETE
```

Figure 3-6 Message IEA196I indicating ALTGRP is not recognized

Removed console commands

In z/OS V1R8, with the removal of console switching, the following commands are removed:

SWITCH CN

VARY CN(xxxx),ALTGRP

With z/OS V1R8, in the CNGRPxx parmlib member, groups are not used for alternate console selection but they are still used for SYNCHDEST and the system console AUTOACT keyword.

3.6 System console (SYSCONS)

The system console function is provided as part of the hardware management console (HMC). An operator can use the system console to initialize MVS and other system software and during recovery situations when other consoles are unavailable.

You can define the system console to MVS in the CONSOLxx parmlib member. You can define the system console on a CONSOLE statement by specifying SYSCONS for DEVNUM. You can specify the CONSOLE keywords NAME, ROUTCODE, LEVEL, MONITOR, MSCOPE, AUTOACT, and CMDSYS. The system ignores other CONSOLxx keywords. The system console always has master authority.

With the removal of the master console, the system console (SYSCONS) takes on more significance. In z/OS V1R8, since there is no master console, the system console (SYSCONS) has been enhanced to ensure it is always available. In z/OS V1R8, the system console (SYSCONS) always has master authority and the authority cannot be changed. The SYSCONS will always be treated as being LOGON(OPTIONAL) even if every console must be LOGON(REQUIRED) to prevent an installation from not defining the correct security product profiles, which prevents the system console from being usable.

During system initialization, the system accepts commands only from a master authority console (or the system console) until a security product is fully initialized and able to process LOGON requests. Allowing commands from a master authority console before a security product is fully initialized allows an operator to intervene if required to complete the security product initialization.

With z/OS V1R8, since there is no master console, the system console (SYSCONS) has been enhanced to ensure it is always available. The system console now always has master authority and the authority can not be changed. The system console is now always treated as being LOGON(OPTIONAL) even if every console must be LOGON(REQUIRED). This is to prevent an installation from not defining the correct security product profiles and thereby preventing the system console from being usable.

3.6.1 Considerations using consoles to display synchronous messages

Use the SYNCHDEST keyword on the DEFAULT statement of CONSOLxx parmlib member, to handle the display of synchronous messages. SYNCHDEST specifies the name of the console group whose members can receive a synchronous message. MVS searches for an eligible console based on the order of the console members specified in the group. You can specify valid MCS console names as members of the group. You can also specify *SYSCON*, the system console. To receive synchronous messages, the console must be attached to the system that issues the message.

If you do not specify a console group on SYNCHDEST or none of the consoles on SYNCHDEST are active, the system that issues the message tries to display the message on the system console.

The SYNCHDEST console group is an ordered list of consoles where MVS is to attempt to display synchronous messages. The system console can be specified in the list. If an MCS console in the list is not attached to the system where the message is issued, it is skipped. So, the same SYNCHDEST group can be used for all systems.

Note: The master console (*MSTCON*) could be specified in the SYNCHDEST group as a destination for synchronous messages.Since the master console function is removed in z/OS V1R8, the system console remains the synchronous message (SYNCHDEST) destination of last resort. You can define a full-capacity MCS console, or the system console, as members of a console group in CNGRPxx parmlib member to receive synchronous messages.

AUTOACT keyword

The AUTOACT keyword for the system console specifies a console group. The consoles in this group can replace the system console. The AUTOACT support is especially useful if you have no MCS consoles (for example, all of them are SMCS consoles). With this support, the system console can be used during NIP processing, and activated automatically at the end of NIP.

Attention: To ensure that you have the ability to operate your installation at all times, you should define multiple consoles with master authority. Note that the system console is forced to have master authority.

3.7 Hardcopy switch elimination

Pre-z/OS V1R8 systems have a hardcopy switch function such that if the OPERLOG is active as the hardcopy medium and fails, z/OS switches the hardcopy function to the SYSLOG. If the SYSLOG is the hardcopy medium and fails, the hardcopy medium switches to the OPERLOG. In both cases, if the other log type is not available, the hardcopy function is suspended.

With z/OS V1R8, the hardcopy switch support is eliminated. Therefore, if only one hardcopy medium is defined and if it fails, the system suspends hardcopy processing. To reduce the likelihood of losing hardcopy, you should define both SYSLOG and OPERLOG as hardcopy medium via the DEVNUM statement in the CONSOLxx parmlib member. You can get information about which type of log is hardcopy using the D C,HARDCOPY command, as shown in Figure 3-7.

```
D C,HARDCOPY
IEE889I 11.30.43 CONSOLE DISPLAY 279
MSG: CURR=0    LIM=1500 RPLY:CURR=0    LIM=999  SYS=SC75
 CONSOLE           ID  --------------- SPECIFICATIONS------
 SYSLOG                COND=H      AUTH=CMDS NBUF=N/A
                       ROUTCDE=ALL
 OPERLOG               COND=H      AUTH=CMDS NBUF=N/A
                       ROUTCDE=ALL
LOG BUFFERS IN USE:        0  LOG BUFFER LIMIT:     6000
```

Figure 3-7 Command to display hardcopy medium defined

3.8 One-byte console-ID elimination completed

Console IDs are 4 bytes in length since MVS/ESA. The process to eliminate support for 1-byte console ID began with z/OS V1R4 and continued in z/OS V1R5 and V1R7. With z/OS V1R8, the removal is complete.

In z/OS V1R4 and z/OS V1R5

Code was added to track its usage. 1-byte console IDs continued to be supported.

Note: For complete details on the 1-byte Console ID Tracking Facility, see APAR II13752.

In z/OS V1R7

- The macros, commands and WTO that support 1-byte console IDs and migration IDs were removed. The WTO and WTOR macros no longer support the MCSFLAG (REG0 or QREG0). A severity 12 MNOTE is issued if the field is used.
- The MCSOPER macro has the MIGID keyword removed. A severity 1 MNOTE indicates that the MIGID is unsupported.

  Existing code would continue to work and is tracked by the tracker. But once the code is recompiled, the compilation would fail with MNOTEs.
- All commands supporting L=cc and L=name support only L=name. The commands that use a console ID receive message IEE274I. These commands are:
  - D C,CN=nn
  - D PFK,CN=nn
  - D R,CN=nn
  - RESET CN(nn)
  - SWITCH CN=nn

    This command is totally removed in z/OS V1R8.
  - VARY CN(nn)

See Figure 3-8 as an example.

```
D C,CN=01
IEE274I DISPLAY  CONSOLE 01        NOT VALID
```

Figure 3-8 Removal of console ID usage from commands

Finally, in z/OS V1R8

In z/OS V1R8, the removal of 1-byte console IDs is completed. In some cases, z/OS V1R8 might still continue to accept 1-byte console IDs but will treat them differently, so programs that use 1-byte console IDs might behave unexpectedly. Because some control blocks and macros have been changed to remove 1-byte console ID support, avoid using 1-byte console ID fields when referencing these control blocks. For a list of changed control blocks and macros, see “Modify programs that reference unsupported console functions” in *z/OS V1R8 Migration Guide*.

The Console ID Tracking facility is still available to help identify 1-byte console ID usage.

In addition, in z/OS V1R8 a console has the ability to receive messages issued by an ID that cannot be resolved to a console (UNKNIDS attribute. Messages still issued with a 1-byte console ID will fall into this category. If a console has attribute UNKNIDS=Y, it receives messages issued by an ID that cannot be resolved to any console. An example of an unknown console ID is one which was formerly a migration ID.

Attention: If you have not run the console ID tracker facility yet, see APAR II13752 for detailed information about this facility.

3.9 Enhanced CONVCON and new CnzConv interfaces

z/OS V1R8 has a new CnzConv service and enhancements have been made to the existing CONVCON service.

3.9.1 Enhanced CONVCON

Prior to z/OS V1R8, a query of the console names INTERNAL and INSTREAM using the CONVCON service would return RC=x’8’ (invalid console name) and RSN=x’C’ (console name reserved). In z/OS V1R8, CONVCON returns RC=X'0' with the appropriate value in the 4-byte console ID field (CONVID). The query console names INTERNAL and INSTREAM now returns the same results as a query for the console ID INTERNAL (0x) and INSTREAM (80x). Therefore, if you depend on CONVCON returning RC=X'8' and RSN=X'C', you must make appropriate changes to your code.

Also, CONVCON would return an unpredictable value in register 0. In z/OS V1R8, register 0 now contains the reason code from the CONVCON query. The reason code is still available in field CONVRSN.

The following optional keywords are added to the CONVCON macro:

- RTNCODE - Save RC from Register 15 into a variable or register
- RSNCODE - Save RSN from Register 0 into a variable or register

Return and reason code constants are available in the IEZVG200 mapping macro.

3.9.2 CnzConv interface

In z/OS V1R8, a new interface, CnzConv service is introduced. This is a keyword-driven macro interface that supports 64-bit mode callers, a 64-bit parameter list, and also input/output values. For all these reasons, we recommend that you use CnzConv instead of CONVCON.

CnzCONV and CONVCON differences

The largest difference is that CnzConv supports more environments than CONVCON. CnzConv can be invoked in 64-bit mode and can have keyword values. The remaining environment attributes are the same, as shown in Table 3-1.

Table 3-1 Attributes of CnzConv

Attribute	Value
Dispatchable unit mode	Task
Minimum authorization	Problem state
Cross memory mode	Any primary, any secondary, and any home address space
AMODE	31 or 64 bit
ASC mode	Primary or access register
Interrupt status	Enabled for I/O and external interrupts
Locks	No locks held
Control parameters	Control parameters must be in the primary address space

There is a standard, list, and execute form of the CnzConv macro. InConsoleName or InConsoleId are mutually exclusive input keywords; one is required. The remaining keywords are outputs that can be specified on any CnzConv invocation.

There is no error checking for keywords that do not make sense when specified together. For example, if a program issued a CnzConv query for a console name of ITSO, and asked for the SMCS_LU, SubsysOwnerName, ConsoleStatus, and ConsoleType, the CnzConv query would complete successfully. Since a console cannot be an SMCS and a subsystem console, asking for SMCS_LU and SubsysOwnerName, that combination of parameters does not make sense. However, it will not result in a compile or assembly error. The idea is that if binary zeros are returned in a field, that particular piece of data was not returned by the CnzConv service.

The CnzConv service does not support the “Area ID syntax validation” supported by CONVCON.

Future enhancements will only be provided for the CnzConv macro. CnzConv supports the additional outputs listed in Table 3-2.

Table 3-2 Additional outputs of CnzConv over CONVCON

Output Name	Meaning
Console status	The status of the console queried: Active, Inactive
Console type	The console type of the console queried: MCS, EMCS, SMCS, Special, Subsystem
Console subtype	The console subtype of the console queried: Internal (Special), Unknown (Special), Instream (Special)
SMCS_LU	The logical unit (LU) name of the queried SMCS console
SubsysOwnerName	The subsystem owner name of the queried subsystem console
SubsysASID	The ASID of the queried subsystem console

ConsoleStatus was supported on the CONVCON macro in the form of return code 0, which indicated the console was active, or return code 4, which indicated the console was inactive. ConsoleType is new on the CnzConv macro. Via the CONVCON macro, you can only tell if a console is an SMCS console. A console type of `Special` will be returned for an input console name or ID that is reserved for use by the system. These consoles do not fit into any of the other console type categories, but are recognized as defined consoles by the system. For example, invoking CnzConv with a console name or ID of internal (x’0’) or instream (x’80’) will return a console type of `Special`. Currently, ConsoleSubType only returns subtypes for a console type of `Special`.

3.10 MSGRT elimination

For infrastructure simplification, the MSGRT command has been removed in z/OS V1R8. See Figure 3-9 on page 57. Although MSGRT no longer exists, you can add the L= operand to the command that was to be message routed to achieve similar results.

The MSGRT keyword on the CONSOLE statement in the CONSOLxx parmlib member in parmlib is not supported on z/OS V1R8. If not removed from the CONSOLxx member, the keyword is ignored during initialization.

```
CONSOLE DEVNUM(3FE) NAME(OPER2)
   UNIT(3270-X)
   AUTH(MASTER)
   ROUTCODE(1,2,4,6,8,65-96,9,10)
   MSGRT('D=(A,C,CONSOLES,D,DUMP,GRS,M,MPF,PFK,R,S,
                              SMF,U,OPDATA),L=OPER1-B')
   USE(FC) DEL(RD) AREA(18,12)
   INTIDS(Y) UNKNIDS(Y)
   RNUM(15) RTME(1) MSCOPE(*ALL)
```

Figure 3-9 MSGRT is removed from the CONSOLE statement

Note: The MSGRT command should not be confused with the ROUTE command. The ROUTE command is still supported.

3.11 Changes to operator commands

In z/OS V1R8, as a result of elimination of some functions in console restructure, some commands are removed and some are changed. You should notify your operators about these removed commands.

3.11.1 Removed commands

With the changes to support of the master console, console switching, and alternate consoles, the following commands are removed. When entering one of these commands, an invalid command message is received in z/OS V1R8 systems. There are also new console warning messages that indicate the functions that are no longer supported when you issue the commands in a sysplex with older z/OS systems together with z/OS V1R8 systems.

- VARY XXXX,MSTCONS
- DISPLAY CONSOLES,MCONLY
- SWITCH CN
- VARY CN(nnnnnn),ALTGRP
- MSGRT

In addition, the following keywords cannot be used to specify options in the CONSOLxx and CNGRPxx parmlib members:

- ALTGRP
- NOCCGRP
- MSGRT

3.11.2 Changed commands

The VARY CN command is changed to add the new keywords INTIDS and UNKNIDS, as shown in Figure 3-10 on page 58.

```
VARY CN(SC75),UNKNIDS=Y
IEE712I VARY CN  PROCESSING COMPLETE
```

Figure 3-10 VARY CN command to set UKNIDS to Y for one console

DISPLAY CONSOLES command

The command DISPLAY CONSOLES is changed to show the new attributes INTIDS and UNKNIDS on MCS and SMCS consoles, as shown in Figure 3-11.

```
D C
IEE889I 15.43.28 CONSOLE DISPLAY 778
MSG: CURR=0    LIM=1500 RPLY:CURR=0    LIM=999  SYS=SC75      PFK=00
 CONSOLE           ID  --------------- SPECIFICATIONS ---------------
 SYSLOG                COND=H      AUTH=CMDS         NBUF=N/A
                       ROUTCDE=ALL
 OPERLOG               COND=H      AUTH=CMDS         NBUF=N/A
                       ROUTCDE=ALL
 CNEF00            08  COND=A      AUTH=MASTER       NBUF=N/A
  EF00                 AREA=Z          MFORM=T,J,X
  SC74                 DEL=R    RTME=1/4    RNUM=28   SEG=28    CON=N
                       USE=FC   LEVEL=ALL             PFKTAB=MCONPFK0
                       ROUTCDE=ALL
                       LOGON=OPTIONAL
                       CMDSYS=SC74
                       MSCOPE=*ALL
                       INTIDS=N UNKNIDS=N
```

Figure 3-11 D C command changed to show new attributes

The DISPLAY EMCS command is changed to also show the new attributes INTIDS and UNKNIDS on an EMCS console, as shown in Figure 3-12.

```
D EMCS,FULL,CN=ROGERS
CNZ4101I 15.47.30 DISPLAY EMCS 782
  DISPLAY EMCS,FULL,CN=ROGERS
  NUMBER OF CONSOLES MATCHING CRITERIA: 1
  CN=ROGERS   STATUS=A    CNID=01000009 KEY=NONE
    SYS=SC75     ASID=005A JOBNAME=ROGERS   JOBID=JOB05352
    HC=N AUTO=N DOM=NORMAL TERMNAME=SC38TC67
    MONITOR=--------
    CMDSYS=SC75
    LEVEL=ALL         AUTH= INFO
    MSCOPE=*ALL
    ROUTCDE=NONE
    INTIDS=N UNKNIDS=N
    ALERTPCT=100
    QUEUED=0          QLIMIT=59998
    SIZEUSED=540K      MAXSIZE=1024K
```

Figure 3-12 D EMCS command changed to show new attributes

The IEEQEMCS macro service is changed to add the values of the new attributes to the returned output. It also has them as filters.

The VARY CN command is also changed to be able to set new keywords for console definitions. See Figure 3-13 for an example.

```
VARY CN(MERAL),INTIDS=Y
 IEE712I VARY CN  PROCESSING COMPLETE
```

Figure 3-13 VARY CN command is changed to set new keywords

3.12 Migration and coexistence considerations

In order to coexist in a sysplex with z/OS V1R8, previous z/OS systems need to install the PTF for APAR OA10632.

This APAR is needed to:

- Synchronize functionality between systems.
- Provide a consistent view of console data.
- Provide message delivery based on new routing attributes.

The following message in Figure 3-14 is seen if the systems in a sysplex are not compatible:

```
IEA002I APAR OA10632 IS NOT INSTALLED ON <sys1>.  SYSTEM <sys2> IS BEING
PARTITIONED.
```

Figure 3-14 Message IEA002I when coexistence APAR OA10632 is not installed

3.12.1 z/OS V1R8 system joins sysplex

During the IPL of a z/OS V1R8 system to join the sysplex with pre-z/OS V1R8 systems only, the following actions occur within 5 seconds of notification of a z/OS V1R8 system joining the sysplex:

- An attempt is made to reset any switched consoles.
- In this mixed-level sysplex containing a z/OS V1R8 system, console switching is disabled.
- The master console designation is dropped (consoles continue to have master authority). The removal of the master console also means that a *no consoles* condition or *no master consoles* condition is no longer recognized.

Console switch function

When a z/OS V1R8 system joins a sysplex with earlier versions (z/OS V1R7 and below), the console switch function is disabled in the sysplex. A CNZ4202I message is issued, as shown in Figure 3-15.

```
CNZ4202I CONSOLE SWITCH FUNCTION NOW INOPERATIVE BECAUSE SYSTEM SC75
       JOINED THE SYSPLEX
```

Figure 3-15 The console switch function is disabled when a z/OS V1R8 system joins the sysplex

SWITCH CN command

In a mixed sysplex, that is, with z/OS V1R8 and also earlier versions, when the command SWITCH CN is used, you will get message CNZ0004I informing you that console switch is not supported in the sysplex because at least one z/OS V1R8 system exists in the sysplex.

```
SWITCH CN=CNE201
CNZ0004I SWITCH CN NOT SUPPORTED DUE TO CURRENT CONFIGURATION.
        REASON=z/OS V1R8 OR HIGHER SYSTEM IN SYSPLEX
```

Figure 3-16 CNZ0004I: console switch command cannot be used in a mixed sysplex

Additional messages in a mixed sysplex

The following messages are issued when certain commands are issued:

```
VARY E201,MSTCONS
CNZ0004I VARY MSTCONS NOT SUPPORTED DUE TO CURRENT CONFIGURATION.
        REASON=z/OS V1R8 OR HIGHER SYSTEM IN SYSPLEX
```

Figure 3-17 CNZ0004I: Master console role disabled when z/OS V1R8 joined sysplex

```
CNZ4204I AT THE TIME CONSOLE SWITCH BECAME INOPERATIVE,
        THE FOLLOWING CONSOLES WERE SWITCHED AND HAVE BEEN RESET:
        CONSOLE     SWITCHED TO
             CN20 -> CN10
```

Figure 3-18 CNZ4204I: console switch inoperative

```
CNZ4205I CONSOLE CN30 IS NO LONGER THE MASTER CONSOLE BECAUSE
         SYSTEM SC75 JOINED THE SYSPLEX.
         THE MASTER CONSOLE IS NO LONGER SUPPORTED
```

Figure 3-19 CNZ4205I: Master console disabled

3.12.2 z/OS V1R8 system leaves a mixed sysplex

When the last z/OS V1R8 system leaves the sysplex, console switching and master console support are enabled. Messages CNZ4203I and CNZ4206A are issued, as shown in Figure 3-20 and Figure 3-21.

```
 CNZ4203I CONSOLE SWITCH FUNCTION NOW OPERATIVE
```

Figure 3-20 CNZ4203I: Console switch enabled after last z/OS V1R8 leaves sysplex

```
*CNZ4206A NO MASTER CONSOLE IS ACTIVE. ISSUE VARY MSTCONS OR 085
 LOG ON TO A MASTER AUTHORITY SMCS CONSOLE
```

Figure 3-21 CNZ4206A: Master console enabled after last z/OS V1R8 leaves sysplex

3.13 One-byte console ID tracker

The Console ID Tracking facility is designed to assist with the identification and removal of 1-byte console IDs and 1-byte migration IDs. In this release, only four-byte IDs are accepted. While four-byte IDs have generally replaced 1-byte IDs, some services still accept 1-byte IDs. The users of services that still accept 1-byte console and migration IDs are now known as violators, and instances of 1-byte ID usage are known as violations.

Because no interfaces are being changed, the Console ID Tracking facility does not present any compatibility issues. It tracks 1-byte users on the following macro services:

WTO
MPF
SSI
MGCR/MGCRE
CONVCON
MCSOPER

To prepare for the removal of 1-byte console IDs, the Console ID Tracking facility provides the following new functions:

- The SETCON operator command, which is used to activate and deactivate the Console ID Tracking facility
- The DISPLAY OPDATA,TRACKING operator command, which is used to display the current status of the Console ID Tracking facility, along with any recorded instances of violations
- The CNIDTRxx parmlib member, which is used to list violations that have already been identified in order to prevent them from being recorded again
- The CNZTRKR macro, which is used to invoke the Console ID Tracking facility

3.13.1 The SETCON operator command

The SETCON operator command allows the Console ID Tracking facility to be activated with an ABEND option, or deactivated, as follows.

```
SETCON TRACKING=ON
```

Use this command to activate the tracking facility.

You can issue this command manually, or it can be automated by placing the command in a COMMNDxx member of parmlib:

```
SETCON TRACKING=ONWITHABEND
```

Use this command to activate the tracking facility and cause violators to be ABENDed with ABEND code 077, reason code 0034. This ABEND is designed to allow an installation to set a SLIP trap to obtain a dump, as follows:

```
SLIP SET,ENABLE,ID=TRAK,COMP=077, REASON=34,ACTION=SVCD,END
```

If no SLIP is set, an entry in Logrec is made, but no dump is taken. You can toggle between SETCON TRACKING=ON and SETCON TRACKING=ONWITHABEND without any loss of recorded instances.

Note: If the track value is 0 or 128, no ABEND is issued even when you specify ONWITHABEND.

Use this command to deactivate the tracking facility:

```
SETCON TRACKING=OFF
```

Before turning the facility off, the SETCON TRACKING=OFF command issues a DISPLAY OPDATA,TRACKING command to capture all recorded instances of violations in the hardcopy log.

The tracking facility then deletes all recorded violations and does not record any more violations.

After issuing a SETCON TRACKING=OFF command, wait for message IEE7121 SETCON PROCESSING COMPLETE to appear, to ensure that the facility has had time to fully deactivate before you attempt to reactivate it.

If you do not wait, the activation command may complete before the deactivation command finishes, leaving the facility off when you expect it to be on.

Note: The maximum number of unique instances that the Console ID Tracking facility can record is 1,000.

3.13.2 Using the DISPLAY OPDATA,TRACKING command

The DISPLAY OPDATA,TRACKING command displays the current status of the Console ID Tracking facility, along with information about each recorded instance of 1-byte console ID or migration ID usage.

The DISPLAY OPDATA,TRACKING command displays the following information about the Console ID Tracking facility:

- Whether the facility is ON, ONWITHABEND, or OFF.
- The number of unique violations that have been recorded.
- The maximum number of unique violations that is accepted.
- The suffix of the active CNIDTRxx parmlib member (if any).
- The number of violations that were not recorded because they were excluded.
- The number of violations that were not recorded because the facility was full, because of timing issues, or because serialization of the facility could not be obtained.
- An indication if the facility is full.

The DISPLAY OPDATA,TRACKING command displays information about a violation, as shown in Figure 3-22 on page 63. The information displayed in this example of the command response when tracking is active and violations have been recorded is as follows:

- The tracking information (up to 28 characters) that was provided when the tracking request was made.
- The four-byte track value (typically a console ID) that was provided when the tracking request was made.
- This value is displayed in hexadecimal, with leading zeros suppressed.
- If no value was provided, zero is displayed, since the zero could be valid (for instance, if the console ID is zero).
- The name of the job from which the tracking request was made.
- The name of the program from which the tracking request was made.

- The offset into the program where the violation occurred and ASID of the violator.

```
CNZ1001I 10.53.40 TRACKING DISPLAY
STATUS=ON,ABEND NUM=19   MAX=1000 MEM=n/a EXCL=0     REJECT=0
----TRACKING INFORMATION---- -VALUE-- JOBNAME  PROGNAME+OFF-- ASID NUM
Parmlib Reader: ADYSET00           00 *MASTER* ADYSETP   1BD8   01   1
Parmlib Reader: COFVLF04           00 VLF      COFMINIT  2EFE   18   1
Parmlib Reader: IEFSSN00           00 *MASTER* IEEMB860  9E2A   01   1
Parmlib Reader: SMFPRM00           00 SMF      IFASMF    ECBE   19   1
WTO: $HASP000 OK                   00 JES2     HASJES20 1B0AC   14   2
WTO: $HASP003         SPECIF       00 JES2     HASJES20 1B0AC   14   2
WTO: $HASP003 RC=(52),             00 JES2     HASJES20 1B0AC   14   2
WTO: $HASP003 RC=(52),S1-999       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP003 RC=(52),T1-999       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP604 ID 0007 T=***.       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP604 ID 0008 T=***.       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP604 ID 0010 T=***.       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP604 ID 0011 T=***.       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP646 0.5714 PERCENT       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP650 Q,Q=W     INVA       00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP893                      00 JES2     HASJES20 1B0AC   14   1
WTO: $HASP893 VOLUME(SPOOL1)       00 JES2     HASJES20 1B0AC   14   2
WTO: IEC350I CATALOG ADDRESS       00 CATALOG  IGG0CLX0 1BBC2   1B   1
WTO: IEF677I WARNING MESSAGE       00 JES2     IEFNB903  BF12   14   1
```

Figure 3-22 Example of the display tracking command

3.13.3 Using the CNIDTRxx parmlib member

The CNIDTRxx parmlib member allows you to exclude specific violation instances of 1-byte console IDs from being recorded by the Console ID Tracking facility. This exclusion allows the facility to ignore violations that have already been reported but not yet corrected.

Use the SET CNIDTR=xx command to activate the CNIDTRxx parmlib member.

If no CNIDTRxx parmlib member is active, then all instances of 1-byte console ID violations will be recorded. Customers are expected to update the CNIDTRxx parmlib member with reported violations. Violations should be reported to IBM, and updates to CNIDTRxx must also be made available.

Data to add to the CNIDTRxx member

CNIDTRxx is used to list the following information about each tracking instance for exclusion processing (the wild cards * and ? are supported):

- Tracking information (up to 28 characters)
- Job name (up to 8 characters)
- Program name (up to 8 characters)

Be aware that once a violation is recorded, changing the exclusion list in CNIDTRxx does not remove the instance from the list of recorded instances displayed by the DISPLAY OPDATA,TRACKING command.

- The Console ID Tracking facility must be restarted to exclude any new additions to the CNIDTRxx parmlib member.

Note: Many of the services that track console IDs also invoke the CONVCON service. This means that one violation might cause two instances to be recorded in the Console ID Tracking facility: one for CONVCON and one for the service that called CONVCON.

Sample output using CNIDTRxx to exclude violations

Figure 3-23 is an example of using the CNIDTRxx parmlib member.

```
*                              Jobname  Pgmname
* Tracking Information Mask    Mask     Mask     Comments (ignored)
*+-------------------------+ +-------+ +-------+ +----------------------+
 WTO: $HASP*                   JES*     HAS*     Ignore JES2 messages
```

Figure 3-23 CNIDTRxx parmlib member example

Using the D OPDATA,TRACKING command after the parmlib member is created is shown in Figure 3-24. See the contrast between this and Figure 3-22 on page 63.

```
CNZ1001I 10.53.40 TRACKING DISPLAY
STATUS=ON       NUM=6    MAX=1000 MEM=00  EXCL=13    REJECT=0
----TRACKING INFORMATION---- -VALUE-- JOBNAME  PROGNAME+OFF-- ASID NUM
Parmlib Reader: ADYSET00           00 *MASTER* ADYSETP   1BD8   01   1
Parmlib Reader: COFVLF04           00 VLF      COFMINIT  2EFE   18   1
Parmlib Reader: IEFSSN00           00 *MASTER* IEEMB860  9E2A   01   1
Parmlib Reader: SMFPRM00           00 SMF      IFASMF    ECBE   19   1
WTO: IEC350I CATALOG ADDRESS       00 CATALOG  IGG0CLX0 1BBC2   1B   1
WTO: IEF677I WARNING MESSAGE       00 JES2     IEFNB903  BF12   14   1
```

Figure 3-24 D OPDATA,TRACKING command after the CNIDTRxx parmlib member is created

Note: After you issue the SET CNIDTR=xx command to activate the CNIDTRxx member, JES2 violations are no longer recorded or displayed.

3.13.4 Using the CNZTRKR macro

You can use the CNZTRKR macro to invoke the Console ID Tracking facility, which records violations of 1-byte console ID usage.

Before issuing the CNZTRKR macro, you must do the following:

- Include the CNZTRPL mapping macro in your program.
- Obtain storage for the CNZTRKR parameter list.
- TRPL_LEN in CNZTRPL contains the length of the parameter list. The parameter list can be in any type of storage.
- Clear the entire parameter list by setting it to binary zeros.
- Initialize the following fields in the parameter list mapped by macro CNZTRPL:

TRPL_Acro	The TRPL acronym.
TRPL_Version	The current version level of the parameter list. The CNZTRPL mapping macro contains the current version level in TRPL_K_Curr_Version.

TRPL_Track_Info	Text that describes the occurrence of this instance. This text can be from 1 to 28 characters in length. Any EBCDIC value is allowed, although you should use displayable characters because undisplayable characters may be changed to blanks when displayed on an operator's console or in the hardcopy log. The text cannot be all blank or all hexadecimal zeros.
TRPL_Track_Data	Four bytes of data associated with this track instance. This data could be the 1-byte console ID that was used by the violator. Zero is a valid value. The DISPLAY OPDATA operator command will display this value as a hexadecimal number.
TRPL_Violators_Addr	While optional, this field should contain the address where the violation occurred (perhaps the address to which the service invoking CNZTRKR will return). If set to zero, the Console ID Tracking facility will attempt to determine the violation address but may not be able to determine the exact violation location. This address is assumed to be a 31-bit address. If a 24-bit address is provided, you must ensure that the high-order byte of the address is zero.

3.14 z/OS V1R8 console restructure migration checklist

Table 3-3 shows the checklist for console restructure in z/OS V1R8.

Table 3-3 Checklist for console restructure in z/OS V1R8

Step	Action	Done
1	Apply the PTFs for OA10632 to the previous z/OS systems in order to coexist withthe z/OS V1R8 system.	
2	Obtain the latest product and service levels for IBM and vendor products that used 1-byte console IDs.	
3	Notify the operators that the following commands are no longer supported: VARY CN(....),MSTCONS VARY CN(....),ALTGRP DISPLAY C,MCONLY SWITCH CN(....) MSGRT ...	
4	Remove the following keywords from the CONSOLxx parmlib member: NOCCGRP ALTGRP MSGRT keyword on the CONSOLE statement You can keep these keywords if you plan to share these members with z/OS levels pre-V1R8. z/OS V1R8 issues warning messages about unsupported keywords but the system continues with no errors.	
5	Remove the following keywords from the CNGRPxx parmlib member: *MSTCON* Remove groups that are only used for alternate console selection. Groups are still used for SYNCHDEST and system console AUTOACT. You can keep these keywords if you plan to share these members with pre-z/OS V1R8. z/OS V1R8 issues warning messages about unsupported keywords but the system continues with no errors.	

Step	Action	Done
6	Since there is no longer an alternate console or switch mechanism, explicitly configure backup consoles with the same attribute if one specific function is desired to be always available.	
7	Specify master authority for MCS or SMCS consoles if desired, because no MCS and SMCS console will be forced to have master authority.	
8	Specify some consoles to receive routing codes 1 and 2 if desired, because there will be no master console to receive these by default.	
9	Define both SYSLOG and OPERLOG as hardcopy medium, since there will be no switch mechanism if one fails.	
10	Use the Console ID Tracker to get information on the usage of 1-byte console ID in your installation. It will not be supported anymore. See APAR II13752 for detailed information.	
11	Examine your automation procedures for commands, functions and messages that no longer exist.	
12	Determine if any new messages need to be automated.	
13	Use the recommended CNZCONV macro instead of the CONVCON macro where possible.	
14	Examine programs that reference the control blocks that have been changed to remove support for 1-byte console IDs, console switching, and master consoles. These control blocks are listed in the chapter “BCP Migration Actions” in the z/OS V1R8 migration book section “Modify programs that reference unsupported console functions.”	
15	Ensure that the programs will continue to work now that 1-byte ID console support has been removed from the control blocks. Because the 1-byte fields have been removed, a clean assembly or compilation will assure you that you were successful in removing all references to 1-byte fields.	
16	Add the INTIDS keyword in the CONSOLxx parmlib member when defining consoles that you want to receive messages that are directed to console ID zero.	
17	Add the UNKNIDS keyword in CONSOLxx parmlib member during defining consoles that you want to receive messages that have unresolved console IDs. In z/OS V1R8, any consoles that request this UNKNIDS attribute receive messages directed to 1-byte console IDs.	

4

CFRM performance enhancements

This chapter describes CFRM performance enhancements in z/OS V1R8. They are a continuation of the CFRM performance enhancements delivered in z/OS V1R4.

The following topics are discussed:

- Introduction to new enhancements
- CFRM performance enhancements history
- Message-based protocol
  - Policy-based versus message-based
  - Choosing an event manager system in the sysplex
- Interactions and dependencies
- Implementing the enhancements
- Migration and coexistence considerations
- Performance comparisons
- Updated commands and parameters reference
- Messages

4.1 Introduction to the enhancements

In z/OS V1R8, with the newly introduced CFRM performance enhancements, Parallel Sysplex environments can realize significant improvement in availability during recovery processing of Coupling Facility resources. This improvement is achieved by a new event processing protocol called *message-based protocol*.

APAR OW48624 to z/OS V1R2

Previously, support was added to z/OS V1R2 with APAR OW48624 to optimize some of the system failure cleanup processing for CF structures. As the number of connections to CF structures increased, a great need appeared to enhance and optimize processing of these connectors during system failure recovery and system failure cleanup. The following changes were introduced:

- Changes were made to have only one system in the sysplex do the determination of the connectors that were active on the failed system, and initiating cleanup events for those connectors, instead of all systems in the sysplex doing these processes.
- A DiscFailConn confirmation look-ahead algorithm for processing the confirmations from all the other active connectors was updated to improve its efficiency.
- Critical structures were given priority by the functions that process the system failure events.
- CFRM I/O processing was reduced for user sync point (IXLUSYNC) event processing.

z/OS V1R4 enhancements

Then, in z/OS V1R4, processing optimizations to CF structure rebuild processing and IXCQUERY processing were added, as follows:

- Rebuild confirmation look-ahead provided CFRM I/O contention relief and enabled optimized structure confirmation processing. This enhancement was done by handing all queued event confirmations in the same call function. This reduced CFRM contention, enabling much faster CF structure rebuilds. However, serialization of the CFRM CDS was still needed, and the CDS still had to be accessed by every system in the sysplex in case of events such as structure rebuilds, CF connectivity failovers, system failovers, and sysplex partitioning.
- The amount of I/O to the CFRM CDS for IXCQUERY commands was minimized. This was done by reducing the amount of policy data that should be retrieved to satisfy requests by analyzing that specific IXCQUERY request first.

In the existing protocol called *policy-based protocol*, every surviving system needs to access the CFRM CDS using serialization in case of a CF or system failure. In order to do the necessary cleanup processing, they need to access the CFRM CDS one system at a time. This causes contention on the CDS and slows the recovery process. Depending on the environment, this recovery process can take many minutes. This impacts the availability of CF structure data and thus the availability of products and subsystems using the structures.

4.2 z/OS V1R8 CFRM performance enhancements

The enhancements in z/OS v1R8 are intended to improve performance across all products and subsystems that use resources managed by CFRM. The CFRM Couple Data Set (CDS) is the centralized control point for CFRM and can become a bottleneck during recovery actions. To address these problems, the following enhancements were added in z/OS 1.8:

- Introduced a new message-based protocol to minimize CDS access and thereby improve recovery time. This new protocol optionally replaces the existing policy-based protocol.
- The new protocol defines a single system as the event manager, responsible for coordinating events and confirmations with participating systems and updating the CFRM CDS when needed.
- Most CFRM CDS I/O are now initiated from a single system, greatly reducing CFRM CDS I/O and contention.
- The message-based protocol is initiated on a structure-by-structure basis:

  A structure is eligible to transition to message-based when it has no events in progress.
- Communication between the manager system and the participant systems uses XCF signalling to send messages between the systems.
- XCF signalling structures are not eligible to use the new message-based protocol.

CFRM performance benefits

CFRM performance enhancements update the current implementation to avoid using the CFRM CDS as much as possible for functions that have been bottlenecked by CFRM I/O in the past.

Customers can benefit from the new enhancements in the following situations:

- Coupling Facility and Coupling Facility connectivity failovers
- System failovers
- Rebuild process of any structure except XCF signalling list structures
- Duplexing failovers
- Sysplex partitioning actions

Note: Since the XCF REALLOCATE command uses the same modules for rebuilding structures, benefits can be achieved here also.

CFRM active policy

The CFRM active policy defines and maintains status for CFRM resources. Each system accesses the policy possibly many times. In a sysplex environment every system should communicate with every other and should be notified in case of any CFRM resource state changes. The types of failures may be caused by a command, by resource owners on purpose, or be the result of an XES event. An XES event is the communication mechanism between XES and the CFRM and the connections to CF structures.

4.2.1 Policy-based protocol flow

To understand the changes in z/OS V1R8 with the optional message-based protocol, here is an example of how the current policy-based protocol is designed, as shown in the following steps in Figure 4-1 on page 70:

1. Operator issued a command in system SYS1 to rebuild a structure.

2. System SYS1 creates an event in the event stack. The event stack is simply an area in CFRM CDS that is used by all systems to do the communication for notifying of events. This means SYS1 does an update to CFRM CDS.
3. SYS1 then notifies all other participant systems in the sysplex by using XCF signalling. It simply sends GAT signals which have no data information about details of events that it created recently. By this GAT signal it simply says that it recently created an event and tells the systems to read the policy and examine the event stack for new events.
4. Each participant system (a system is participant if it has an active connector to structure) then reads the CFRM policy to discover events that have not been presented to local connectors. Local connectors of a system are applications and subsystems running on that system that are using structures.
5. Each participant updates the data area in CFRM CDS to say that it processed the event. When all systems have “seen” the event, the last system that has “seen” the event pops up the event (removes) from the event stack.

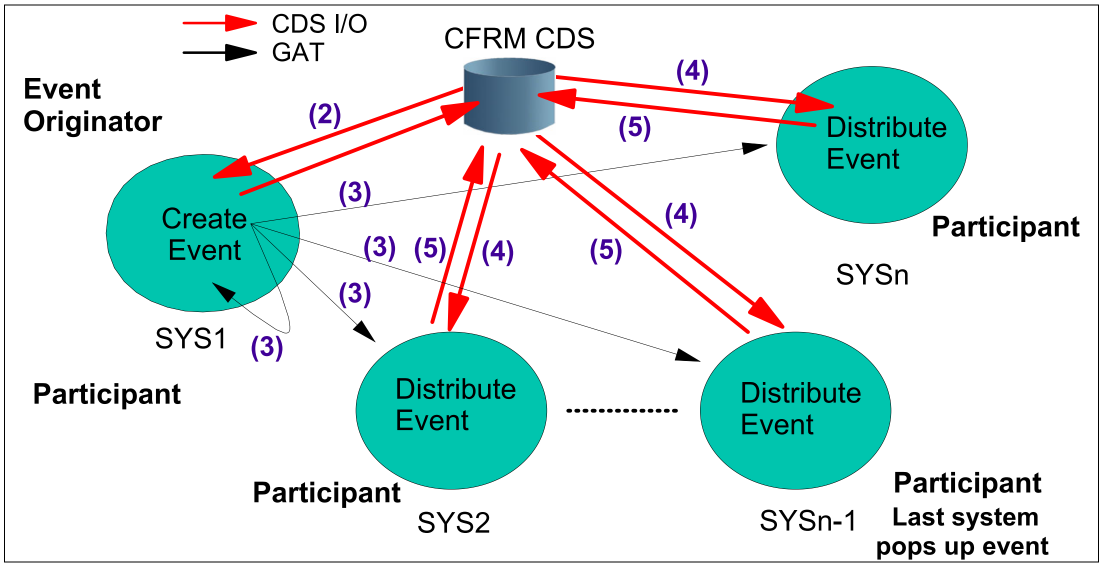


Figure 4-1 Event management flow in policy-based protocol

Important: Updating the policy requires access serialized by policy lock. Since every participant attempts to read and potentially update the policy at about the same time, contention on the CFRM couple data set results. A policy lock is used instead of GRS ENQ, because GRS has structure itself that needs to be managed by CFRM. In order not to go into a deadlock situation, CFRM CDS serialization uses the policy lock mechanism.

Imagine how this process causes events like system failure (multiple simultaneous connector failures) or loss of CF connectivity (multiple simultaneous structure rebuilds) because it greatly increases CFRM accesses and contention.

4.2.2 New message-based protocol

The message-based protocol allows the XES event processing (both event delivery and response processing) to be accomplished without all systems trying to access the CFRM CDS at the same time, thus eliminating the bottleneck on the CFRM CDS and reducing the elapsed time for recovery. Message-based protocol is initiated on a structure-by-structure basis.

Basically, with this new protocol, a system in the sysplex is chosen as the event manager. Only the event-originating system and the event manager system will need to access the CFRM CDS, and will *not* need to access it at the same time. A system is a participant when structures are being used by subsystems running on that system. The manager is responsible for coordinating events and confirmations with participating systems and updating CFRM CDS when needed. This reduces CFRM CDS I/O to a central point. Communication between the manager and the participant systems uses XCF signalling to send messages between the systems.

CFRM CDS format

In order to use the new message-based protocol, the CFRM couple data set should be formatted with a new statement:

```
ITEM NAME(MSGBASED) NUMBER(1)
```

To be able to use this new formatted CFRM couple data set, every system in the sysplex should be running z/OS V1R8.

Attention: The message-based protocol optionally replaces the policy-based protocol. You can put it into use any time you want to. We recommend that you enable the new message-based processing protocol in order to obtain better performance, availability, and scalability benefits.

4.2.3 Message-based protocol flow

Now let us see how this process works in the message-based protocol. When the sysplex message-based protocol is enabled and SYSn has been chosen as the event manager system, as shown in Figure 4-2 on page 72, the processing flow is as follows:

1. An operator issues a command in system SYS1 to rebuild a structure.
2. System SYS1 creates an event in the event stack. To do this an update in CFRM CDS is made.
3. Instead of sending GAT signals to other participants to notify them about the new event, SYS1 now sends “event signals” using XCF signalling. This differs from GAT signals because it is no longer a simple “look in the policy” notification. This signal now contains data related to this event. Therefore, there is no need for other systems to access and read the CFRM CDS to see what event it is and who created it. Also, there is no need for them to update the CDS to say they are done with this event information. They have already received the information they need to deal with this event.
4. The event manager system is always notified whether or not it is a participant of that event. In other words, according to this example, the event manager is notified whether or not the local connectors have active connections to the related structure.
5. For participant systems and the system that created the event, they simply notify the event manager system when their processing is done. This is done by just sending an ACK signal using XCF signalling paths.
6. When the event manager system gets acknowledgement signals from all participant systems, it updates the CFRM policy to remove the event from the event stack. Only the event manager reads and updates the policy when all participants have done their work.
7. After removing the event from the event stack, the event manager sends a discard signal to all participants including the event creator and the event manager. The aim of this signal is to inform the interested systems that the event is removed from the event stack so that they can discard their local representations of the event.

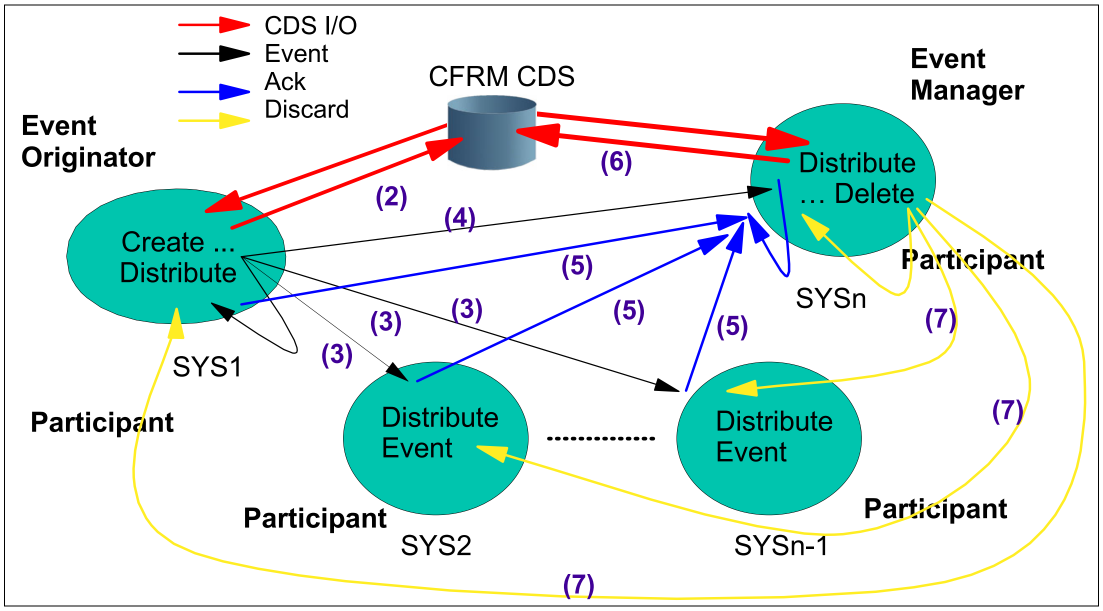


Figure 4-2 Event management flow in the message-based protocol

Restriction: XCF signalling structures cannot use the message-based protocol. This is because the message-based protocol uses XCF signalling to send information related to an event. In order not to get into a deadlock situation, XCF signalling structures are not allowed to use this protocol.

4.2.4 Choosing an event manager system in a sysplex

During the process of placing the message-based protocol into use in a sysplex, one of the systems is chosen as event manager. The event manager is responsible for the following:

- Event coordinations and confirmations with participant systems
- Updating CFRM CDS when needed during these processes
- Initiating most CFRM CDS I/O in the sysplex

With the new algorithm, the CFRM CDS IXCLOSUP sub record, which previously was empty, now contains information about the event manager system and the type of protocol in use. The first system selected to update this record becomes the new event manager.

Important: All processes that initiate the message-based protocol try to make the current system the event manager.

Defining the message-based protocol

A sysplex can start using the message-based protocol when a message-based protocol capable CFRM CDS is put into use, as follows:

- During an IPL via the COUPLEXX parmlib member, as follows:

  The first system, normally the one that is IPLed, is the one that updates this record and then this system becomes the event manager.
- Issue the SETXCF PSWITCH,SETXCF PCOUPLE command.

The first system, normally the one that sends this command, updates the record and becomes the event manager.

- The protocol can be started explicitly using the SETXCF START,MSGBASED command.

  This command should be used if you start with the policy-based protocol and then decide to use the message-based protocol. In this case, the system that the command is issued on becomes the event manager.

Important: If the event manager system fails, the first system to succeed in accessing and updating the CFRM CDS is the next event manager. This becomes the system that handles termination of the current manager. Remember that system failures are also an event, which means other systems are notified. If no event manager can be found, the sysplex goes back to the policy-based protocol. In this case, all unfinished events and signals are redriven.

In all situations where an event manager could not be chosen for any reason, the sysplex goes back to the policy-based protocol.

Flow for changing protocols

At any time installations can change the event manager system using the following flow:

1. Send the SETXCF STOP,MSGBASED command from any system in the sysplex.
2. Log on to the system which you want to be the new event manager. Issue the SETXCF START,MSGBASED command. That system becomes the new event manager.

Important: In summary, the event manager is generally the winner of a race—first to IPL, or first to complete processing to bring a MSGBASED-capable CDS into use, or the system that handles termination of the current manager. If you are unhappy about the choice of manager system for imbalance, performance, or other reasons, you have the ability to change it.

4.3 Implementing new enhancements

A new version of the CFRM CDS is required for systems to enable message-based protocol. A CFRM CDS must be formatted specifying the new ITEM MSGBASED. The new formatted CFRM couple data set for message-based processing must be the primary CFRM CDS for the sysplex in order for the message-based protocol to become enabled. As soon as the new CFRM CDS is put into use as the primary CDS, the message-based protocol will start to be used. This CFRM CDS must be specified either in the COUPLExx member or in a SETXCXF COUPLE command to activate new CFRM CDS.

Figure 4-3 on page 74 shows the sample JCL for the format utility to introduce the message-based protocol. Sample JCL to run the format utility for formatting couple data sets for CFRM is shipped in the SYS1.SAMPLIB member IXCCFRMF.

```
//DEFCFRM   JOB (999,POK),'DEFINE CFRM CDS',CLASS=A,REGION=4M,
//             MSGCLASS=T,TIME=10,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//STEP1    EXEC PGM=IXCL1DSU
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
     DEFINEDS SYSPLEX(PLEX75)
              MAXSYSTEM(2)
              DSN(SYS1.XCF.CFRM10) VOLSER(BH5CD2)
              CATALOG
          DATA TYPE(CFRM)
               ITEM NAME(POLICY) NUMBER(10)
               ITEM NAME(CF) NUMBER(8)
               ITEM NAME(STR) NUMBER(512)
               ITEM NAME(CONNECT) NUMBER(32)
               ITEM NAME(SMREBLD) NUMBER(1)
               ITEM NAME(SMDUPLEX) NUMBER(1)
               ITEM NAME(MSGBASED) NUMBER(1)/*
```

Figure 4-3 JCL to format the CFRM CDS for the message-based protocol

4.3.1 Enabling the message-based protocol

The message-based protocol can be enabled using one of the following methods:

- During system initiation

  When a system is IPLed with a new primary CFRM CDS that is formatted to include the ITEM(MSGBASED) statement, the sysplex will start using the new message-based protocol. During CFRM initialization processing, the system sends message IXC549I, which indicates that the message-based protocol is active, as shown in Figure 4-4 on page 74.

```
IXC286I COUPLE DATA SET
SYS1.XCF.CFRM10,
VOLSER BH5CD2, HAS BEEN ADDED AS THE PRIMARY
FOR CFRM ON SYSTEM SC74
IXC286I COUPLE DATA SET
SYS1.XCF.CFRM11,
VOLSER BH5CD1, HAS BEEN ADDED AS THE ALTERNATE
FOR CFRM ON SYSTEM SC74
IXC549I EVENT MANAGEMENT: MESSAGE-BASED  MANAGER SYSTEM NAME:
SC75
IEE252I MEMBER CTIGRS00 FOUND IN SYS1.IBM.PARMLIB
IXC517I SYSTEM SC74 ABLE TO USE
        COUPLING FACILITY  002094.IBM.02.00000002991E
                           PARTITION: 0D    CPCID: 00
        NAMED CF7A
IXC517I SYSTEM SC74 ABLE TO USE
        COUPLING FACILITY  002094.IBM.02.00000002991E
                           PARTITION: 1D    CPCID: 00
        NAMED CF7B
```

Figure 4-4 Message IXC549I issued indicating CFRM event management protocol

If this was not the first use of new ITEM(MSGBASED) in CFRM CDS, systems will start using the protocol that was last active. In this case, the IXC549I message will indicate the CFRM event management protocol according to the CFRM that was the last active policy.

- Using the SETXCF COUPLE,PSWITCH or SETXCF COUPLE,PCOUPLE commands

  After you decide to enable the message-based protocol, the CFRM CDS that is formatted to support the new protocol can be put into use by issuing the SETXCF COUPLE,PSWITCH command. After this a new CFRM CDS becomes the new alternate CDS data set, then the SETXCF COUPLE,PSWITCH,TYPE=CFRM command can be issued and the sysplex now starts using the new protocol, as shown in Figure 4-5 on page 75.

```
SETXCF CPL,PSWITCH,TYPE=CFRM
 IXC309I SETXCF COUPLE,PSWITCH REQUEST FOR CFRM WAS ACCEPTED
 IXC257I PRIMARY COUPLE DATA SET
 SYS1.XCF.CFRM00 FOR CFRM
 IS BEING REPLACED BY
 SYS1.XCF.CFRM10 DUE TO OPERATOR REQUEST
 IEF196I IEF285I   SYS1.XCF.CFRM00
 IEF196I IEF285I   VOL SER NOS= BH5CD2.
 IXC263I REMOVAL OF THE PRIMARY COUPLE DATA SET
 SYS1.XCF.CFRM00 FOR CFRM IS COMPLETE
 IXC267E PROCESSING WITHOUT AN ALTERNATE
 COUPLE DATA SET FOR CFRM.
 ISSUE SETXCF COMMAND TO ACTIVATE A NEW ALTERNATE.
 IXC548I CFRM EVENT MANAGEMENT ENVIRONMENT UPDATED
 EVENT MANAGEMENT PROTOCOL: MESSAGE-BASED
 REASON FOR CHANGE: FIRST USE OF MSGBASED-CAPABLE COUPLE DATA SET
 TRANSITION SEQUENCE NUMBER: 00000001
 TRANSITION TIME: 05/18/2006 14:42:10.766547
 MANAGER SYSTEM NAME: SC75
 MANAGER SYSTEM NUMBER: 0100002E
```

Figure 4-5 Activating the message-based protocol using the CFRM CDS switch process

- Using the SETXCF START,MSGBASED command

  Assuming that the active primary CFRM couple data set is message-based protocol enabled and for some reason the sysplex was put back into policy-based protocol, the message-based protocol can be put back into use with the new SETXCF START,MSGBASED command, as shown in Figure 4-6 on page 75.

```
SETXCF START,MSGBASED
IXC548I CFRM EVENT MANAGEMENT ENVIRONMENT UPDATED
EVENT MANAGEMENT PROTOCOL: MESSAGE-BASED
REASON FOR CHANGE: SETXCF COMMAND
TRANSITION SEQUENCE NUMBER: 00000007
TRANSITION TIME: 05/25/2006 16:31:56.519269
MANAGER SYSTEM NAME: SC74
MANAGER SYSTEM NUMBER: 0200003A
IXC547I THE SETXCF START,MSGBASED REQUEST COMPLETED
```

Figure 4-6 Going back to the message-based protocol via the SETXCF START,MSGBASED command

The message-based protocol is initiated on a structure-by-structure basis. After it is activated in the whole sysplex, an individual structure may continue to use policy-based processing until all events already in process have completed.

Note: When structure information is displayed using the D XCF,STR,STRNAME command, it is normal to see some structures still using the policy-based protocol, although the message-based protocol is activated on that sysplex. A structure is eligible to transition to the message-based protocol when it has no events in progress. The transition occurs on the first event processed after the message-based protocol is selected, which may be quite some time.

4.4 Migration and coexistence considerations

All systems in the sysplex must be at the z/OS V1R8 level. Systems that do not support message-based processing, z/OS V1R7 and below, are not able to join a sysplex that is using a CFRM CDS that was formatted for message-based processing.

A CFRM CDS formatted for message-based processing cannot be brought into use by a sysplex while there are one or more down-level systems, z/OS V1R7or lower, in the sysplex.

Important: Pre-z/OS V1R8 systems cannot use the new formatted CFRM CDS. Any older system who tries to join the sysplex during the IPL will fail.

For example, if a sysplex is using a new formatted CDS and there is one z/OS V1R7 system trying to join the sysplex, this z/OS V1R7 system receives the error messages shown in Figure 4-7 on page 77 during system initialization and cannot join the sysplex.

We recommend that you start using the message-based protocol when you are sure that there will be no fallback to pre-z/OS V1R8 systems in any image in the sysplex. Because in order to go back to the old CDS, you need to do a sysplex-wide IPL. But after you have started using the message-based protocol once, you always have a chance to go back to the policy-based protocol any time you want. In order to go back to the policy-based protocol, you only need to use the SETXCF STOP,MSGBASED command from any system in the sysplex.

```
SETXCF CPL,TYPE=CFRM,ACOUPLE=SYS1.XCF.CFRM10
IXC309I SETXCF COUPLE,ACOUPLE REQUEST FOR CFRM WAS ACCEPTED
IXC260I ALTERNATE COUPLE DATA SET REQUEST FROM SYSTEM
SC75 FOR CFRM IS NOW BEING PROCESSED.
DATA SET:  SYS1.XCF.CFRM10
IEF196I IEF237I 813C ALLOCATED TO SYS00063
*IXC248E COUPLE DATA SET
SYS1.XCF.CFRM10 ON VOLSER BH5CD2
FOR CFRM MAY BE IN USE BY ANOTHER SYSPLEX.
*008 IXC247D REPLY U TO ACCEPT USE OR D TO DENY USE OFTHE COUPLE DATA SET
FOR CFRM.
R 8,U
IEE600I REPLY TO 008 IS;U
IEF196I IEF237I 813C ALLOCATED TO SYS00089
IXC520I SYSTEM SC74 NOT USING COUPLE DATA SET FOR CFRM
REASON: ALTERNATE COUPLE DATA SET HAS THE WRONG VERSION
IXC255I UNABLE TO USE DATA SET
SYS1.XCF.CFRM10
AS THE ALTERNATE FOR CFRM: REJECTED BY CFRM
IXC250I ALTERNATE COUPLE DATA SET REQUEST FAILED FOR DATA SET
SYS1.XCF.CFRM10 FOR CFRM:
CONSISTENCY CHECKING FAILED FOR THE NEW ALTERNATE DATA SET
IEF196I IEF285I   SYS1.XCF.CFRM10                              KEPT
IEF196I IEF285I   VOL SER NOS= BH5CD2.
IXC261I ALTERNATE COUPLE DATA SET REQUEST REJECTED
BY SYSTEM SC74 FOR CFRM
DATA SET NAME:  SYS1.XCF.CFRM10
IEF196I IEF285I   SYS1.XCF.CFRM10                              KEPT
IEF196I IEF285I   VOL SER NOS= BH5CD2.
IXC253I ALTERNATE COUPLE DATA SET
SYS1.XCF.CFRM10 FOR CFRM
IS BEING REMOVED BECAUSE OF A SETXCF COUPLE,ACOUPLE OPERATOR COMMAND
DETECTED BY SYSTEM SC75
IXC263I REMOVAL OF THE ALTERNATE COUPLE DATA SET
SYS1.XCF.CFRM10 FOR CFRM IS COMPLETE
```

Figure 4-7 Messages IXC520I and IXC255I during process of enabling new CDS in a mixed sysplex

Command to switch couple data sets

When a command to switch to a new CFRM CDS fails, message IXC520I is issued. Figure 4-8 on page 78 is an example of a sysplex with two images, one image, SC75, running z/OS V1R8 and a second image running z/OS V1R7. In this sysplex, a command was issued to switch the CFRM couple data set to the one formatted in z/OS V1R8 with the following statement:

```
ITEM NAME(MSGBASED) NUMBER(1)
```

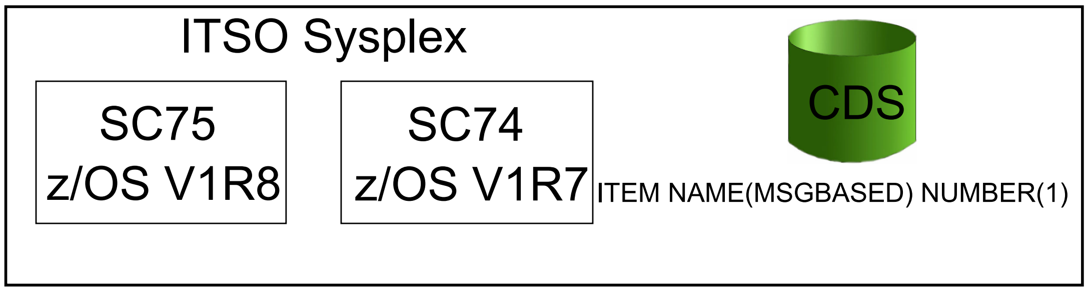


Figure 4-8 Sysplex with mixed z/OS levels

The command failed with message IXC520I, as shown in Figure 4-9. In order to use a new formatted version of CFRM couple data set to use the message-based protocol, there should not be any pre-z/OS V1R8 systems in the sysplex.

```
SETXCF CPL,TYPE=CFRM,ACOUPLE=SYS1.XCF.CFRM10
IXC309I SETXCF COUPLE,ACOUPLE REQUEST FOR CFRM WAS ACCEPTED
IXC260I ALTERNATE COUPLE DATA SET REQUEST FROM SYSTEM
SC75 FOR CFRM IS NOW BEING PROCESSED.
DATA SET:  SYS1.XCF.CFRM10

IXC520I SYSTEM SC74 NOT USING COUPLE DATA SET FOR CFRM
REASON: ALTERNATE COUPLE DATA SET HAS THE WRONG VERSION
IXC255I UNABLE TO USE DATA SET
SYS1.XCF.CFRM10
AS THE ALTERNATE FOR CFRM: REJECTED BY CFRM
IXC250I ALTERNATE COUPLE DATA SET REQUEST FAILED FOR DATA SET
SYS1.XCF.CFRM10 FOR CFRM:
CONSISTENCY CHECKING FAILED FOR THE NEW ALTERNATE DATA SET
IEF196I IEF285I   SYS1.XCF.CFRM10
IEF196I IEF285I   VOL SER NOS= BH5CD2.
IXC261I ALTERNATE COUPLE DATA SET REQUEST REJECTED
BY SYSTEM SC74 FOR CFRM
DATA SET NAME:  SYS1.XCF.CFRM10
*IXC267E PROCESSING WITHOUT AN ALTERNATE
COUPLE DATA SET FOR CFRM.
```

Figure 4-9 Message IXC520I from a PSWITCH command in a sysplex with lower-level systems

New CFRM format

With the new CFRM format, the size of subrecord IXCLOSUP is increased. Therefore, once an installation brings into use a CFRM CDS formatted with support for the message-based protocol, it cannot revert to a CDS formatted without this support without a sysplex-wide IPL. An attempt to switch to the old CDS will fail with error message IXC255I, as shown in Figure 4-10 on page 79.

```
SETXCF CPL,TYPE=CFRM,ACOUPLE=SYS1.XCF.CFRM01
IXC309I SETXCF COUPLE,ACOUPLE REQUEST FOR CFRM WAS ACCEPTED
IXC260I ALTERNATE COUPLE DATA SET REQUEST FROM SYSTEM
SC75 FOR CFRM IS NOW BEING PROCESSED.
DATA SET:  SYS1.XCF.CFRM01
IEF196I IEF237I 803C ALLOCATED TO SYS00064
IXC255I UNABLE TO USE DATA SET
SYS1.XCF.CFRM01
AS THE ALTERNATE FOR CFRM:
ALLOWABLE SIZE OF IXCLOSUP RECORDS IS LESS THAN CURRENT PRIMARY
RELEVANT CFRM COUPLE DATA SET FORMAT INFORMATION
PRIMARY
  FORMAT KEYWORDS: POLICY(10) CF(8) STR(512) CONNECT(32)
                   SMREBLD(1) SMDUPLEX(1) MSGBASED(1)
ALTERNATE
  FORMAT KEYWORDS: POLICY(10) CF(8) STR(512) CONNECT(32)
                   SMREBLD(1) SMDUPLEX(1)
IXC250I ALTERNATE COUPLE DATA SET REQUEST FAILED FOR DATA SET
SYS1.XCF.CFRM01 FOR CFRM:
CONSISTENCY CHECKING FAILED FOR THE NEW ALTERNATE DATA SET
IEF196I IEF285I   SYS1.XCF.CFRM01
IEF196I IEF285I   VOL SER NOS= BH5CD1.
```

Figure 4-10 Message IXC255I issued during an attempt to switch back to an old CDS

Switch back to policy-based protocol

If you decide to go back to the old policy-based protocol at any time, use the SETXCF STOP,MSGBASED command. Once this command is executed, transition from message-based to policy-based occurs. The CFRM CDS that supports the message-based protocol will remain in use by the sysplex, as shown in Figure 4-11. This transition has no impact on any processing in the sysplex. This command is implemented to do a fallback since there is no way to go back to old non-message-based-capable CDS other than a sysplex-wide IPL.

```
SETXCF STOP,MSGBASED
IXC548I CFRM EVENT MANAGEMENT ENVIRONMENT UPDATED
EVENT MANAGEMENT PROTOCOL: POLICY-BASED
REASON FOR CHANGE: SETXCF COMMAND
TRANSITION SEQUENCE NUMBER: 00000008
TRANSITION TIME: 05/25/2006 16:39:24.815160
IXC547I THE SETXCF STOP,MSGBASED REQUEST COMPLETED
```

Figure 4-11 Transition from message-based to policy-based protocol using the SETXCF command

Unplanned system outages

In the event of an unplanned outage in a system that was the event manager, the first system that updates the subrecord will be the new event manager system in the sysplex. Message IXC548I is issued indicating the event manager system is changed, as shown in Figure 4-12 on page 80. In this example, the event manager was system SC74. This system was reset and system SC75 became the event manager.

```
IEE600I REPLY TO 003 IS;DOWN
IXC101I SYSPLEX PARTITIONING IN PROGRESS FOR SC74 REQUESTED BY XCF
REASON: SYSTEM STATUS UPDATE MISSING
IEA257I CONSOLE PARTITION CLEANUP IN PROGRESS FOR SYSTEM SC74.
IXC307I STOP PATHOUT REQUEST FOR STRUCTURE IXC_DEFAULT_4
        LIST 8 TO COMMUNICATE WITH SYSTEM SC74 COMPLETED
        SUCCESSFULLY: SYSPLEX PARTITIONING OF REMOTE SYSTEM
IXC548I CFRM EVENT MANAGEMENT ENVIRONMENT UPDATED
EVENT MANAGEMENT PROTOCOL: MESSAGE-BASED
REASON FOR CHANGE: FAILURE OF MANAGING SYSTEM
TRANSITION SEQUENCE NUMBER: 00000004
TRANSITION TIME: 05/31/2006 10:35:17.631652
MANAGER SYSTEM NAME: SC75
MANAGER SYSTEM NUMBER: 01000043
IXC307I STOP PATHOUT REQUEST FOR STRUCTURE IXC_DEFAULT_1
        LIST 8 TO COMMUNICATE WITH SYSTEM SC74 COMPLETED
        SUCCESSFULLY: SYSPLEX PARTITIONING OF REMOTE SYSTEM
*$HASP493 JES2 MEMBER-SC74 RESTART IS IN PROGRESS
$HASP406 SYSLOG   WAS EXECUTING
$HASP406 INIT     WAS EXECUTING
IXL030I CONNECTOR STATISTICS FOR LOCK STRUCTURE ISGLOCK,
CONNECTOR ISGLOCK#SC75:
      00010004
      00000000 00000000 00000000 00000000
      00000000 00000000 00000000 00000000
```

Figure 4-12 Event manager system changed as a result of previous event manager system outage

Planned system outage

When a system outage is planned in an event manager system, you have the opportunity to choose the new event manager system using the SETXCF START,MSGBASED command on the desired system, as shown in the example on system SC74. First, you must stop the message-based protocol by using the SETXCF STOP,MSGBASED command from any system in the sysplex—system SC75 in our example . Message IXC548I is issued indicating the event manager system is changed, as shown in Figure 4-13 on page 81. If these commands are not used and the event manager system is removed from the sysplex, the same rules as an unplanned outage are applied, where the first system in the sysplex that updates the subrecord becomes the new event manager. In this case, message `IXC548I` is issued, as shown in Figure 4-12.

```
SC75 00000290  SETXCF STOP,MSGBASED
         00000090  IXC548I CFRM EVENT MANAGEMENT ENVIRONMENT UPDATED
     893 00000090  EVENT MANAGEMENT PROTOCOL: POLICY-BASED
     893 00000090  REASON FOR CHANGE: SETXCF COMMAND
     893 00000090  TRANSITION SEQUENCE NUMBER: 00000008
     893 00000090  TRANSITION TIME: 05/25/2006 16:39:24.815160
         00000090  IXC547I THE SETXCF STOP,MSGBASED REQUEST COMPLETED
SC74 00000290  SETXCF START,MSGBASED
         00000090  IXC548I CFRM EVENT MANAGEMENT ENVIRONMENT UPDATED
     472 00000090  EVENT MANAGEMENT PROTOCOL: MESSAGE-BASED
     472 00000090  REASON FOR CHANGE: SETXCF COMMAND
     472 00000090  TRANSITION SEQUENCE NUMBER: 00000009
     472 00000090  TRANSITION TIME: 05/25/2006 16:43:36.485559
     472 00000090  MANAGER SYSTEM NAME: SC74
     472 00000090  MANAGER SYSTEM NUMBER: 0200003A
TEMEL    00000090  IXC547I THE SETXCF START,MSGBASED REQUEST COMPLETED
```

Figure 4-13 Message as a result of changing the event manager system manually

4.5 Performance comparisons

By changing from policy-based to message-based protocol, Parallel Sysplex installations with a large number of CF structures or CF structure connectors should see a major improvement in sysplex availability (reduced recovery times), especially during occurrences of a system failure where connection cleanups are done, or CF recoveries where CF structure rebuild and duplexing failovers are done.

For normal day processes, since this protocol also affects any structure rebuild process, except signalling structures, all structure rebuild processing, including that triggered by the SETXCF START, REALLOCATE command, is improved.

Parallel Sysplex installations that have only two images in the sysplex can also benefit from this message-based protocol. Although, in these Parallel Sysplexes, both images need to update the CFRM CDS for event management. According to the new protocol, these updates are not done at the same time. However, there is still minimized CFRM CDS contention in these installations.

4.5.1 Failover performance examples

Following are two duplexing failover performance test results with different numbers of systems in a sysplex having different numbers of structures.

In Figure 4-14 on page 82, test results can be found about duplexing performance failover tests in a sysplex with 4 systems using 16, 72, and 144 structures. The amount of time to process the failover is dramatically reduced, as shown in both examples.

As you can see, failover times are significantly reduced when using the message-based protocol that is available with z/OS V1R8.

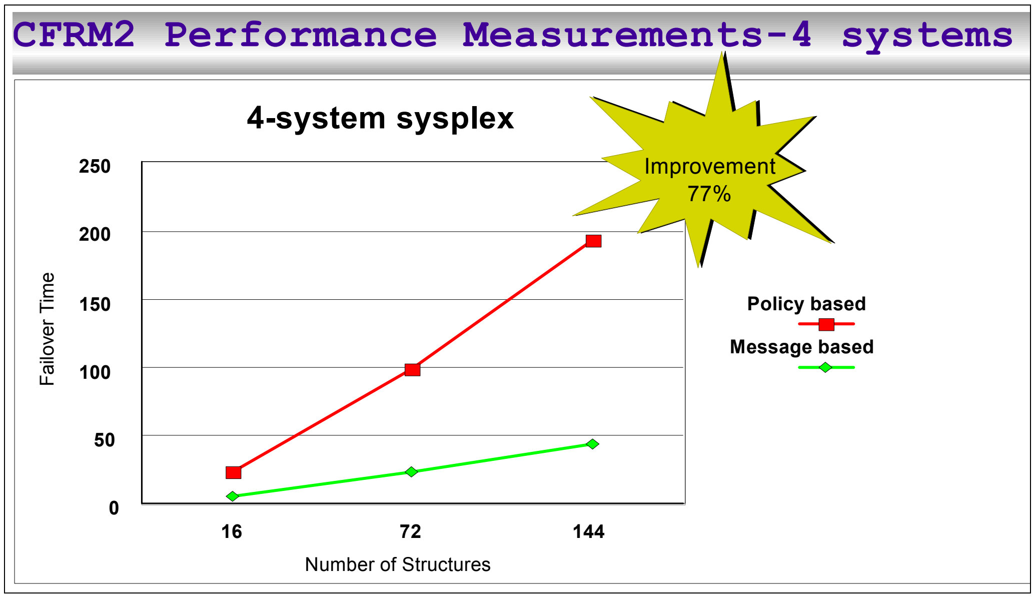


Figure 4-14 Duplexing failover performance comparison of two protocols in a 4-system sysplex

In Figure 4-15 test results are shown about duplexing performance failover tests in a sysplex with 12 systems using 24, 72, and 144 structures.

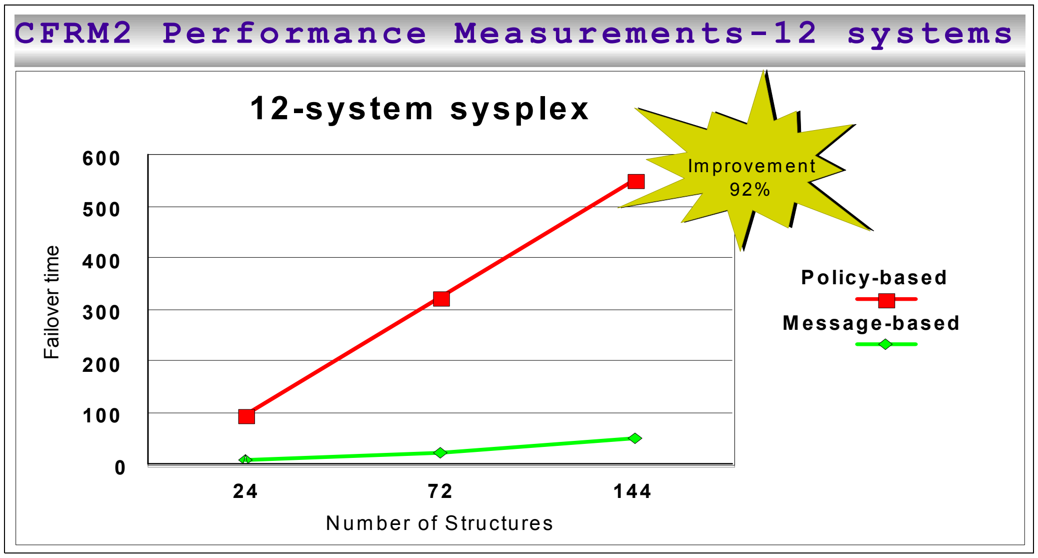


Figure 4-15 Duplexing failover performance comparison of two protocols in 12-system sysplex

4.6 Messages changed

In this section we introduce some new messages and some updated ones not previously shown in the chapter.

Message IXC358I

When the D XCF,CPL,TYPE=CFRM command is issued, this message is changed to show the CDS formatting information; see Figure 4-16 on page 83.

```
IXC358I  14.44.05  DISPLAY XCF 939
CFRM COUPLE DATA SETS
PRIMARY    DSN: SYS1.XCF.CFRM10
           VOLSER: BH5CD2     DEVN: 813C
           FORMAT TOD         MAXSYSTEM
           05/18/2006 14:34:55       2
           ADDITIONAL INFORMATION:
            FORMAT DATA
             POLICY(10) CF(8) STR(512) CONNECT(32)
             SMREBLD(1) SMDUPLEX(1) MSGBASED(1)
CFRM IN USE BY ALL SYSTEMS
```

Figure 4-16 IXC358I indicates CFRM CDS format information including msgbased

Message IXC359I

This message is changed to show summary information about the protocol being used in the sysplex as a result of the D XCF,STR= command; see Figure 4-17.

```
D XCF,STR
IEF196I IEF237I D501 ALLOCATED TO SYS00068
IXC359I  17.22.59  DISPLAY XCF 339
STRNAME             ALLOCATION TIME   STATUS
ISGLOCK           05/26/2006 17:29:48 ALLOCATED
ISTGENERIC        05/19/2006 11:42:34 ALLOCATED
IXC_DEFAULT_1     05/26/2006 17:29:49 ALLOCATED
IXC_DEFAULT_2     05/26/2006 17:29:48 ALLOCATED
IXC_DEFAULT_3     05/26/2006 17:29:49 ALLOCATED
IXC_DEFAULT_4     05/26/2006 17:29:48 ALLOCATED
SYSIGGCAS_ECS     05/26/2006 17:30:22 ALLOCATED
SYSTEM_OPERLOG    05/23/2006 09:08:13 ALLOCATED
SYSZWLM_WORKUNIT 05/26/2006 17:30:22 ALLOCATED
SYSZWLM_991E2094 05/26/2006 17:30:23 ALLOCATED
EVENT MANAGEMENT: MESSAGE-BASED MANAGER SYSTEM NAME:  SC74
IEF196I IEF285I   SYS1.MIGLIB
IEF196I IEF285I   VOL SER NOS= Z18RA1.
```

Figure 4-17 IXC359I indicating summary information about the protocol

Message IXC360I

This message is changed to show detailed information about protocol usage of the structure that is being displayed with the D XCF,STR,STRNAME= command; see Figure 4-18 on page 84.

```
D XCF,STR
IEF196I IEF237I D501 ALLOCATED TO SYS00068
IXC359I  17.22.59  DISPLAY XCF 339
STRNAME           ALLOCATION TIME   STATUS
ISGLOCK         05/26/2006 17:29:48 ALLOCATED
ISTGENERIC      05/19/2006 11:42:34 ALLOCATED
IXC_DEFAULT_1   05/26/2006 17:29:49 ALLOCATED
IXC_DEFAULT_2   05/26/2006 17:29:48 ALLOCATED
IXC_DEFAULT_3   05/26/2006 17:29:49 ALLOCATED
IXC_DEFAULT_4   05/26/2006 17:29:48 ALLOCATED
SYSIGGCAS_ECS   05/26/2006 17:30:22 ALLOCATED
SYSTEM_OPERLOG  05/23/2006 09:08:13 ALLOCATED
SYSZWLM_WORKUNIT 05/26/2006 17:30:22 ALLOCATED
SYSZWLM_991E2094 05/26/2006 17:30:23 ALLOCATED
EVENT MANAGEMENT: MESSAGE-BASED MANAGER SYSTEM NAME:  SC74
```

Figure 4-18 IXC360I indicating which protocol is being used by that structure

4.7 CFRM site awareness and GDPS enhanced recovery

Coupling Facility structure duplexing (CF duplexing) is designed to address structure failures, CF failures, or loss of connectivity failures between systems and CFs and between CFs. When you use CF duplexing, the system maintains a duplexed copy of a structure in a remote Coupling Facility, so that in the event of a failure, a viable copy remains available for the application.

CFRM initiates a duplexing failover when z/OS cross-system extended services (XES) detects a loss of connectivity error that affects a duplexed CF structure. When duplexing failover is initiated, CFRM stops CF duplexing, keeps one structure instance and deletes the other instance. Which instance is kept and which is deleted depends on the type of the first failure observed by XES.

Loss of connectivity errors may be the first sign of a site failure. In a Geographically Dispersed Parallel Sysplex™ (GDPS®) environment, z/OS XES might detect the error before GDPS does, and decide to delete a structure inconsistently with the policy defined to GDPS (or any other recovery manager program). For example, in the FREEZE=STOP freeze policy it is desired to keep the structure at the recovery site, but a loss of connectivity error due to a site failure might result in the structure instance in the recovery site being deleted, instead of the structure instance in the primary site.

Because the duplexed CF structures are not necessarily available at the recovery site, GDPS does not allow duplexed structures to be used in disaster recovery operations. Without CF structures, log-based methods are needed for disaster recovery. Log-based recovery can be time consuming, and application availability or performance may suffer during this time.

4.7.1 SITE keyword in the CFRM policy

To cope with the problem described above, z/OS V1R8 provides *site awareness* to the CFRM policy. Site awareness means that CFRM is aware of which CFs exist in every site, using a new keyword in the CFRM policy. This enables CFRM to make duplexing failover decisions consistent with the recovery manager decisions (failover to a recovery site). The new support guarantees that the duplexed structure is available in the recovery site, allowing GDPS to use it for disaster recovery operations, which saves the need to perform log-based recovery, and shortens recovery time.

CFRM site awareness is enabled by specifying the `SITE` keyword in the CFRM policy. For each CF definition, you can specify whether it resides in `SITE(SITE1)` or in `SITE(SITE2)`. When specified, site information along with status information from GDPS or another recovery manager, is used in duplexing failover decisions to keep the structure instance at the recovery site. `SITE` changes take effect when the policy is activated.

Figure 4-19 shows an example of using the `SITE` keyword in a CFRM policy.

```
DATA TYPE(CFRM)

  DEFINE POLICY NAME(POLICY1)

  CF NAME(FACIL01) SITE(SITE1)
    TYPE(002084) MFG(IBM) PLANT(EN) SEQUENCE(111111111111)
    PARTITION(0) CPCID(00)

  CF NAME(FACIL02) SITE(SITE2)
    TYPE(002084) MFG(IBM) PLANT(EN) SEQUENCE(222222222222)
    PARTITION(0) CPCID(00)

  STRUCTURE NAME(LIST_01) SIZE(150000)
    DUPLEX(ENABLED)
    PREFLIST(FACIL01,FACIL02)
```

Figure 4-19 Using the SITE keyword in a CFRM policy

Implementing site awareness

Site awareness must be installed on all systems in the sysplex. If there are systems running earlier releases of z/OS in your sysplex, APAR OA11719 is required to make them support the new function.

Normally, the IXCMIAPU administrative data utility rejects requests to define a CFRM policy having CFs on the same processor associated with different sites. However, such a policy definition may be required for the purpose of testing CF duplexing failover or GDPS. APAR OA15340 will allow you to define CFs in a single processor associated with different sites.

Site information is also displayed in the output of operator commands, as shown in Figure 4-20.

```
D XCF,CF
IXC361I  11.42.55  DISPLAY XCF 784
CFNAME     COUPLING FACILITY              SITE
CF7A       002094.IBM.02.00000002991E     SITE1
           PARTITION: 0D   CPCID: 00
CF7B       002094.IBM.02.00000002991E     SITE2 - RECOVERY SITE
           PARTITION: 1D   CPCID: 00
```

Figure 4-20 Output of the D XCF,CF operator command

GDPS enhanced recovery

GDPS 3.3 uses duplexed structures at the recovery site for restarted workload, avoiding time consuming log-based recovery and improving application availability and consistency. GDPS uses the Coupling Facility configuration management service, IXCCFCM, to inform CFRM that it is an active recovery manager. GDPS also uses IXCCFCM to inform CFRM which site is the recovery site. The IXCCFCM service provides an interface to allow GDPS or other recovery managers to provide recovery manager status to CFRM to assist in making the appropriate CF duplexing failover decisions in such configurations. IXCCFCM can be used by a recovery manager to inform CFRM that it is active and that the recovery site is SITE2, as follows:

```
IXCCFCM RMACTIVE,RECOVERYSITE=‘SITE2’
```

Once this is done, CFRM is able to use GDPS enhanced recovery for duplexing failover decisions.

Note: Existing CF duplexing failover decisions are unaffected unless the recovery manager is active and site information is in the CFRM active policy.

RMF enhancements

In this chapter we describe the following enhancements to RMF in z/OS V1R8:

- Enable the overview condition for duration processing
- Remove the sort exit ERBPPSRT
- RMF support for real storage > 128 GB
- HyperPAV support
- RMF support for ZIIP
- DDS enhancements - OMEGAMON integration
- RMF Spreadsheet Reporter enhancements

5.1 Introduction to RMF z/OS V1R8

Figure 5-1 shows the RMF components and elements that are part of RMF products. Many enhancements have been made in the past few years. With z/OS V1R8, RMF has several enhancements for its components in order to achieve the required qualities of service and also to support enhancements made to other subsystems. In this chapter, we describe these enhancements in detail.

The RMF Spreadsheet Reporter, a component of RMF that runs on the workstation, converts Postprocessor listings and Overview records into spreadsheets. At your workstation, independent of the systems you are monitoring, you can use one of several familiar spreadsheet applications to manipulate the data as you wish. In addition, the Spreadsheet Reporter provides sample macros to help you in presenting and analyzing performance data at a glance.

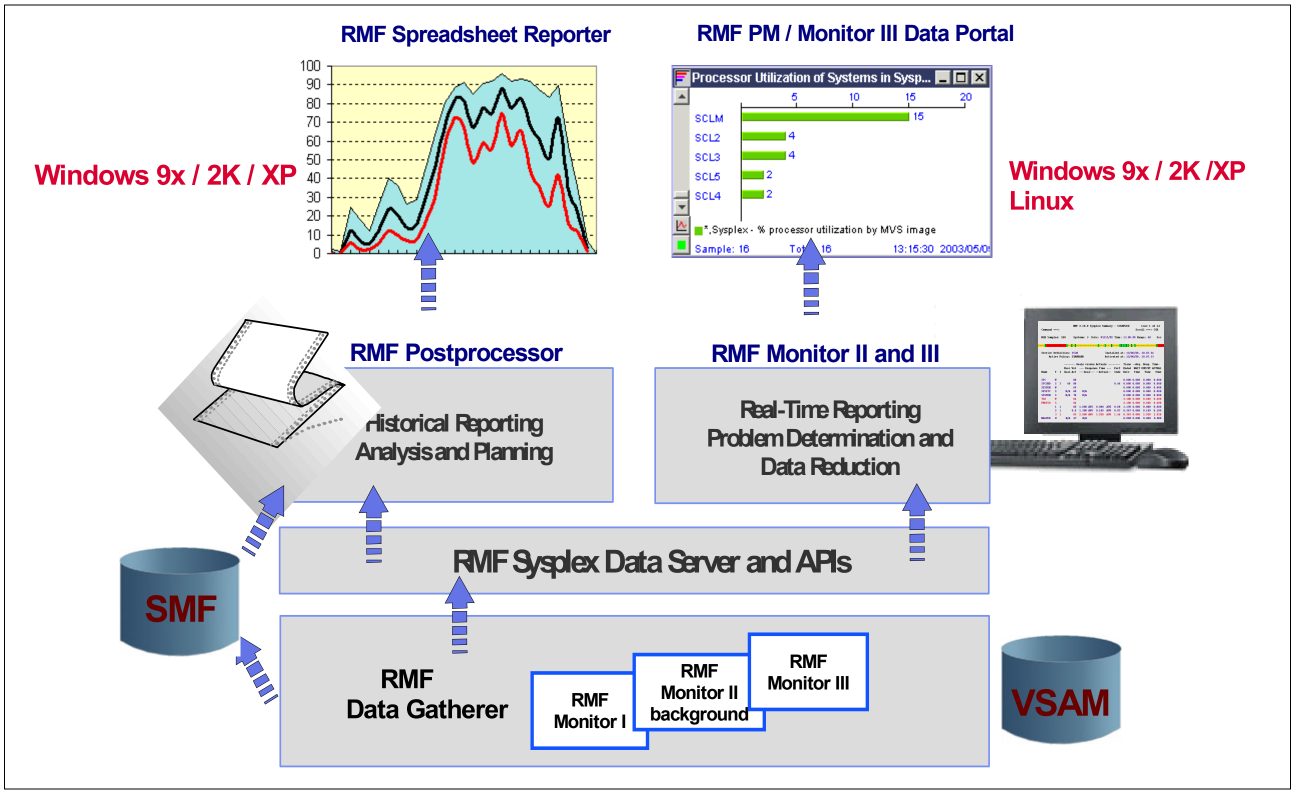


Figure 5-1 RMF components

5.2 RMF enhancements in z/OS V1R8

The following enhancements are described in this chapter:

- Enable overview conditions for duration processing

  The RMF Postprocessor basically supports the aggregation of multiple intervals to so-called duration periods. While one single interval is limited to a maximum of 60 minutes, duration periods up to 100 hours are possible. A duration period is specified by the DINTV(hhmm) Postprocessor control statement.

  With the exploitation of overview control statements, you can generate reports that contain multiple intervals. This is especially valuable for trend analysis.

However, today the DINTV control is ignored for Overview processing. In other words, the interval length for Overview data is automatically determined by the length of the gathering interval, which is typically 15 or 30 minutes. Therefore, the interval-oriented Overview Reports are not well suited for the analysis of longer periods such as multiple days or weeks.

This limitation is removed in z/OS v1R8. The RMF Postprocessor is enhanced to honor duration intervals DINTV(hhmm) for Overview processing in addition to using standard intervals. Now, you can produce trend reports over long periods of time by means of overview control statements.

- RMF support for real storage greater than 128 GB

  On a pre-z/OS V1R8 system, the highest system UIC gets calculated every 10 seconds. The UIC represents the oldest frame of all IN address spaces. To find the oldest, RSM needs to check the reference bit for each pageable frame that belongs to an address space on the IN queue. This is a very CPU-time consuming process. If many large address spaces are active, this process can take more than 0.2 seconds. This UIC update processing is even done when the available frame queue has a lot of available frames.

  With z/OS V1R8, the UIC algorithm changes. The UIC is defined as a single walk through the entire real storage. The storage is split into logical segments. For each of these segments, a UIC is calculated. These logical segment UICs are used by the SRM to calculate a minimum, a maximum, and a current UIC. This new UIC can vary from 0 to 65535 (which is 18 hours).

- HyperPAV support

  HyperPAV is the name of a new feature of Parallel Access Volumes (PAV) that reduces the number of PAV-aliases needed per logical subsystem (LSS) by an order of magnitude and still maintain optimal response times.

  This is accomplished by no longer statically binding PAV-aliases to PAV-bases and adjusting the bindings via the Workload Manager. Instead, in HyperPAV mode, PAV-aliases are bound to PAV-bases only for the duration of a single I/O operation. This reduces the number of aliases required per LSS significantly.

- Monitor I and III channel data gathering changes

  Today, RMF measures channel path activity only at the end of a Monitor I interval by reading the counts from the channel path measurement blocks (CPMB). With each new channel type generation, the capacity and throughput of channels increases significantly, so that the CPMB counters may wrap more than once during an RMF interval. This may lead to invalid (too low) data in the Channel Path Activity report.

  To ensure correct channel path data in SMF records 73 in the future, RMF is increasing its measurement frequency for CPMB data by introducing cycle gathering in addition to mere interval gathering.

  The Monitor III channel data is currently obtained from Monitor II. With RMF for z/OS V1R8, Monitor III channel data gathering changes such that Monitor II channel data is no longer obtained and a new Monitor III channel data gatherer is introduced instead. The format and content of the Monitor III Channel Report does not change.

  The Monitor II Channel Report is used to report on channel data obtained from Monitor II in prior releases.

- RMF support for zIIPs

  RMF supports the IBM System z9 Integrated Information Processor (zIIP) by extending the Postprocessor CPU Activity Report, the Postprocessor Workload Activity report, the Monitor III CPC Capacity report, the Monitor III System Information Report, and the Monitor III Enclave report.

RMF distinguishes between general purpose CP and special purpose processors (zAAP and zIIP) where necessary and collects and reports about zIIP consumption and collects and reports about using and delay states for the zIIP.

RMF monitors the zIIP in the same manner as the zAAP. For consistent naming, RMF uses the terms AAP and IIP. That is, the previous term IFA that was used for the zAAP is replaced by the term AAP.

- OMEGAMON integration

  The RMF Distributed Data Server (DDS) runs in a separate address space and provides the API to the performance data contained in the Monitor III ISPF Reports. DDS data is exploited today by the RMF Performance Monitoring Java™ client as well as by the RMF CIM Provider. The OMEGAMON XE data collector also accesses DDS data starting with Coupling Facility statistics in z/OS V1R8. DDS uses internally the RMF Sysplex Data Server API ERB3XDRS. This function returns sysplex-wide data. Therefore, just one DDS instance is needed across the sysplex. In the past, the system where the DDS instance resided had to be selected manually.

- RMF Spreadsheet Reporter enhancements

  The Spreadsheet Reporter handles the changed RMF Postprocessor Reports. It is available as Version 5.2.1 shipped with APAR OA12865 and is available via the RMF home page. The Spreadsheet Reporter uses an FTP connection to the FTP server on the host, for example, to submit Postprocessor JCL and to download the Postprocessor Listing or Overview records. Because currently only active FTP is supported, some customers are not able to use the Spreadsheet Reporter FTP connection. Active FTP access may fail, because firewalls block that kind of access. Non-secure FTP connections may be restricted, due to security policy reasons on the customer site. Therefore, the Spreadsheet Reporter now supports passive FTP connection mode and SSL FTP connection.

5.3 Enable overview condition for duration processing

In z/OS V1R8, the postprocessor can use the value specified with the DINTV control statement for processing the Overview and Exception Reports. This allows you to produce trend reports over long time periods.

DINTV control statement

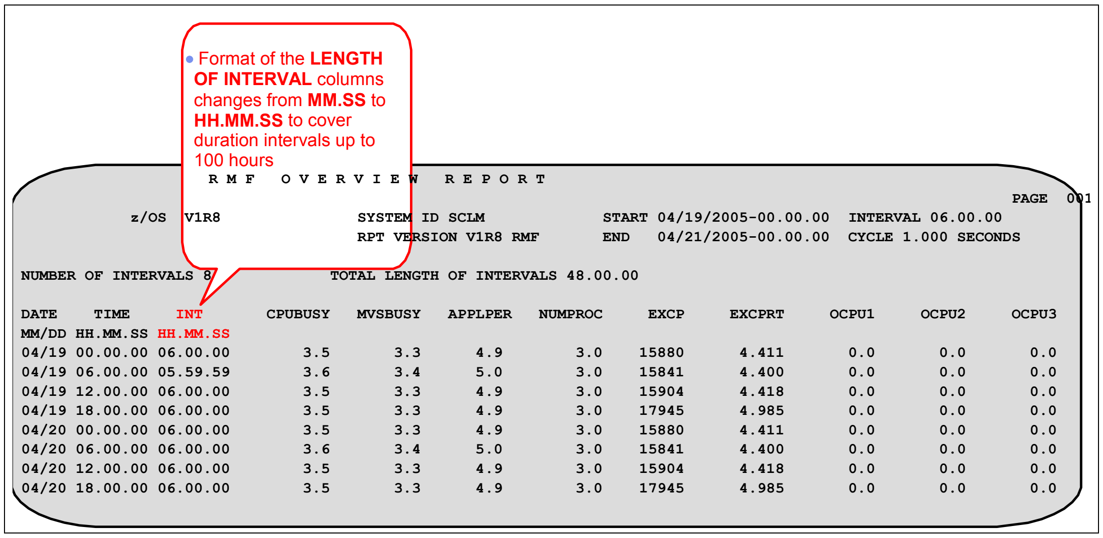
The DINTV control statement specifies that the Postprocessor is to generate duration reports and indicates the length of the duration interval.The duration interval is the length of time each report can cover and should be a multiple of the measurement interval. The syntax of the statement is:

```
DINTV(hhmm)
```

The maximum is 9960, which is equivalent to 100 hours.

Figure 5-2 on page 91 shows an Overview Report for a reporting period of two days with a duration interval length of 6 hours; DINTV(0600) was specified.

Attention: The format of the interval length in the Exception and Overview reports is changed from MM.SS to HH.MM.SS. The additional 3 spaces used for the interval length require a shift of the data columns by 3 spaces also. This change is done to cover duration intervals of up to 100 hours. For Exception reports, this is no problem. However, the Overview reports which provided up to 11 data columns lose one data column and contain only up to 10 data columns now.


Format of the LENGTH OF INTERVAL columns changes from MM.SS to HH.MM.SS to cover duration intervals up to 100 hours

```
                         R M F    O V E R V I E W    R E P O R T
                                                                                                  PAGE   001
            z/OS   V1R8                 SYSTEM ID SCLM              START 04/19/2005-00.00.00  INTERVAL 06.00.00
                                        RPT VERSION V1R8 RMF        END   04/21/2005-00.00.00  CYCLE 1.000 SECONDS

NUMBER OF INTERVALS 8                TOTAL LENGTH OF INTERVALS 48.00.00

DATE    TIME      INT        CPUBUSY   MVSBUSY   APPLPER   NUMPROC     EXCP     EXCPRT     OCPU1     OCPU2     OCPU3
MM/DD HH.MM.SS HH.MM.SS
04/19 00.00.00 06.00.00          3.5       3.3       4.9       3.0    15880      4.411       0.0       0.0       0.0
04/19 06.00.00 05.59.59          3.6       3.4       5.0       3.0    15841      4.400       0.0       0.0       0.0
04/19 12.00.00 06.00.00          3.5       3.3       4.9       3.0    15904      4.418       0.0       0.0       0.0
04/19 18.00.00 06.00.00          3.5       3.3       4.9       3.0    17945      4.985       0.0       0.0       0.0
04/20 00.00.00 06.00.00          3.5       3.3       4.9       3.0    15880      4.411       0.0       0.0       0.0
04/20 06.00.00 06.00.00          3.6       3.4       5.0       3.0    15841      4.400       0.0       0.0       0.0
04/20 12.00.00 06.00.00          3.5       3.3       4.9       3.0    15904      4.418       0.0       0.0       0.0
04/20 18.00.00 06.00.00          3.5       3.3       4.9       3.0    17945      4.985       0.0       0.0       0.0
```

Figure 5-2   RMF Overview Report showing a DINTV interval of 6

Important: It is recommended to specify, for duration reports, not only the reporting interval, but also the date (also if the SMF data set contains only records for those days you want to report on) because of performance reasons as well as for internal processing reasons.

5.3.1 Overview duration processing

The duration processor uses the suboptions that were specified with the REPORTS options to select the SMF records to be included for duration processing. For exception/overview duration processing, all SMF records that fall into the specified exception report time frame are considered to be candidates for exception/overview processing.

Recommendation/Restriction: Therefore, the REPORTS option should not be used at all or should be specified very carefully in order to avoid unwanted report output.

To demonstrate this, consider a specification for the device duration report for just one device by using the REPORTS(DEVICE(NMBR(2222))) control statement. Duration processing used to exclude all devices but the device 2222. This produced a duration device report just for this one device.

With the exception/overview duration support this is no longer true. For exception/overview duration processing the devices are no longer eliminated and you will get a duration device report for all devices even if the request was just one device number.

RTOD control statement

The RTOD control statement specifies the starting time and ending time of the reporting period for interval or duration reporting for each day included in the reporting period. The syntax of the statement is:

```
RTOD(hhmm,hhmm)
```

ETOD control statement

The ETOD control statement specifies the starting time and ending time of the reporting period for an Exception or Overview report for each day in the reporting period. The syntax of the statement is:

```
ETOD(hhmm,hhmm)
```

In z/OS V1R8, overview duration processing is based on the existing duration. This requires that the RTOD and ETOD control statements specify the same starting and ending time as soon as the DINTV option is specified. The postprocessor compares the RTOD and ETOD values and if they are different, RTOD values are replaced by ETOD values, as shown in Figure 5-3. When this occurs, message ERB469I is issued, as follows:

```
ERB469I – PPS: FOR EXCEPTION/OVERVIEW DURATION PROCESSING, THE RTOD VALUE IS
SET TO ETOD
```

```
MERALRMF JOB04634       <       .POSTRMF .MFPMSGDS>           Line 1 of 34
ERB103I PPS: OPTIONS IN EFFECT
ERB103I PPS:   NODELTA  -- DEFAULT
ERB103I PPS:   NOEXITS  -- DEFAULT
ERB103I PPS:   MAXPLEN(50)  -- DEFAULT
ERB103I PPS:   ETOD(0000,2400)  -- DEFAULT
ERB103I PPS:   STOD(0000,2400)  -- DEFAULT
ERB103I PPS:   PTOD(0000,2400)  -- DEFAULT
ERB103I PPS:   DINTV(0400)  -- SYSIN
ERB103I PPS:   RTOD(0800,2000)  -- SYSIN
ERB103I PPS:   OVW(APPLPER(APPLPER(POLICY)))  -- SYSIN
ERB103I PPS:   EXCEPT(AVGINRDY(AVGIARDY,GE,0))  -- SYSIN
ERB103I PPS:   EXCEPT(MVSBUSY(MVSBSY,GE,0))  -- SYSIN
ERB103I PPS:   EXCEPT(NUMPROC(NUMPROC,GE,0))  -- SYSIN
ERB103I PPS:   EXCEPT(LPARBUSY(CPUBSY,GE,0))  -- SYSIN
ERB103I PPS:   OVERVIEW(REPORT)  -- SYSIN
ERB103I PPS:   NOSUMMARY  -- SYSIN
ERB103I PPS:   SYSOUT(X)  -- SYSIN
ERB469I PPS: FOR EXCEPTION/OVERVIEW DURATION PROCESSING, THE RTOD
ERB469I PPS:     VALUE IS SET TO ETOD.
ERB103I PPS:   DATE(01011956,12312055) -- DEFAULT
```

Figure 5-3 Message ERB469I is issued when ETOD and RTOD are not equal

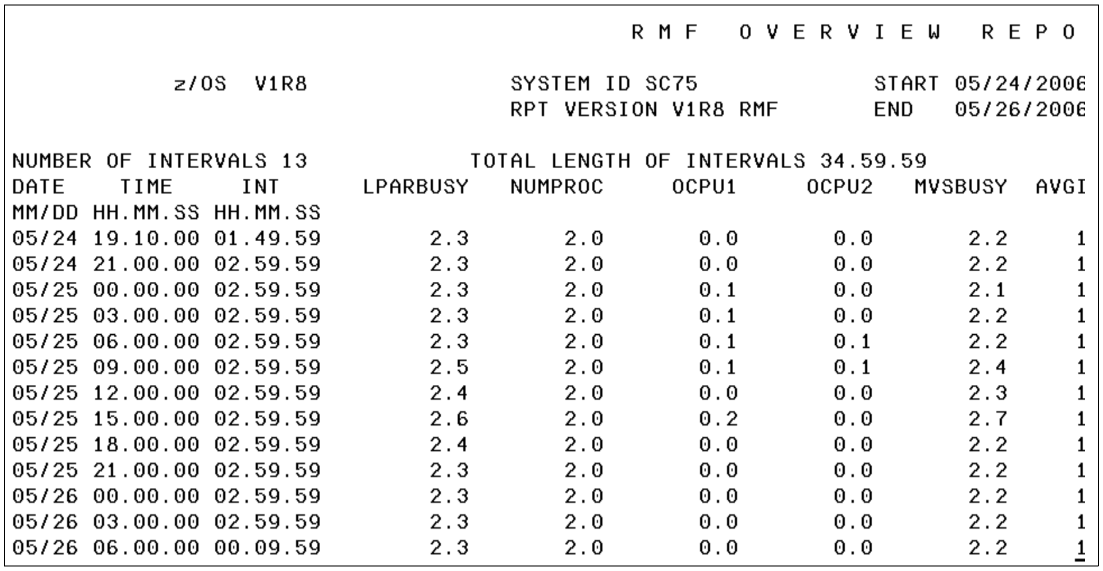
DINTV interval examples

In pre-z/OS V1R8 systems, the DINTV interval was ignored without creating a warning message. The increase in the interval does not cause any restriction in Exception reports but that is not the case in Overview reports. In pre-z/OS V1R8, Overview reports provided up to 11 data columns. With the increase in length, reports lose one data column and can contain up to 10 data columns in z/OS V1R8.

Figure 5-5 on page 93 shows an RMF Overview report with a duration interval length of 3 hours using the JCL statements in Figure 5-4 on page 93. In pre-z/OS V1R8 systems, the DINTV parameter was ignored without sending any warning message.

```
//POSTRMF  EXEC PGM=ERBRMFPP,REGION=32M
//MFPINPUT DD  DSN=&&SO,DISP=(OLD,DELETE)
//MFPMSGDS DD SYSOUT=*
//PPOVWREC DD SYSOUT=*
//SYSIN    DD  DDNAME=SREP
//SREP     DD  *
  SYSOUT(X)
  NOSUMMARY
  OVERVIEW(REPORT)
  OVW(LPARBUSY(CPUBSY))
  OVW(NUMPROC(NUMPROC))
  OVW(OCPU1(OCPU1))
  OVW(OCPU2(OCPU2))
  OVW(MVSBUSY(MVSBSY))
  OVW(AVGINRDY(AVGIARDY))
  OVW(APPLPER(APPLPER(POLICY)))
  OVW(AVGBATCH(AVGBATCH))
  OVW(AVGSQA(AVGSQA))
  OVW(AVGCSA(AVGCSAT))
  OVW(PYSCP(PBUSYL(SC75)))
  ETOD(0000,2400)
  RTOD(0000,2400)
  DINTV(0300)
/*
```

Figure 5-4 Sample Overview postprocessor step using DINTV



```
                                                     R M F   O V E R V I E W   R E P O

            z/OS  V1R8              SYSTEM ID SC75            START 05/24/2006
                                    RPT VERSION V1R8 RMF      END   05/26/2006

NUMBER OF INTERVALS 13           TOTAL LENGTH OF INTERVALS 34.59.59
DATE    TIME     INT      LPARBUSY   NUMPROC     OCPU1      OCPU2   MVSBUSY  AVGI
MM/DD HH.MM.SS HH.MM.SS
05/24 19.10.00 01.49.59        2.3        2.0       0.0        0.0       2.2     1
05/24 21.00.00 02.59.59        2.3        2.0       0.0        0.0       2.2     1
05/25 00.00.00 02.59.59        2.3        2.0       0.1        0.0       2.1     1
05/25 03.00.00 02.59.59        2.3        2.0       0.1        0.0       2.2     1
05/25 06.00.00 02.59.59        2.3        2.0       0.1        0.1       2.2     1
05/25 09.00.00 02.59.59        2.5        2.0       0.1        0.1       2.4     1
05/25 12.00.00 02.59.59        2.4        2.0       0.0        0.0       2.3     1
05/25 15.00.00 02.59.59        2.6        2.0       0.2        0.0       2.7     1
05/25 18.00.00 02.59.59        2.4        2.0       0.0        0.0       2.2     1
05/25 21.00.00 02.59.59        2.3        2.0       0.0        0.0       2.2     1
05/26 00.00.00 02.59.59        2.3        2.0       0.0        0.0       2.2     1
05/26 03.00.00 02.59.59        2.3        2.0       0.0        0.0       2.2     1
05/26 06.00.00 00.09.59        2.3        2.0       0.0        0.0       2.2     1
```

Figure 5-5 Sample Overview report using DINTV run in z/OS V1R8

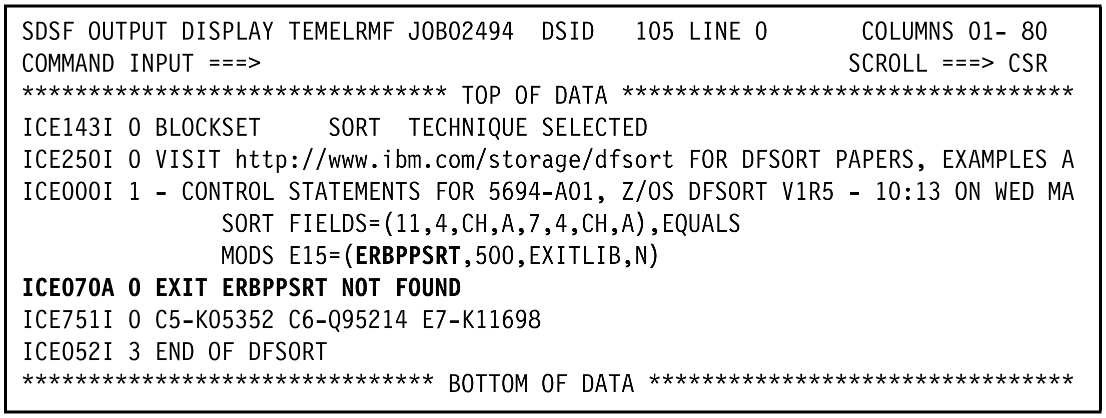
With z/OS V1R8, the Enqueue Activity report provides no duration processing. Thus, there is also no duration processing for OVW conditions based on SMF record type 77 (Enqueue Activity).

Restriction: The Enqueue Activity report does not support duration processing.

5.3.2 ERBPPSRT sort exit removed

The RMF postprocessor requires sorted RMF records as input. Previously, RMF provided two sort exits, ERBPPE15 and ERBPPE35, to be used when running the sort program. The previous sort exit, ERBPPSRT, did not follow the rules for sort exits and sometimes caused errors in Spreadsheet reports.

Starting with z/OS V1R8, the ERBPPSRT is no longer shipped. If you are running the RMF postprocessor job that still specifies ERBPPSRT as a sort exit, change the specification to use ERBPPE15 and ERBPPE35. If not, an ICE070A error message is received as shown in Figure 5-6.



```
SDSF OUTPUT DISPLAY TEMELRMF JOB02494  DSID   105 LINE 0       COLUMNS 01- 80
COMMAND INPUT ===>                                            SCROLL ===> CSR
******************************** TOP OF DATA **********************************
ICE143I 0 BLOCKSET     SORT  TECHNIQUE SELECTED
ICE250I 0 VISIT http://www.ibm.com/storage/dfsort FOR DFSORT PAPERS, EXAMPLES A
ICE000I 1 - CONTROL STATEMENTS FOR 5694-A01, Z/OS DFSORT V1R5 - 10:13 ON WED MA
              SORT FIELDS=(11,4,CH,A,7,4,CH,A),EQUALS
              MODS E15=(ERBPPSRT,500,EXITLIB,N)
ICE070A 0 EXIT ERBPPSRT NOT FOUND
ICE751I 0 C5-K05352 C6-Q95214 E7-K11698
ICE052I 3 END OF DFSORT
****************************** BOTTOM OF DATA ********************************
```

Figure 5-6 Message ICE070A when ERBPPSRT is used in the sort step

5.4 RMF support for real storage > 128 GB

With z/OS V1R8, real storage up to 4 TB is now supported. In order to achieve this support, some internal changes have been made related to usage of UIC values and calculations. RMF is also enhanced to support the new changes. RMF replaces the highest system UIC by the current system UIC in the following reports:

- Header area of any Monitor II report
- Monitor II SRCS in the Central Storage/Processor/SRM report
- Monitor II SPAG in the Paging Activity report
- Monitor III STORS in the Storage Delay Summary report
- Monitor III STORR in the Storage Resource Delays report
- Postprocessor Paging report in the Central Storage section

Monitor II and III reports displaying the UIC are also changed to handle a 5-digit value.

SMF record type 71

SMF record type 71 (paging activity) is extended. SMF record type 79 subtype 3 (storage/processor data) and subtype 4 (paging activity data) are changed.

The SMF record type 71 changes are handled consistently in RMF. Additional fields are added at the end of the paging data section. The postprocessor paging activity report formats AVG, MIN and MAX HIGH UIC values based on the new Current system UIC values SMF71UAC, SMF71ULC, and SMF71UHC instead of the old UIC fields SMF71ACA, SMF71LIC, and SMF71HIC. The highest possible value changes from 2540 to 65535. If SMF

71 records from a previous release are used as input, the old UIC fields are continued to be formatted.

RMF Overview report

In the RMF Overview report, there are two conditions related to UIC values. These are AVGHUIC and MXHUIC. AVGHUIC represents the average high UIC for central storage frames. AVGHUIC was kept in the SMF71ACA field in pre-z/OS V1R8 systems. In z/OS V1R8, this value is stored in the SMF71UAC field. The other condition is MXHUIC which represents the maximum high UIC for central storage frames. This value was driven from SMF71HIC in pre-z/OS V1R8. In z/OS V1R8, this value is stored in the SMF71UHC field.

The RMF postprocessor only formats the current UIC field (MCTCurSystemUIC). MCTMinSystemUIC and MCTMaxSystemUIC are fields in SMF type 71 and are not formatted in the report. The highest possible value changes from 2540 to 65535.

Monitor II now displays UIC values from 0 to 9999. Values greater than 9999 are displayed as 10K up to 65K.

Monitor III displays UIC values up to 65535.

5.5 HyperPAV support

HyperPAV support is the name of a new feature of parallel access volumes (PAV) that will reduce the number of PAV-aliases needed per logical subsystem (LSS) by an order of magnitude and still maintain optimal response times.

This is accomplished by no longer statically binding PAV-aliases to PAV-bases and adjusting the bindings via WLM. Instead, in HyperPAV mode, PAV aliases are bound to PAV-bases only for the duration of a single I/O operation. This will reduce the number of aliases required per LSS significantly. WLM will have no have control of PAV management other than sending priority values.

Important: The HyperPAV support is shipped as SPE with APAR OA12865.

5.5.1 RMF support for HyperPAV

In order to support HyperPAV in RMF, reports and data gathering processes are changed in the following areas:

- Device Activity
- IOQ Activity
- RMF Spreadsheet Reporter

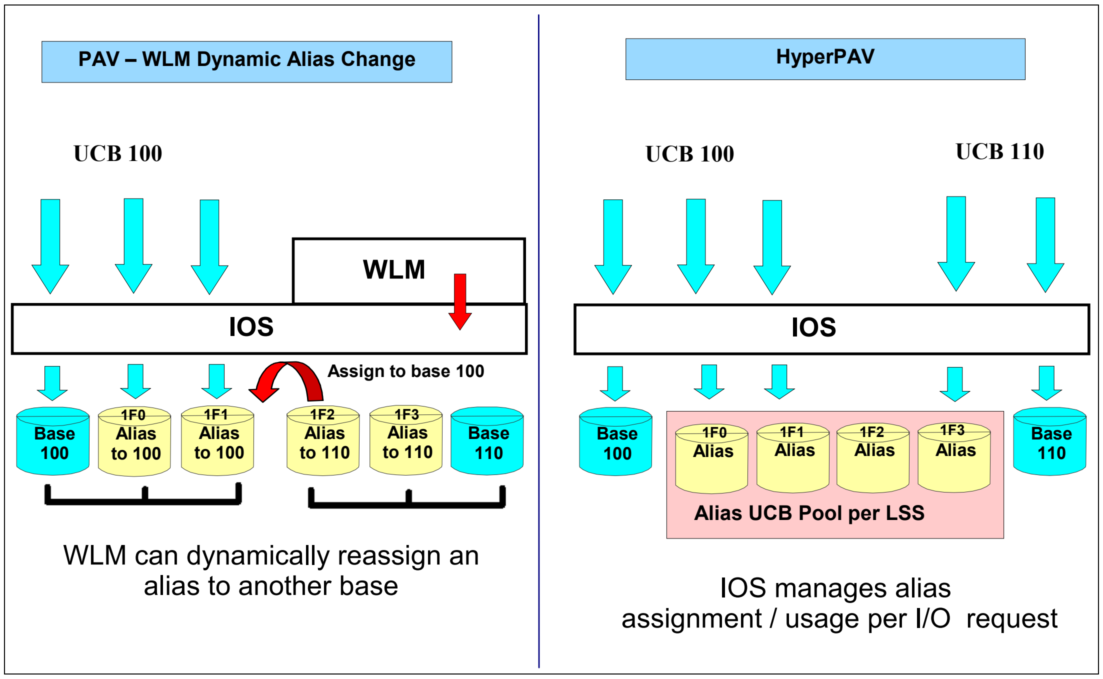
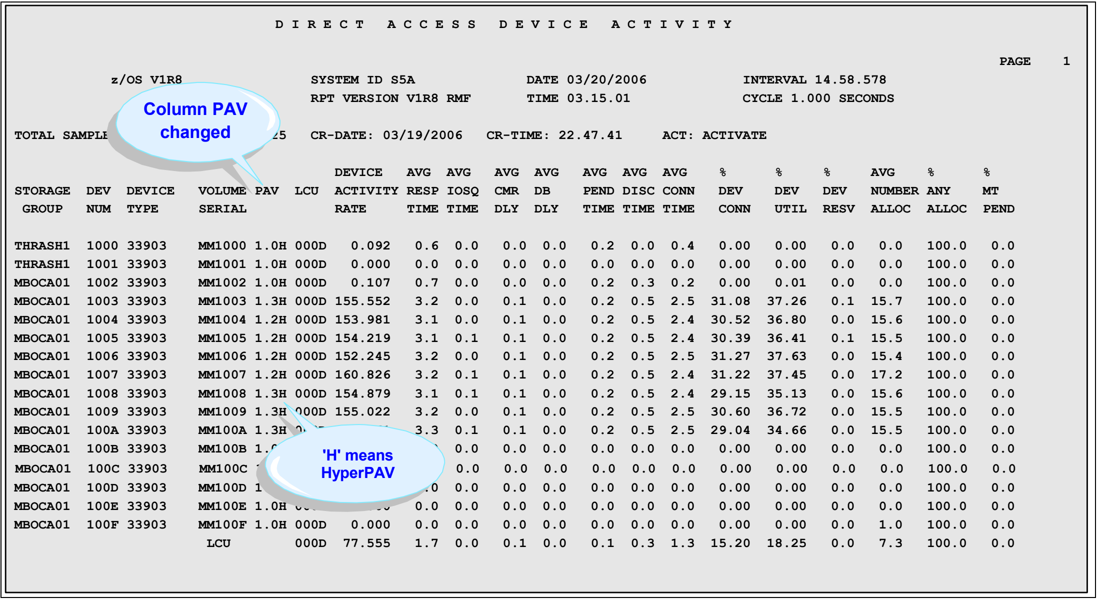


Figure 5-7 Overview of HyperPAV support

Monitor I-II-III Device Activity Report changes

The PAV column in the RMF Monitor I Device Activity Report now shows either a number of PAV aliases or an average number of HyperPAV aliases. The character ‘H for HyperPav’ next to the PAV values represents whether this is a PAV or HyperPAV type, as shown in Figure 5-8 on page 96. The Monitor II and III reports are also changed accordingly.

```
Average # of HPAV = (Accumulated # of HyperPAV devices)/ (Number of samples)
```



```
                              D I R E C T   A C C E S S   D E V I C E   A C T I V I T Y

                                                                                                        PAGE    1
        z/OS V1R8                        SYSTEM ID S5A            DATE 03/20/2006           INTERVAL 14.58.578
                                         RPT VERSION V1R8 RMF     TIME 03.15.01             CYCLE 1.000 SECONDS

TOTAL SAMPLE            25               CR-DATE: 03/19/2006    CR-TIME: 22.47.41      ACT: ACTIVATE

                                      DEVICE   AVG  AVG   AVG  AVG   AVG  AVG  AVG    %      %      %     AVG     %      %
STORAGE  DEV   DEVICE    VOLUME  PAV   LCU    ACTIVITY RESP IOSQ  CMR  DB    PEND DISC CONN   DEV    DEV    DEV   NUMBER  ANY    MT
 GROUP   NUM   TYPE      SERIAL               RATE     TIME TIME  DLY  DLY   TIME TIME TIME   CONN   UTIL   RESV  ALLOC   ALLOC  PEND

THRASH1  1000 33903    MM1000 1.0H 000D   0.092   0.6  0.0   0.0  0.0   0.2  0.0  0.4   0.00   0.00   0.0   0.0    100.0   0.0
THRASH1  1001 33903    MM1001 1.0H 000D   0.000   0.0  0.0   0.0  0.0   0.0  0.0  0.0   0.00   0.00   0.0   0.0    100.0   0.0
MBOCA01  1002 33903    MM1002 1.0H 000D   0.107   0.7  0.0   0.0  0.0   0.2  0.3  0.2   0.00   0.01   0.0   0.0    100.0   0.0
MBOCA01  1003 33903    MM1003 1.3H 000D 155.552   3.2  0.0   0.1  0.0   0.2  0.5  2.5  31.08  37.26   0.1  15.7    100.0   0.0
MBOCA01  1004 33903    MM1004 1.2H 000D 153.981   3.1  0.0   0.1  0.0   0.2  0.5  2.4  30.52  36.80   0.0  15.6    100.0   0.0
MBOCA01  1005 33903    MM1005 1.2H 000D 154.219   3.1  0.1   0.1  0.0   0.2  0.5  2.4  30.39  36.41   0.1  15.5    100.0   0.0
MBOCA01  1006 33903    MM1006 1.2H 000D 152.245   3.2  0.0   0.1  0.0   0.2  0.5  2.5  31.27  37.63   0.0  15.4    100.0   0.0
MBOCA01  1007 33903    MM1007 1.2H 000D 160.826   3.2  0.1   0.1  0.0   0.2  0.5  2.4  31.22  37.45   0.0  17.2    100.0   0.0
MBOCA01  1008 33903    MM1008 1.3H 000D 154.879   3.1  0.1   0.1  0.0   0.2  0.5  2.4  29.15  35.13   0.0  15.6    100.0   0.0
MBOCA01  1009 33903    MM1009 1.3H 000D 155.022   3.2  0.0   0.1  0.0   0.2  0.5  2.5  30.60  36.72   0.0  15.5    100.0   0.0
MBOCA01  100A 33903    MM100A 1.3H                3.3  0.1   0.1  0.0   0.2  0.5  2.5  29.04  34.66   0.0  15.5    100.0   0.0
MBOCA01  100B 33903    MM100B 1.                       0.0   0.0  0.0   0.0  0.0  0.0   0.00   0.00   0.0   0.0    100.0   0.0
MBOCA01  100C 33903    MM100C                          0.0   0.0  0.0   0.0  0.0  0.0   0.00   0.00   0.0   0.0    100.0   0.0
MBOCA01  100D 33903    MM100D 1                   .0   0.0   0.0  0.0   0.0  0.0  0.0   0.00   0.00   0.0   0.0    100.0   0.0
MBOCA01  100E 33903    MM100E 1.0H                0.0  0.0   0.0  0.0   0.0  0.0  0.0   0.00   0.00   0.0   0.0    100.0   0.0
MBOCA01  100F 33903    MM100F 1.0H 000D   0.000   0.0  0.0   0.0  0.0   0.0  0.0  0.0   0.00   0.00   0.0   1.0    100.0   0.0
                        LCU        000D  77.555   1.7  0.0   0.1  0.0   0.1  0.3  1.3  15.20  18.25   0.0   7.3    100.0   0.0
```

Figure 5-8 Monitor I Device Activity report showing hyperPAV values

Attention: SMF record 74 subtype 1 and SMF record 79 subtype 9 are changed to support HyperPAV.

5.5.2 Monitor I-II-III I/O Activity Report changes

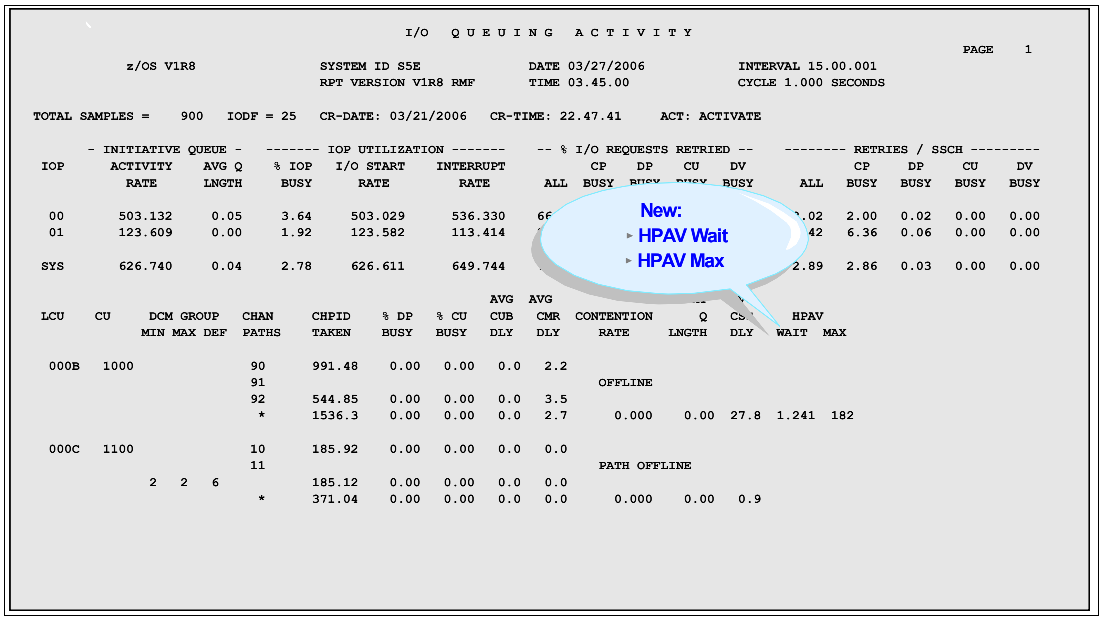
Figure 5-9 shows two new fields, HyperPAV Wait and HyperPAV Max, in the Monitor I I/O Activity Report example. Monitor II and III are also changed accordingly.

HyperPAV Wait

This is the ratio of the number of times an I/O could not start and the total number of I/O requests. The ratio of HyperPAV Wait might indicate that the pool of alias devices was not large enough. Expanding the pool may reduce I/O Queue time and CPU overhead.

HyperPAV Max

The maximum number of concurrently in-use HyperPAV alias devices for that LCU within that interval. In this case, HyperPAV Max approaches the number of configured HyperPAV alias devices for the LSS pool for the peak intervals, so you can reduce the I/O Queue time by adding more alias devices to the LSS pool.



```
                                         I/O   Q U E U I N G   A C T I V I T Y
                                                                                                         PAGE    1
             z/OS V1R8              SYSTEM ID S5E              DATE 03/27/2006           INTERVAL 15.00.001
                                    RPT VERSION V1R8 RMF       TIME 03.45.00             CYCLE 1.000 SECONDS

TOTAL SAMPLES =     900    IODF = 25    CR-DATE: 03/21/2006    CR-TIME: 22.47.41      ACT: ACTIVATE

       - INITIATIVE QUEUE -    ------- IOP UTILIZATION -------     -- % I/O REQUESTS RETRIED --    -------- RETRIES / SSCH ---------
 IOP      ACTIVITY    AVG Q     % IOP   I/O START     INTERRUPT          CP    DP    CU    DV             CP     DP     CU     DV
            RATE      LNGTH     BUSY      RATE          RATE       ALL  BUSY  BUSY  BUSY  BUSY      ALL  BUSY   BUSY   BUSY   BUSY

  00      503.132      0.05     3.64    503.029       536.330      66                               .02  2.00   0.02   0.00   0.00
  01      123.609      0.00     1.92    123.582       113.414                                        42  6.36   0.06   0.00   0.00

 SYS      626.740      0.04     2.78    626.611       649.744                                      2.89  2.86   0.03   0.00   0.00

                                                          AVG   AVG
 LCU    CU      DCM GROUP   CHAN      CHPID    % DP   % CU   CUB   CMR  CONTENTION       Q    CS       HPAV
              MIN MAX DEF   PATHS     TAKEN    BUSY   BUSY   DLY   DLY     RATE      LNGTH   DLY   WAIT   MAX

  000B   1000                 90      991.48    0.00   0.00   0.0   2.2
                              91                                          OFFLINE
                              92      544.85    0.00   0.00   0.0   3.5
                               *      1536.3    0.00   0.00   0.0   2.7      0.000     0.00  27.8  1.241  182

  000C   1100                 10      185.92    0.00   0.00   0.0   0.0
                              11                                          PATH OFFLINE
               2   2   6              185.12    0.00   0.00   0.0   0.0
                               *      371.04    0.00   0.00   0.0   0.0      0.000     0.00   0.9
```

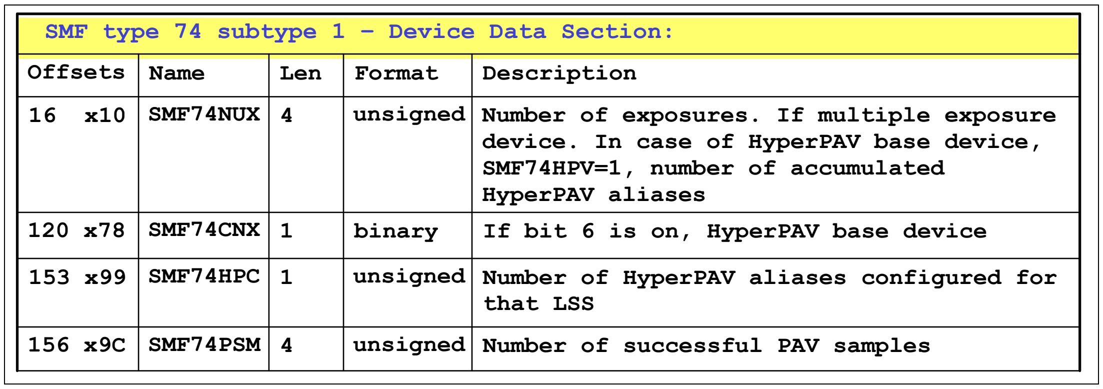
Figure 5-9 RMF Monitor I I/O Queuing Activity to support HyperPAV

SMF type 74 record

The UCB flag, UCBDPAVH, indicates whether this is a PAV or HyperPAV device. The Monitor I cycle gatherer ERBMFEDV is changed to accumulate the number of currently used HyperPAV aliases and to count the number of PAV samples.

The Monitor I interval gatherer processes PAV or HyperPAV devices and NON PAV devices. In case of HyperPAV devices, SMF74NUX contains the accumulated number of HyperPAV aliases of that interval or the last number of PAV alias devices. SMF74PSM saves the number of successful PAV samples.

The SMF type 74 fields for HyperPAV are shown in Figure 5-10 on page 98.

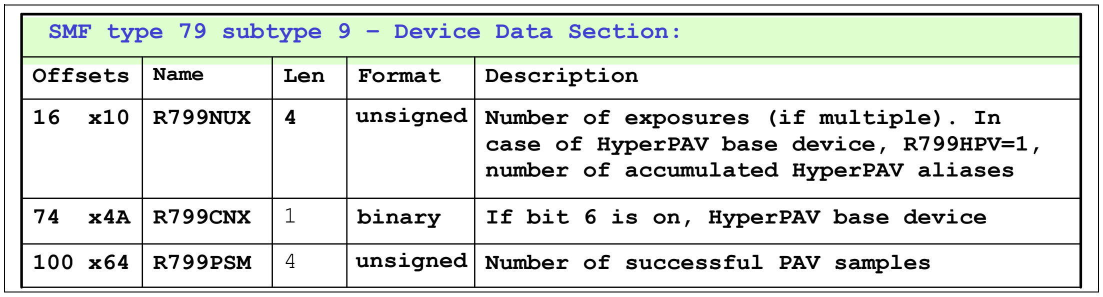


SMF type 74 subtype 1 - Device Data Section:				
Offsets	Name	Len	Format	Description
16 x10	SMF74NUX	4	unsigned	Number of exposures. If multiple exposure device. In case of HyperPAV base device, SMF74HPV=1, number of accumulated HyperPAV aliases
120 x78	SMF74CNX	1	binary	If bit 6 is on, HyperPAV base device
153 x99	SMF74HPC	1	unsigned	Number of HyperPAV aliases configured for that LSS
156 x9C	SMF74PSM	4	unsigned	Number of successful PAV samples

Figure 5-10 SMF type 74 fields for HyperPAV

SMF type 79 records

RMF Monitor II gatherer determines whether this is a PAV or HyperPAV device by checking the UCBDPAVH bit. In case of a HyperPAV device, the actual number of HyperPAV aliases is accumulated and saved in R799NUX, as shown in the SMF type 79 fields for HyperPAV in Figure 5-11 on page 98.



SMF type 79 subtype 9 - Device Data Section:				
Offsets	Name	Len	Format	Description
16 x10	R799NUX	4	unsigned	Number of exposures (if multiple). In case of HyperPAV base device, R799HPV=1, number of accumulated HyperPAV aliases
74 x4A	R799CNX	1	binary	If bit 6 is on, HyperPAV base device
100 x64	R799PSM	4	unsigned	Number of successful PAV samples

Figure 5-11 SMF type 79 fields for HyperPAV

5.6 RMF channel data gathering enhancements

With the Monitor I channel data cycle gathering, the channel data table CPDT (mapping macro ERBCPDT) is extended to include a base and accumulator count for each channel measurement item.

Monitor I channel data cycle gathering now collects the data from the CPMB every 120 seconds, calculates the deltas and accumulates the deltas to the accumulator counts of the current interval. After that, the base counts in the CPDT are set to the current counts from CPMB so that they can be used as new base in the next cycle.

The Monitor III channel data is currently obtained from Monitor II. With RMF for z/OSV1R8, Monitor III channel data gathering changes such that Monitor II channel data is no longer obtained and a new Monitor III channel data gatherer is introduced instead. The format and content of the Monitor III channel report does not change. Monitor II channel reporter is used to report on channel data obtained from Monitor II in prior releases.

5.7 RMF enhancements for zIIPs

These enhancements are described in “RMF support for zIIPs” on page 165.

5.8 DDS enhancements in z/OS V1R8

RMF PM takes its input data from a single data server on one system in the sysplex, which gathers the data from the RMF Monitor III distributed on all systems in the sysplex. Therefore, this is called the Distributed Data Server (DDS).

In z/OS V1R8, DDS has enhancements related to the following:

- Sysplex-wide management for RMF Distributed Data Server (DDS)
- WLM sysplex routing services to query DDS location

  DDS uses the IWMSRSRG service to register itself for sysplex routing. This service is an authorized service. Therefore, the calling DDS user ID GPMSERVE must either have the attribute TRUSTED or must have explicit READ access to the BPX.WLMSERVER resource profile in the FACILITY class. Otherwise, the DDS cannot propagate hostname and port number for potential exploiters. If you did not mark the GPMSERVE task as TRUSTED(YES), you must grant access for this task to the RACF Facility BPX.WLMSERVER with the following:

  ```
  PERMIT BPX.WLMSERVER CLASS(FACILITY) ID(GPMSERVE) ACCESS(READ)
  ```
- The OMEGAMON XE data collector now can access DDS data starting with Coupling Facility statistics in z/OS V1R8.

5.8.1 Distributed Data Server (DDS) overview

The RMF Distributed Data Server (DDS) provides the API to the performance data contained in Monitor III ISPF reports and runs in a separate address space. DDS data is exploited today by the following:

- RMF Performance Monitoring Java client
- The HTTP client
- The RMF CIM provider

5.8.2 Sysplex-wide management

DDS uses the RMF Sysplex Data Server API ERB3XDRS. This function returns sysplex-wide data. Therefore, just one DDS instance is needed across the sysplex.

Support prior to z/OS V1R8

In the past, the system where the DDS instance resides had to be selected manually and the following guidelines applied:

- Monitor III gatherer should be active
- You need RMF to be active
- SMF buffers must be active
- The DDS had to be started manually on the appropriate system by means of the START GPMSERVE command.

z/OS V1R8 DDS enhancements

The RMF session control task is enhanced to manage the Distributed Data Server address space across the sysplex, as follows:

- When the RMF initialization is complete, DDS is started automatically on the best suited system in the sysplex.

- If the DDS system is removed from the sysplex, a new DDS instance is started on the best suited candidate of the remaining systems.
- If Monitor III gathering is stopped on the DDS system, the DDS instance switches to the next appropriate system.
- If a new system joins the sysplex and this system is the best suited candidate, the DDS instance switches to this system.

WLM sysplex routing

WLM sysplex routing services are used to retrieve the hostname where the DDS is active and the port number where DDS is listening. Any user-written application can now determine the DDS location and obtain performance data from DDS, regardless of which system in the sysplex the program is running on. The following information is provided by DDS in the data areas when an application uses the WLM service:

hostname	The TCP/IP name of the host
userdata	The IP address and port number

For more information about WLM sysplex routing services, refer to *z/OS MVS Programming: Workload Management Services*, SA22-7919.

Important: The Distributed Data Server uses the WLM IWMSRSRG service to register itself for sysplex routing. This service is an authorized service. Therefore, the calling DDS user ID GPMSERVE must either have the attribute TRUSTED or must have explicit READ access to the BPX.WLMSERVER profile in the FACILITY class. For more information about RACF definitions, see the *z/OS Resource Measurement Facility User's Guide*, SC33-7990.

5.8.3 New options to stop and start DDS

With the new DDS options, no more user interaction is needed to start and stop the Distributed Data Server. This is now automatically executed by the RMF session control task.

The RMF session control is responsible for handling the start and stop processing for the Distributed Data Server address space. RMF session control maintains a list to react to one of the following events that could occur:

- A command has been entered at the operator console.
- The DDS timer has been posted and every 20 seconds DDS status checking processing executes.
- The collector service subtask that collects RSM data every 3 seconds (independent of other gathering routines) has terminated.
- A monitor II background session has terminated. The session control module has to wait for the termination of up to 32 background sessions when RMF is stopped.

As shown in Figure 5-12 on page 101, when the DDS option is either activated or deactivated from the console, this is recognized by the RMF session control. The current setting is then propagated to all members of the sysplex where RMF is running via the sysplex data server group event handler.

Since the DDS status is checked every 20 seconds, DDS is started when all of the following conditions apply:

- The DDS option is active.

- The local system is the appropriate system for the DDS (master system).
- DDS is not already active on the local system.

DDS is stopped when one of the following conditions applies:

- The DDS option is inactive but the DDS is running on the local system.
- The DDS option is active but the local system is not the master system.
- RMF is terminating.

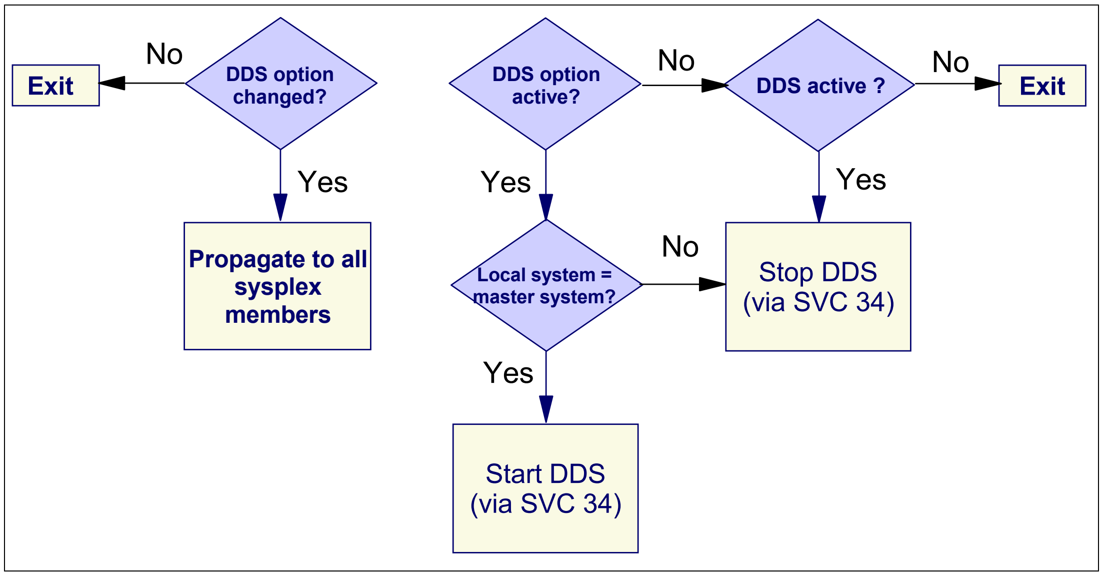


Figure 5-12 DDS sysplex-wide session control

Specifying DDS options

The DDS options can be specified in one of the following ways:

- PARM='DDS' to activate the sysplex-wide DDS management with the EXEC statement of the RMF procedure
- S RMF,,,DDS to activate the sysplex-wide DDS management with the RMF start command
- F RMF,DDS to activate the sysplex-wide DDS management with the RMF modify command

During initialization RMF starts DDS, as shown in Figure 5-13 on page 102.

```
S RMF,,,DDS
IRR812I PROFILE RMF.* (G) IN THE STARTED CLASS WAS USED 570
IAT6100 ( DEMSEL ) JOB RMF      (JOB04455), PRTY=15, ID=RMF
ICH70001I RMF      LAST ACCESS AT 11:55:22 ON THURSDAY, MAY RMF
IEF695I START RMF      WITH JOBNAME RMF      IS ASSIGNED TO
ERB100I RMF: ACTIVE
IEE252I MEMBER ERBRMF00 FOUND IN SYS1.PARMLIB
IERB325I ZZ : OLD WKLD GATHERER SUBOPTION(S) '(PERIOD)' IN RMF
ERB325I ZZ :     INPUT IGNORED.
ERB450I RMF: SMF DATA BUFFER INITIALIZED
ERB100I ZZ : ACTIVE
S GPMSERVE.RMFDDS01
ERB141I RMF: STARTING DISTRIBUTED DATA SERVER
IAT6100 ( DEMSEL )JOB GPMSERVE(JOB04456),PRTY=15,ID=GPMSERVE
ICH70001I GPMSERVE LAST ACCESS AT 11:55:42 ON THURSDAY, MAY GPMSERVE 00000290
IEF695I START GPMSERVE WITH JOBNAME GPMSERVE IS ASSIGNED TO
GPMSERVE, GROUP OMVSGRP
IEE252I MEMBER GPMSRV00 FOUND IN SYS1.PARMLIB
GPM060I RMF DISTRIBUTED DATA SERVER READY FOR COMMANDS
```

Figure 5-13 Using the DDS option with the START RMF command

- Using the F RMF,DDS command

  If not started using the S RMF command, sysplex-wide DDS management can be activated with an RMF modify (F) command. This automatically starts the DDS address space, as shown in Figure 5-14 on page 102.

```
OPERATOR 00000290  F RMF,DDS
RMF      00000090  ERB140I RMF: DDS OPTION ACCEPTED
RMF      00000290  S GPMSERVE.RMFDDS01
RMF      00000090  ERB141I RMF: STARTING DISTRIBUTED DATA SERVER
JES3     00000090  IAT6100 ( DEMSEL ) JOB GPMSERVE (JOB04414), PRTY=15, ID=GPMS
GPMSERVE 00000090  ICH70001I GPMSERVE LAST ACCESS AT 15:37:41 ON WEDNESDAY, MAY
GPMSERVE 00000290  IEF695I START GPMSERVE WITH JOBNAME GPMSERVE IS ASSIGNED TO
                   GPMSERVE, GROUP OMVSGRP
GPMSERVE 00000290  IEE252I MEMBER GPMSRV00 FOUND IN SYS1.PARMLIB
GPMSERVE 00000090  GPM060I RMF DISTRIBUTED DATA SERVER READY FOR COMMANDS
```

Figure 5-14 Starting DDS using the RMF modify command

- Via the PARM statement of the RMF procedure

```
//IEFPROC EXEC PGM=ERBMFMFC,REGION=4096K,DPRTY=(10,10),
//     PARM='SMFBUF(SPACE(2G),RECTYPE(70:78)),DDS'
//IEFPARM DD DDNAME=IEFRDER
//IEFRDER DD DSN=SYS1.PARMLIB,DISP=SHR
//SYSMDUMP DD SYSOUT=Z
```

Figure 5-15 RMF procedure to start DDS during RMF initialization

```
S RMF
IRR812I PROFILE RMF.* (G) IN THE STARTED CLASS WAS USED 120
IAT6100 ( DEMSEL ) JOB RMF      (JOB04425), PRTY=15, ID=RMF
ICH70001I RMF      LAST ACCESS AT 08:40:31 ON THURSDAY, MAY RMF
IEF695I START RMF      WITH JOBNAME RMF      IS ASSIGNED TO
ERB100I RMF: ACTIVE
IEE252I MEMBER ERBRMF00 FOUND IN SYS1.PARMLIB
ERB325I ZZ : OLD WKLD GATHERER SUBOPTION(S) '(PERIOD)' IN RMF
ERB325I ZZ :     INPUT IGNORED.
ERB450I RMF: SMF DATA BUFFER INITIALIZED
ERB100I ZZ : ACTIVE
S GPMSERVE.RMFDDS01
ERB141I RMF: STARTING DISTRIBUTED DATA SERVER
IAT6100 ( DEMSEL )JOB GPMSERVE(JOB04426),PRTY=15,ID=GPMSERVE GPMSERVE 00000090
ICH70001I GPMSERVE LAST ACCESS AT 08:40:51 ON THURSDAY, MAY GPMSERVE 00000290
IEF695I START GPMSERVE WITH JOBNAME GPMSERVE IS ASSIGNED TO GPMSERVE 00000290
IEE252I MEMBER GPMSRV00 FOUND IN SYS1.PARMLIB
GPM060I RMF DISTRIBUTED DATA SERVER READY FOR COMMANDS
ERB101I ZZ : REPORT AVAILABLE FOR PRINTING
```

Figure 5-16 Starting DDS automatically during start of the RMF procedure

Once the DDS option is specified on one single system, it is valid across the entire sysplex.

Note: The DDS option can be specified on any system in the sysplex. It is recognized on all images where RMF is active.

Stopping DDS

You can stop usage of DDS with the following command:

```
F RMF,NODDS
```

The command terminates the DDS address space; it will not be started until RMF is recycled. An RMF restart will also not start DDS if DDS is not added in the PARM parameter in the procedure. If you want to stop DDS in the sysplex, use the F RMF,NODDS command. Do not use the STOP GPMSERVE.RMFDDS001 command, as shown in Figure 5-17 on page 103, because until you remove the DDS enable option using this modify command, RMF will try to start it again.

```
F RMF,NODDS
ERB144I RMF: NODDS OPTION ACCEPTED
P GPMSERVE.RMFDDS01
ERB145I RMF: STOPPING DISTRIBUTED DATA SERVER
GPM066I RMF DISTRIBUTED DATA SERVER HAS TERMINATED
```

Figure 5-17 Stopping DDS using the RMF modify command

5.9 RMF Performance Monitoring - Java Technology Edition

RMF Performance Monitoring - Java Technology Edition (hereafter called RMF PM) allows you to monitor the performance of your z/OS from one focal point, as follows:

- A workstation running Windows® 2000, Windows XP, Windows ME

- Linux through a TCP/IP interface to one or more z/OS sysplexes

RMF PM Java Technology Edition is a graphical platform-independent enhancement for RMF online monitoring that allows you to analyze RMF Monitor III data right from your workstation.

Download RMF PM

You can download the current version to your Windows 2000/XP/ME or Linux workstation and explore the capabilities and functionality of this graphical performance monitoring tool.

You can manage z/OS sysplexes and LINUX images from a single point of control by monitoring the resources of the corresponding system. RMF PM takes input data from a single data server on one system and gathers the data from the RMF monitor II on each image. RMF PM supports the complete set of metrics provided by the RMF Monitor III gatherer. RMF PM offers powerful analysis functions to drill down problems. RMF PM is available for Microsoft® and also for LINUX. In figure Figure 5-18 on page 105, a typical RMF PM user interface is shown. Check for updates and downloads on the following RMF homepage:

```
www.ibm.com/servers/eserver/zseries/zos/rmf
```

From the RMF homepage, select the following from the available button to download RMF PM:

```
New Spreadsheet Reporter Version 5.2.0
```

After downloading the gpmwinv2.exe file, simply run it on your workstation to install the workstation client. Once it is installed, you can then start RMF PM from Start Programs.

Starting RMF PM

RMF PM takes its input data from a single data server on one system in the sysplex, which gathers the data from the RMF Monitor III on each MVS image. Therefore, this is called the Distributed Data Server (DDS). You can analyze and monitor data from the present or the recent past.

When you start RMF PM, you will receive the panel shown in Figure 5-18 on page 105 with the default specifications for the display.

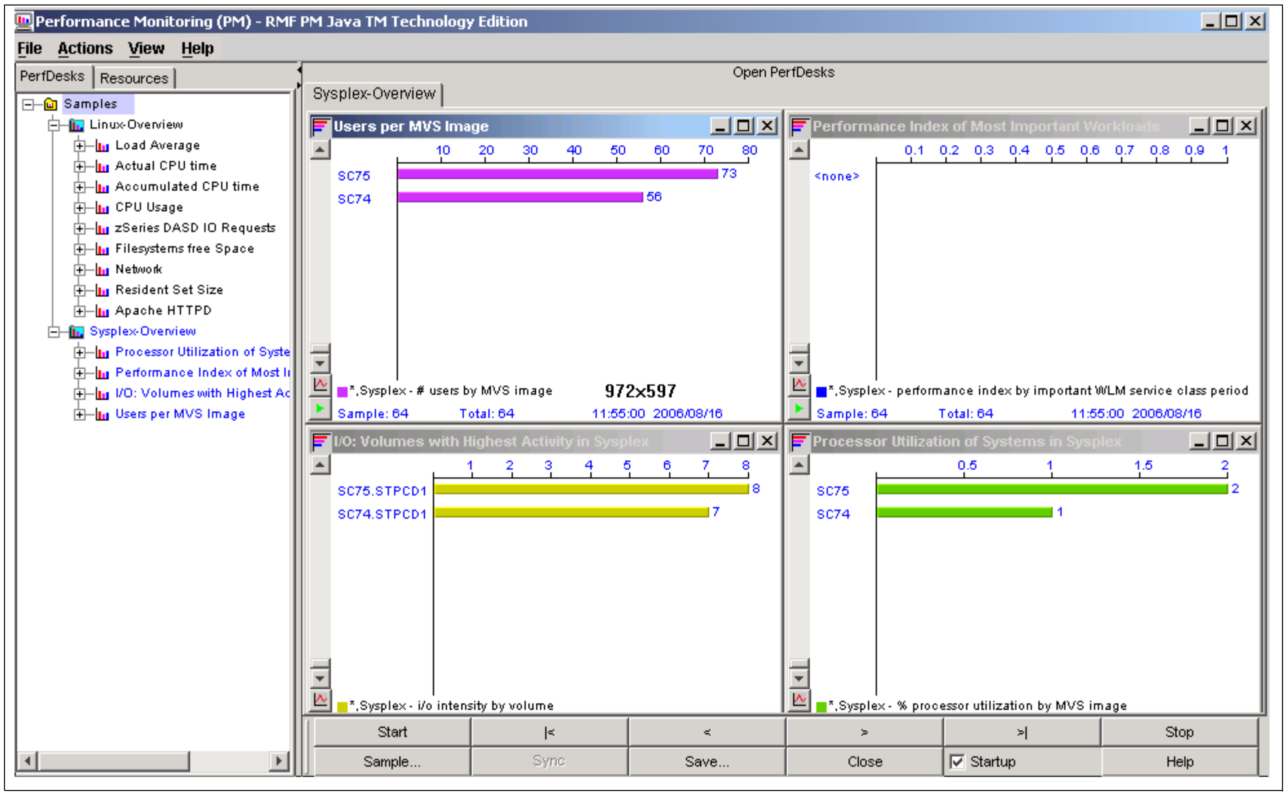

Performance Monitoring (PM) - RMF PM Java TM Technology Edition
File Actions View Help
PerfDesks Resources
Samples
Linux-Overview
Load Average
Actual CPU time
Accumulated CPU time
CPU Usage
zSeries DASD IO Requests
Filesystems free Space
Network
Resident Set Size
Apache HTTPD
Sysplex-Overview
Processor Utilization of Syste
Performance Index of Most I
I/O: Volumes with Highest Ac
Users per MVS Image
Open PerfDesks
Sysplex-Overview
Users per MVS Image
SC75 73
SC74 56
*,Sysplex - # users by MVS image
Sample: 64 Total: 64 11:55:00 2006/08/16
Performance Index of Most Important Workloads
<none>
*,Sysplex - performance index by important WLM service class period
Sample: 64 Total: 64 11:55:00 2006/08/16
I/O: Volumes with Highest Activity in Sysplex
SC75.STPCD1 8
SC74.STPCD1 7
*,Sysplex - i/o intensity by volume
Processor Utilization of Systems in Sysplex
SC75 2
SC74 1
*,Sysplex - % processor utilization by MVS image
Start
|<
<
>
>|
Stop
Sample...
Sync
Save...
Close
Startup
Help

Figure 5-18 RMF PM Java Technology Edition

5.9.1 RMF Monitor III Data Portal for z/OS

Instead of using the RMF PM, you can access RMF Monitor III data using DDS via any Web browser. The RMF Distributed Data Server (DDS) has been enhanced to respond directly to HTTP requests so that DDS behaves like an HTTP server. Therefore, if RMF is currently running and you have not started DDS, as explained in “New options to stop and start DDS” on page 100, you must start DDS as follows, as an example:

```
F RMF,DDS
```

You can then open a Web browser and simply type the following URL by specifying the hostname of a system in your sysplex and the port number 8803:

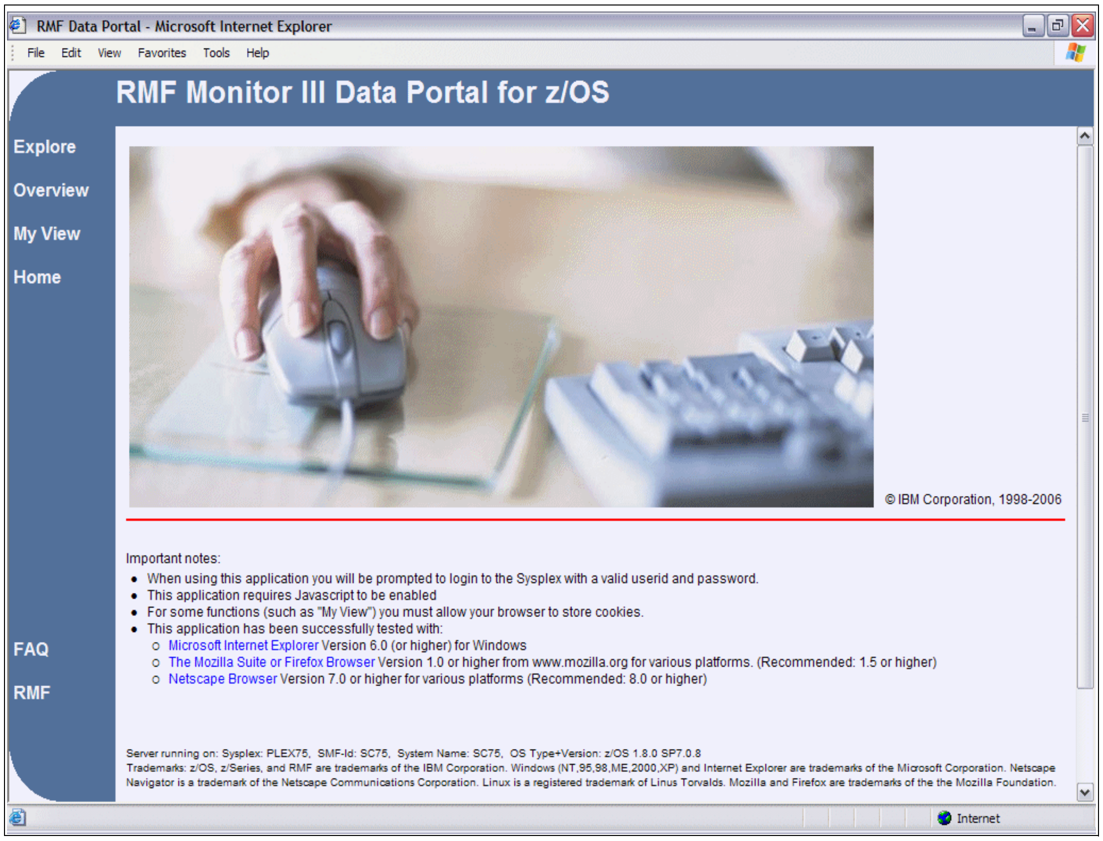
```
http://<hostname>:8803
```

For example:

```
http://wtsc74.itso.ibm.com:8803.
```

DDS behaves as an HTTP server for port 8803.

Once you have customized the first window that appears, Figure 5-19 on page 106 is the first panel that you see.


RMF Data Portal - Microsoft Internet Explorer
File Edit View Favorites Tools Help
RMF Monitor III Data Portal for z/OS
Explore
Overview
My View
Home
FAQ
RMF
Internet

© IBM Corporation, 1998-2006

Important notes:

- When using this application you will be prompted to login to the Sysplex with a valid userid and password.
- This application requires Javascript to be enabled
- For some functions (such as "My View") you must allow your browser to store cookies.
- This application has been successfully tested with:
  - Microsoft Internet Explorer Version 6.0 (or higher) for Windows
  - The Mozilla Suite or Firefox Browser Version 1.0 or higher from www.mozilla.org for various platforms. (Recommended: 1.5 or higher)
  - Netscape Browser Version 7.0 or higher for various platforms (Recommended: 8.0 or higher)

Server running on: Sysplex: PLEX75, SMF-Id: SC75, System Name: SC75, OS Type+Version: z/OS 1.8.0 SP7.0.8
Trademarks: z/OS, z/Series, and RMF are trademarks of the IBM Corporation. Windows (NT,95,98,ME,2000,XP) and Internet Explorer are trademarks of the Microsoft Corporation. Netscape Navigator is a trademark of the Netscape Communications Corporation. Linux is a registered trademark of Linus Torvalds. Mozilla and Firefox are trademarks of the the Mozilla Foundation.

Figure 5-19 RMF Monitor III Data Portal

RMF PM panels

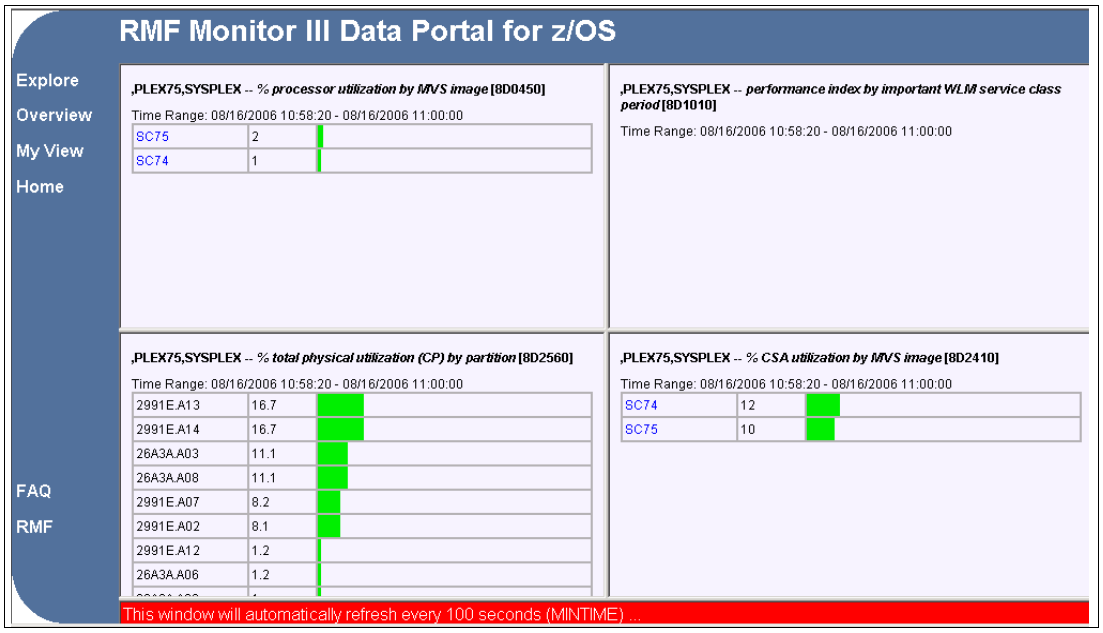
On each panel on the left side in Figure 5-19, you have the following options:

Overview	Shows you a health check overview of your system
My View	Views where you can see your preferred metrics
Explore	Explores the resources and performance metrics of your system
RMF	Leads you to the RMF Home page
Home	Leads you back to the welcome screen

Shown in Figure 5-21 on page 107 is the Welcome screen that also offers these functions. You can get a full Monitor III report or add metrics to customize your own view by selecting the Explore option.

Overview panel

When you click **Overview** in Figure 5-19, you receive the health check overview panel shown in Figure 5-20 on page 107.



,PLEX75,SYSPLEX -- *% processor utilization by MVS image* [8D0450]

Time Range: 08/16/2006 10:58:20 - 08/16/2006 11:00:00

SC75	2
SC74	1

,PLEX75,SYSPLEX -- *performance index by important WLM service class period* [8D1010]

Time Range: 08/16/2006 10:58:20 - 08/16/2006 11:00:00

,PLEX75,SYSPLEX -- *% total physical utilization (CP) by partition* [8D2560]

Time Range: 08/16/2006 10:58:20 - 08/16/2006 11:00:00

2991E.A13	16.7
2991E.A14	16.7
26A3A.A03	11.1
26A3A.A08	11.1
2991E.A07	8.2
2991E.A02	8.1
2991E.A12	1.2
26A3A.A06	1.2

,PLEX75,SYSPLEX -- *% CSA utilization by MVS image* [8D2410]

Time Range: 08/16/2006 10:58:20 - 08/16/2006 11:00:00

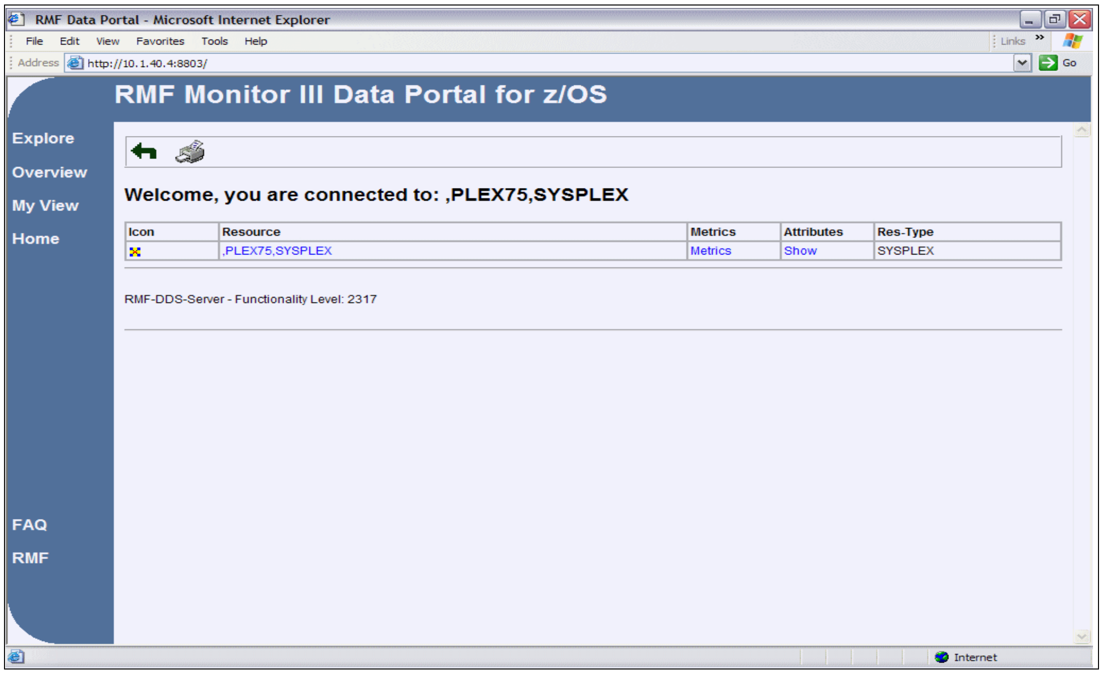
SC74	12
SC75	10

This window will automatically refresh every 100 seconds (MINTIME) ...

Figure 5-20 Overview panel

Explore panel

By selecting **Explore**, you get the panel shown in Figure 5-21.


RMF Data Portal - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links
Address http://10.1.40.4:8803/
Go
RMF Monitor III Data Portal for z/OS
Explore
Overview
My View
Home
FAQ
RMF
Internet

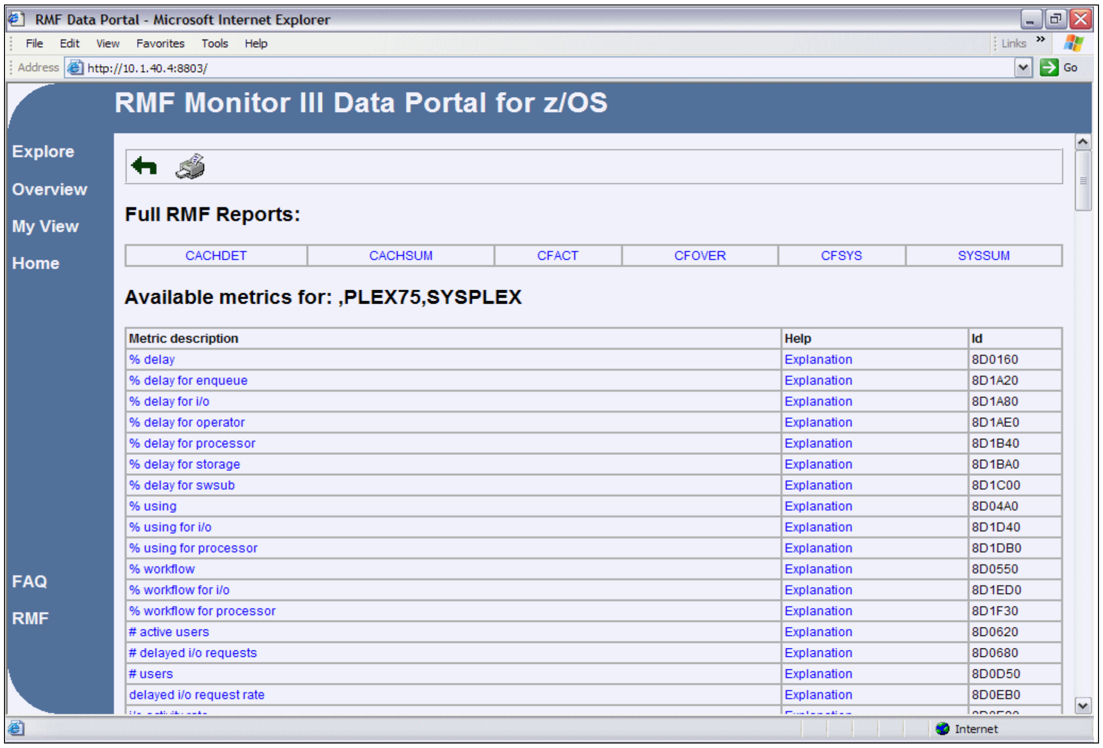
Welcome, you are connected to: ,PLEX75,SYSPLEX

Icon	Resource	Metrics	Attributes	Res-Type
	,PLEX75,SYSPLEX	Metrics	Show	SYSPLEX

RMF-DDS-Server - Functionality Level: 2317

Figure 5-21 Monitor III Data Portal Explore panel

By selecting the Metrics button in the Explore panel you can choose a full RMF Monitor III report type such as CACHDET or SYSSUM, or available metrics, as shown in Figure 5-22 on page 108.


RMF Monitor III Data Portal for z/OS

Full RMF Reports:

CACHDET	CACHSUM	CFACT	CFOVER	CFSYS	SYSSUM

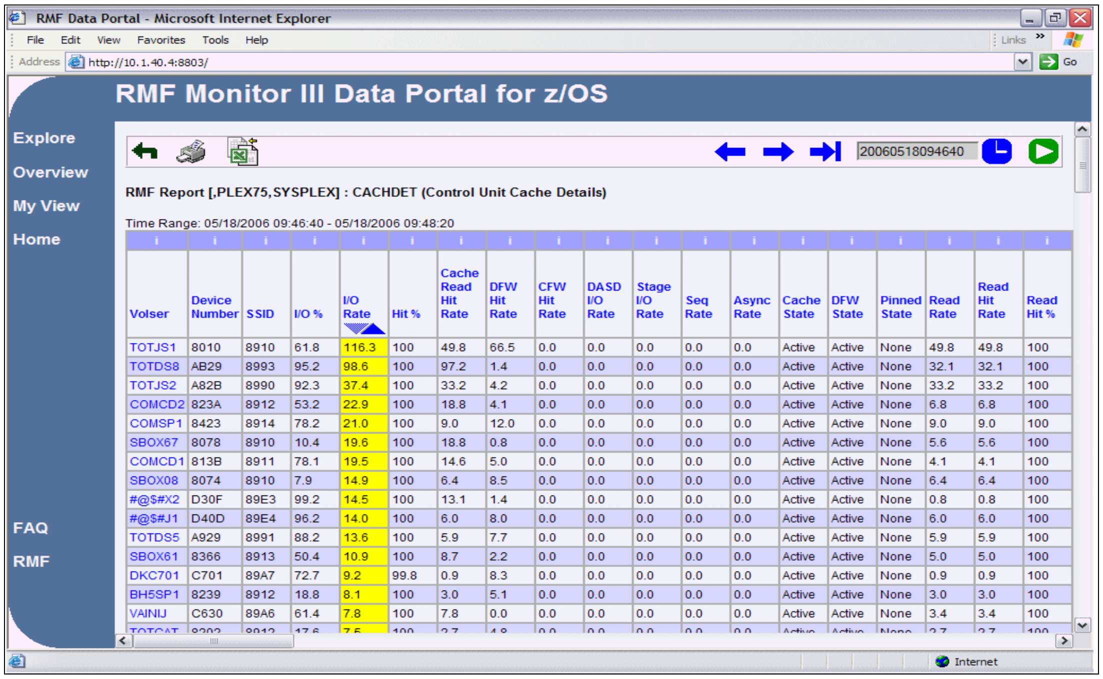
Available metrics for: ,PLEX75,SYSPLEX

Metric description	Help	Id
% delay	Explanation	8D0160
% delay for enqueue	Explanation	8D1A20
% delay for i/o	Explanation	8D1A80
% delay for operator	Explanation	8D1AE0
% delay for processor	Explanation	8D1B40
% delay for storage	Explanation	8D1BA0
% delay for swsub	Explanation	8D1C00
% using	Explanation	8D04A0
% using for i/o	Explanation	8D1D40
% using for processor	Explanation	8D1DB0
% workflow	Explanation	8D0550
% workflow for i/o	Explanation	8D1ED0
% workflow for processor	Explanation	8D1F30
# active users	Explanation	8D0620
# delayed i/o requests	Explanation	8D0680
# users	Explanation	8D0D50
delayed i/o request rate	Explanation	8D0EB0

Figure 5-22 Monitor III Data Portal panel showing full RMF report options and metrics

Report options and metrics

From this panel, when you select a full RMF report type such as CACHDET and CFSYS, you get to a panel that is similar to the host Monitor III panel. You can do what you do in a host terminal from here, such as going backward in interval time periods as much as your SMF buffer limits. This is shown in Figure 5-23 on page 108 and Figure 5-24 on page 109.

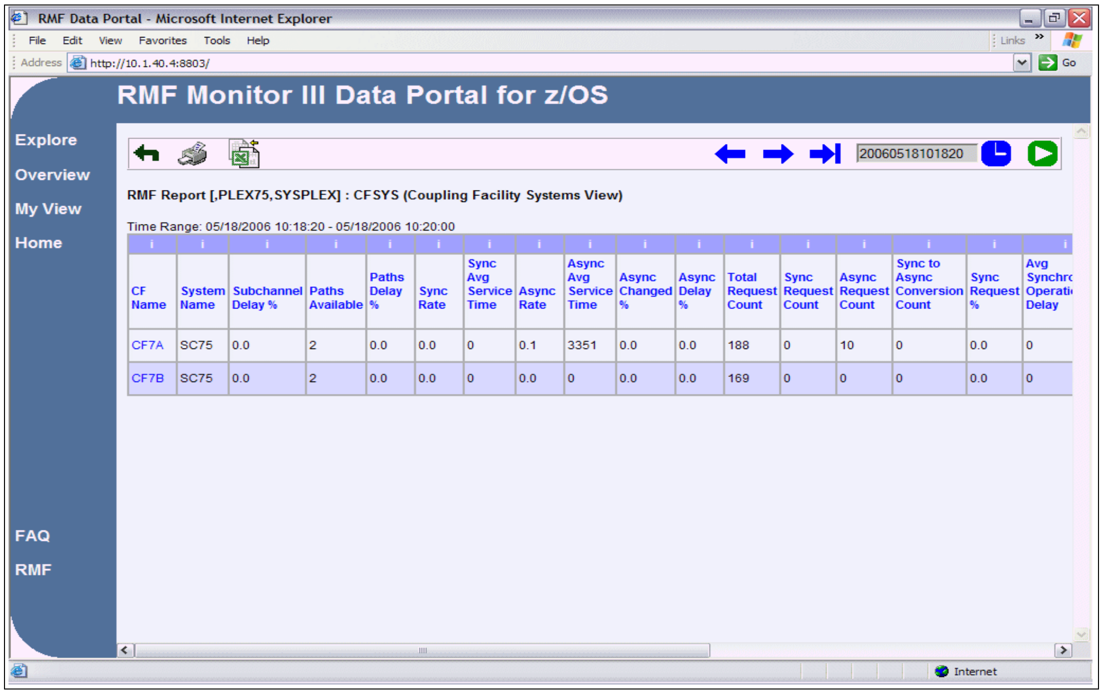

RMF Monitor III Data Portal for z/OS

RMF Report [,PLEX75,SYSPLEX] : CACHDET (Control Unit Cache Details)

Time Range: 05/18/2006 09:46:40 - 05/18/2006 09:48:20

Volser	Device Number	SSID	I/O %	I/O Rate	Hit %	Cache Read Hit Rate	DFW Hit Rate	CFW Hit Rate	DASD I/O Rate	Stage I/O Rate	Seq Rate	Async Rate	Cache State	DFW State	Pinned State	Read Rate	Read Hit Rate	Read Hit %
TOTJS1	8010	8910	61.8	116.3	100	49.8	66.5	0.0	0.0	0.0	0.0	0.0	Active	Active	None	49.8	49.8	100
TOTDS8	AB29	8993	95.2	98.6	100	97.2	1.4	0.0	0.0	0.0	0.0	0.0	Active	Active	None	32.1	32.1	100
TOTJS2	A82B	8990	92.3	37.4	100	33.2	4.2	0.0	0.0	0.0	0.0	0.0	Active	Active	None	33.2	33.2	100
COMCD2	823A	8912	53.2	22.9	100	18.8	4.1	0.0	0.0	0.0	0.0	0.0	Active	Active	None	6.8	6.8	100
COMSP1	8423	8914	78.2	21.0	100	9.0	12.0	0.0	0.0	0.0	0.0	0.0	Active	Active	None	9.0	9.0	100
SBOX67	8078	8910	10.4	19.6	100	18.8	0.8	0.0	0.0	0.0	0.0	0.0	Active	Active	None	5.6	5.6	100
COMCD1	813B	8911	78.1	19.5	100	14.6	5.0	0.0	0.0	0.0	0.0	0.0	Active	Active	None	4.1	4.1	100
SBOX08	8074	8910	7.9	14.9	100	6.4	8.5	0.0	0.0	0.0	0.0	0.0	Active	Active	None	6.4	6.4	100
#@$#X2	D30F	89E3	99.2	14.5	100	13.1	1.4	0.0	0.0	0.0	0.0	0.0	Active	Active	None	0.8	0.8	100
#@$#J1	D40D	89E4	96.2	14.0	100	6.0	8.0	0.0	0.0	0.0	0.0	0.0	Active	Active	None	6.0	6.0	100
TOTDS5	A929	8991	88.2	13.6	100	5.9	7.7	0.0	0.0	0.0	0.0	0.0	Active	Active	None	5.9	5.9	100
SBOX61	8366	8913	50.4	10.9	100	8.7	2.2	0.0	0.0	0.0	0.0	0.0	Active	Active	None	5.0	5.0	100
DKC701	C701	89A7	72.7	9.2	99.8	0.9	8.3	0.0	0.0	0.0	0.0	0.0	Active	Active	None	0.9	0.9	100
BH5SP1	8239	8912	18.8	8.1	100	3.0	5.1	0.0	0.0	0.0	0.0	0.0	Active	Active	None	3.0	3.0	100
VAINIJ	C630	89A6	61.4	7.8	100	7.8	0.0	0.0	0.0	0.0	0.0	0.0	Active	Active	None	3.4	3.4	100

Figure 5-23 Monitor III Data Portal - CACHDET full RMF report panel


RMF Data Portal - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://10.1.40.4:8803/
RMF Monitor III Data Portal for z/OS
Explore
Overview
My View
Home
FAQ
RMF
20060518101820
Internet

RMF Report [,PLEX75,SYSPLEX] : CFSYS (Coupling Facility Systems View)

Time Range: 05/18/2006 10:18:20 - 05/18/2006 10:20:00

CF Name	System Name	Subchannel Delay %	Paths Available	Paths Delay %	Sync Rate	Sync Avg Service Time	Async Rate	Async Avg Service Time	Async Changed %	Async Delay %	Total Request Count	Sync Request Count	Async Request Count	Sync to Async Conversion Count	Sync Request %	Avg Synchro Operati Delay
CF7A	SC75	0.0	2	0.0	0.0	0	0.1	3351	0.0	0.0	188	0	10	0	0.0	0
CF7B	SC75	0.0	2	0.0	0.0	0	0.0	0	0.0	0.0	169	0	0	0	0.0	0

Figure 5-24 Monitor III Data Portal CFSYS panel

Using the Distributed Data Portal

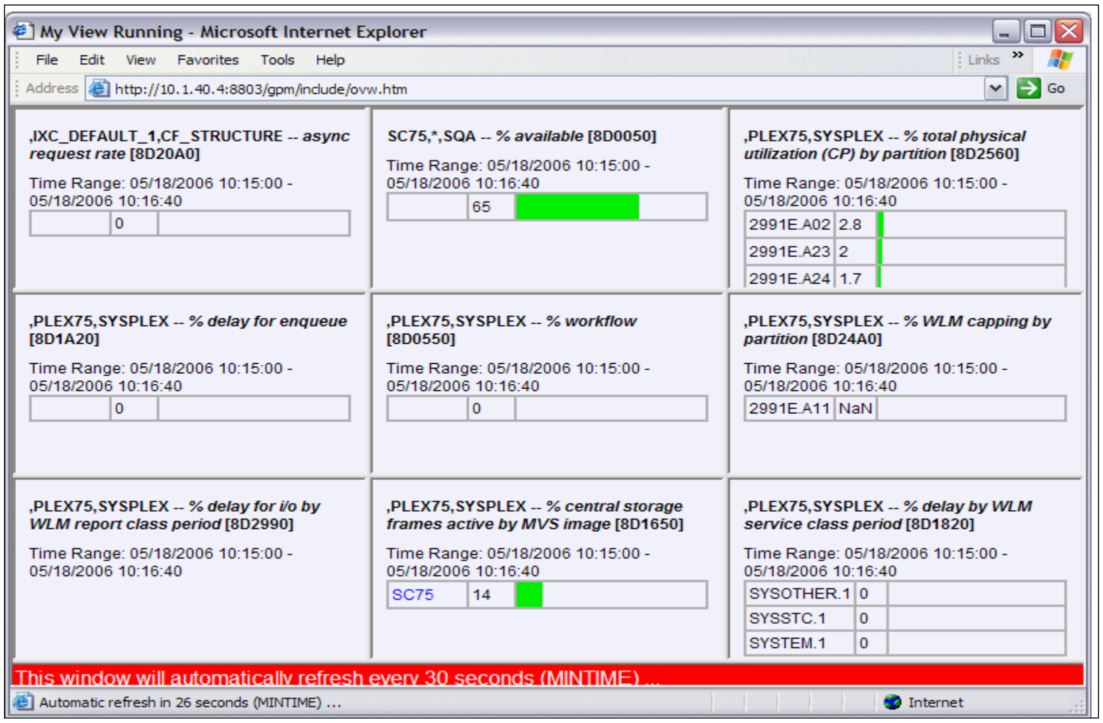
The nice part about the Distributed Data Portal is that you can choose your own report by selecting which metrics you want. About 80% of all data fields that exist in Monitor III can be chosen as metrics. In the future, all fields in Monitor III can be chosen as metrics.

The other most useful opportunity that you have using the RMF Monitor III data portal is the ability to get data you can view in an Excel® worksheet. You may use it from any panel where you can see an Excel logo. Just click on it and you can view all the data on that page on an Excel worksheet.

To use these functions, you need to customize DDS in your system, which is a simple process, and start using the Internet Explorer®. The only necessary subsystem other than RMF is TCP/IP. You should be sure that TCP/IP is configured and up and running in the system. This allows you to have sysplex-managed Distributed Data Server, and you do not need to have a DDS in every image in your sysplex.

Using the option My View

Figure 5-25 on page 110 shows an example of a panel when you select My View from any of the currently displayed panels.


My View Running - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://10.1.40.4:8803/gpm/include/ovw.htm

,IXC_DEFAULT_1,CF_STRUCTURE -- *async request rate* [8D20A0]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

	0	

SC75,*,SQA -- *% available* [8D0050]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

	65	🟩

,PLEX75,SYSPLEX -- *% total physical utilization (CP) by partition* [8D2560]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

2991E.A02	2.8	🟩
2991E.A23	2	🟩
2991E.A24	1.7	🟩

,PLEX75,SYSPLEX -- *% delay for enqueue* [8D1A20]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

	0	

,PLEX75,SYSPLEX -- *% workflow* [8D0550]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

	0	

,PLEX75,SYSPLEX -- *% WLM capping by partition* [8D24A0]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

2991E.A11	NaN	

,PLEX75,SYSPLEX -- *% delay for i/o by WLM report class period* [8D2990]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

,PLEX75,SYSPLEX -- *% central storage frames active by MVS image* [8D1650]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

SC75	14	🟩

,PLEX75,SYSPLEX -- *% delay by WLM service class period* [8D1820]
Time Range: 05/18/2006 10:15:00 - 05/18/2006 10:16:40

SYSOTHER.1	0	
SYSSTC.1	0	
SYSTEM.1	0	

This window will automatically refresh every 30 seconds (MINTIME) ...

Automatic refresh in 26 seconds (MINTIME) ... Internet

Figure 5-25 Monitor III Data Portal My View example

5.10 RMF Spreadsheet Reporter enhancements

The RMF Spreadsheet Reporter uses an FTP connection to the FTP server on the host, for example to submit postprocessor JCL and to download the postprocessor listing or overview records. Since currently only active FTP is supported, some installations are not able to use the Spreadsheet Reporter FTP connection. Active FTP access may fail, because firewalls block that kind of access. None-secure FTP connections may be restricted, due to security policy reasons on an installation site.

Therefore, in z/OS V1R8 the RMF Spreadsheet Reporter is enhanced to now support passive FTP connections as well as secure FTP connections.

5.10.1 RMF Spreadsheet Reporter overview

The Spreadsheet Reporter is the powerful workstation solution for graphical presentation of long-term postprocessor data. Use Spreadsheet Reporter to convert your postprocessor reports to spreadsheet format for further processing with spreadsheet macros. Spreadsheet macros generate representative charts for all performance-related areas. Also, they help you view and analyze performance data at a glance with guidance to drill down to performance problems.

Spreadsheet Reporter uses an FTP connection to submit postprocessor JCL, download the postprocessor listing, or to download overview records from the host.

5.10.2 Support for passive FTP connections

In pre-z/OS V1R8 systems, only active FTP was supported. Because in some installations firewalls block access, active FTP connections may fail and cannot be used by some installations. With z/Os V1R8, passive FTP connections are now available.

Active FTP connections

With active FTP connections, the client tells the server to create a data connection. The FTP server is the active part, and creates a data connection to a data port on the client site. Then, the flow is as follows:

1. The client connects from a command port to the command port on the server site.
2. The client requests a data connection from the server client data port.
3. The client waits and listens for a connection on the specified client data port.
4. The server creates a data connection from the server data port to the given client data port on the client.

This means that when someone from outside tries to open a connection to a non-standard port on the client, which is not usual, firewalls may treat this connection as an attack and may block the connection. The result is that the client waits until it receives a timeout because the server data connection has never reached the client. Firewalls need to be configured to allow that type of connection, but this may interfere with security policies.

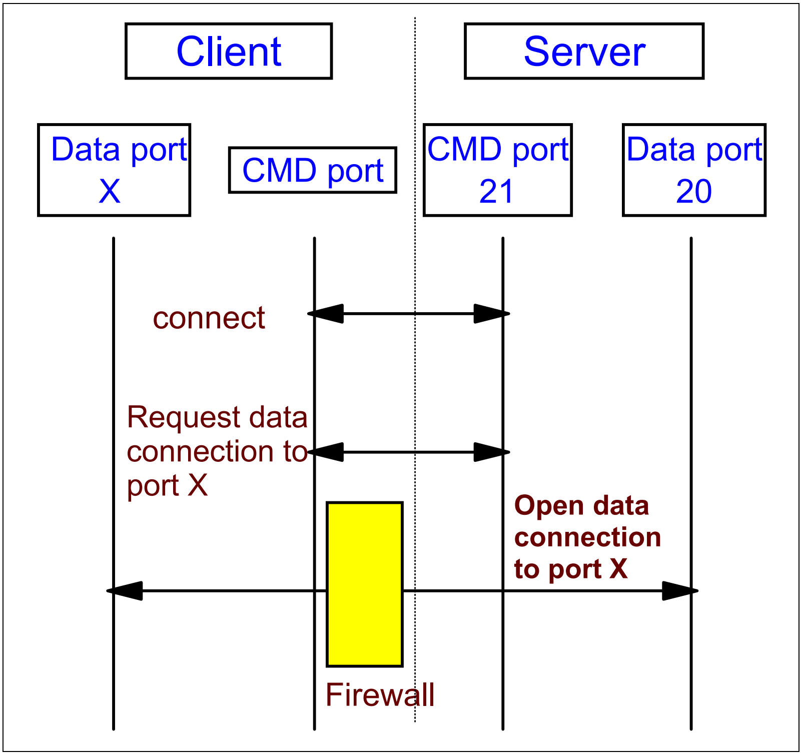


Figure 5-26 RMF active FTP connection

Passive FTP connections

For a passive FTP connection, the main idea is that the client creates a data connection to the server and chooses the passive mode. Then the flow is as follows:

- The client connects from the command port to the command port on the server.

- The client requests to switch to passive mode.
- The server opens a data port and gives it to the client.
- The server waits and listens for a connection on the specified data port.
- The client opens a data connection to the server data port.

The problem with the active mode request is that the server opens a data connection and waits for a connection. Since it is a server, normally no firewall will block access to non-standard ports. Therefore it's possible that another program can open the data port.

Since this is a passive FTP security issue, local client security may restrict passive FTP connections.

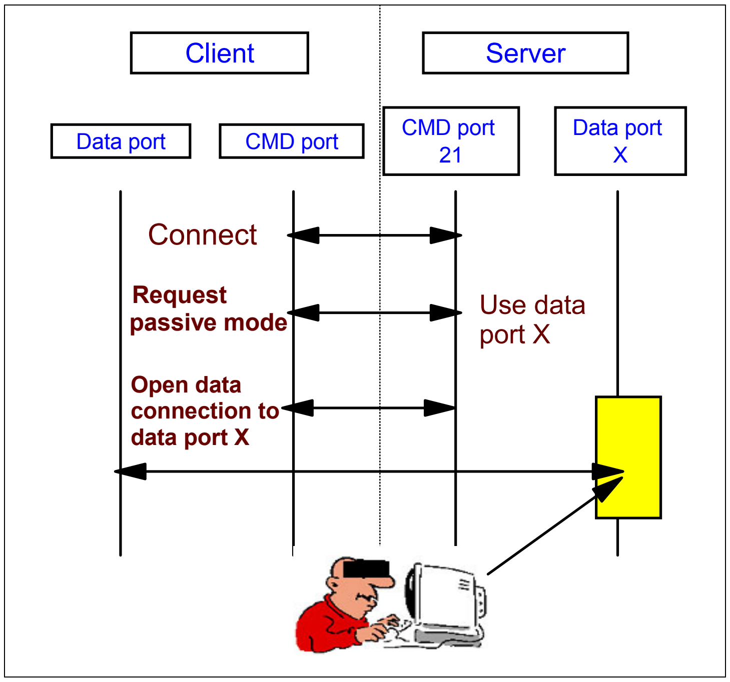
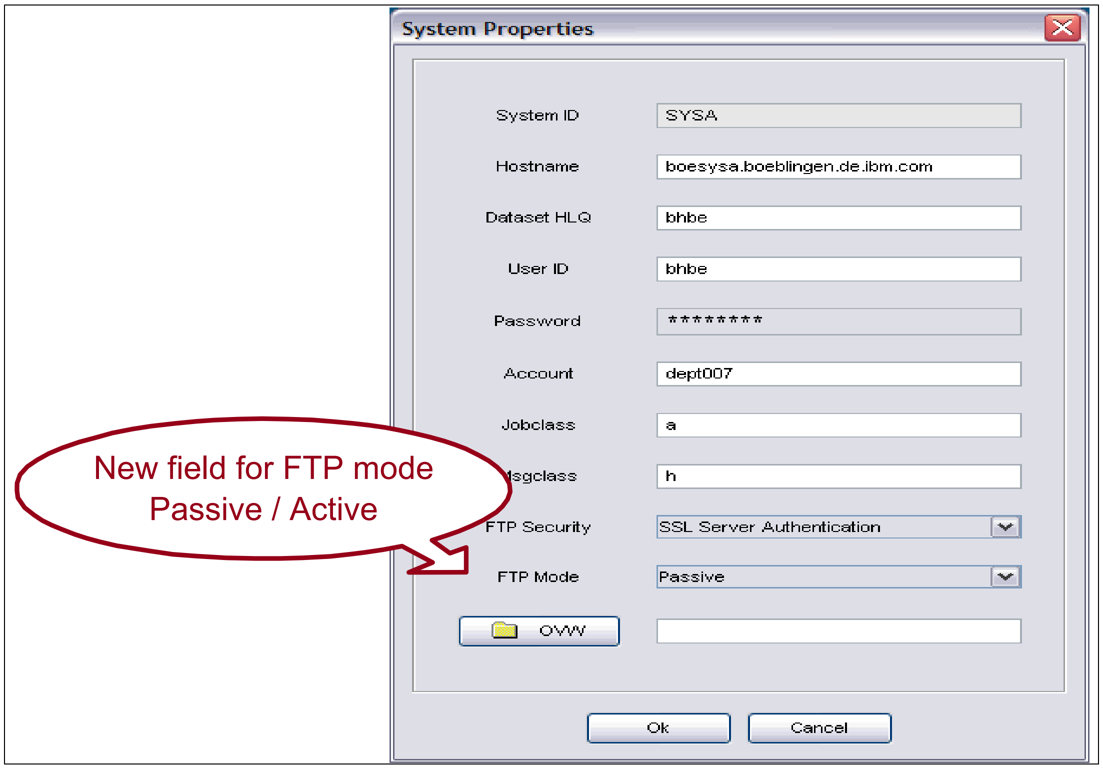


Figure 5-27 RMF passive FTP connection

In the system definitions in the Spreadsheet Reporter configuration, there is a new field that you can use to select the FTP connection type:

```
FTP mode: Active / Passive
```

See Figure 5-28 on page 113 for specification of the FTP mode.


System Properties

System ID	SYSA
Hostname	boesysa.boeblingen.de.ibm.com
Dataset HLQ	bhbe
User ID	bhbe
Password	*********
Account	dept007
Jobclass	a
Msgclass	h
FTP Security	SSL Server Authentication
FTP Mode	Passive
OVW	

Figure 5-28 FTP configuration new mode field

5.10.3 Support for SSL FTP connection with server authentication

In z/OS V1R8, an SSL FTP connection is supported. If the Spreadsheet Reporter connects to a server using SSL FTP with server authentication for the first time, the certificate is unknown. The Spreadsheet Reporter displays the certificate information, who issued the certificate for whom, the valid date, and additionally the hostname of the host who sends the certificate.

The user can now decide to trust the certificate or not. If the certificate is trusted and valid, it is stored. The next time the user restarts the Spreadsheet Reporter and connects to the server, the Spreadsheet Reporter knows the certificate and trusts the server.

If the certificate is trusted but invalid, for example if it has expired, this certificate is only valid for the current Spreadsheet Reporter session. This means that in the current Spreadsheet Reporter session, SSL FTP connections to the FTP server are allowed. But when you restart the Spreadsheet Reporter and connect to the server, you must again decide if you want to trust the certificate.

Non-secure FTP connection

When a non-secure FTP connection is used, all communication is sent as plain text. This means that within the TCP/IP packets, data such as the user ID and password are also available as plain text. It’s possible to use a sniffer program that sniffs the TCP/IP packets, which means that if a sniffer is connected to the network, the sniffer can easily read the user ID and the password.

Note: An Intrusion Detection System can provide detection of some types of attacks. Common intrusion detection system types currently deployed are network sniffers or sensors and vulnerability scanners. Sniffers, placed at strategic points in the network (in front or behind a firewall, in the network, or in front of a host), operate in promiscuous mode, examining traffic real-time that passes through on the local network. Sniffers use pattern matching to try to match a packet against a known attack, which is expressed as an attack signature.

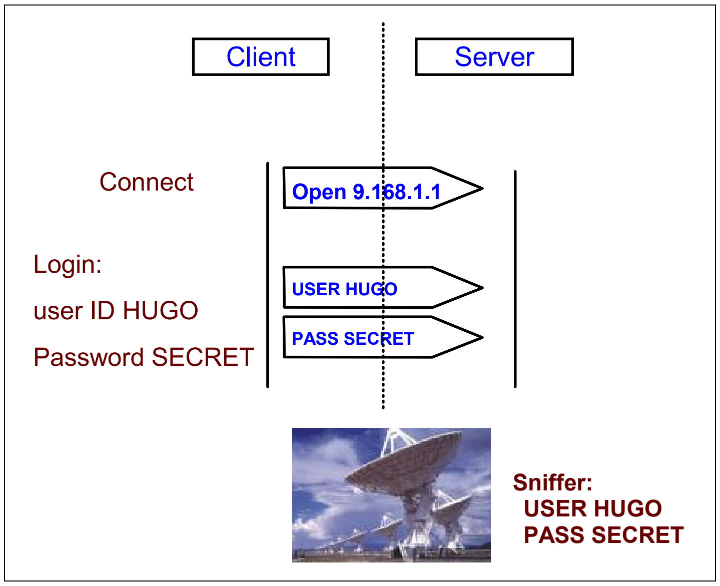


Figure 5-29 A non-secure FTP connection

Secure FTP connection

When a Secure Socket Layer (SSL) FTP connection with server authentication is used, after initiating communication, all further communication is encrypted and the server authenticates itself to the client with a certificate, as shown in Figure 5-30 on page 115.

There are two big security advantages:

- After initial communication, all further communication is ciphered before it is sent to the server.
- The server identifies itself with a server certificate, which helps the client to ensure that the client is talking to the right server.

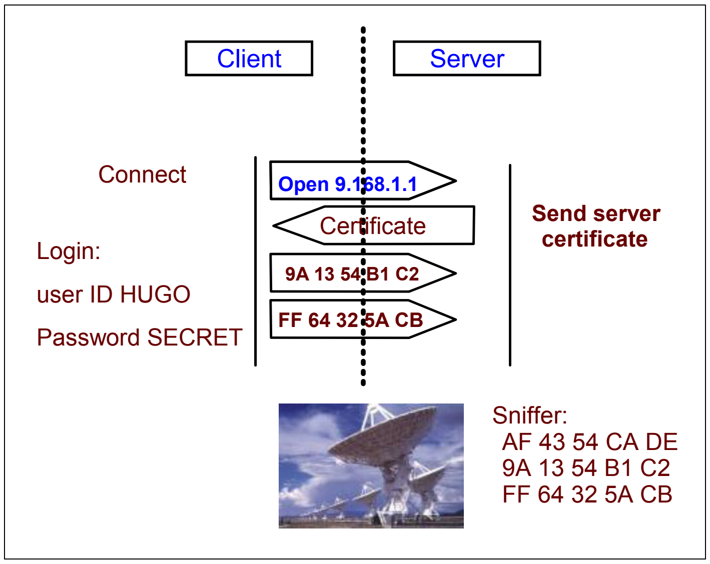


Figure 5-30 A secure FTP connection

Secure FTP connection flow

The detailed flow is as follows:

- The client connects to the server and requests to switch to SSL mode.
- The client and the server negotiate a cipher algorithm; after that, all further communication is now ciphered.
- The server sends a server certificate to the client.
- This server certificate contains the information as to who issued the certificate, for whom the certificate is issued, and a valid date.
- The client retrieves the certificate and verifies it: Is the certificate expired? Does the server hostname match the certificate?
- The client can decide to trust the certificate and continue the communication (even if the certificate is invalid; for example, it's expired).

If a sniffer is used, the sensitive user ID and password data are ciphered and cannot be read as clear text by the sniffer. The server certificate gives additional security that the client is not talking to a fake host, as shown in Figure 5-30.

In the system definitions in the Spreadsheet Reporter configuration, there is a new field where you can select a secure FTP connection type, as shown in Figure 5-28 on page 113.

```
FTP Security: None / SSL with server authentication
```

6

WLM enhancements

In this chapter we introduce the enhancements in WLM with z/OS V1R8.

We describe the following:

- WLM routing services problem
- Resource group enhancement
- WLM support for ZIIPs
- Improved management for zAAP workloads
- WLM I/O priority for tape I/O
- Database buffer pool management by WLM
- Enhanced JES2 WLM-managed classes
- WLM- EWLM enhancements
- Migration and coexistence

6.1 WLM routing services

The sysplex routing services allow work associated with a server to be distributed across a sysplex. They are intended for use by clients and servers. A client is any application or product in the network that requests a service. The service could be a request for data, a program to be run, or access to a database or application.

In terms of the sysplex routing services, a client is any program routing work to a server. A server is any subsystem address space that provides a service on an MVS image. The sysplex routing services provide the following main functions:

- The IWMSRSRG macro lets a caller register as a server and is replaced by a new macro service, IWM4SRSC, in z/OS V1R7.
- The IWMSRSRS macro provides a caller with a list of registered servers and the number of requests that should be routed to each server.

Web application example

Assume you have a Web application. On each system of a sysplex you have one Web server that is able to handle the incoming requests to one externalized IP address. The routing manager wants to distribute the incoming requests among the Web servers in a balanced way. That means the percentage of the requests being sent to one Web server should be equivalent to its ability to handle those requests at the given time—taking into account the system utilization and the server's behavior. The WLM routing services give you a recommendation for this distribution by returning a weight for each server, that is registered for routing.

The Sysplex Distributor of Communication Server, DB2, and other router manufacturers exploit those services, as shown in Figure 6-1.

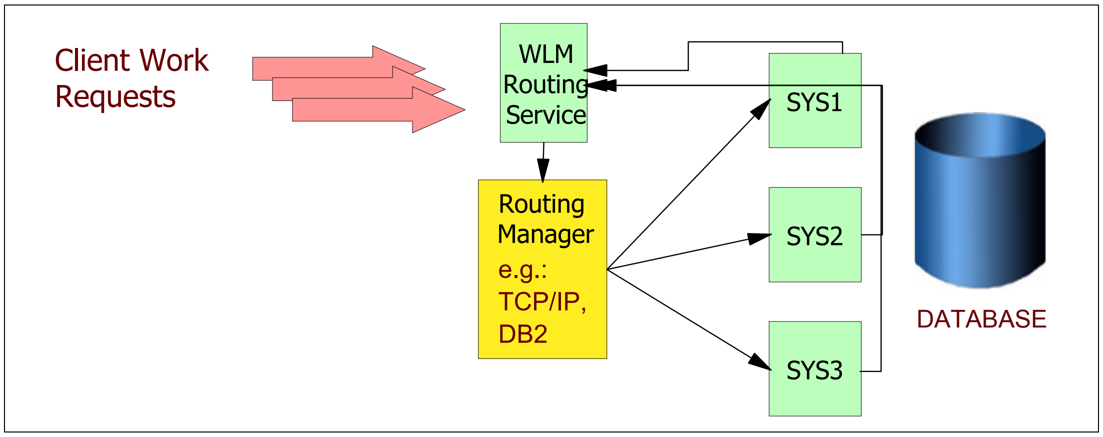


Figure 6-1 WLM sysplex routing

IWMSRSRG macro lets a caller register as a server

Before being able to use the service for routing recommendations, IWMSRSRS, you have to register the servers through the IWMSRSRG service. You group servers during registration by associating them with an identifier called LOCATION.

Figure 6-2 on page 119 gives an example of how the servers are registered using IWMSRSRG. The list of servers registered on the different systems of a sysplex is communicated between the systems every 10 seconds.

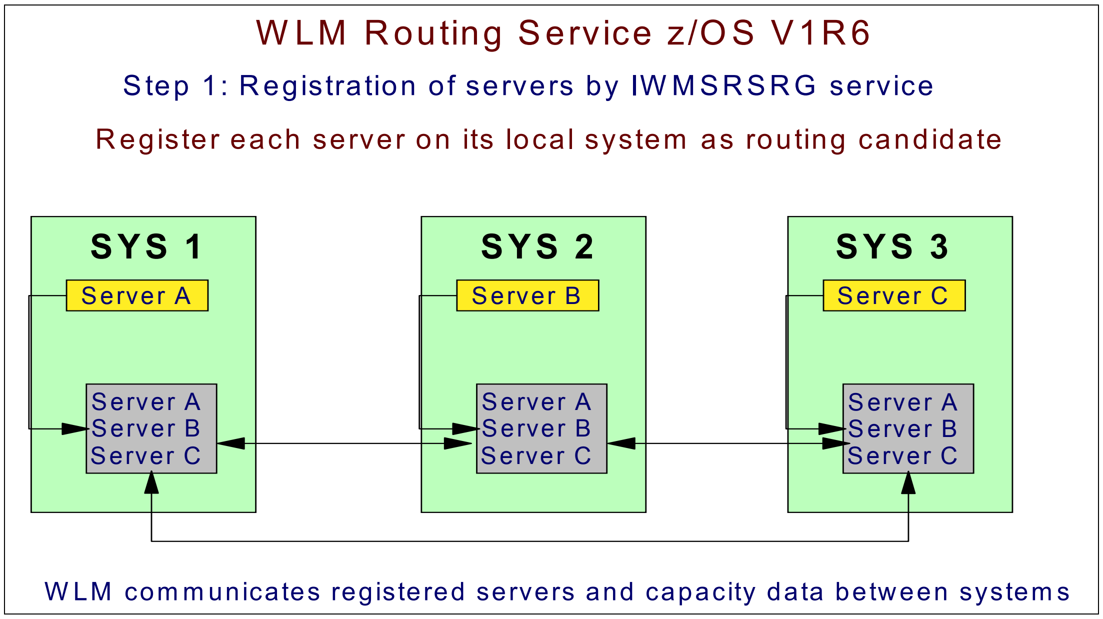

WLM Routing Service z/OS V1R6
Step 1: Registration of servers by IWMSRSRG service
Register each server on its local system as routing candidate
SYS 1
Server A
Server A
Server B
Server C
SYS 2
Server B
Server A
Server B
Server C
SYS 3
Server C
Server A
Server B
Server C
WLM communicates registered servers and capacity data between systems

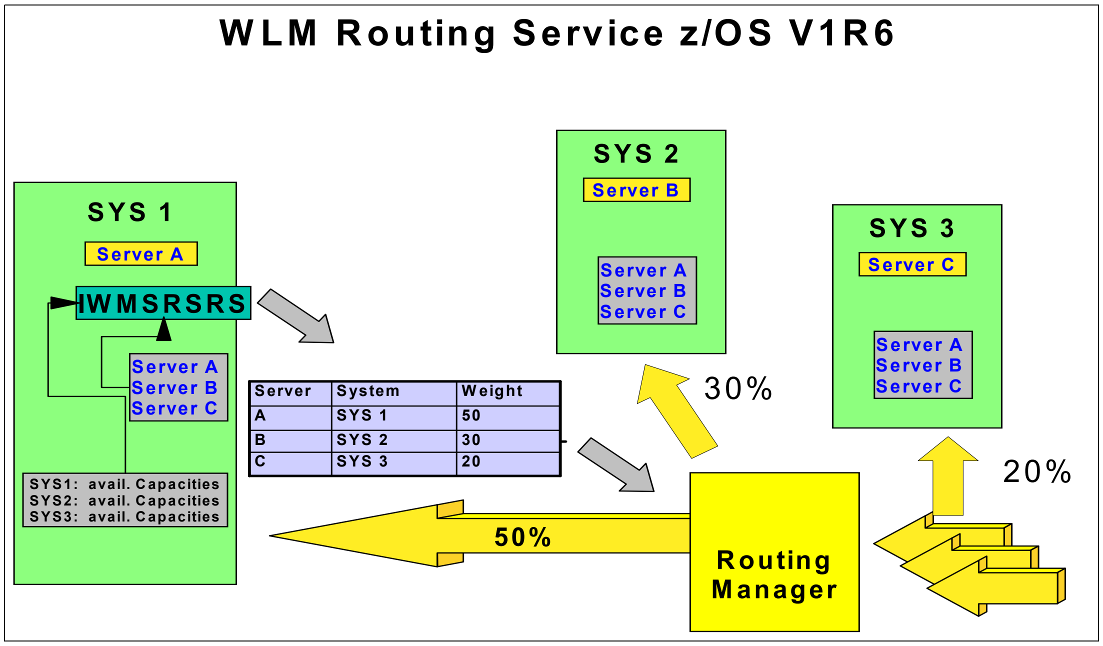
Figure 6-2 Registration of servers by the IWMSRSRG macro

Ask WLM for routing recommendations by the IWMSRSRS service

To get a routing recommendation, you can call the IWMSRSRS service with the LOCATION parameter on any system in the sysplex. It returns recommendations, called weights, which are numbers between 1 and 64 for each server that was registered under the LOCATION ID.

Then you would use those weights to distribute the incoming requests among the servers according to the size of their weights.

Figure 6-3 on page 119 gives an idea of how it works.



Server	System	Weight
A	SYS 1	50
B	SYS 2	30
C	SYS 3	20

Figure 6-3 IWMSRSRS provides a caller with a list of registered servers and the number of requests that should be routed to each server

Weight calculation

To assign weights to the three systems, WLM scans the CPU consumption table from the bottom up, looking for a level where at least one system has 5% or greater cumulative CPU consumption. In this case, the level 5 row is used.

The sum of all SUs used by importance level 5 or lower (called *displaceable capacity*) is 200+400+300 = 900.

By dividing each system's displaceable capacity of level 5 by this sum, we get the system weights.

The resulting server weights are the result of dividing each system weight by the number of registered servers on that system.

Weakness of WLM routing service before z/OS V1R7

The problem of pre-z/OS V1R7 WLM routing service is that the routing recommendation, the weight, for a server is only based on the available capacity of the system (LPAR). It does not take the specific information about the performance-related behavior of the servers itself into account.

This could lead to unexpected results. For example, consider a server running on a system that is fully loaded with low importance work, but the server has high importance. In this case the server would be able to displace other work and handle more requests than the routing recommendation specified to send there.

It might also happen that the server runs on a low utilized system, but does not have enough needed resources to run much work. (For instance, a DB2 server could run out of DBATs.) But due to the low system load it is recommended to send much more work to this server.

6.1.1 z/OS V1R7 enhancements

With z/OS V1R7 and higher, two WLM routing services are available:

- IWM4SRSC - A service for non-registered servers with LPAR scope
- IWMSRSRS - A sysplex-wide service for registered servers

6.1.2 New WLM routing services

A new routing service, IWM4SRSC, and a new function for the old routing service, IWMSRSR, were implemented in z/OS V1R7 in order to solve the routing services problems detected in previous releases. These changes are both system- and server-specific.

The new function SPECIFIC for the IWMSRSRS service is called in the same way as the old functions SELECT and QUERY. It requires having the servers registered in advance (through the registration service IWMSRSRG). It can be called on any system and takes all registered servers on all systems into account. DB2 is the first exploiter of this new function.

The other new service, IWM4SRSC, does not require registration of the servers. But it can only return a recommendation for one server per call, so this routine has to be called for each server that a recommendation is wanted for, on the system that server is running on. Communication Server is the first exploiter of this new function (Sysplex Distributor for TCP/IP).

New routing service IWM4SRSC

IWM4SRSC has the STOKEN parameter to identify the server address space that the callers want information about. This new service gives the weight parameter as output, which is a number between 1 and 64; it indicates the relative performance behavior of the server so that it can be compared to other servers.

The weight calculation takes into account two factors, scaled by 64:

- Performance Indicator (PI) factor

  PI gives an indication of how well this server is achieving its goals as defined in the active WLM policy.

- The importance factor

  This is a measurement of how much CPU capacity is displaceable by work of the server‘s importance, respective to the work that is related to this server.

New function code SPECIFIC in routing service IWMSRSRS

The weight calculation for the new function code, SPECIFIC, in routing service is a product of three factors:

- System utilization factor

  This is the same as the resulting system weight for the old SELECT function in pre-z/OS V1R7 releases.

- PI factor

  This is calculated in the same way as the PI factor of the IWS4SRSC.

- Queue time ratio

  If the server owns independent enclaves, this is the ratio of queue time to elapsed time of those enclaves.

  The queue time is the time between the time given in the ARRIVALTIME parameter of the enclave create service and the start time of the enclave.

  The queue times and the total elapsed times of the enclaves owned by the server are stored. The average ratio of the queue times versus the elapsed times is calculated to get the queue time ratio of the server.

  Note: If two or more servers are registered on the same system, the weight is divided by the number of those servers. This is the same as for the old SELECT function for that service, to avoid overloading a system that has many servers.

6.1.3 Routing service problems with z/OS V1R7

In some situations, a server for which WLM has to give routing recommendations for, starts processing the incoming requests with error codes, but finishes each request very fast. This could lead to a situation in which this server has a good PI and its LPAR usage is low, as shown in Figure 6-4 on page 122. When this occurs, WLM could calculate a high weight as a routing recommendation for that server. This would cause the routing service to send many requests to that server, instead of stopping or at least reducing requests for this server.

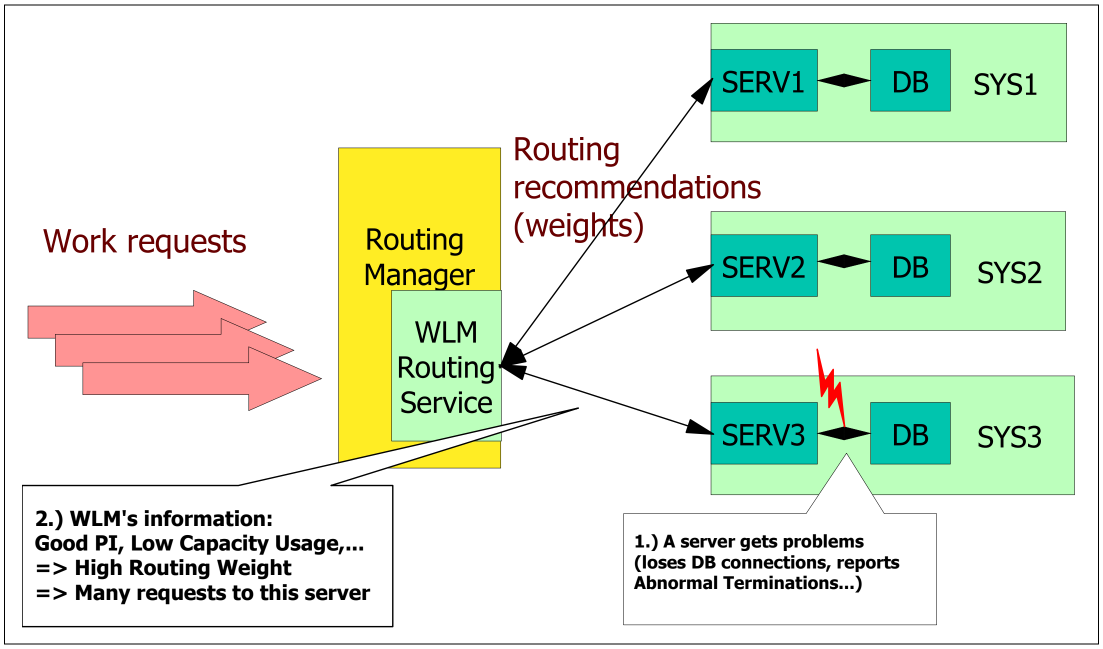

SERV1
DB
SYS1
Routing recommendations (weights)
Work requests
Routing Manager
WLM Routing Service
SERV2
DB
SYS2
SERV3
DB
SYS3
2.) WLM's information:
Good PI, Low Capacity Usage,...
=> High Routing Weight
=> Many requests to this server
1.) A server gets problems
(loses DB connections, reports
Abnormal Terminations...)

Figure 6-4 WLM routing manager problem

6.1.4 WLM routing service enhancements with z/OS V1R8

Because WLM is not aware of abnormal conditions of a server, it can arrive at bad recommendations. A method to inform WLM about abnormal conditions is now implemented to make sure WLM should take these abnormal conditions into account. Therefore, the following two situations are now considered by WLM when making routing recommendations:

- Health value of a server
- Abnormal termination rate of a server in which WLM is informed about these conditions

By taking the abnormal termination rate and the health value of a server into account in routing recommendations, the routing recommendations are more adequate in abnormal situations.

The planned exploiters of these enhancements are Communication Server for z/OS V1R8, DB2 V8, and CICS. For more detailed information about services and enhancements, see *z/OS V1R8 MVS: Planning for Workload Manager*, SA22-7602-12 and *z/OS MVS Programming: Workload Management Services*, SA22-7619.

Important: This solution will be rolled back to z/OS V1R7 with APAR OA14310. The enhanced IMW4SRSC service can be used on any system with z/OS V1R7 or higher. IWMSRSRS with enhanced function SPECIFIC can be called when all servers registered under the given LOCATION run on systems with z/OS V1R7 or higher. The services can be called as assembler macros by any application program.

6.1.5 Health value of a server

In WLM z/OS V1R8, every server now has a health value associated with it. The health value can be between 0 and 100 and shows the health status as a percentage. For example, a health value percentage is used as follows:

100	Means the server is fully healthy.

0	Means the server is not able to do any work.
60	Means the server is able to do 60% of its regular ability.

The default health value is 100.

Setting the health value of a server

The health value of a server is set either by a new IWM4HLTH service, as follows:

```
IWM4HLTH stoken=token,health=health_value
```

or by a new input parameter of the IMWSRSRG registration service:

```
IWMSRSRSG stoken=token,health=health_value,location=......
```

In the routing services IWM4SRSC and IWMSRSRS, the weight is multiplied by the additional health factor, which reflects the health value of the related server, as follows:

```
health factor = health value / 100
```

Weight calculation

In the IWM4SRSC service:

```
Weight = PI Factor  *  Importance factor * Health Factor
```

In the IWMSRSRS service:

```
Weight = LPAR Utilization Factor * PI Factor * Enclaves Queue Time Ratio *
Health Factor
```

6.1.6 Abnormal termination rate of a server

With this enhancement, WLM now takes abnormal conditions into account. The ratio of abnormal terminations versus total terminations are reported by the WLM Report service IWMRPT. They are accumulated by service class per LPAR. The long term average of abnormal termination rate results in an additional factor in the weight calculation of the IWM4SRSC service as follows:

```
Weight = PI Factor  *  Importance factor * Health Factor * Abnormal Termination
Factor
```

Where the Abnormal Termination Factor = 1 - Abnormal Termination Rate.

6.2 Resource groups

A resource group is an amount of processor capacity. It is optional. Unless you have some special need to limit or protect processor capacity for a group of work, you should skip defining resource groups and let workload management manage all of the processor resources to meet performance goals. You use a resource group to:

- Limit the amount of processing capacity available to one or more service classes
- Set a minimum processing capacity for one or more service classes in the event that the work is not achieving its goals
- Define a minimum and maximum amount of capacity sysplex-wide, or on the system level

Using resource groups

Prior to z/OS V1R8, resource groups were a way of limiting or guaranteeing resource capacity. A resource group is a named amount of CPU capacity that you can assign to one or more service classes. For most systems, you can let workload management decide how to manage the resources in the sysplex and not use resource groups. You set performance goals for work and let workload management adjust to meet the goals. You can specify a minimum and maximum amount of capacity to a resource group. You can assign only one resource group to a service class. You can define up to 32 resource groups per service definition.

Resource groups are used if it is necessary that for a certain capacity that is required by an application or department, you may need to assign a fixed amount of CPU capacity and want to ensure that no more is used. Or, you may want to ensure that some work with discretionary goals receives some minimum amount of capacity. For both cases, a resource group is appropriate. For other cases, it is probably not, so you do not need to use resource groups.

In pre-z/OS V1R8 systems, resource groups were always sysplex-wide and no resource entitlements were on a system basis. Resource groups should be defined in service units (which change with the model of the processor).

Current resource group specifications

As shown in Figure 6-5, multiple service classes can be assigned to a resource group, RGTE_A, with different utilizations on different systems. Each system may have a different capacity. As a result, it is not easy to understand how much is consumed on which system. The consumption depends highly on the capacity of the systems. In reality, resource groups need to be revisited when systems are upgraded or when workload utilization changes.

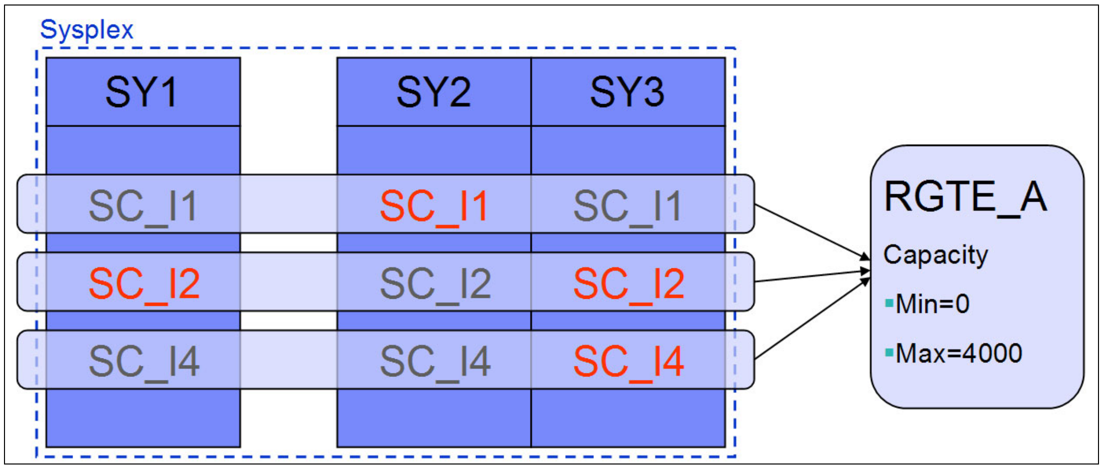


Figure 6-5 Resource group usage in pre-z/OS V1R8 systems

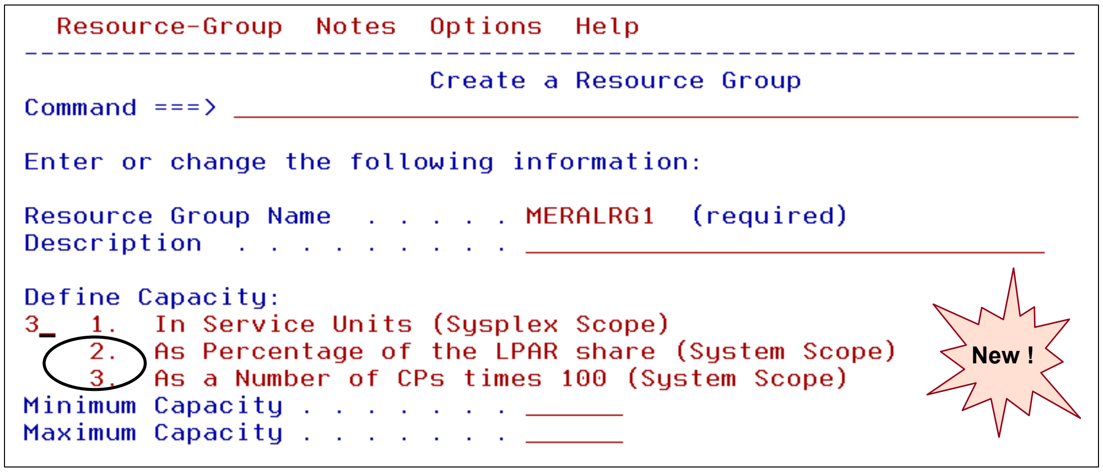
6.2.1 Resource group enhancements in z/OS V1R8

In z/OS V1R8, two additional resource group types are available when defining resource groups. The scope of the definitions has also changed. In addition to the one already existing group type, type 1, the new resource group types are 2 and 3, as shown in Figure 6-6 on page 125. For all 3 types you can define a minimum and maximum capacity.

Attention: On back-level systems the new type 2 and 3 resource groups are converted to type 1 resource groups without a minimum and a maximum.

The enhancements in resource group definitions in z/OS V1R8 now let you define three variations for resource groups when you select the Define Capacity type options, as shown in Figure 6-6.

1. In service units with a sysplex-wide scope - This is the existing support.
2. As a percentage of the LPAR capacity - This is new in z/OS V1R8. It is based on LPAR share or weight. The value must be in the range of 0 to 99, and this type is managed system-wide.
3. As a percentage of CP capacity - This is also new in z/OS V1R8. This type is also managed system-wide and the value must be in the range of 0 to 6400. 6400 is used in support of the future existence of 64 processors in one image.



```
  Resource-Group  Notes  Options  Help
 -----------------------------------------------------------------
                        Create a Resource Group
 Command ===> ____________________________________________________

 Enter or change the following information:

 Resource Group Name . . . . . MERALRG1  (required)
 Description . . . . . . . . . ________________________________

 Define Capacity:
 3_  1.  In Service Units (Sysplex Scope)
     2.  As Percentage of the LPAR share (System Scope)
     3.  As a Number of CPs times 100 (System Scope)
 Minimum Capacity . . . . . . . ______
 Maximum Capacity . . . . . . . ______
```

Figure 6-6 New resource group definition types

The resource group types are described in Table 6-1.

Table 6-1 Explanation of each resource group definition type

Resource group type	Explanation
1	The capacity is specified in unweighted CPU service units per second; the values must be between 0 and 999999. Minimum and maximum capacity have a sysplex scope. WLM ensures that the limits are met within the sysplex.
2	The capacity is specified as a percentage of the LPAR share; the value must be between 0 and 99, and the sum of all minimum LPAR shares for all resource groups of this type should not exceed 99 (warning message in AA). Minimum and maximum capacity have system scope. WLM ensures that the limits are met on each system within the sysplex.
3	The capacity is specified as a number of General Purpose Processors (CPs); a number of 100 represents the capacity of 1 CP. The number should be between 0 and 999999. Minimum and maximum capacity have system scope. WLM ensures that the limits are met on each system within the sysplex

The table in Appendix B, “CPU Capacity Table” of the *z/OS V1R8 MVS: Planning for Workload Manager*, SA22-7602-12 shows the service units per second by processor model. For more information on unweighted CPU service units, see the “System Resources Manager” chapter of *z/OS MVS Initialization and Tuning Guide*, SA22-7591.

Note: Capacity includes cycles in both TCB and SRB mode.

6.2.2 Scope of resource group definitions

In pre-z/OS V1R8 systems, resource groups defined a minimum and maximum limit in absolute amounts of service units in a sysplex environment. In order to meet the requirements, WLM exchanged information in the sysplex about the CPU consumption of all service classes in the resource group and attempted to distribute the available capacity to the service classes based on their goal achievement. This behavior can produce the effect that a service class could get a very high part of CPU usage on some systems, and less or no capacity on others in the same sysplex.

The existing resource group concept is:

- Sysplex-wide defined in service units
- Sysplex-wide managed

z/OS V1R8 enhancement

Therefore, in z/OS V1R8 with the new resource group types, the definition is in a sysplex-wide context, meaning that the resource group's minimum and maximum limits apply to all systems of the sysplex. Also, the association of service classes to the resource group is sysplex-wide, but the fulfilment of the limits is managed independently by each system, so a uniformly distributed disposition of CPU usage can be ensured over all systems.

In z/OS V1R8, multiple service classes can be assigned to a resource group, but this has no sysplex-wide effect any longer. In other words, resource groups are sysplex-wide defined but the definition applies to each system and is managed by each system also.

The new resource group concept is:

- Sysplex-wide defined, but definition applies to each system
- Managed by system

Important: In z/OS V1R8, the fulfillment of the limits for resource groups is managed independently by each system in order to distribute CPU usage uniformly over all systems in the same sysplex.

With the new enhancement, the benefit is dynamic and allows for system-wide resource entitlements that can grow with capacity upgrades. The most important benefit should be that it is now much easier to understand and much easier to use.

As shown in Figure 6-7 on page 127, multiple service classes can be assigned to a resource group, RGTE_A, but this has no sysplex-wide effect anymore. The definitions are in system units, as follows:

- LPAR capacity based on system weights
- LCP capacity

As a result, new resource groups are managed by system, and thus they must be evaluated on a per system basis. Resource groups grow automatically if systems are upgraded, making it much easier to understand and much easier to use to define system-wide resource entitlements.

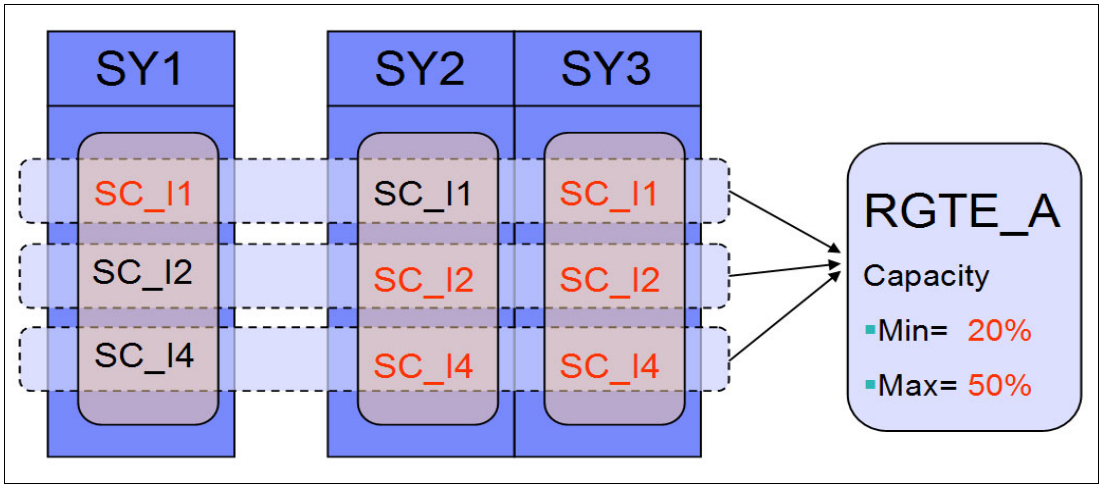


Figure 6-7 Resource group usage in z/OS V1R8

Details about LPAR share-based type

For the type 2 resource group, the minimum and maximum capacity is defined as a percentage of the share for the logical partition, namely LPAR share. The LPAR share is defined as the percentage of the weight definition for the logical partition to the sum of all weights for all active partitions on the processor. Calculation details for LPAR share of one system based on its LPAR weight are shown in Figure 6-8.

$$RG_{[SUs]} = \begin{cases} Softcap_{[SUs]} \bullet RG_{Limit[\%]} & \text{if softcap} < \text{ShareCapacity or LCPCapacity} \\ LCPCapacity_{[SUs]} \bullet RG_{Limit[\%]} & \text{if LCPCapacity} < \text{ShareCapacity or softcap} \\ ShareCapacity_{[SUs]} \bullet RG_{Limit[\%]} & \text{if ShareCapacity} < \text{LCPCapacity or softcap} \end{cases}$$

$$ShareCapacity_{[SUs]} = CECCapacity_{[SUs]} \bullet LPARShare$$

$$CECCapacity_{[SUs]} = \text{Capacity based on shared physical processors for the CEC}$$

$$LPARShare = \frac{\text{Weight(Current Partition)}}{\sum_{i}^{\text{all active partitions}} \text{Weight}(i)}$$

$$LCPCapacity_{[SUs]} = \text{Capacity based on shared processors available to the LPAR}$$

Figure 6-8 Calculation of LPAR share based on LPAR weight

6.2.3 Resource group example

The calculation for the new resource group limits is based on the LPAR weight as shown in Figure 6-8. It is based on the LPAR weight, but it is considered if a *softcap* is defined or if there are not enough logical CPs defined to support the weight definition.

An example for a partition running on a 2084-306 with a small weight (resulting in a 7.7% share) and a resource group with a maximum of 60% is shown in Figure 6-9 on page 129.

Note: The partition can use more than its share based on the weight so it can service classes on the partition which are not capped. But the service classes in the resource group cannot exceed the limit.

Description of the sample system

Let us look at an example for the calculation and usage of resource type 2. In our example we have the following scenario:

- Three workloads, one of them running in a resource group called ELPMAX, which is defined with a maximum of 60% of the LPAR share.
- The processor is a 2084-310 with 10 processors. Three of the processors are dedicated, so we have seven shared processors on the processor left.
- LPAR IRD1 has 6 logical shared processors assigned. It has a weight of 120 defined; the total weight of all active LPARs using shared processors is 1758.

A step-by-step explanation of Figure 6-9 on page 129 follows:

1. Look up the capacity table of your processor in

   `http://www.ibm.com/servers/eserver/zseries/srm/`
2. Take the row in the table that fits the SHARED processors that are assigned to this processor and get the SU/sec value. In our example we take the row 2084-307 (because of the 7 shared processors for the processor) which has 18202.5028 SU/second.
3. Multiply that number by the number of shared processors of the processor. In our example: 18202.5028*7 ~ 127417
4. Multiply the result by the weight value for the LPAR (taken from the HMC). In our example 127417*120=15290040
5. Divide the result by the sum of all weights of all active partitions on the processor. In our example, we get 15290040 / 1758 ~ 8697 SU/sec. This number is the LPAR share (in SU/sec).
6. Now we have the resource group ELPMAX of type 2 (% LPAR share) with a value of 60%, which means the resource group is allowed to consume 8697*60/100 ~ 5218 SU/sec on this system of the sysplex.

Figure 6-9 on page 129 shows the chart of this resource group's usage statistics. As shown in Figure 6-9 on page 129, after a short period of adjustment the resource group is really capped to a value below that maximum. In the small picture you also see the resource consumption of the other two workloads on the system. Even after these workloads have been stopped, the work that is running in ELPMAX is still capped.

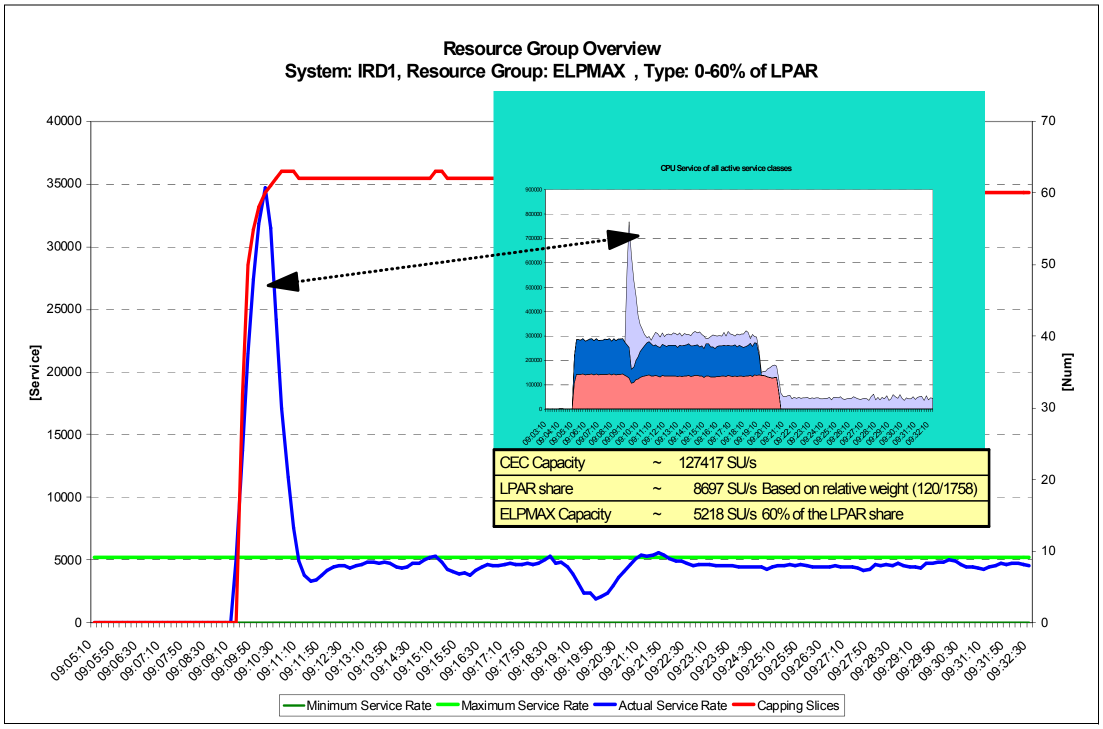


Figure 6-9 Resource group type 2 usage statistics

Important: The calculation is done on every member of the sysplex independently. Different members can produce different results, based on their relative weight. And the LPAR share (and thus the RG limit) can dynamically change as LPARs go online or offline, or if weight changes or capacity upgrades are done. Thus, the calculations are repeated by WLM every 10 seconds.

But your LPAR may in fact consume much more resources than LPAR share, if other LPARs are not using their weight. This is important to understand. And also, if you are running work that is capped by a %LPAR resource group at 60%, you will not see 60% CPU utilization in SDSF or RMF, probably much less.

If you have defined many LPARs and thus have “overcommitted” the processor, the LPAR share and the resulting SU for the resource group can be very low.

There are two additional special cases that may affect the calculation of the LPAR share:

- If you have defined a softcap, the softcap will override the calculation above.
- If there are not enough logical CPs defined for the LPAR to support the weight definition, the share of the logical CPs for the LPAR overrides the calculation.

Note: You may also find a different example in *z/OS V1R8 MVS: Planning for Workload Manager*, SA22-7602-12.

6.2.4 Migration and coexistence considerations

WLM migration and coexistence support is provided for the WLM component when running z/OS V1R8 as part of a mixed sysplex with APAR OA13837. Systems in the sysplex that are running at releases prior to z/OS V1R8 need this support for WLM to operate properly.

z/OS V1R8 made incompatible changes to the WLM service definition and WLM policy. The structure of the information that WLM exchanges in the sysplex has changed as well. Without this compatibility APAR, you can experience severe problems on the pre-z/OS V1R8 systems, such as the following:

- ABEND15F reason code 156
- Failures in LPAR management indicated by the message

  ```
  IWM054I FAILURE IN LPAR CPU MANAGEMENT, PROCESSING DISABLED
  ```

- When activating a WLM policy on a z/OS V1R8 system, the following message is received:

  ```
  IWM010I VARY WLM COMPLETED, BUT POLICY NOT ACTIVATED ON ALL SYSTEMS
  ```

  When issuing the D WLM,SYSTEMS command to display the Workload Management Status of systems without this compatibility APAR, the following message will show:

  ```
  ACTIVE, NOT RUNNING WITH ACTIVE POLICY
  ```

- WAIT STATE WAIT064-9 (WAIT064 reason code 9 rsn9 rc9) during IPL of a pre-z/OS V1R8 system when the WLM couple data set contains a WLM service policy that was activated from a z/OS V1R8 system. The wait state is due to an ABEND0C4 in IRATXCHP.

6.3 WLM support for ZIIPs

IBM has introduced a new special purpose processor called zSeries Integrated Information Processor (zIIP). This new special engine is available to be used with the z9 EC and z9 BC processors. It is designed to help integrate data across an enterprise and improve resource optimization to lower the cost of ownership for eligible data-serving workloads. z/OS manages and directs work between the general purpose processor and the zIIP. The first exploiter of the zIIP is DB2 UDB for z/OS V8 (with PTFs) and with z/OS V1R6 or later (with PTFs). The z/OS V1R8 base is the first release to have the support included in the base. Enclave SRBs are the types of workloads that have some portions of their work eligible for zIIPs. For more detailed information about zIIPs see Chapter 7, “zSeries Integrated Information Processors (zIIPs)” on page 149.

WLM support for zIIP usage is conceptually similar to the zAAP support. Alternate Wait Management (AWM) should be activated in order to have general purpose processor help for zIIPs. AWM is active by default in the IEAOPTxx parmlib member. For zIIPs, the IFAHONORPRIORTY parameter is always on. There is no zIIP equivalent to the zAAP specification of IFAHONORPRIORITY=YES.

zIIP processors are not actively managed by WLM. However, the zIIP using and delay samples are added to the set-of-samples and thereby influence the policy calculations. They are also included in the number of total usings and delays. zIIP processors have to be taken out of all calculations where only the amount of regular CPs is required.

6.3.1 Services changed for zIIPs

The following services have been modified to support zIIPs in z/OS V1R8:

- The TIMEUSED macro has been modified to allow zIIP execution time to be requested in addition to the standard CP consumption.
- The IWMEQTME service has been enhanced to include zIIP usage.
- The IWM4EDEL service has been enhanced to include zIIP usage.
- The SMF type 30 record (IFASMFR3) has been updated to include new zIIP consumption fields.
- The SMF type 99 record has new fields related to zIIPs.
- A new CVTZIIP bit can be used by software to determine if the z/OS release supports the new zIIP execution environment.
- The new single PROJECTCPU=YES parameter enables z/OS to collect zIIP usage as if one were configured, when the target workload is being run. This projection capability can be run at any time on a production environment if desired. RMF and SMF now show this calculated zIIP time so that an accurate zIIP projection can be made.

6.4 Improved management for zAAP workloads

There are two main improvements related to the management of zAAP workloads in z/OS V1R8. These enhancements are also available for pre-z/OS V1R8 systems via APAR OA14131(supervisor) and APAR OA13953 (SRM). They are:

- Improved IFAHONORPRIORTY=YES processing
- Improved WLM zAAP support

6.4.1 Improved IFAHONORPRIORTY=YES processing

There are two parameters in the IEAOPTxx parmlib member to control the zAAP dispatching of work, IFAHONORPRIORTY and IFACROSSOVER. In z/OS V1R8, the meaning of these parameters is changed in order to run more zAAP work on zAAP engines if there are still zAAP CP resources available. This support also removes the cases where zAAP work runs in general purpose processors although zAAPs does not need any help. In other words, it removes the problem of running zAAP work more than desired in general purpose CPs.

Pre-z/OS V1R8 support

IFACROSSOVER=YES is needed to specify IFAHONORPRIORTY=YES, and when IFAHONORPRIORTY=YES was specified, at each dispatch event, general purpose processors evaluated the zAAP eligible work and the priority of jobs was considered. When the highest priority unit of work was zAAP-eligible work, the general purpose CPs dispatched it. This resulted in more zAAP work run in general purpose CPs, although zAAPs did not need any help.

When there were no zAAPs online you may have needed to set both parameters to YES.

z/OS V1R8 support

IFACROSSOVER=YES is no longer needed to specify IFAHONORPRIORTY=YES, and they are not related to each other any longer. If you specify IFAHONORPRIORITY=YES (the default) in the IEAOPTxx parmlib member, you indicate that standard CPs may execute both Java and non-Java work in priority order, if zAAP processors are unable to execute all zAAP-eligible work. If you specify IFAHONORPRIORITY=NO, then zAAP-eligible work can execute on standard CPs, but at a lower priority than non-Java work.

Because standard CPs ask for help, not all CPs may process zAAP work at the same time. This change is intended to allow more zAAP-eligible work to run on zAAP processors while still remaining responsive to the zAAP demand.

The need for help is determined by the Alternate Wait Management (AWM) function of SRM for both standard CPs and zAAPs. Only standard CPs help each other, but they can also help zAAPs if YES is in effect. Note that specifying YES does not mean the priorities will always be honored. The system manages dispatching priorities based on the goals provided in the WLM service definition. AWM should not be disabled when IFAHONORPRIORITY=YES is in effect.

Note: In the IEAOPTxx parmlib member, the CCAWMT parameter specifies whether to activate or deactivate Alternate Wait Management (AWM). If AWM is active, SRM and LPAR cooperate to reduce low utilization effects and overhead. In an LPAR, some n-way environments with little work appear to require more capacity than expected because of the time spent waking up idle logical and physical processors to compete for individual pieces of work. If AWM is active, SRM and LPAR will reduce this unproductive use of processor so that capacity planning is more accurate and CPU overhead is reduced. Any value from 1 to 499999 makes AWM active.

- There can be a very significant effect on performance by using large values (such as 490,000) to activate AWM. IBM suggests accepting the default value. Specify a value for CCCAWMT in order to override the default after consulting with the IBM Support Center. Do not specify what is thought to be the default value as a means of disabling any OPT control. Default values sometimes change over time. Always convert a control to a comment (see the following) if the objective is to use the default value while retaining a reminder of the last value used.
- Specify a value greater than or equal to 500,000 to deactivate AWM. AWM should be active when Integrated Facility for Applications (IFAs) is present and IFAHONORPRIORITY=YES is in effect.

We recommend that you specify IFAHONORPRIORITY=YES or the default, which is the same.

With IFACROSSOVER=NO and IFAHONORPRIORTY=NO, general purpose processors do not examine zAAP work directly, but there are cases where zAAP-eligible work will run on general purpose CPs where necessary to handle conflicts for system resources between zAAP work and non-zAAP work.

When there are no zAAPs online, both parameters are ignored. If zAAPs are defined to the LPAR but are not online, zAAP work is processed by standard CPs in priority order. The system ignores the IFACROSSOVER and IFAHONORPRIORITY parameters in this case and handles the work as if it had no eligibility to zAAPs. zAAP times are reported in RMF and SMF for planning purposes.

The new behavior of general purpose processors according to a combination of values of these two parameters is shown in Table 6-2.

Table 6-2 General purpose processor behavior in the new support

CROSS	HONOR	Old Behavior	New Behavior
YES	YES	General purpose processors will run zAAP work in priority order, whether it is necessary or not.	General purpose processors will run zAAP work when help is needed and also when no non-zAAP work is ready (no regular CP work available).

CROSS	HONOR	Old Behavior	New Behavior
NO	YES	No zAAP work can ever run on general purpose CPs.	General purpose processors will run zAAP work *only* when help is needed by zAAPs.
YES	NO	General purpose processors will run zAAP work *only* when there is no non-zAAP work ready (no regular CP work available).	Not changed. General purpose processors will run zAAP work ONLY when there is no non-zAAP work ready (no regular CP work available)
NO	NO	General purpose processors will not run zAAP work.	Not changed. General purpose processors will not run zAAP work.

Note: An IFA is a z/OS internal designation for zAAPs.

Java work in priority order

Based on your IEAOPTxx settings for zAAPs, there are some automation changes you could make to ensure that Java work will continue to execute in priority order. If you have specified IFACROSSOVER=NO, Java work is not allowed to move to standard CPs if the last zAAP is not removed from the configuration. It also means that IFAHONORPRIORITY is not relevant. When the last zAAP is made unavailable, you may want to set IFAHONORPRIORITY=YES. In this situation, you may want to have Java work executed in priority order with other non-Java work. Automation detecting that the last zAAP was made unavailable could issue the SET OPT command with an IEAOPTxx parmlib member that has IFACROSSOVER=YES and IFAHONORPRIORITY=YES specified.

6.4.2 Improved WLM zAAP support

Support for zAAPs began in z/OS V1R6 and included resource monitoring and resource consumption reported via RMF. Today, zAAP workloads are managed based on CP utilization.

In z/OS V1R8, workload delays imposed by zAAPs are handled in the same way they are for standard CPs. Before this support, SRM was just passing RMF the workload delays. It did not take into account workload delays. However, as zAAP utilizations climb and the ratio between an application’s zAAP-resident work and its CP-resident work changes to use zAAPs more and CPs less, an improvement was needed, as follows:

- The processor projection and assessment logic is extended to project the change of dispatch priorities for zAAPs in the same way as for regular processors. This includes that a service class is helped because zAAPs state samples indicate that a change for this resource is beneficial for it.
- This change is related to optimization. The overall design idea for the zAAP support is to modify the Policy Adjustment (PA) routines such that zAAPs are taken in consideration when fixing CPU delays.
- The SRM logic remains unchanged but is applied separately for regular CPs and for zAAPs, and the assessment is done with the using and delay deltas of both processor types added together.
- The SRM logic is extended to project the change of dispatch priorities for zAAPs in the same way as for regular processors. Consumption and demand of zAAPs are included in WLM’s processor management algorithms. This includes that a service class is helped because zAAP state samples indicate that a change for this resource is beneficial for it.

Note: Previously, the title line of the SDSF DA panel showed the system start I/O rate in addition to MVS and LPAR views of CPU utilization. With z/OS V1R8, the system start I/O rate is omitted to make room for a zAAP view of CPU utilization.

6.4.3 IEAOPTxx parmlib member enhancements

The IEAOPTxx parmlib member provides three new options for zIIPs and zAAPs, as follows:

PROJECTCPU Specifies whether to activate or deactivate the projection of how work could be offloaded from regular CPs to special assist processors like zAAPs and zIIPs. When active, any work that is eligible for being offloaded to a special assist processor will be reported as Processor_on_CP work. This information can be used to understand the benefit of adding special processors into the configuration.

Specifying the PROJECTCPU parameter allows you to project zIIP (and zAAP) consumption when a zIIP (or zAAP) processor is not yet defined to the configuration. RMF and SMF show the potential calculated zIIP time, so that an accurate zIIP projection can be made. The PROJECTCPU option can be used while running the target workload, once all software is installed that enables hardware sizing data to be produced.

Value Range: YES/NO

Default Value: NO: Projection will not be done.

ZAAPAWMT Specifies an Alternate Wait Management (AWM) value for IBM zSeries Application Assist Processors (zAAPs) to minimize SRM and LPAR low utilization effects and overhead. In an LPAR, some n-way environments with little work may appear to have little capacity remaining because of the time spent waking up idle zAAPs to compete for individual pieces of work. The ZAAPAWMT parameter allows you to reduce this time so that capacity planning is more accurate and CPU overhead is reduced, even though arriving work might wait a longer time before being dispatched.

Value Range: 1-499000 microseconds (up to 1/5 second)

Default Value: 12000 (12 milliseconds)

ZIIPAWMT Specifies an Alternate Wait Management (AWM) value for IBM System z9 Integrated Information Processors (zIIPs) to minimize SRM and LPAR low utilization effects and overhead. In an LPAR, some n-way environments with little work may appear to have little capacity remaining because of the time spent waking up idle zIIPs to compete for individual pieces of work. The ZIIPAWMT parameter allows you to reduce this time so that capacity planning is more accurate and CPU overhead is reduced, even though arriving work might wait longer before being dispatched.

Value Range: 1-499000 microseconds (up to 1/5 second)

Default Value: 12000 (12 milliseconds)

6.5 WLM I/O priority for tape I/O

If you want workload management to manage I/O priorities for you, specify YES for I/O priority management. Workload management then manages your I/O priorities and includes I/O usings and delays in its execution velocity calculation. When you specify NO (which is the default), workload management sets the I/O priority to be the same as the dispatching priority, and I/O usings and delays are not included in the execution velocity calculation.

Prior to z/OS V1R8, SRM provided two I/O priorities for use by IOS, as follows:

- Channel path I/O priority
- DASD I/O priority

Both were dynamically calculated. However, there were cases where SRM did not provide any dynamic I/O priority, for example with tape I/O. IOS in z/OS V1R8 now has a new I/O priority scheme for tape based on a static mapping of importance. SRM now calculates a new static I/O priority for all address spaces and enclaves for tape devices to be used when no dynamic I/O priority has been assigned. The priority is derived from the importance of the unit of work when no WLM I/O priority is provided.

Dispatching priority by service class

Every I/O request now gets a priority value according to the service class of the work from where this I/O is initiated. A static mapping of dispatching priority assignment is given in Table 6-3.

Table 6-3 WLM tape I/O priority dispatching priority by service class

Dispatching Priority	**Service Class**
FF	SYSTEM, SYSSTC
FE	RESERVED
FD	Service Classes with Importance=1
FC	Service Classes with Importance=2
FB	Service Classes with Importance=3
FA	Service Classes with Importance=4
F9	Service Classes with Importance=5
F8	Discretionary work

6.6 Database buffer pool management by WLM

In z/OS V1R8, database buffer pool management by WLM is introduced. The aim is to help data managers benefit from the performance-driven storage management of WLM. Buffer pools occupy large amounts of central storage. With this change, buffer pool management is integrated with WLM’s management of storage resources in order to use storage resources more efficiently and assist customers in adjusting the size of a buffer pool. Buffer pool management is now based on the goal achievement of work according to the WLM policy. The buffer pool size is coordinated dynamically with the requirements of data requesters and the system’s sysplex workload.

6.6.1 New WLM services

In support of database buffer pool management in z/OS V1R8, a new delay type is being reported to WLM. A new mechanism is introduced to identify the user of buffer pools and report a user’s current state.

The prerequisites for buffer pool management with WLM and SRM are: the buffer pool is registered with WLM, and a Performance Block for the Database Manager exists. The buffer pool registration informs the data collection exit and adjustment exits of size limits using the new IWMMXDC and IWMMXRA services, respectively.

The following new WLM services support database buffer pools:

IWM4MREG The purpose of the IWM4MREG service is to register a resource type for delay monitoring. The service allows the caller to identify a resource type which may be involved in delays to work requests. The caller's home address space is assumed to be the owner of the resource to be monitored. The system may decide to increase or decrease the size of the resource to balance the associated delays.

This allows a database manager to register a resource type such as a buffer pool for delay monitoring, define size limits, and create internal representation of buffer pools to WLM and SRM.

IWM4MDRG The purpose of the IWM4MDRG service is to deregister a resource from monitoring which was previously registered via IWM4MREG. The service allows the caller to identify a resource that is no longer involved in delays to work requests. Thus, the system may no longer alter the size of the resource to balance associated delays.

This allows a database manager to deregister a resource, such as a buffer pool, for delay monitoring and remove its internal representation from WLM and SRM.

IWMMXDC The IWMMXDC service invokes the resource data collection exit specified. This exit returns information about the usage of the buffer pool or other resources which may be responsible for delays to work requests. The return and reason codes for IWMMXDC are those set by the exit invoked.

The requestor provides a parameter list format used by the data collection exit. The parameters are, among others, the current buffer pool size and its current hit-ratio needed by WLM and SRM.

IWMMXRA The IWMMXRA service invokes the resource adjustment exit. This exit makes the adjustments indicated for the buffer pool or other resources that may be responsible for delays to work requests. The return or reason codes for IWMMXRA are those set by the exit invoked.

The requestor provides a parameter list format used by the resource adjustment exit implementation containing the buffer pool for use by WLM and SRM.

6.6.2 Changed WLM services

The following WLM services have changed in support of database buffer pool management in z/OS V1R8:

IWM4MCHS The IWM4MCHS service reflects the current work request state about a particular report of buffer pool delays.

IWM4MCRE The IWM4MCRE service creates a delay monitoring environment to create a specific buffer pool management-only monitoring environment.

IWM4MINI The IWM4MINI service provides further information on the monitoring environment to associate a buffer pool management-only monitoring environment to an enclave that uses the buffer pool.

IWMMRELA The IWMMRELA service relates different monitoring environments to relate a buffer pool monitoring environment to an enclave's buffer pool management-only monitoring environment.

A new mechanism is introduced to associate a buffer pool with a user. Buffer pool users are CICS or IMS transactions represented by a PB (performance block) and DDF enclaves represented by an enclave. At present, buffer pool delay monitoring is only possible through performance blocks associated to enclaves, not for address spaces. This is also a DB2 requirement to be made available in a future release of DB2.

A new mechanism is introduced to track and report a user's current state. These new delays are reported to dedicated performance blocks (PBs) of data requesters suffering performance delays while waiting for data that still has to be loaded into the buffer pool. WLM tracks when users have to wait for data in the buffer pool.

6.6.3 Database buffer pool management flow

Figure 6-10 on page 138 shows a typical sequence of a database manager reporting buffer pool delays that have been triggered by an application's request. The prerequisites for database buffer pool management together with WLM and SRM are: the database buffer pool is registered with WLM, and a performance block for the database manager exists.

The new WLM services described in 6.6.1, "New WLM services" on page 135 are created to perform buffer pool registration and to use the new data collect and adjustment exits, as well as controlling the size limits.

Database applications

The following steps show the flow for buffer pool management for an application:

1. The database application registers its buffer pool with the database manager using the new WLM service as follows:

```
IWM4MREG
   RESOURCE_TYPE <- BUFFER_POOL,
   RESOURCE_TKN -> res_tkn,
   RES_ADJ_SIZE <- adjustment size,
   RES_MIN_SIZE <- minimum size,
   RES_MAX_SIZE <- maximum size,
   RES_ADJ_EXIT <- exit pointer,
   RES_DATA_EXIT <- exit pointer
```

   A buffer pool performance block (PB) is created and the data collection and adjustment exits are established.

2. The application creates an enclave, and a buffer pool PB is created.

Steps 1 and 2

The prerequisites for database buffer pool management with WLM are: the database buffer pool is registered with WLM, and a performance block for the database manager exists.The buffer pool vector table gathers all the buffer pool management blocks. The buffer pool management blocks are created at registration time. The buffer pool registration informs about data collect and adjustment exists, as well as size limits.

3. The application task joins the enclave.
4. The application starts to use the buffer pool. At this time, the buffer pool performance block is related to the enclave performance block. During this period state changes are recorded in the enclave's performance block.

5. WLM collects performance block states.

Steps 3, 4, and 5

The database manager creates the performance block that will be exclusively used to record buffer pool delays. The performance block is associated with the requesting enclave and the buffer pool performance block is related to the dedicated enclave's performance block.

6. WLM collects buffer pool size and hit ratio.
7. WLM algorithms verify buffer pool size.

Steps 6 and 7

The database manager reports any buffer pool delays in the buffer pool performance block. Delays are caused when requested data is not available in the buffer pool that must be retrieved by the buffer pool manager through I/O operations. The delays are considered in the performance calculations of the enclave's classification. These will directly affect the enclave's service class period PI.

8. If necessary, WLM calls the adjustment exit to change the buffer pool size.

Step 8

After having satisfied the enclave's request, the dedicated buffer pool performance block is released from its association with the enclave. The buffer pool performance block can be reused for reporting states of another requesting enclave.

Once delay values are collected, WLM gathers performance data to recognize whether a buffer pool size change is necessary. WLM collects the performance block states. The collected states are reported to the adjustment algorithms. Periodically, WLM polls the current buffer pool size. If buffer pool delays are a problem, a calculation is made and the buffer pool manager is instructed to adjust the buffer pool size.

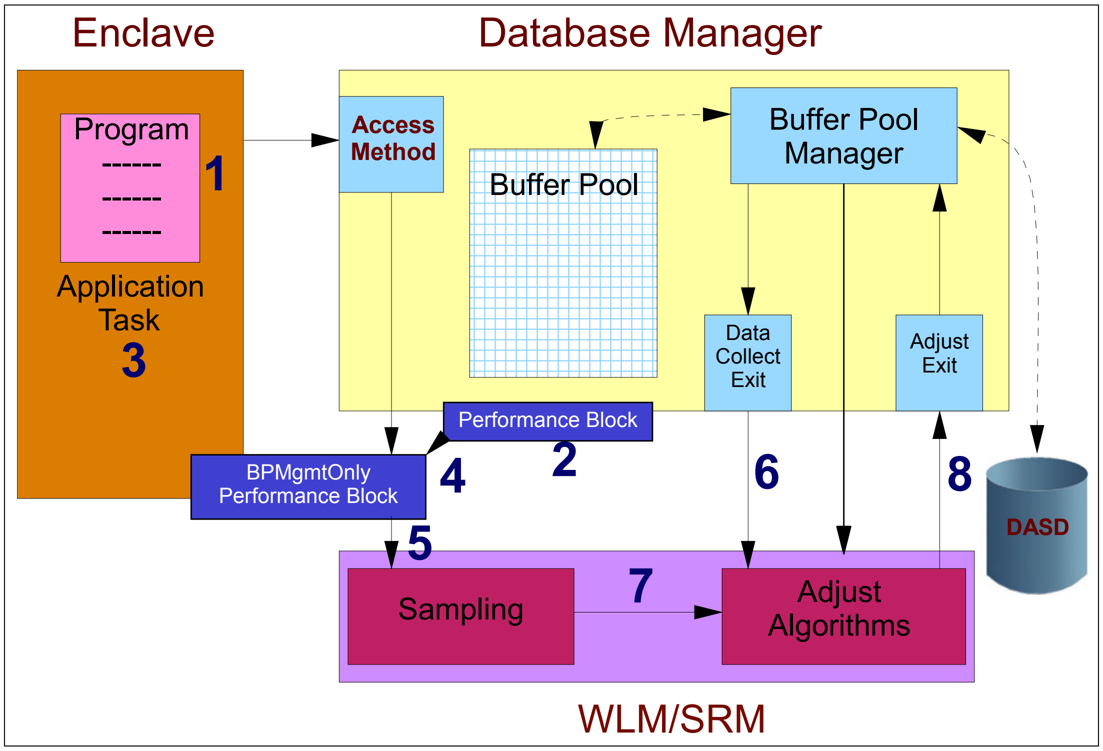


Figure 6-10 WLM database buffer pool management flow

Buffer pool management considerations

At present, buffer pool delay monitoring is only possible through performance blocks associated to enclaves, which is a requirement for DB2. With the current implementation, enhancements can easily be made to also support buffer pool delay monitoring through performance blocks associated to address spaces.

6.6.4 SMF type 99 record

The SMF type 99 record subtype 2 is extended to include fields with buffer pool delay samples gathered in the last interval, and the total or accumulated buffer pool delay samples for a specific service class period.

6.7 WLM EWLM enhancements

EWLM is a product intended to deliver levels of performance management not previously achievable for distributed systems by bringing self-tuning technology to the set of servers, routers and other Java technology-capable devices supporting multi-tiered applications. Its management reach can encompass your entire TCP/IP infrastructure, extending even to Internet servers over which you have some control. Or, it can be deployed to handle a critical subset of your IT infrastructure.

EWLM allows you to set response time and other goals for end-user service, and takes actions such as routing, so that they are met. It can make possible cross-platform performance analysis and capacity planning. All of this can help to substantially reduce the number of people and systems needed to meet the goals for end-user service required by the business.

The following enhancements to WLM have been made to EWLM:

- EWLM performance enhancements for DB2 DDF
- EWLM support for WLM execution delay monitoring services
- Assignment of WLM service and report classes based on EWLM classification
- EWLM zAAP support
- Process entitlement groups

Important: The acronym “ARM” in the context of EWLM does not mean “Automatic Restart Manager”. For EWLM it means “Application Response Measurement” (ARM). ARM is a standard of the Open Group for instrumentation of applications. EWLM has chosen to use the ARM 4.0 standard as the suggested way for middleware applications to instrument their software for EWLM. But in z/OS, existing middleware such as DB2, WAS, CICS, and IMS has used WLM proprietary instrumentation interfaces for years. The idea is to map this existing instrumentation to ARM calls as much as possible, so that EWLM can simply pick up that data.

6.7.1 WLM performance enhancement for DB2 DDF

Previously, DB2 DDF EWLM support was only available as a USERMOD and was not integrated into DB2. This support has a large amount of overhead for the EWLM exploitation for DB2 DDF. Also, the implementation did not include support for EWLM V2.1.

With z/OS V1R8, there are APARs to provide enhancements for WLM and DB2 that now ensure an official EWLM exploitation of DB2 DDF. This new support reduces the previous EWLM overhead within DB2. For EWLM exploitation, the overhead is significantly reduced down to 1-4%, depending on workload characteristics.

DB2 DDF overview

In a distributed data environment, DB2 applications can access data at many different DB2 sites and at remote relational database systems. A distributed environment relies on the distributed data facility (DDF), which is part of DB2 UDB for z/OS. DB2 applications can use DDF to access data at other DB2 sites and at remote relational database systems that support Distributed Relational Database Architecture™ (DRDA®). DRDA is a standard for distributed connectivity. All IBM DB2 servers support this DRDA standard.

DDF also enables applications that run in a remote environment that supports DRDA. These applications can use DDF to access data in DB2 servers. With DDF, you can have up to 150 000 distributed threads connect to a DB2 server at the same time. A thread is a DB2 structure that describes an application's connection and traces its progress.

DB2 DDF instrumentation

Instrumentation provides the data to create the topology of a transaction. It describes what server instances or application instances a transaction flows through. In a typical Web transaction (for example, buy books) the transaction starts at a Web server (edge server), flows to an application server, then to a database server. Without instrumentation, the transaction would look like three separate processes with three separate response times rather than one transaction. This is key to EWLM, providing end-to-end topology views and statistics for business transactions. To report on these transactions and potentially manage the EWLM management domain based on these statistics, Application Response Measurement (ARM) 4.0 has been implemented for application instrumentation. Currently there are three types of applications instrumented with ARM 4.0:

- WebSphere® Application Server
- DB2 UDB
- Web servers

To monitor work requests in a multi-tiered heterogeneous environment, you must identify and track the performance of those requests across server boundaries. It is possible to collect this data using versions of middleware that have been instrumented with the ARMV4.0 standard.

z/OS V1R8 support for DB2 DDF instrumentation

With z/OS V1R8, DB2 across all platforms made the decision that the EWLM instrumentation should be on the network flow level and not on the commit scope level (as it was on z/OS with the existing WLM instrumentation), as shown in Figure 6-11. In this example, the number of WLM service calls has increased from 2 calls to 10 calls. The benefit is the increased granularity of measurements in the EWLM Control Center (measurements are done on the network flow level instead of the commit level). This eliminates network delays and delays that are actually caused by the client WAS requesting database services. Because the increased number of WLM calls results in a higher CPU consumption, APAR OA12005 addresses that problem and significantly reduces that overhead. The overhead is dependent on the application itself, as a DB2 transaction with many network flows and short database interactions, as shown in Figure 6-11, tends to have a larger overhead.

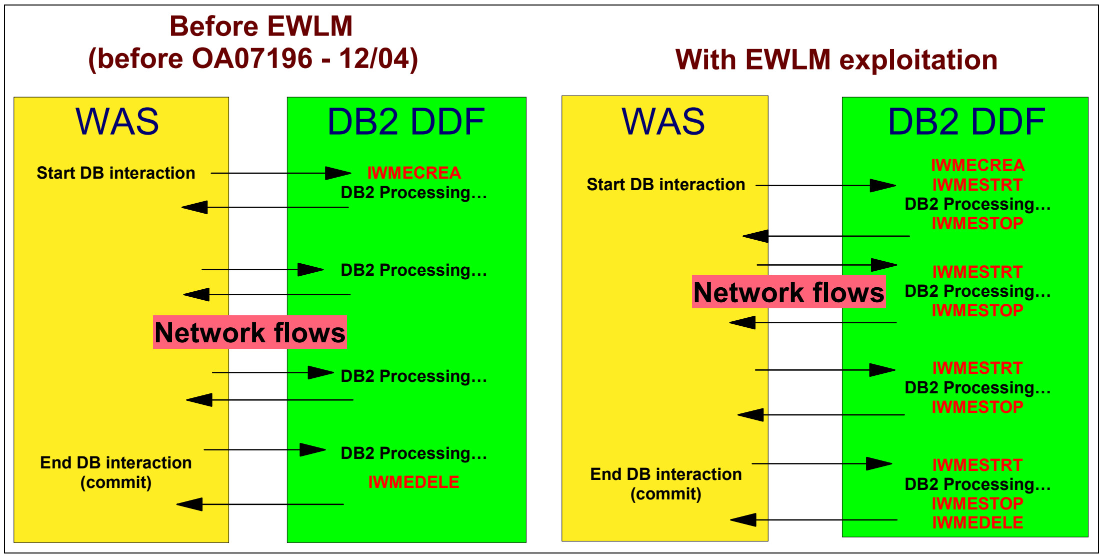


Figure 6-11 DB2 DDF instrumentation

Note: To use this new support, install EWLM V2.1 and apply APAR PK11801.

6.7.2 EWLM support for WLM execution delay monitoring services

From the execution delay monitoring services, workload management knows how well work is executing, and where any delays are occurring. The execution delay monitoring services are for complex work manager configurations that process across systems in a sysplex, but do not allow MVS to individually manage resource consumption of the transactions. The services allow MVS to recognize additional address spaces that are processing transactions.

When the execution delay monitoring services are used, MVS can allocate resources for address spaces based on the behavior of the transactions being serviced by them. The services also provide execution delay information, so that your customers can determine where work is being delayed. They can then adjust the work manager configuration to consistently meet the goals. Only response time goals can be used with execution delay services.

Server regions using the enhanced execution delay monitoring services are now be fully integrated into EWLM.

Current support for execution delay monitoring services

Today, middleware using the WLM delay monitoring services is not able to participate in EWLM. The most prominent exploiters of these services are CICS and IMS. There are also missing display commands for EWLM/ARM instrumented processes and missing IPCS support.

Changes to WLM to participate in EWLM

With z/OS V1R8, support is provided for EWLM platform support on z/OS for WLM execution delay monitoring services. Now, exploiters such as CICS and IMS have to enhance their instrumentation to exploit the new features. In addition, there are enhanced display facilities and IPCS support.

Note: APAR OA12935 for z/OS V1R7 is available, and this support is integrated into z/OS V1R8.

Changed WLM services

The problem that customers faced was that the WLM execution delays monitoring services could not participate in EWLM. Thus, customers could not see what was going on on the z/OS platform via EWLM. To fix this, the interfaces of the execution delay monitoring services were created. The following services are changed for this new support:

- IWM4MCRE
- IWM4MINI
- IWMRPT
- IWMMNTFY
- IWMMEXTR
- IWMMXFER
- IWMCLSFY

Operator command change

The MVS operator command D WLM,AM,ALL, shown in Figure 6-12 on page 142, and the IPCS command WLMDATA ARMAGENT, shown in Figure 6-13 on page 143, to provide detail and summary reports, are new with z/OS V1R8.

```
10.09.04 WLMH           D,WLM,AM,ALL           ? New suboption ALL displays registered
AS
10.09.04 WLMH           IWM075I  10.09.04  WLM DISPLAY 108
  EWLM ARM SERVICES ARE ENABLED
  EWLM MANAGED SERVER JOBNAME=WLJEFTZO ASID=002E
  EWLM POLICY NAME=POLICY-WITH-ZOS-ATTRIBUTES-06JUL2005
  NUMBER OF REGISTERED PROCESSES=2, APPLICATIONS=2
  ADDRESS SPACES CURRENTLY REGISTERED WITH EWLM ARM:
    JOBNAME=MYDDF ASID=0030
      APPLICATION=DDF
        IDENTITY PROPERTIES=0 CONTEXT NAMES=0
        STARTED APPLICATION INSTANCES:
          EWLMSUBS
            TRAN=1 GROUP=DDFGroup1
        REGISTERED TRANSACTIONS:
          SYS_DefaultZWLMTransactionName
    JOBNAME=WLJEWR72 ASID=002F
      APPLICATION=Web-Server
        IDENTITY PROPERTIES=7 CONTEXT NAMES=1
        STARTED APPLICATION INSTANCES:
          <name omitted>
            TRAN=1 GROUP=DDSGroup
        REGISTERED TRANSACTIONS:
          DDS-HttpTransaction
```

Figure 6-12 Enhanced D WLM,AM,ALL operator command

For the IPCS support, enter the WLMDATA ARMAGENT command to display the content of EWLM/ARM control blocks, as follows:

```
WLMDATA ARMAGENT [SUMMARY] displays short information.
WLMDATA ARMAGENT DETAIL for extensive information.
```

```
------------------------- IPCS Subcommand Entry ----------------------
 Enter a free-form IPCS subcommand or a CLIST or REXX exec invocation

===> wlmdata armagent detail

----------------------- IPCS Subcommands and Abbreviations --------------------
 ADDDUMP           ! DROPDUMP, DROPD   ! LISTDUMP, LDMP    ! RENUM,    REN
 ANALYZE           ! DROPMAP,  DROPM   ! LISTMAP,  LMAP    ! RUNCHAIN, RUNC
 ARCHECK           ! DROPSYM,  DROPS   ! LISTSYM,  LSYM    ! SCAN
 ASCBEXIT, ASCBX   ! EPTRACE           ! LISTUCB,  LISTU   ! SELECT
 ASMCHECK, ASMK    ! EQUATE,   EQU, EQ ! LITERAL           ! SETDEF,   SETD
 CBFORMAT, CBF     ! FIND,     F       ! LPAMAP            ! STACK
 CBSTAT            ! FINDMOD,  FMOD    ! MERGE             ! STATUS,   ST
 CLOSE             ! FINDUCB,  FINDU   ! NAME              ! SUMMARY,  SUMM
 COPYDDIR          ! GTFTRACE, GTF     ! NAMETOKN          ! SYSTRACE
 COPYDUMP          ! INTEGER           ! NOTE,     N       ! TCBEXIT,  TCBX
 COPYTRC           ! IPCS HELP, H      ! OPEN              ! VERBEXIT, VERBX
 CTRACE            ! LIST,     L       ! PROFILE,  PROF    ! WHERE,    W
```

Figure 6-13 IPCS command support

6.7.3 Assign a service and report class on EWLM classification

When running a transaction across different platforms, EWLM is able to view how the transaction is working on every platform, whereas WLM can only see what is going on on the z/OS platform. To associate a work request to a service class, a work manager passes some attribute values that EWLM matches against filters and z/OS WLM matches against subsystem work qualifiers. The risk is that the attribute values for work requests received by the entry application of an EWLM Management Domain might be different from those passed to z/OS WLM, especially if the classification on the distributed platform happened in an application environment different from the entry point on z/OS. For this reason it may not always be possible to have a one-to-one mapping for EWLM service classes and z/OS WLM service classes. Currently, there is no way to use the EWLM classification information to manage transactions on z/OS.

EWLM policy versus WLM policy

There are two workload policies: EWLM and WLM. The EWLM policy manages the workload on different platforms whereas the WLM policy manages the workload on the z/OS platform only. Thus, the scope of the EWLM policy is broader than the scope of the WLM policy. The changes in z/OS V1R8 are as follows:

- Allow association of an EWLM transaction class with local z/OS WLM service class and/or report class.
- The mapping between EWLM and WLM is defined in the WLM service definition.
- If a mapping exists for a EWLM transaction class, then WLM assigns the specified service class instead of performing the normal classification process as it does today.
- Management is still controlled by WLM and aims to satisfy the goal of the service class assigned to that work on z/OS.
- To enable the definition of mappings in the WLM administrative application, a new subsystem type, EWLM, is introduced. For this subsystem type classification rules can be defined that correlate EWLM service classes (ESC rule) or EWLM transaction classes

(ETC rule) with WLM service classes and report classes, as shown in Figure 6-14 on page 145.

As usual, wildcards in the WLM qualifier names (EWLM transaction/service class name) are allowed.

This new support allows WLM classification to use the EWLM transaction class determined at the edge server for assignment of a WLM service and or report class.This is a first step towards management of end-to-end goals.

Note: This support is available with APAR OA12784 (UA23218 for z/OS V1R6 and UA23219 for z/OS V1R7 and is fully integrated into z/OS V1R8.

Subsystem EWLM in the WLM policy

The subsystem type EWLM comprises a special function of WLM: it assigns EWLM transaction classes and service classes to WLM service classes. Thus, WLM supports the following:

- Assigning EWLM service classes to WLM service or report classes
- Assigning EWLM transaction classes to WLM service or report classes

Assigning several EWLM service or transaction classes to a WLM service class means:

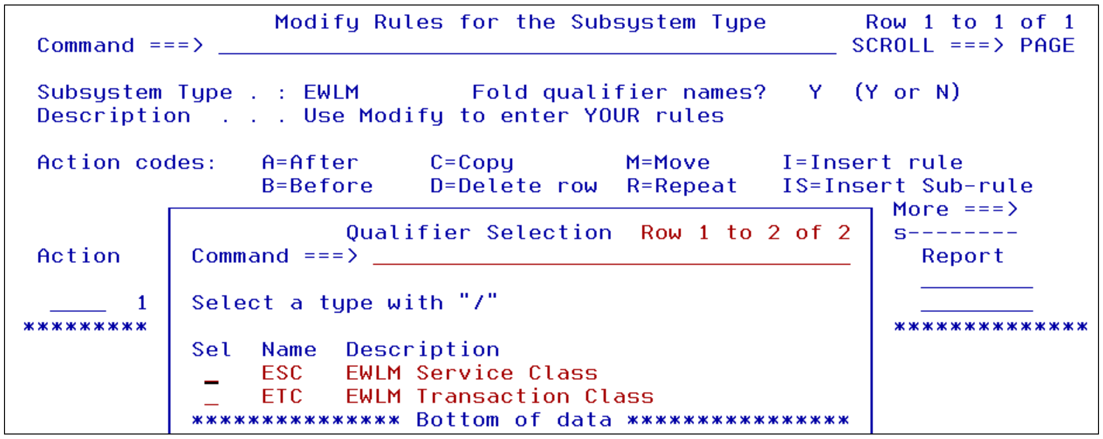
- End-to-end work requests that are already classified in one of those EWLM service or transaction classes are managed on z/OS according to the goals of the assigned WLM service class.
- If an EWLM service or transaction class is correlated with a WLM service class, WLM assigns the specified service class right away instead of going through the z/OS classification process.

Thus, end-to-end work processed by a z/OS subsystem can be managed towards different goals than local z/OS work, and can also be reported in different report classes than local z/OS work. This function has the following advantages:

- Management is still controlled by WLM and is done according to the goal of the service class assigned to that work on z/OS.
- On the EWLM control center, the performance administrator has access to a report that integrates the end-to-end work within the same transaction class for reporting and management.

New work qualifiers

In order to assign a WLM service class and a WLM report class according the EWLM classification, the work qualifiers EWLM service class (ESC) and the EWLM transaction class (ETC) are implemented.



```
                  Modify Rules for the Subsystem Type       Row 1 to 1 of 1
Command ===> ____________________________________________ SCROLL ===> PAGE

Subsystem Type . : EWLM        Fold qualifier names?   Y  (Y or N)
Description  . . . Use Modify to enter YOUR rules

Action codes:   A=After     C=Copy        M=Move     I=Insert rule
                B=Before    D=Delete row  R=Repeat   IS=Insert Sub-rule
                                                                More ===>
                        Qualifier Selection  Row 1 to 2 of 2    s--------
Action          Command ===> _________________________________    Report

  ____  1       Select a type with "/"                           ________
*********                                                        ________
                Sel  Name  Description                          *************
                 _   ESC   EWLM Service Class
                 _   ETC   EWLM Transaction Class
                *************** Bottom of data ****************
```

Figure 6-14 New work qualifiers for EWLM

Using the new work qualifiers

ESC and ETC are only supported for subsystem EWLM. The maximum length of an EWLM ESC or ETC name is 64 characters. To specify the entire name, eight rules would be required. Thus, the maximum length for ESC and ETC is 32 characters.

There is no service class required for the classification rule and there is no default service class required for subsystem EWLM. If the processing of the classification rules does not result in an assignment of a valid WLM service class, the usual subsystem type-specific WLM classification rules are applied to assign a WLM service class.

EWLM service or transaction class names can only be specified in EBCDIC characters. Therefore, to allow for defining a correlation, the transaction class names in the EWLM domain policy should only contain characters that have an EBCDIC representation.

EWLM service or transaction class names are specified in mixed case (fold qualifier name = N) and the comparison is also done case-sensitive.

Subrules must use the same qualifier as their parents, that is, either ESC or ETC rules.

The specified correlation rules are processed top down. This means that the first correlation found for an EWLM service or transaction class is applied.

WLM IWMCLSFY service changes for EWLM

The purpose of this service is to factor in available information about an arriving work request in order to associate a service class and possibly a report class with it. To assign a work request a WLM service class and a report class based on the EWLM service class or on the EWLM transaction class, the following changes were made to the IWMCLSFY service:

EWLM_CHCORR	An optional output parameter, which contains the cross-platform Enterprise Workload Management (EWLM) child correlator associated with the instantiated sub work request.
EWLM_CHCTKN	An optional output parameter, which contains the cross-platform Enterprise Workload Management (EWLM) child correlator token associated with the instantiated sub work request.
PLISTVER=7	An optional input parameter that specifies the version of the macro. The range version of the macro is now from 0 to 7. PLISTVER determines which parameter list the system generates. PLISTVER=7 supports EWLM_CHCORR, EWLM_CHCTKN and EWLM_OUTCORR in addition to the parameters supported by versions 0 to 7.

Migration considerations for EWLM

In order to use this feature in prior releases of z/OS V1R8, install APAR OA12784 to increase the WLM functionality level of the policy to LEVEL017. The APAR does the following:

- The functionality level is increased to 17 if the policy contains the subsystem EWLM with specified default service/or report classes and the new work qualifier rules.
- The WLM level in the policy header is set to LEVEL017 if the policy is activated from a system with APAR OA12784 applied.
- Lower-level systems in the same sysplex without APAR OA12784 applied will ignore the ESC/ETC rules. Classification will only use the subsystem-specific classification rules as before.
- Lower-level systems in the same sysplex without APAR OA12784 applied will not be able to activate, extract, or modify a new policy that is at LEVEL017 with the new constructs that is installed by a system that has APAR OA12784 applied.
- Lower-level systems in the same sysplex (without OA12784 applied) will be able to activate, extract, or modify a new policy that is at LEVEL017 without the new constructs that is installed by a system that has OA12784 applied.
- A policy ISPF data set with the new constructs that is at LEVEL017 cannot be modified with a WLM administrative application that does not have APAR OA12784 applied.

Important: Until you have installed APAR OA12784 on all the systems in the sysplex, avoid any change to modify, install, or activate a service definition policy from a system that has APAR OA12784 installed.

6.7.4 EWLM zAAP support

In z/OS V1R8, the changes to WLM now include zAAP data in the CPU using and delay samples as well as in the CPU service times to be reported to EWLM for zAAP processors. But EWLM cannot differentiate between CP, zAAP, or zIIP processor activities. This provides consistent CPU measurements in both WLM and EWLM.

Migration considerations

To use this feature on pre-z/OS V1R8 releases, apply APAR 0A12786 as follows:

- For zAAP support:

  UA23190 for z/OSV1R6 and UA23209 for z/OS V1R7
- zIIP support:

  Available with the appropriate FMIDs for z/OS 1R6 and z/OS V1R7

The new support is fully integrated into z/OS V1R8.

6.8 New defined service classes

Until now, it was not possible to prioritize system-like work at a granular level because everything either had to go to service class SYSSTC or installations had to define high important service classes with very stringent goals to cope with system work. In addition, resource groups were always sysplex-wide and it was not possible to define resource entitlements on a system basis. Resource groups were defined in service units that need adjustments when systems grow.

In z/OS V1R8, two new concepts are added to the WLM policy as follows:

Process ranking	There are five new predefined system service classes, SYSSTC1 to SYSSTC5
Enhanced resource groups	There are two new types of resource groups in the service class definitions: ► Min/Max capacity specified as a percentage of the "LPAR capacity" ► Or as percentage of a "single processor capacity" (also called: "Number of CPs times 100")

Note: Enhanced resource groups are described in 6.2.1, "Resource group enhancements in z/OS V1R8" on page 124.

6.8.1 New service classes

The service classes SYSSTC1, SYSSTC2, SYSSTC3, SYSSTC4, and SYSSTC5 are provided mostly for future z/OS and EWLM support. Service class SYSSTC and the SYSSTCx service classes are congruent as far as management of work is concerned. Work assigned to any of these service classes is managed identically to work assigned to any other. Currently, there is no technical reason to choose SYSSTCx as an alternative to SYSSTC. Many displays, for example SDSF displays, provide the service class assigned to an address space. It is possible that one might assign SYSSTCx to convey some meaning that would be similar to a report class.

These new service classes allow installations to group system work into different levels based on their relative importance. This would allow for dynamic and system-wide resource entitlements that can grow with capacity upgrades.

6.8.2 Migration considerations for WLM enhancements

This support is available with z/OS V1R8 only. For previous releases for coexistence, the compatibility APAR OA13837 is required.

You should ensure that z/OS V1R7 releases in a sysplex can tolerate the new constructs in the z/OS V1R8 workload manager (WLM) service definition. The PTFs that are required for z/OS V1R7, z/OS V1R6, and z/OS V1R5 are mentioned in APAR OA13837. Consider the following:

- If the WLM service definition in the couple data set has not been, and will not be, created by a z/OS V1R8 system, the PTF is not required.
- If the WLM service definition in the couple data set has been created by a z/OS V1R8 system, an earlier system without the PTF installed will fail during IPL with a wait state.
- If a WLM service definition is activated from a z/OS V1R8 system, any earlier systems currently running without the PTF installed will not be able to activate the new policy.

In order to use the two new types of resource group definitions, the WLM couple data set should be reformatted with the LEVEL019, which is introduced with z/OS V1R8. APAR OA13837 allows z/OS V1R6 and z/OS V1R7 to use this new functionality by increasing the format level from LEVEL017 to LEVEL019.

7

zSeries Integrated Information Processors (zIIPs)

This chapter discusses the new, special purpose processors known as zSeries Integrated Information Processors (zIIPs). The z9 BC and z9 EC servers support zIIPs.

This chapter explains the following topics:

- History of special purpose processors
- How zIIPs work
- How zIIP exploitation can be activated
- Capacity planning for zIIPs
- RMF support for zIIPs
- Instances where zIIP would not be exploited
- DRDA
- Star schema
- Index maintenance
- SRBs

7.1 History of special purpose processors

With the new z9 EC and BC servers, a new special purpose processor is introduced, known as the System z9 Integrated Information Processor (zIIP); see Figure 7-1.

Prerequisites for initial zIIP usage are:

- z/OS V1R6 or higher
- DB2 for z/OS V8 or higher
- z9 EC or z9 BC with zIIP processors

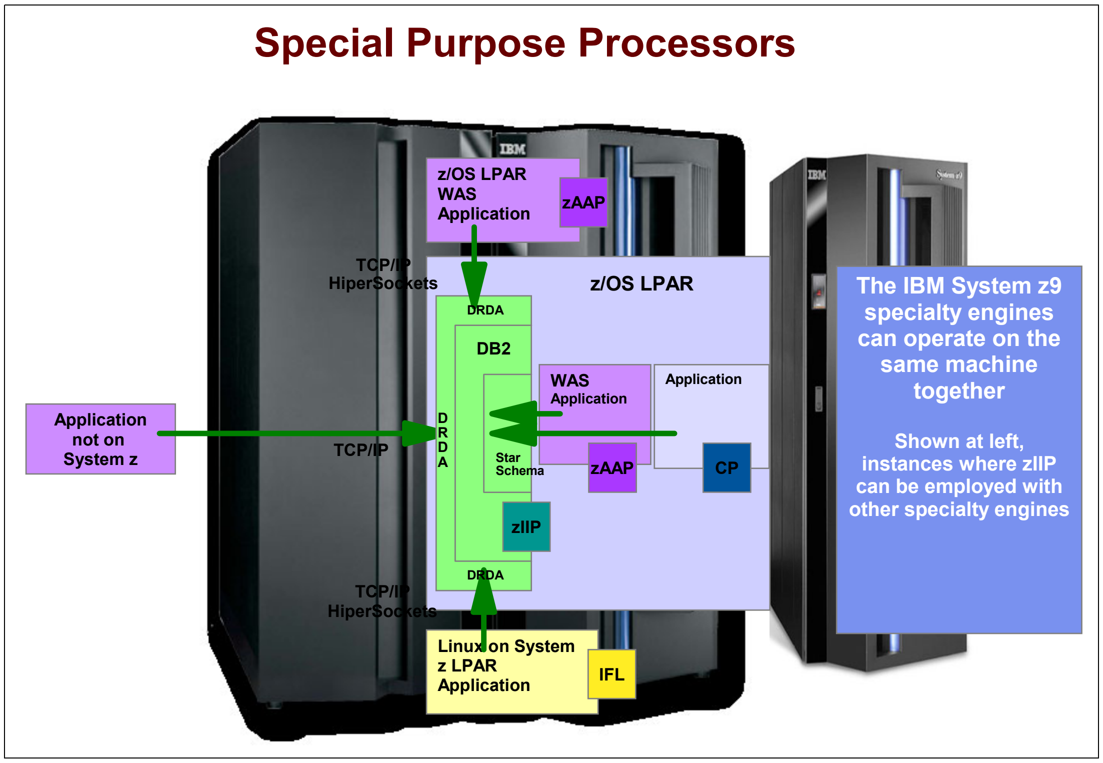


Figure 7-1 Special purpose processors

zIIPs are not the first special purpose processors. In the following sections, we describe other IBM special purpose processors that have been developed through the years.

SAP special purpose processors

The first special purpose processor was the System Assistant Processor (SAP), which was implemented to perform I/O processing. It was introduced in 1994 with the first 9672 processors.

ICF special purpose processor

The second special purpose processor was an ICF. Because the CF was implemented using 9672 CP chips, it was possible to implement “internal CFs”. The introduction of ICFs in 1997 allowed the backup CF to be an LPAR on a production processor. Also, with the introduction of CF duplexing, internal CFs began to be used on both production processors.

IFL special purpose processor

The third special purpose processor was the zSeries Integrated Facility for Linux (IFL). IFLs were introduced in 2001. IFLs are processors dedicated to Linux-only workloads. They can not be mixed with general purpose processors in an LPAR.

zAAP special purpose processor

The fourth special purpose processor was the zAAP, or System z9 Application Assist Processor. Introduced in 2004, zAAPs are a type of special purpose processors used for the IBM Java Virtual Machine on z/OS to support Java workloads. They can be used with the z990, z890, z9 EC, and z9 BC.

Following are some zAAP limitations:

- zAAPs cannot be IPLed.
- zAAPs only execute z/Architecture® mode instructions.
- zAAPs do not support all manual operator commands like PSW restart, LOAD or LOAD derivatives.
- zAAPs have no I/O interrupts, no clock comparator interrupts, and no affinity scheduling.

7.1.1 zIIP special purpose processor

In 2006, IBM introduced the fifth special purpose processor, zIIP. A zIIP is designed to help improve resource optimization for eligible data workloads within an enterprise. It most closely resembles the existing zSeries Application Assist Processor (zAAP).

zIIPs are orderable on System z9, as follows:

- By feature code 7815 on the z9 EC
- By feature code 7868 on the z9 BC

To place your order or for more information, work through your normal IBM Sales Channel.

A zIIP is designed to enable redirection of appropriate system overhead functions to zIIP engines. The initial IBM exploiter is DB2.

zIIP processing

Note that the same limitations exist for zIIPs and zAAPs:

- Neither can be used to IPL an operating system or sustain system operation by itself
- General interrupts cannot be handled by these processors
- I/O cannot be managed by these processors
- Timers cannot be set by these processors

zIIP processor pool

zIIPs have their own processor pool, which is Pool 6. The number of zIIPs per z9-109 cannot exceed the number of general purpose CPs. No changes are anticipated to DB2 for z/OS V8 applications.

The use of zAAPs and zIIPs by a single transaction flow is not mutually exclusive. The two specialty engines are designed to run two different types of work. For example, it is possible for transactions or batch to use a zIIP to handle eligible portions of the DRDA over TCP/IP processing for DB2 for z/OS V8, while also running Java stored procedures on a zAAP on the same System z9.

Customers are able to have a zIIP defined as a capacity backup (CBU) processor. The maintenance price for the zIIPs is lower than that of a general purpose CP, and it is expected to be similar to the maintenance price for other specialty engines. IBM does not impose software charges on zIIP capacity. The amount of general purpose processor savings varies, based on the amount of workload executed by the zIIPs.

Note: The overall impact that a zIIP will make on your LPAR, machine, or sysplex depends on your workload.

7.1.2 z/OS support for zIIP

The zIIP execution environment accepts eligible work from z/OS. z/OS is designed to manage and direct the work between the general purpose processor and the zIIP engine. DB2 UDB for z/OS V1R8 exploits the zIIP capability for eligible workloads. Similar to a zAAP, the z/OS dispatcher can direct certain work to a zIIP. In more specific terms, z/OS uses a zIIP only to handle SRB routines that are part of a defined enclave.

However, other than these operating system-related functions, the full instruction set is available to zAAP and zIIP processors. zIIP support is incorporated in z/OS V1R8 and is available as a Web-installable FMID for z/OS V1R6 and zOS V1R7.

zIIP support is based on the changes implemented for dispatcher affinity (off) in z/OS V1R8. z/OS V1R8 is the first z/OS operating system that includes zIIP support in the base. For previous releases, the support is via FMIDs, as follows:

- z/OS V1R6 - JBB77S9
- z/OS V1R7 - JBB772S

CVT bit

A new CVTZIIP bit can be used by software to determine whether the z/OS release supports the new zIIP execution environment.

TIMEUSED macro

The TIMEUSED macro enables you to record execution times, and to measure performance. TIMEUSED returns the amount of processor or vector time a task has used since being created (attached). The TIMEUSED macro has been modified to allow zIIP execution time to be requested, in addition to the standard CP consumption.

7.1.3 DB2 support for zIIPs

z/OS dispatches work in either Task Control Block (TCB) mode or Service Request Block (SRB) mode. DB2 parallel tasks use SRB mode and are assigned the same importance as the originating address space.

Preemptible enclaves are used to perform work on behalf of the originating TCB or SRB address space. Enclaves are grouped by common characteristics and service requests and since they are preemptible, the z/OS dispatcher (and WLM) can interrupt these tasks for more important ones (that is, to manage a transaction end-to-end). There are two types of preemptible SRBs:

- Client SRBs
- Enclave SRBs

If the DB2 for z/OS V8 request arrives over distributed, DRDA over TCP/IP, then most of that work (other than stored procedures and user-defined functions) is executed in enclave SRBs.

If the request arrives over a local or native connection, then that work is dispatched between TCBs, client Sarabande enclave SRBs. Star schema parallel queries and some utility index maintenance now use enclave SRBs.

As for the zIIP, only the enclave SRB work can be used. The client SRB work, non-preemptible SRB work, and TCB work are not eligible to be redirected to the zIIP. DB2 V8 knows how its work is dispatched and directs z/OS V1R6 or later to dispatch or redirect a portion of the eligible work to the zIIP.

DB2 monitoring of zIIPs

DB2 accounting trace records can provide information on the zIIP processor. IBM Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS, DB2 Performance Expert, or IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS can monitor the zIIP information.

7.1.4 How zIIPs work

A zIIP is designed so that a program can work with z/OS to have eligible portions of its enclave service request block (SRB) work directed to the zIIP. The z/OS operating system, acting on the direction of the program running in SRB mode, controls the distribution of the work between the general purpose processor and the zIIP. Using zIIPs can help free up capacity on the general purpose processor. No anticipated changes to applications that access DB2 UDB for z/OS V8 data is necessary. Utilization of the zIIP is transparent to the application.

Types or DB2 UDB workloads

For applications, running on z/OS, UNIX, Linux, Intel®, or Linux on System z that access DB2 UDB for z/OS V8 on a System z9, via DRDA over a TCP/IP connection, DB2 gives z/OS the necessary information to have portions of these SQL requests directed to the zIIP.

The types of DB2 UDB for z/OS workloads listed here are those executing in enclave SRBs, portions of which can be directed to zIIPs:

- Distributed SQL requests (DRDA)
- Complex parallel query
- DB2 utilities for index maintenance

z/OS dispatching of zIIP work

z/OS dispatches work running in either TCB mode or SRB mode. DB2 parallel tasks use SRB mode where they are assigned the same importance as the originating address space.

Preemptible enclaves are used to do the work on behalf of an originating TCB or SRB. As described previously, enclaves are grouped by common characteristics and service requests. Because they are preemptible, z/OS can interrupt these tasks for more important ones. There are two types of preemptible SRBs:

- Client SRBs
- Enclave SRBs

If a DB2 for z/OS V8 request is coming from a DRDA over TCP/IP, then most of that work other than stored procedures and user-defined functions is executed in enclave SRBs.

If a DB2 for z/OS V8 request is coming over a local connection, then work is dispatched between TCBs, client SRBs and enclave SRBs. Star schema parallel queries and some utility index maintenance now use enclave SRBs.

Only enclave SRB work is eligible to be redirected to zIIPs. DB2 for z/OS V8 knows how its work is dispatched and directs z/OS to dispatch a portion of eligible work to zIIPs.

7.1.5 Distributed SQL requests (DRDA)

Distributed Relational Database Architecture (DRDA) is one of the two types of remote access that DB2 supports between the requesting database management system (DBMS) and the serving relational database management system. The other remote access type is DB2 private protocol access.

When three-part named objects are referenced, DB2 chooses between the two connection types based on the bind option that you choose or the default protocol set at your site. It is recommended that you use DRDA if new applications are being developed, because no enhancements are planned for private protocol.

Note: A database access thread is created when an SQL request is received from the requester. For every database access thread, an independent enclave is created.

DRDA is native to DB2. By using DRDAs, the need for additional gateway products is reduced; such products may affect performance and availability. The Open Group adopted DRDA in 1998 as the open standard for database access interoperatively.

An application uses a DRDA application requestor or server to access a remote database. DB2 connect is an example of a DRDA application requestor. Applications use TCP/IP or SNA as a network protocol to flow DRDA commands. However, only requests that come via TCP/IP are zIIP-eligible.

Regarding the zIIPs, if DB2 for z/OS V8 workload comes over TCP/IP and is DRDA-compliant, then all or a portion of that DB2 workload is eligible to be redirected to the zIIP. Both TCP/IP and DRDA are needed.

Queries that access DB2 for z/OS V8 via DRDA over TCP/IP connections are dispatched within z/OS as enclave SRBs. z/OS directs a portion of this work to the zIIP. Only DRDAs that are coming via TCP/IP are zIIP-eligible. DRDAs that come via SNA are not zIIP-eligible.

DB2 stored procedures

The amount of DRDA via TCP/IP to be redirected might be affected by the DB2 Stored procedures and user-defined functions in the application. DB2 stored procedures and user-defined functions run under TCBs, not SRBs; thus, they are not eligible for zIIPs. The call to the stored procedures, the commit processing and the result of processing may be redirected, but the primary process which is in TCB mode cannot be redirected.

Figure 7-2 on page 155 shows DRDA zIIP usage. Note that the figure is used for illustrative purposes only and shows only a single application; actual workload redirects may vary.

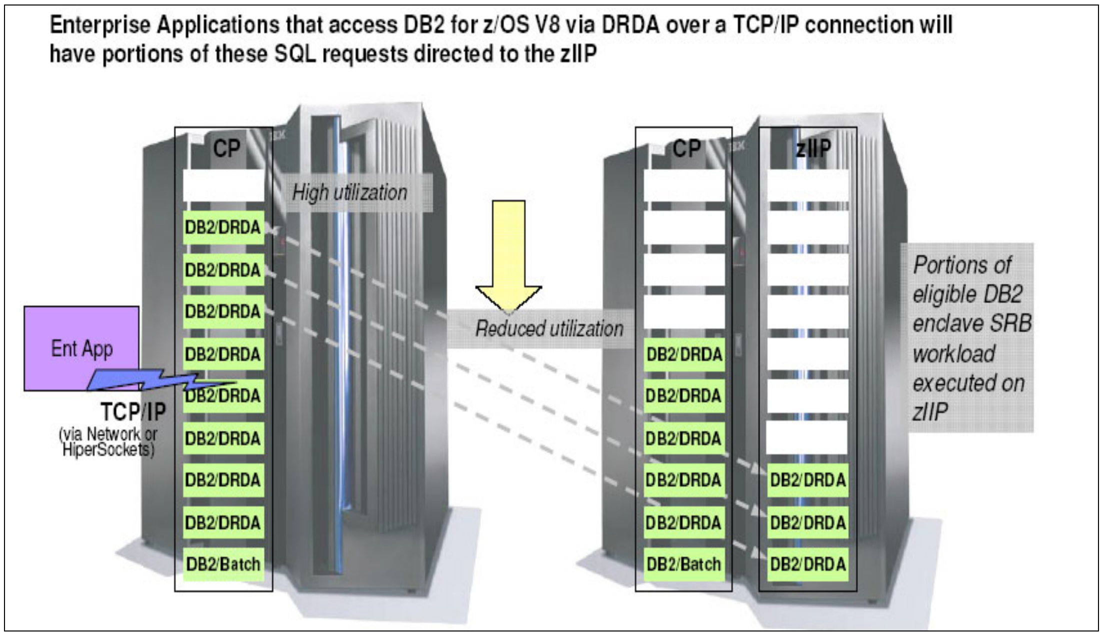

Enterprise Applications that access DB2 for z/OS V8 via DRDA over a TCP/IP connection will have portions of these SQL requests directed to the zIIP
CP
High utilization
DB2/DRDA
DB2/Batch
Ent App
TCP/IP
(via Network or HiperSockets)
Reduced utilization
CP
zIIP
Portions of eligible DB2 enclave SRB workload executed on zIIP

Figure 7-2 DRDA zIIP usage

7.1.6 Complex star schema parallel queries

Data warehousing applications that send requests that utilize DB2 UDB for z/OS V8 complex star schema parallel queries may have portions of these SQL requests directed to the zIIP when DB2 gives z/OS the necessary information.

Star join is a special join technique that DB2 uses to efficiently join tables that form a star schema. A *star schema* is a logical database design that is included in decision support applications. A star schema is composed of a fact table where the data is stored and a number of dimension tables that are connected to it, as shown in Figure 7-3 on page 156. A *dimension table* contains several values that are given an ID, which is used in the fact table instead of all the values.

You can think of the fact table, which is much larger than the dimension tables, as being in the center surrounded by dimension tables; the result resembles a star formation.

Complex star schema parallel queries now use enclave SRBs. z/OS directs a portion of this work to the zIIP. The following different star schema parallel queries can both have portions of their work directed to a zIIP:

- Complex star schema parallel queries via DRDA over TCP/IP connection
- Complex star schema parallel queries via local connection

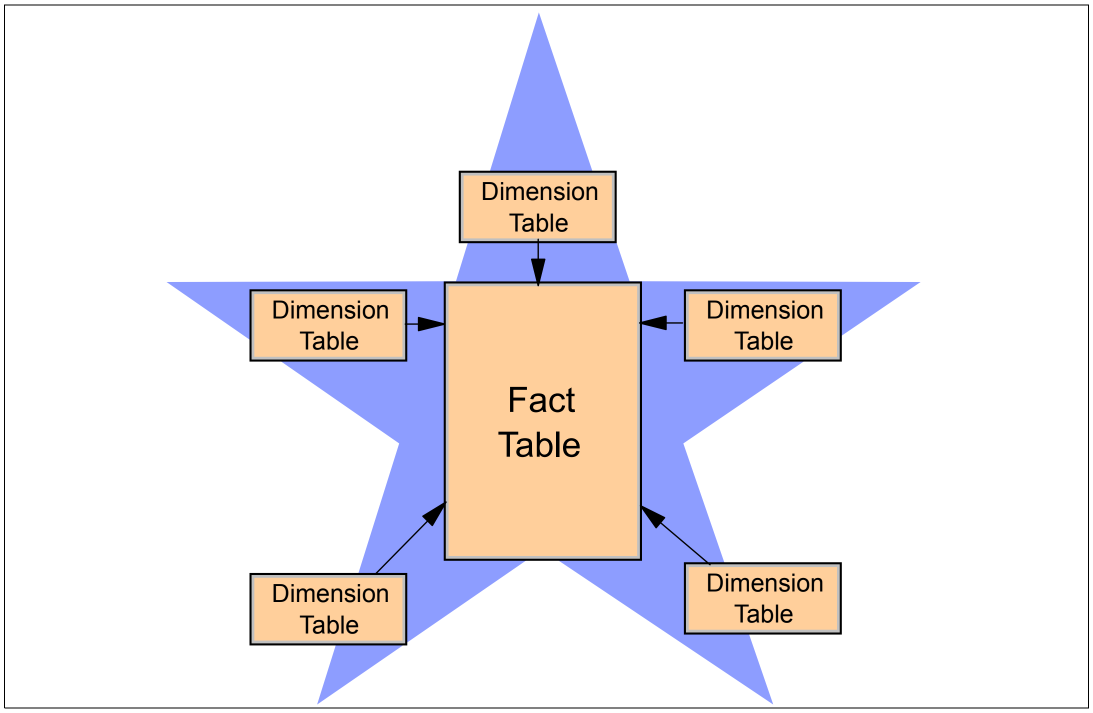


Figure 7-3 Star schema graphical view

Star schema workloads

Star schema workloads may benefit from two redirected tasks:

- The main task, which is a DRDA request, and
- A child task, which is the star schema parallel query

The child and main tasks are "additive", which means that a combination of them is expected to yield a larger amount of redirect than that of DRDA via TCP/IP alone. Longer-running queries see greater benefit. Benefits to a data warehousing application may vary significantly, depending on the details of that application.

Figure 7-4 on page 157 illustrates an example of a business intelligence application type in which complex star schema parallel queries via DRDA are being used.

Important: Only parallel queries can be zIIP-eligible. Any parallel star join can exploit zIIPs in any environment (BATCH, IMS, CICS, JDBC™ and so on).

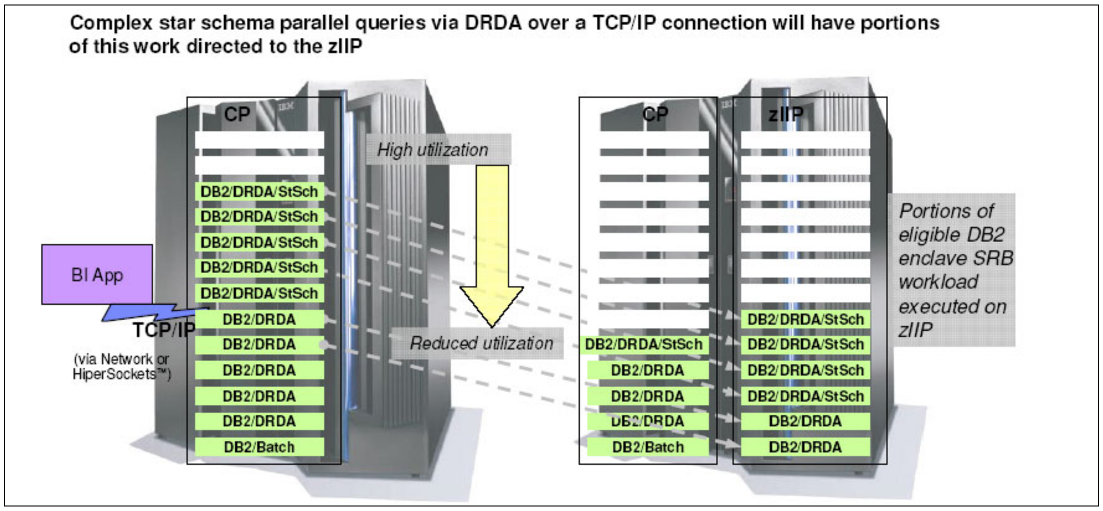

Complex star schema parallel queries via DRDA over a TCP/IP connection will have portions of this work directed to the zIIP
CP
High utilization
DB2/DRDA/StSch
DB2/DRDA
DB2/Batch
BI App
TCP/IP
(via Network or HiperSockets™)
Reduced utilization
CP
zIIP
Portions of eligible DB2 enclave SRB workload executed on zIIP

Figure 7-4 Complex star schema parallel queries via DRDA- business intelligence application

Figure 7-5 illustrates an example of a business intelligence application type in which complex star schema parallel queries via LOCAL connection are being used. No DRDA is being used.

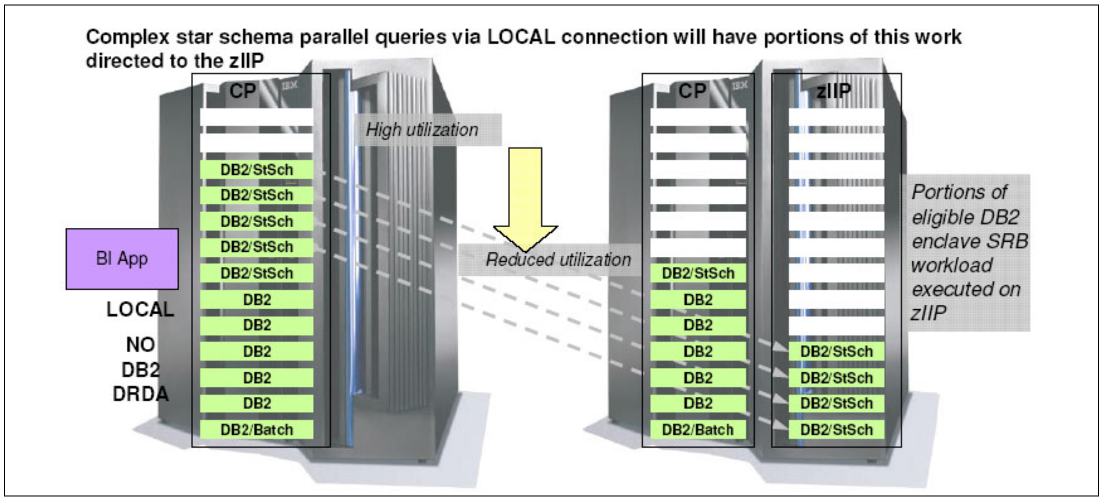

Complex star schema parallel queries via LOCAL connection will have portions of this work directed to the zIIP
CP
High utilization
DB2/StSch
DB2
DB2/Batch
BI App
LOCAL
NO DB2 DRDA
Reduced utilization
CP
zIIP
Portions of eligible DB2 enclave SRB workload executed on zIIP

Figure 7-5 Complex star schema parallel queries via LOCAL - business intelligence application

The actual work of complex star schema parallel queries redirects may vary, depending on the following:

- How long the queries run
- How much parallelism is used
- The table design being used
- The number of zIIPs and CPs employed

zIIP exploitation is limited to only CP parallelism for star join queries. The following factors are taken into consideration by DB2 for the exploitation of query parallelism:

- DB2 attempts to exploit sysplex parallelism if the underlying DB2 instance is a DS member (not supported for star join queries).
- DB2 attempts to exploit CP parallelism if there is more than one CP available.
- DB2 attempts to exploit I/O parallelism if neither sysplex parallelism nor CP parallelism is eligible (no zIIP benefit).

- The estimated cost of each query block (needs to exceed the value set for ZPARM SPRMPLIM).
- The number of CPs if CPU-bound (in general, parallelism is disabled if only 1 CP).
- The number of partitions of the leading table if I/O-bound (in general, parallelism is disabled if the table is not partitioned).

7.1.7 DB2 utilities for index maintenance

An index allows quick access to the rows in a table. Indexes are created using one or more columns of a table. Over time, as data in a large database is manipulated, indexes can become less efficient. They need to be updated and maintained, which can be a major task.

The DB2 utilities LOAD, REORG, and REBUILD INDEX now use enclave SRBs for the portion of the process that is eligible to execute in a zIIP. The LOAD utility loads tables. The REORG utility improves index performance. The REBUILD INDEX utility creates or rebuilds indexes. Only the build portion of LOAD, REORG, and REBUILD DB2 utility processing is related to index maintenance, and can be redirected to zIIPs. The amount of workload that is eligible for zIIPs mainly depends on the following:

- How many indexes are defined on the table
- How many partitions are in the table
- If data compression is being used or not

The LOAD and REORG utilities have many indexes or many partitions and are more eligible than tables with fewer indexes. Fewer partitions, and the ones that are using compression, have the lowest eligibility for zIIPs.

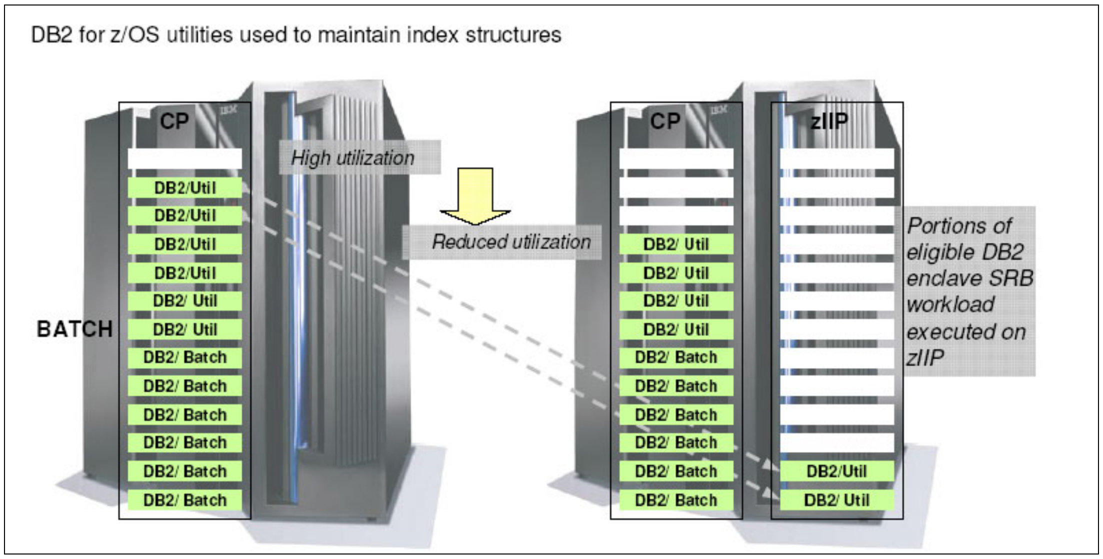

DB2 for z/OS utilities used to maintain index structures
CP
High utilization
Reduced utilization
BATCH
DB2/Util
DB2/Batch
CP
zIIP
DB2/ Util
DB2/ Batch
DB2/Util
Portions of eligible DB2 enclave SRB workload executed on zIIP

Figure 7-6 DB2 for z/OS V8 utilities used to maintain index structures

Important: One objective of zIIPs is to help bring the cost of network access to DB2 more closely in line with the cost of running similar workloads under CICS, IMS, or batch on the mainframe.

zIIP processing flow

A flow of zIIP usage in detail, using DRDA via TCP/IP as an example, is shown in Figure 7-7. The flow proceeds as follows:

1. A DB2 request comes in DRDA over TCP/IP.
2. DB2 schedules an SRB.
3. DB2 creates an enclave and classifies it. One enclave per transaction is created.
4. DB2 notifies WLM that the enclave is eligible to direct a portion of the work to a zIIP.
5. WLM with z/OS Dispatcher dispatches work to either zIIP or general purpose processor.

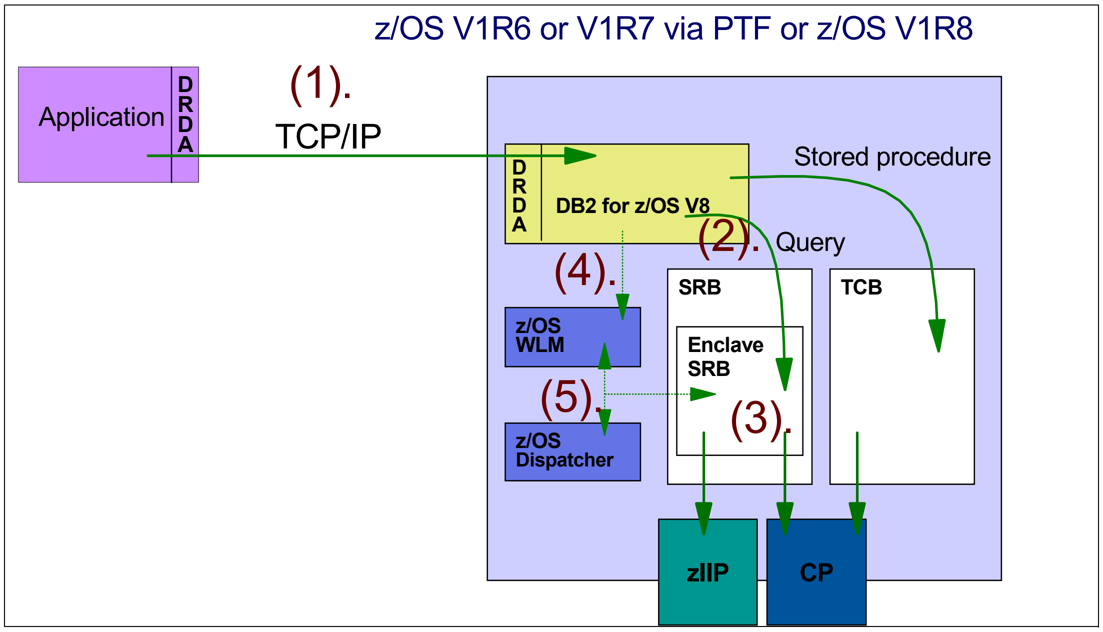


Figure 7-7 Flow of work in DRDA example

Note: Some DB2 processing, like batch and stored procedures, does not occur in SRB mode or in enclave SRBs. Only portions of enclave SRB work is eligible for a zIIP.

7.1.8 Utility processing eligible for IBM zIIP offload

Estimating the amount of processing that is eligible for offload to a zIIP without APAR PK19920, is as follows:

- The following rule of thumb estimates identify the percentage of Load and Reorg CPU time that can be expected to be offloaded.
- The percentages are derived from a set of internal measurements from internal testing. They should be used for capacity planning purposes only.
- The true offload can vary, depending on the customer environment.
- The range of percentage is dependent on the number of partitions in the table.

Test results

The low end of the range is for tables that have few partitions (for example, 1 to 49 partitions) and the high end of the range is for tables that have many partitions (greater than 49 partitions), as follows:

- Load or Reorg of one partition with only one index or entire table space:
  - 10 to 20%
- Load or Reorg of one partition with 2 or more indexes:
  - 60 to 75% if two indexes on a table
  - 70 to 85% if four indexes on a table

The rebuild Index offload behavior is the opposite of what is expected. Instead of having a higher percentage offload with more indexes, there is a lower percentage offload with more indexes. The reason for this is because the sort cost goes up non-linearly as the number of indexes increases, such that the cost of sort starts to dominate as the number of indexes increases, thus reducing the percentage impact of offloading an index build.

- The rebuild index is:
  - 20 to 30% higher if fewer indexes, and lower if more indexes

Note: These values should be used for capacity planning purposes only. The true offload can vary, depending on the customer environment.

7.1.9 SAP workload on zIIPs

IBM System z9 advantage for SAP Applications is available with both the IBM System z9 Enterprise Class (z9 EC) and IBM System z9 Business Class (z9 BC) servers, which are ideal for SAP environments in both medium and large enterprises. These new offerings can provide benefits to companies wishing to deploy their first SAP application functions on System z, as well as to customers running existing SAP application landscapes on System z, especially those planning to install or upgrade to the latest application based on SAP NetWeaver®.

As an enterprise data hub, the IBM System z9 can provide customers with a platform that can help manage costs and improve resource utilization. By leveraging DB2 Version 8, the new zIIP specialty processor for data-centric applications, and the Integrated Facility for Linux (IFL), this new offering can enable consolidation of SAP application landscapes onto a single system.

SAP workloads can benefit from zIIP exploitation. Workload types that are eligible are the same as previously mentioned; refer to Figure 7-8. For estimations, SAP customers should use special SAP data collection procedures.

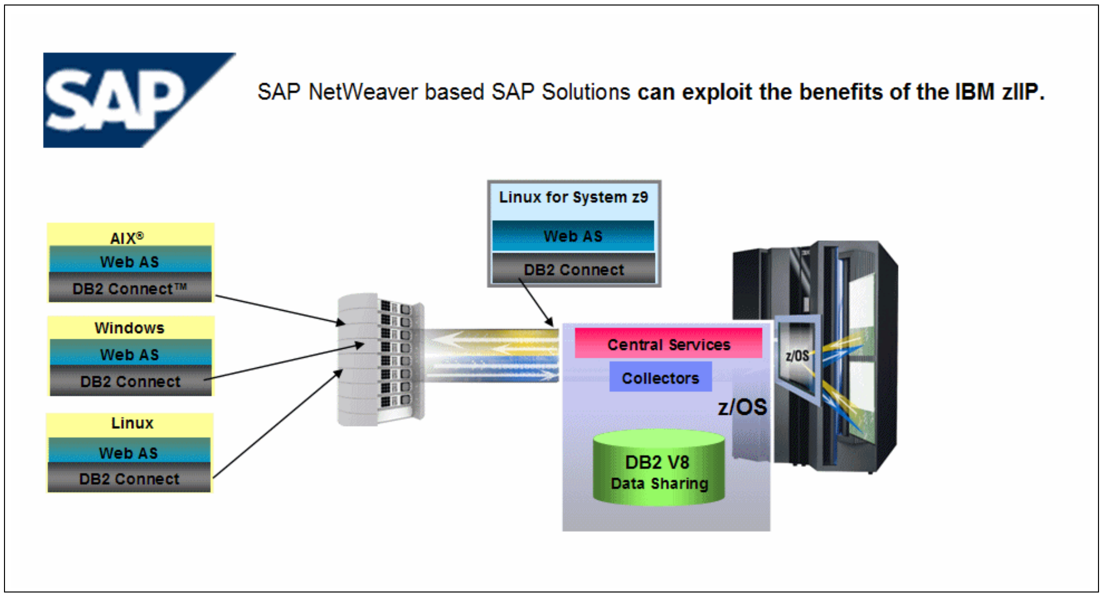


Figure 7-8 SAP workload and zIIPs

7.2 Activating zIIP exploitation

The hardware zIIP engines are available with the z9 BC and z9 EC models. After the appropriate hardware and software is installed, no further action is required to implement zIIP exploitation.

7.2.1 DB2 V8 support for zIIPs

Following is a list of the enabling APARs for this new function in DB2 V8:

PK18454	DB2 z/OS exploitation of the IBM System z9 Integrated Information Processor (IBM zIIP) for DRDA threads
PK19920	DB2 z/OS exploitation of the IBM System z9 Integrated Information Processor (IBM zIIP) for DB2 utilities
PK19921	DB2 exploitation of the IBM System z9 Integrated Information Processor (IBM zIIP) for star join parallel threads
PK20487	DB2 utility toleration of specialty engines such as ICF, IFL, zAAP, and zIIP

7.2.2 IEAOPTxx parmlib member

The OPT parameters allow the installation to change the special assist processor (such as zAAP and zIIP) options, as follows:

- **[IFAHONORPRIORITY=YES|NO]**

  All zIIPs are statically configured with the equivalent of IFAHONORPRIORTY=YES. However, the IFAHONORPRIORITY=YES option still applies to zAAP work. It controls specific help for zAAPs.

  Note: For zIIPs, IFAHONORPRIORITY is not considered. It is always accepted as YES.

- **[PROJECTCPU=YES|NO]**

  The PROJECTCPU=YES option (also available on z/OS V1R6 and z/OS V1R7 as part of the zIIP FMIDs) now also allows zAAP projection to occur, without requiring any per JVM™ configuration changes. Previously, each impacted JVM had to be individually configured to cause zAAP statistics to be collected in RMF and SMF.

  To aid in determining the number of zIIP engines required to satisfy a specific customers usage, this new parmlib option is available after all the software updates have been applied. The PROJECTCPU=YES parameter enables z/OS to collect zIIP usage as if there were one configured, when the target workload is being run.

  This projection capability can be run at any time, on a production environment if desired. RMF and SMF now show this calculated zIIP time so that an accurate zIIP projection can be made.

- **[ZAAPAWMT=xxxxx]**

  It is expected that this parameter will not be specified by customers, unless explicitly directly by IBM support personal.

- **[ZIIPAWMT=xxxxx]**

  It is expected that this parameter will not be specified by customers, unless explicitly directly by IBM support personal.

- **[CCCAWMT=12000]**

The IEAOPTxx parmlib member, CCCAWMT parameter, does not need to be changed. It is strongly recommended that you use the default setting. This parameter is used to activate alternate wait management for processors. This feature is used to remove the overhead of awaking idle CPs when there is no work for them.

Important: The other IEAOPTxx parmlib parameter that is related to zAAP is IFACROSSOVER=YES. This previous zAAP configuration option has become obsolete due to a lack of customer demand and functional complexity. It still exists in IEAOPTxx but is ignored.

7.2.3 zIIP processors

The D M=CPU command output has been modified to use an uppercase letter (I) to represent a zIIP, as shown in Figure 7-9 on page 162.

```
IEE174I 15.39.06 DISPLAY M 135
PROCESSOR STATUS
ID  CPU                  SERIAL
00  +                     01991E2094
01  +                     01991E2094
02  +                     01991E2094
03  +                     01991E2094
04  -
05  -
06  +I                    01991E2094
07  +I                    01991E2094

CPC ND = 002094.S18.IBM.02.00000002991E
CPC SI = 2094.710.IBM.02.000000000002991E
CPC ID = 00
CPC NAME = SCZP101
LP NAME = A01       LP ID =  1
CSS ID  = 0
MIF ID  = 1

+ ONLINE    - OFFLINE    . DOES NOT EXIST    W WLM-MANAGED
N NOT AVAILABLE

I        INTEGRATED INFORMATION PROCESSOR (zIIP)
```

Figure 7-9 D M=CPU command displays zIIPs

7.2.4 SDSF zIIP and zAAP support

SDSF adds information to the DA panel and the ENC panel to show usage of the zAAP (zSeries Application Assist Processor or System z9 Application Assist Processor) and the zIIP (System z9 Integrated Information Processor); refer to Figure 7-10 for an example. Most of the changes for the zAAP were introduced in APAR PK06616, which has been incorporated into z/OS V1R8 SDSF.

```
  Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF ENCLAVE DISPLAY  SC74     ALL                     LINE 0-0 (0)
COMMAND INPUT ===>                                            SCROLL ===> HALF
NP   TOKEN            SysName  Subsys   zAAP-Time zACP-Time zIIP-Time zICP-Time
```

Figure 7-10 SDSF ENC panel showing new columns for zAAP and zIIP times

Note: SDSF support for zIIPs is similar to zAAP.

7.2.5 WLM services for zIIP execution

The following WLM services have been enhanced to support zIIPs with the specified new parameters:

IWMEQTME The IWMEQTME service has been enhanced to include zIIP usage, as follows:

ZIIPONCPTIME=ziiponcptime - An optional output parameter, which will contain the total accumulated time spent on a standard processor for zIIP eligible work for the enclave that is associated with the current dispatchable work unit. The time will be in TOD clock format, normalized to standard processor speed.

To code, specify the RS-type address, or address in register (2)-(12), of an 8-character field.

ZIIPQUALTIME=ziipqualtime - An optional output parameter, which will contain the total time the enclave that is associated with the current dispatchable work unit was qualified to run on an integrated information processor (zIIP). The time will be in TOD clock format.

To code, specify the RS-type address, or address in register (2)-(12), of an 8-character field.

ZIIPTIME=ziiptime - An optional output parameter, which will contain the total accumulated time spent on an integrated information processor (zIIP) for the enclave that is associated with the current dispatchable work unit. The zIIP time will be in TOD clock format, normalized to standard processor speed.

To code, specify the RS-type address, or address in register (2)-(12), of an 8-character field.

IWM4EDEL The IWM4EDEL service has been enhanced to include zIIP usage, as follows:

ZIIPSERVICE=ziipservice - An optional output parameter, which contains the integrated information processor (zIIP) service accumulated by the enclave on the local system. The service is normalized to standard processor speed.

To code, specify the RS-type address, or address in register (2)-(12), of a 64-bit field.

ZIIPTIME=ziiptime - An optional output parameter, which contains the total integrated information processor (zIIP) time accumulated by the enclave on the local system. The time is normalized to standard processor speed.

To code, specify the RS-type address, or address in register (2)-(12), of a 64-bit field.

Mixing zIIP-enabled sysplex members with non-zIIP-enabled members is fully supported. IBM will not charge any software licensing fees for offloaded zIIP work, but customers should contact their ISVs to learn their zIIP licensing policies.

7.3 Capacity planning for zIIPs

In order to help customers to determine the number of zIIP engines required to satisfy a specific customer usage, a new parmlib member option is available after all the software updates have been applied. Software requirements are mentioned in 7.2, “Activating zIIP exploitation” on page 161.

The PROJECTCPU=YES parameter enables z/OS to collect zIIP usage as if there was one configured, when the target workload is being run. By specifying the PROJECTCPU option in the IEAOPTxx parmlib member, zIIP consumption can be projected without any new hardware or software support applied. The PROJECTCPU=YES option enables RMF to monitor DB2 for how zIIP consumption would be.

This projection capability can be run at any time, on a production environment if desired. RMF and SMF will show the calculated zIIP time so that an accurate zIIP projection can be made.

This single PROJECTCPU=YES option, which is also available on z/OS V1R6 and z/OS V1R7 as part of the zIIP FMID, allows zAAP projection to occur without requiring any per JVM configuration changes. (Previously, each impacted JVM had to be individually configured to cause zAAP statistics to be collected in SMF and RMF.) In summary, the PROJECTCPU parmlib member in IEAOPTxx projects zIIP or zAAP consumption without the hardware installed.

7.3.1 Data collection procedures for zIIP utilization

After all software and hardware requirements are installed, SMF and RMF collect the necessary data. The procedures mentioned in this section are not applicable for SAP use. SAP data collection procedures can be given to SAP customers by IBM. Data collection procedures are estimates for the re-direct of DRDA workload. For procedures related to estimating the re-direct of DB2 parallel star schema and DB2 utilities, consult your IBM representative.

Collecting performance data

By using data provided in the SMF 70 and 72 records, as well as data in the SMF 101 (DB2 accounting) records, it is possible to estimate the amount of CPU capacity that is eligible to run on a zIIP CP. This process has been developed to make the analysis relatively easy for both the customer and account team, while providing the most accurate estimate possible. It is accomplished by sharing the work between the customer, the IBM account team, and the IBM DRDA sizing team.

The following records are used, so you need to collect SMF record types 70 and 72 for RMF reports.

For the DB2 data:

- Select one hour's worth of type 101 data within the 24-hour period chosen for the RMF data. This will be used to find stored procedure and UDF workloads within the DDF workloads that are not eligible for re-directing onto the zIIP processor.
- Accounting Class 1 and 2 must be turned on, either by default, or via the following DB2 command (prefixed by the command recognition character):

  ```
  START TRACE (ACCTG) DEST (SMF) CLASS(1,2)
  ```

> **Note:** The IBM DRDA sizing team will supply a data collection guide to the IBM account team to be given to the customer. This data collection guide will contain directions on what data needs to be sent to IBM, and how the data is to be delivered.
>
> The IBM DRDA sizing team will use the tools developed by IBM to process the customer's data, to determine an estimate of the amount of DRDA work that is eligible to execute on a zIIP.

There are two options for the collection of performance data:

- You can choose to provide both RMF and DB2 data. The analysis is then performed once, using both sets of data.
- You can provide only RMF and SMF data for initial analysis. If the percentage of zIIP eligible data is significant, a special program provided by IBM with the data collection procedure and analysis can be performed a second time, using both RMF and DB2 data.

DB2 V8 customers who are concerned with the volume of DB2 accounting trace records can minimize the DB2 accounting trace records gathered by using the following DRDA and RRS attach accounting roll-up DSNZPARM option:

- ACCUMACC=10 or higher (10 is the default). With this setting, DB2 will cut one rolled-up accounting record for every 10 DRDA or RRS attach occurrences or transactions.

> **Note:** The amount of DRDA work that uses stored procedures, UDF, or enters the system via an SNA connection is *not* eligible to execute on a zIIP. DB2 data for this workload must be removed from the estimate, if present.

After this data is collected, customers should fill out a form about their workload and configurations, then FTP the data to the necessary sites. For details and an explanation of the latest procedures, contact IBM and get the data collection guide.

Stored procedure workload

Parts of stored procedure workload processing (such as Call, Results Set, and Commit processing) are eligible for zIIP redirect and are accounted for in the redirect estimation.

> **Note:** If you use stored procedures or UDFs extensively, there are other DB2 DSNZPARM settings that may affect the estimation.

7.4 RMF support for zIIPs

After a zIIP is installed (with z/OS V1R6 or z/OS V1R7 with a PTF) and DB2 V8 (with PTFs) is installed, the monitoring of zIIP activity is similar to monitoring zAAP activity. The monitoring of the zIIP activity is done as follows:

- Set up a WLM policy with service class(es) for SUBSYSTEM TYPE=DDF.
- RMF Monitor 1 Type 70 records will monitor overall zIIP activity, as follows:
  - Logical processor busy as seen by z/OS is reported.
  - Physical processor busy as seen by the LPAR is reported.
- RMF Monitor 1 Type 72 records will show more detail, as follows:
  - The amount of time spent executing on zIIP processors is reported.
  - Usage and delay sample counts for zIIP eligible work is reported.

RMF support for zIIP is introduced with z/OS V1R8. It is also rolled-back as an SPE with APAR OA13499. Table 7-1 on page 166 lists the PTFs needed for RMF to support zIIPs.

Table 7-1 PTFs needed to support zIIPs in RMF

PTF number	FMID	Release
UA90521	HRM7708	z/OS V1R5 RMF Base *
UA90252	JRM77J8	z/OS V1R5 RMF Kanji *
UA90253	HRM7720	z/OS V1R7 RMF Base
UA90254	JRM772J	z/OS V1R7 RMF Kanji
		* Only in combination with z/OS V1R6

Important: RMF uses the term "IIP" to denote zIIP processors in the affected reports, messages, and help panels. To use consistent terms for zAAPs and zIIPs, any IFA fields in affected RMF reports have been renamed to AAP.

7.4.1 RMF zIIP implementation

RMF support for the zIIP is like support for the zAAP. RMF distinguishes between general purpose CPs and special purpose processors (zAAP and zIIP) where necessary, collects and reports about zIIP consumption, and collects and reports about using and delay states for the zIIP. RMF provides measurements about zIIP activity in the following reports:

- Postprocessor:
  - CPU Activity report and its Partition Data Report section
  - Workload Activity report
- Monitor III:
  - CPC Capacity report
  - Enclave Report
  - System Information report
- New Postprocessor overview conditions for zIIP are introduced.
- New fields with zIIP measurements in SMF 70.1, 72.3, 79.1, and 79.2 are introduced.
- New DDS and RMF PM metrics about zIIP activity are introduced.
- Monitor II data services now return the system's zIIP utilization.

7.4.2 RMF CPU Activity for zIIPs

In addition to general purpose CPs and zAAPs (IFA), the CPU Activity section formats additional lines for each zIIP configured with these columns: online, LPAR busy, and MVS busy time percentages; see Figure 7-11 on page 167.

The changes are as follows:

- A summary line is printed with the average percentage values for the zIIPs.
- The term IFA is replaced by AAP (zAAP).

```
                                                 C P U  A C T I V I T Y

            z/OS V1R8             SYSTEM ID S5C              DATE 05/13/2006            INTERVAL 15.0
                                  RPT VERSION V1R8 RMF       TIME 15.00.00              CYCLE 1.000 S
CPU  2094   MODEL  743   H/W MODELS54
---CPU---  ONLINE TIME   LPAR BUSY MVS BUSY     CPU SERIAL  I/O TOTAL          % I/O INTERRUPTS
NUM  TYPE  PERCENTAGE    TIME PERC TIME PERC    NUMBER      INTERRUPT RATE     HANDLED VIA TPI
 0   CP    100.00        78.44     78.44         030CFE       5.63               0.00
 1   CP    100.00        78.03     78.04         030CFE       6.17               0.02
 2   CP    100.00        76.38     76.38         030CFE       5.90               0.00
 3   CP    100.00        75.64     75.64         030CFE       5.88               0.00
 4   CP    100.00        74.50     74.51         030CFE       6.01               0.00
 5   CP    100.00        73.62     73.63         030CFE       5.79               0.02
CP   TOTAL/AVERAGE       74.70     74.71                     83.28               0.00
 F   AAP   100.00        99.87     99.88         030CFE
10   AAP   100.00        99.84     99.86         030CFE
11   AAP   100.00        99.88     99.89         030CFE
12   AAP   100.00        99.86     99.87         030CFE
AAP  AVERAGE             99.87     99.88
13   IIP   100.00        99.99     100.0         030CFE
14   IIP   100.00        99.99     100.0         030CFE
15   IIP   100.00        99.99     100.0         030CFE
16   IIP   100.00        99.99     100.0         030CFE
17   IIP   100.00        99.99     100.0         030CFE
IIP  AVERAGE             99.99     100.0
```

Figure 7-11 CPU Activity report showing zIIP activity

7.4.3 RMF Partition Data Report for zIIPs

The Partition Data section of the CPU Activity Report shows the number of physical zIIPs, as illustrated in Figure 7-12 on page 168. If zIIPs are configured, an additional data block is formatted with one line per LPAR exploiting zIIPs, followed by the PHYSICAL and TOTAL line for the zIIP resource pool. For zIIPs, the same report columns as for other special purpose processors (zAAPs, IFLs, ICFs) are printed.

```
                                          P A R T I T I O N   D A T A   R E P O R T

            z/OS V1R8                SYSTEM ID S5C              DATE 05/13/2006            INTERVAL 15.00.000
                                     RPT VERSION V1R8 RMF       TIME 15.00.00              CYCLE 1.000 SECONDS

MVS PARTITION NAME                       S5C        NUMBER OF PHYSICAL PROCESSORS          54                 GR
IMAGE CAPACITY                           721                   CP                           43
NUMBER OF CONFIGURED PARTITIONS            25                  AAP                           4
WAIT COMPLETION                            NO                  IFL                           0
DISPATCH INTERVAL                     DYNAMIC                  ICF                           2
                                                               IIP                           5

--------- PARTITION DATA -----------------  -- LOGICAL PARTITION PROCESSOR DATA --   -- AVERAGE PROCESSOR UTIL
                   ----MSU----  -CAPPING--  PROCESSOR- ----DISPATCH TIME DATA----    LOGICAL PROCESSORS  --- P
NAME       S   WGT  DEF    ACT  DEF    WLM% NUM   TYPE   EFFECTIVE       TOTAL       EFFECTIVE    TOTAL  LPAR
S5C        A   900    0    539  NO     0.0 15.0   CP    02.48.03.847  02.48.04.742       74.70    74.70      0
S51        A   100    0      0  NO     0.0 10.0   CP    00.00.00.000  00.00.00.000        0.00     0.00      0
S5D        A   100    0      0  NO     0.0   15   CP    00.00.00.000  00.00.00.000        0.00     0.00      0
S5G        A   100    0      0  NO     0.0 15.0   CP    00.00.00.000  00.00.00.000        0.00     0.00      0
S5H        A   100    0      0  NO     0.0 15.0   CP    00.00.00.000  00.00.00.000        0.00     0.00      0
*PHYSICAL*                                                          00.00.02.598                         0
                                                        ------------  ------------                         ---
  TOTAL                                                 02.48.03.847  02.48.07.341                           0

S5C        A   900                            4   AAP   00.59.55.019  00.59.55.147       99.86    99.87      0
S51        A   100                            2   AAP   00.00.00.000  00.00.00.000        0.00     0.00      0
S5D        A   100                            4   AAP   00.00.00.000  00.00.00.000        0.00     0.00      0
S5G        A   100                            4   AAP   00.00.00.000  00.00.00.000        0.00     0.00      0
S5H        A   100                            4   AAP   00.00.00.000  00.00.00.000        0.00     0.00      0
*PHYSICAL*                                                          00.00.00.345                          0
                                                        ------------  ------------                         ---
  TOTAL                                                 00.59.55.019  00.59.55.492                           0

S5C        A   900                            5   IIP   01.14.59.421  01.14.59.558       99.99    99.99      0
S51        A   100                            2   IIP   00.00.00.000  00.00.00.000        0.00     0.00      0
S5D        A   100                            5   IIP   00.00.00.000  00.00.00.000        0.00     0.00      0
S5G        A   100                            5   IIP   00.00.00.000  00.00.00.000        0.00     0.00      0
S5H        A   100                            5   IIP   00.00.00.000  00.00.00.000        0.00     0.00      0
*PHYSICAL*                                                               00.00.00.428
                                                       ------------  ------------                         ---
  TOTAL                                                01.14.59.421 01.14.59.987 -
```

Figure 7-12 RMF Partition Data Report showing zIIPs

7.4.4 RMF Service Class Periods Report for zIIPs

The Resource Consumption and Goal versus Actuals section of the WLMGL report is changed to format the following for zIIPs, as shown in Figure 7-13 on page 169:

- zIIP service times and zIIP using and delay state samples.
- TCB time reflects CPU time spent on regular CPs, as well as on zIIPs and zAAPs.
- The SERVICE TIMES block is extended by a new field called IIP, reporting the zIIP service time in seconds.
- All APPL% values are moved to a new block that is inserted between the SERVICE TIMES and PAGE-IN RATES blocks.
- To gain space for this new block, the STORAGE and TRANS-TIME blocks are squeezed. This is achieved by shortening a couple of field names:
  - R/S AFFINITY is changed into R/S AFFIN in the TRANS-TIME block.
  - In the STORAGE block, TOTAL is changed into TOT, CENTRAL into CEN, EXPAND into EXP and SHARED into SHR.
- The percentage of CPU time on zIIPs is added. This field is called APPL% IIP.
- To assess the portion of IIP work executed on a standard CP, a new field called APPL% IPPCP is added, which is a subset of APPL% CP. The calculation is based on the IIP service time spent on standard CPs. Thus, this value is only reported as percentage of

CPU time, but not as value in time of seconds. However, IIP time on CPs can be formatted by means of overview reporting.

- APPL% AAPCP and APPL% IIPCP might be reported with values greater than zero even without hardware to project zIIP and zAAP consumption (refer to 7.2, “Activating zIIP exploitation” on page 161, for more information about the PROJECTCPU option).
- The USING% block of the Goals versus Actuals section is extended by IIP using samples.
- The EXECUTION DELAY% block is extended. IIP delay samples can appear as new contributors to the total delay samples.

APPL % column

In Figure 7-13, there are two zIIP percentages in this column, which are explained here:

- In the APPL % column in Figure 7-13, the IIP for zIIPs is the percentage of CPU time used by transactions executed on zIIPs in the service or report class period. The calculation is as follows:

$$\text{APPL\% IIP} = \frac{\text{IIP} * 100}{\text{Interval Length}}$$

  APPL% shows the CPU utilization based on uniprocessor capacity. This means that the values can exceed 100% in systems with more than one processor. In a sysplex, the values for seconds and CPU time percentage are meaningful only if all processors have the same speed.

- In the APPL % column in Figure 7-13, the IIPCP for zIIPs is the percentage of CPU time used by zIIP eligible transactions running on standard CPs. This is a subset of APPL% CP.

EXECUTION DELAYS % column

In Figure 7-13, for the EXECUTION DELAYS %, the IIP column is the zIIP-eligible work delayed because it is waiting for a processor that can run zIIP work.

USING % column

In Figure 7-13, the IIP column is the samples of zIIP work executing on standard CPs.

```
-------------------------------------------------------------------------------------------------- SERVICE CLASS PERIODS

REPORT BY: POLICY=DRDAIC1    WORKLOAD=DB2    SERVICE CLASS=DB2ADDRS   RESOURCE GROUP=*NONE     PERIOD=1 IMPORTANCE=1

TRANSACTIONS   TRANS-TIME HHH.MM.SS.TTT  --DASD I/O--  ---SERVICE----  SERVICE TIMES  ---APPL %---  PAGE-IN RATES
AVG      3.00  ACTUAL                0  SSCHRT 585.3  IOC     300191  TCB     18.6  CP     10.68  SINGLE     0.0
MPL      3.00  EXECUTION             0  RESP     1.0  CPU     547934  SRB     11.1  AAPCP   0.00  BLOCK      0.0
ENDED       0  QUEUED                0  CONN     0.7  MSO          0  RCT      0.0  IIPCP   0.01  SHARED     0.0
END/S    0.00  R/S AFFIN             0  DISC     0.1  SRB     329049  IIT      1.5                HSP        0.0
#SWAPS      0  INELIGIBLE            0  Q+PEND   0.2  TOT      1177K  HST      0.0  AAP     0.00  HSP MISS   0.0
EXCTD       0  CONVERSION            0  IOSQ     0.0  /SEC      4024  AAP      0.0  IIP     2.81  EXP SNGL   0.0
AVG ENC  0.00  STD DEV               0                               IIP      8.2                EXP BLK    0.0
REM ENC  0.00                                         ABSRPTN  1341                              EXP SHR    0.0
MS ENC   0.00                                         TRX SERV 1341

GOAL: EXECUTION VELOCITY 80.0%     VELOCITY MIGRATION:   I/O MGMT  85.3%     INIT MGMT 85.7%

         RESPONSE TIME EX   PERF  AVG    ------ USING% ----- ------------ EXECUTION DELAYS % ------------ ---DLY%--
SYSTEM                 VEL% INDX ADRSP   CPU  AAP  IIP  I/O  TOT  I/O  IIP  CPU                          UNKN IDLE
H2           --N/A--   85.7  0.9   3.0   3.3  0.0 10.9 16.9  5.4  3.0  1.4  1.0                           0.0 63.5
```

Figure 7-13   RMF Service Class Periods Report showing zIIPs

7.4.5 RMF Distributed Data Server for zIIPs

The RMF Distributed Data Server (DDS) is extended to provide the same variety of metrics for the zIIP as for the zAAP. These metrics can be displayed either via an RMF PM client, or via a RMF Monitor III Data Portal.

Here are the changes related to metrics to support zIIPs.

- **New metrics for resource sysplex**
- % total utilization (zIIP) by partition
- % total physical utilization (zIIP) by CPC
- % total LPAR management time (zIIP) for PHYSICAL by CPC
- % zIIP by job
- **New metrics for resource processor**
- % effective physical utilization (zIIP)
- % total physical utilization (zIIP)
- % total LPAR management time (zIIP) for PHYSICAL
- % effective physical utilization (zIIP) by partition
- % total physical utilization (zIIP) by partition
- **New metrics for resource processor**
- % zIIP
- % zIIP on CP
- # zIIPs online
- % zIIP by enclave
- % zIIP delay by enclave
- total / delta zIIP seconds by enclave
- % zIIP by job / service or report class
- % zIIP by service or report class period

7.4.6 Additional zIIP monitoring support

In addition, DB2 accounting trace records can provide information on the zIIP and the following products can be used to monitor the zIIP information:

- IBM Tivoli OMEGAMON XE for DB2
- Performance Expert on z/OS
- DB2 Performance Expert
- IBM Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS

7.5 SMF records for zIIP support

The SMF type 30 record (IFASMFR3) has been updated to include new zIIP consumption fields. The SMF type 30 record may also be used to enhance existing charge back applications to include explicit zIIP charge back.

The SMF records for types 70, 72, and 79 have been updated to include information about zIIPs.

7.5.1 SMF type 70 records for zIIPs

The SMF type 70 record type 1 reports on the availability of zIIPs in the processor flags section of the RMF product section and the CPU data section, as shown in Figure 7-14.

Offsets	Name	Len	Format	Description
SMF record types 70 to 79 – RMF product section				
49 x31	PRF	1	binary	Processor flags BIT MEANING WHEN SET 0 System has expanded storage 1 Processor enabled for ESCON 2 ESCON Director in configuration 3 zAAPs available 4 zIIPs available
SMF 70.1 – CPU control section				
64 x40	SMF70SUP	4	Binary	Number of zIIPs online at the end of interval
SMF 70.1 - CPU data section				
15 x0F	SMF70TYP	1	Binary	CPU type (0=CP, 1=zAAP, 2=zIIP)
SMF 70.1 – CPU identification section				
0 x00	SMF70CIN	16	EBCDIC	CPU-identification name
16 x10	SMF70CTN	2	Binary	Number of physical CPUs of this type
18 x12		2		reserved

Figure 7-14 SMF type 70 record for zIIPs

New overview conditions based on SMF type 70 subtype 1 records (CPU activity) are available in z/OS V1R8, as shown in Figure 7-15.

```
              R M F   O V E R V I E W   R E P O R T

NUMBER OF INTERVALS 5
DATE    TIME     INT       NUMIIP    IIPBSY    IIPMBSY
MM/DD HH.MM.SS MM.SS
03/14 12.33.59 06.00            1       0.4        0.2
03/14 12.40.00 19.59            1       0.4        0.2
03/14 13.00.00 20.00            1       0.3        0.2
03/14 13.20.00 20.00            1       0.3        0.2
03/14 13.40.00 19.59            1       0.3        0.2
```

Figure 7-15 RMF Overview Report

As shown in Figure 7-15 and in Figure 7-16 on page 171, IIPBSY and IIPMBSY can be specified with a processor identifier as qualifier. If omitted, the value represents the average for all zIIPs.

Condition	Name	Algorithm
Number of zIIPs available at the end of the reporting interval	NUMIIP	SMF70SUP
CPU processor busy for zIIPs	IIPBSY	Like CPUBSY but for zIIPs instead of standard CPs
MVS processor busy for zIIPs	IIPMBSY	Like MVSBSY but for zIIPs instead of standard CPs

Figure 7-16 New overview conditions for zIIPs

7.5.2 SMF type 72 record for zIIPs

The SMF type 72 record subtype 3 (Workload Activity), as shown in Figure 7-17, is extended as follows:

- The workload manager control section is extended by the zIIP normalization factor. In addition, the description of the zAAP normalization factor (R723NFFI) is updated.
- The service report class period data section is extended starting at offset x214.

Offsets	Name	Len	Format	Description
SMF 72.3 - Workload manager control section				
240 xF0	R723NFFI	4	Binary	Normalization factor for zAAP (IFA). Multiply zAAP times or service units with this value and divide by 256 to calculate the CP equivalent value.
244 xF4	R723NFFS	4	Binary	Normalization factor for zIIP. Multiply zIIP service units with this value and divide by 256 to calculate the CP equivalent value.
SMF 72.3 - Service / Report class period data section				
532 x214	R723SUPU	4	Binary	zIIP using samples
536 x218	R723SUCU	4	Binary	zIIP on CP using samples (included in R723CCUS)
540 x21C	R723SUPD	4	Binary	zIIP delay samples
544 x220	R723CSUP	8	Float	zIIP service units. Multiply with R723NFFS and divide by 256 to calculate the CP equivalent value
552 x228	R723CSUC	8	Float	zIIP service units spent on CPs
560 x230	R723CIFA	8	Float	zAAP service units. Multiply with R723NFFI and divide by 256 to calculate the CP equivalent value
568 x238	R723CIFC	8	Float	zAAP service units spent on CPs

Figure 7-17 SMF type 72 record for zIIPs

Note: zIIP service is provided in terms of service units, and zAAP service is now provided in terms of service units, too. But for purposes of consistency, zAP service in terms of *microsecond units* (R723IFAT and R723IFCT) is also kept.

Overview fields for type 72 records

The new overview condition fields for SMF type 72 are shown in Figure 7-18 on page 173. OVW conditions IIPDLYP, IIPUSGP, and IPCUSGP are specified for a service or report class period. All other new OVW conditions can be specified for service and report classes, service and report class periods, workload groups, or the entire service policy.

To use consistent terms for zIIP- and zAAP-based OVW conditions, an alternate set of condition names for IFASEC, IFANSEC, IFACPSEC, APPLIFA, APPLIFCP, IFAUSGP, IFCUSGP, and IFADLYP is provided. The new names are AAPSEC, AAPNSEC, AAPCPSEC, APPLAAP, APPLAPCP, AAPUSGP, APCUSGP, and AAPDLYP.

zAAP consumption is now also available in terms of service units. Previously, it was only available in terms of service time (IFASEC and IFACPSEC):

- zAAP service units per second: AAPSRV … Sum(R723CIFA) / SMF72INT
- zAAP service units on standard CPs per second: AAPCPSRV … Sum(R723CIFC) / SMF72INT

Condition	Name	Algorithm
zIIP service units per second	IIPSRV	Sum(R723CSUP) / SMF72INT
zIIP service units on standard CPs per second	IIPCPSRV	Sum(R723CSUC) / SMF72INT
zIIP service time in seconds	IIPSEC	Sum((R723CSUP x R723MADJ) / (1600 x R723MCPU))
zIIP service time in seconds (normalized)	IIPNSEC	Sum((R723CSUP x R723MADJ) / (1600 x R723MCPU)) x R723NFFS / 256
zIIP service time in seconds spent on standard CPs	IIPCPSEC	Sum((R723CSUC x R723MADJ) / (1600 x R723MCPU))
zIIP application execution time %	APPLIIP	Like IIPSEC… / SMF72INT x 100
zIIP on CP application execution time %	APPLIPCP	Like IIPCPSEC… / SMF72INT x 100
zIIP delay %	IIPDLYP	R723SUPD / R723CTSA x 100
zIIP using %	IIPUSGP	R723SUPU / R723CTSA x 100
zIIP on CP using %	IPCUSGP	R723SUCU / R723CTSA x 100

Figure 7-18 New overview conditions based on SMF record 72-3 Workload activity

7.5.3 SMF type 79 record for zIIPs

The SMF type 79 record for subtypes 1 and 2, as shown in Figure 7-19, has zIIP information.

Offsets	Name	Len	Format	Description
SMF 79.1 - Address space state data - ASD and ASDJ data section				
208 xD0	R791TSUP	4	Binary	CPU time consumed on zIIPs (ASSB_TIME_ON_zIIP)
212 xD4	R791TSUC	4	Binary	CPU time consumed on standard CPs by zIIP eligible work (ASSB_TIME_zIIP_ON_CP)
216 xD8	R791NFFS	4	Binray	Normalization factor for zIIP time. Used to convert between real and normalized zIIP times, i.e. the equivalent time on a standard CP. Multiply R791TSUP by this value and divide by 256.
SMF 79.2 - Address space resource data - ARD and ARDJ data section				
196 xC4	R792TSUP	4	Binary	CPU time consumed on zIIPs (ASSB_TIME_ON_zIIP)
200 xC8	R792TSUC	4	Binary	CPU time consumed on standard CPs by zIIP eligible work (ASSB_TIME_zIIP_ON_CP)
204 xCC	R792NFFS	4	Binray	Normalization factor for zIIP time. Used to convert between real and normalized zIIP times, i.e. the equivalent time on a standard CP. Multiply R792TSUP by this value and divide by 256.

Figure 7-19 SMF type 79 record for zIIPs

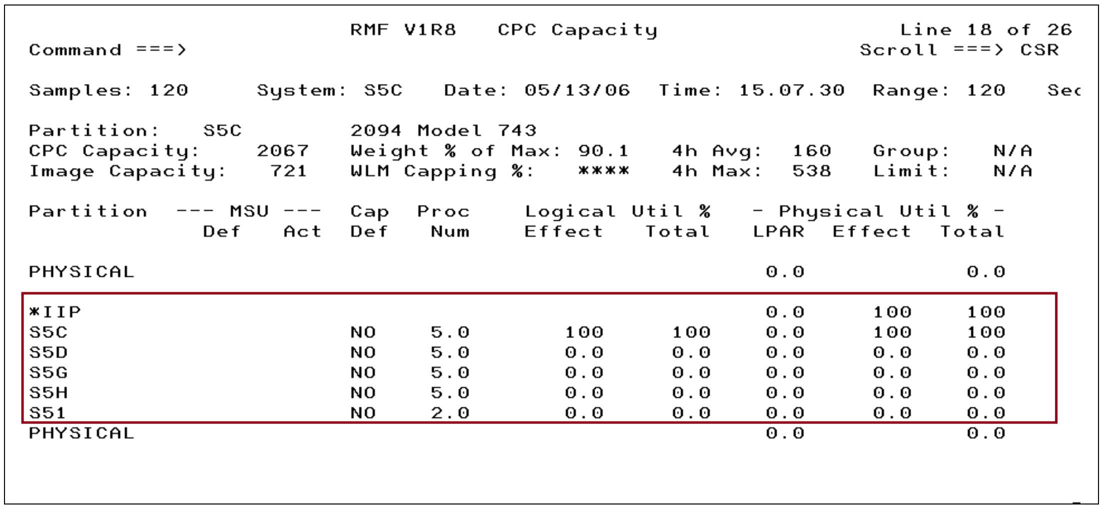
7.6 RMF Monitor III support for zIIPs

RMF Monitor III supports the following reports for zIIPs:

- CPC Capacity report - in Figure 7-20
- Enclave Report - in Figure 7-21 and Figure 7-22 on page 175
- System Information report - in Figure 7-23 on page 175

RMF CPC Capacity report

Figure 7-20 contains a sample CPC Capacity report showing zIIP activity.

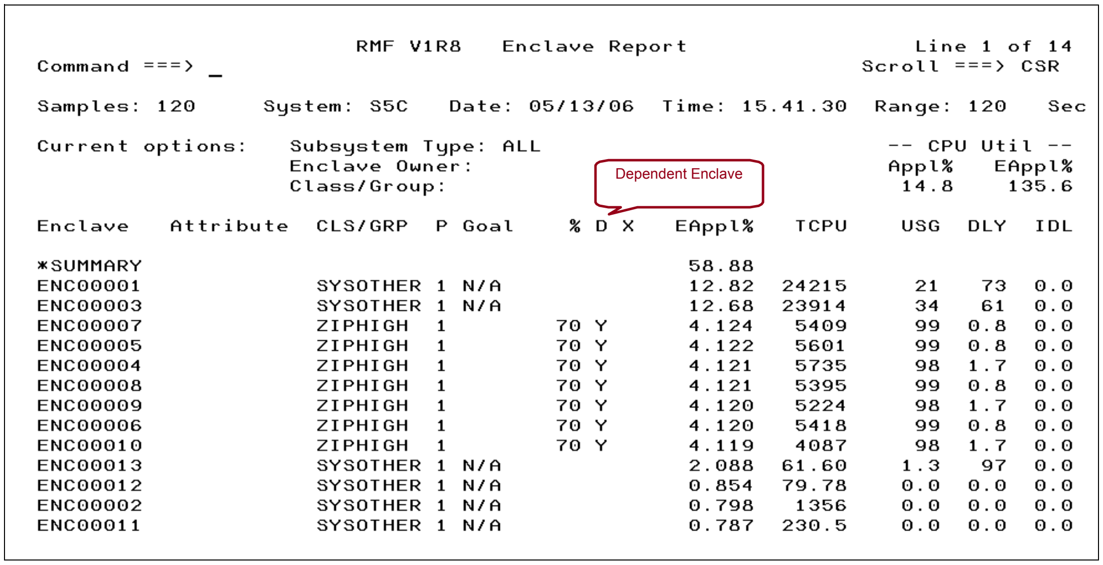


```
                          RMF V1R8   CPC Capacity                    Line 18 of 26
 Command ===>                                                     Scroll ===> CSR

 Samples: 120     System: S5C   Date: 05/13/06  Time: 15.07.30  Range: 120    Sec

 Partition:   S5C         2094 Model 743
 CPC Capacity:    2067    Weight % of Max: 90.1    4h Avg:  160    Group:    N/A
 Image Capacity:   721    WLM Capping %:   ****    4h Max:  538    Limit:    N/A

 Partition    --- MSU ---  Cap  Proc     Logical Util %   - Physical Util % -
                Def   Act  Def   Num     Effect   Total   LPAR  Effect  Total

 PHYSICAL                                                  0.0           0.0

 *IIP                                                      0.0    100    100
 S5C                       NO    5.0      100      100     0.0    100    100
 S5D                       NO    5.0      0.0      0.0     0.0    0.0    0.0
 S5G                       NO    5.0      0.0      0.0     0.0    0.0    0.0
 S5H                       NO    5.0      0.0      0.0     0.0    0.0    0.0
 S51                       NO    2.0      0.0      0.0     0.0    0.0    0.0
 PHYSICAL                                                  0.0           0.0
```

Figure 7-20 Sample CPC capacity report indicating usage of zIIPs

RMF Enclave Report

Enclave reports provide detail information about the activities of enclaves. An *enclave* is a transaction that can span multiple dispatchable units (SRBs and tasks) in one or more address spaces, and is reported on and managed as units. Figure 7-21 shows an example of zIIP usage information on the Monitor III Enclave Reports.



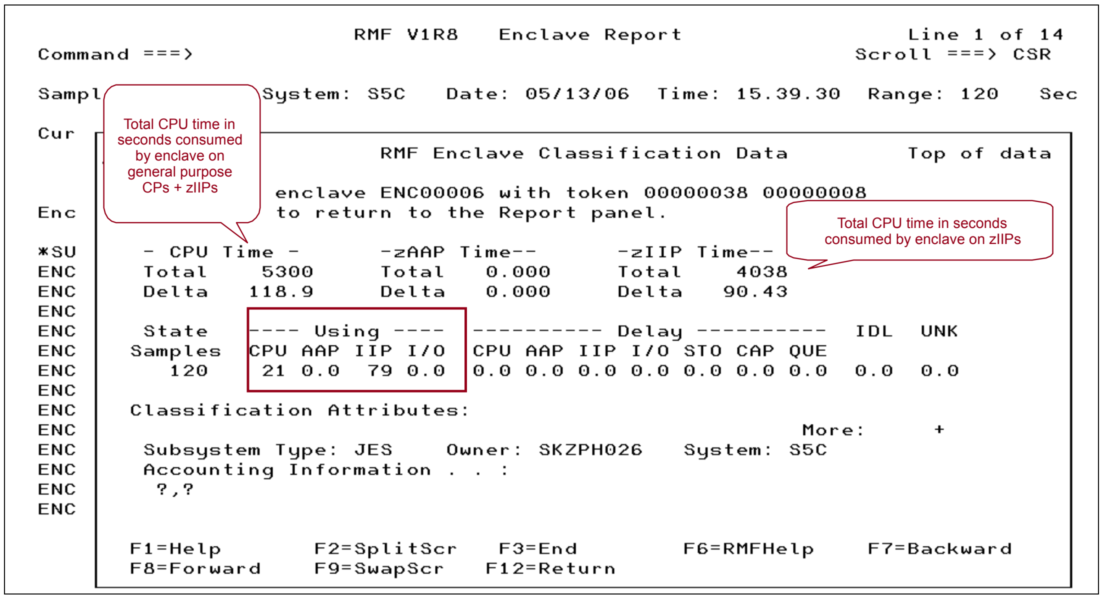
```
                          RMF V1R8   Enclave Report                   Line 1 of 14
 Command ===> _                                                   Scroll ===> CSR

 Samples: 120     System: S5C   Date: 05/13/06  Time: 15.41.30  Range: 120    Sec

 Current options:   Subsystem Type: ALL                              -- CPU Util --
                    Enclave Owner:                                   Appl%   EAppl%
                    Class/Group:                                      14.8    135.6

 Enclave    Attribute  CLS/GRP  P Goal     % D X    EAppl%   TCPU    USG   DLY   IDL

 *SUMMARY                                            58.88
 ENC00001              SYSOTHER 1 N/A                12.82  24215     21    73   0.0
 ENC00003              SYSOTHER 1 N/A                12.68  23914     34    61   0.0
 ENC00007              ZIPHIGH  1         70 Y       4.124   5409     99   0.8   0.0
 ENC00005              ZIPHIGH  1         70 Y       4.122   5601     99   0.8   0.0
 ENC00004              ZIPHIGH  1         70 Y       4.121   5735     98   1.7   0.0
 ENC00008              ZIPHIGH  1         70 Y       4.121   5395     99   0.8   0.0
 ENC00009              ZIPHIGH  1         70 Y       4.120   5224     98   1.7   0.0
 ENC00006              ZIPHIGH  1         70 Y       4.120   5418     99   0.8   0.0
 ENC00010              ZIPHIGH  1         70 Y       4.119   4087     98   1.7   0.0
 ENC00013              SYSOTHER 1 N/A                2.088  61.60    1.3    97   0.0
 ENC00012              SYSOTHER 1 N/A                0.854  79.78    0.0   0.0   0.0
 ENC00002              SYSOTHER 1 N/A                0.798   1356    0.0   0.0   0.0
 ENC00011              SYSOTHER 1 N/A                0.787  230.5    0.0   0.0   0.0
```

Figure 7-21 Monitor III enclave report indicating zIIPs usage

RMF Enclave Classification Data report

If you place your cursor on any data field of an enclave, as shown in Figure 7-21 on page 174, a pop-up panel appears, as shown in Figure 7-22. This panel displays performance statistics and classification attributes for the selected enclave.

The Using percentage of all types of processors is also displayed in this panel. All percentages are based on indicated numbers of state samples (120, in this example).

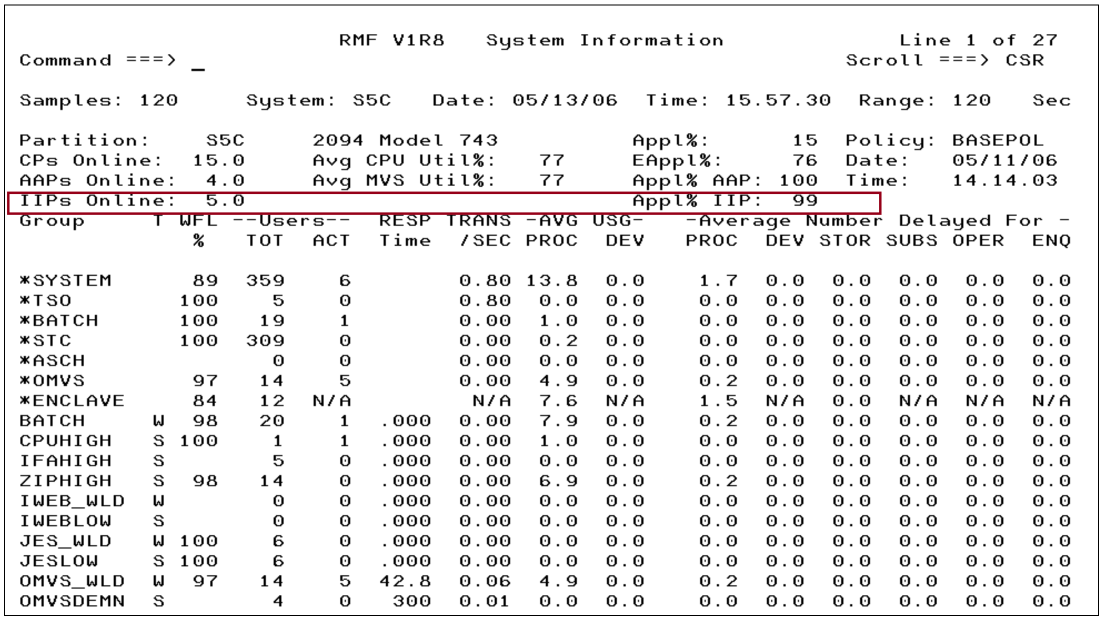

Total CPU time in seconds consumed by enclave on general purpose CPs + zIIPs
Total CPU time in seconds consumed by enclave on zIIPs

```
                         RMF V1R8   Enclave Report                  Line 1 of 14
Command ===>                                                      Scroll ===> CSR

Sampl                   System: S5C   Date: 05/13/06   Time: 15.39.30   Range: 120   Sec

Cur     ┌─────────────────────────────────────────────────────────────────────┐
        │                 RMF Enclave Classification Data          Top of data
        │
Enc     │           enclave ENC00006 with token 00000038 00000008
        │           to return to the Report panel.
*SU     │
ENC     │    - CPU Time -       -zAAP Time--       -zIIP Time--
ENC     │    Total    5300      Total    0.000     Total     4038
ENC     │    Delta   118.9      Delta    0.000     Delta    90.43
ENC     │
ENC     │    State     ---- Using ----  ---------- Delay ----------  IDL  UNK
ENC     │   Samples    CPU AAP IIP I/O  CPU AAP IIP I/O STO CAP QUE
ENC     │     120       21 0.0  79 0.0  0.0 0.0 0.0 0.0 0.0 0.0 0.0  0.0  0.0
ENC     │
ENC     │   Classification Attributes:
ENC     │                                                  More:     +
ENC     │    Subsystem Type: JES     Owner: SKZPH026   System: S5C
ENC     │    Accounting Information . . :
ENC     │     ?,?
ENC     │
        │   F1=Help       F2=SplitScr   F3=End        F6=RMFHelp    F7=Backward
        │   F8=Forward    F9=SwapScr   F12=Return
        └─────────────────────────────────────────────────────────────────────┘
```

Figure 7-22   Monitor III enclave detail report

RMF System Information report

Figure 7-23 is an example of Monitor III system information report showing the usage of zIIP processors. The Appl% IIP field shows the percentage of CPU time on zIIPs used by all address spaces and enclaves during the report interval. This value is divided by the number of zIIPs that have been active during this interval.

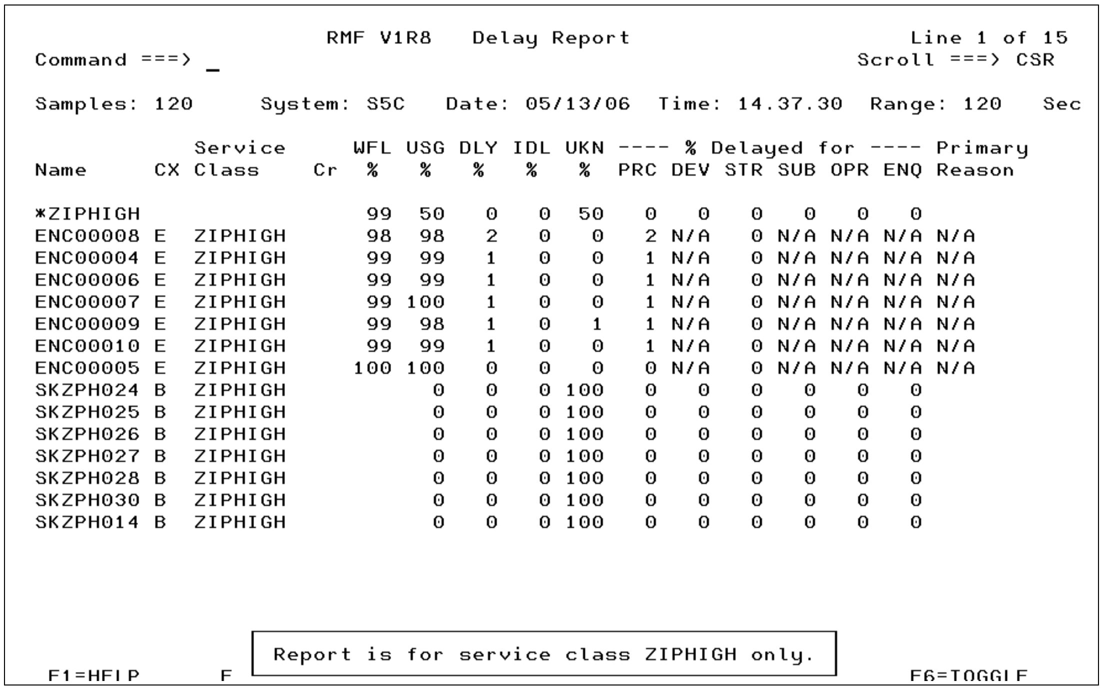


```
                         RMF V1R8   System Information                Line 1 of 27
Command ===> _                                                      Scroll ===> CSR

Samples: 120      System: S5C   Date: 05/13/06   Time: 15.57.30   Range: 120   Sec

Partition:     S5C      2094 Model 743          Appl%:       15  Policy: BASEPOL
CPs Online:   15.0      Avg CPU Util%:    77    EAppl%:      76  Date:   05/11/06
AAPs Online:   4.0      Avg MVS Util%:    77    Appl% AAP:  100  Time:   14.14.03
IIPs Online:   5.0                              Appl% IIP:   99
Group        T WFL --Users--  RESP TRANS -AVG USG-   -Average Number Delayed For -
                %   TOT  ACT  Time  /SEC PROC  DEV   PROC  DEV STOR SUBS OPER  ENQ

*SYSTEM        89   359    6        0.80 13.8  0.0    1.7  0.0  0.0  0.0  0.0  0.0
*TSO          100     5    0        0.80  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*BATCH        100    19    1        0.00  1.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*STC          100   309    0        0.00  0.2  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*ASCH                 0    0        0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*OMVS          97    14    5        0.00  4.9  0.0    0.2  0.0  0.0  0.0  0.0  0.0
*ENCLAVE       84    12  N/A         N/A  7.6  N/A    1.5  N/A  0.0  N/A  N/A  N/A
BATCH        W  98   20    1  .000  0.00  7.9  0.0    0.2  0.0  0.0  0.0  0.0  0.0
CPUHIGH      S 100    1    1  .000  0.00  1.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
IFAHIGH      S        5    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
ZIPHIGH      S  98   14    0  .000  0.00  6.9  0.0    0.2  0.0  0.0  0.0  0.0  0.0
IWEB_WLD     W        0    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
IWEBLOW      S        0    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
JES_WLD      W 100    6    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
JESLOW       S 100    6    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
OMVS_WLD     W  97   14    5  42.8  0.06  4.9  0.0    0.2  0.0  0.0  0.0  0.0  0.0
OMVSDEMN     S        4    0   300  0.01  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
```

Figure 7-23   Sample RMF System Information Report indicating zIIPs usage

RMF Delay Report

When you press any key under the service class of enclaves (ZIPHIGH, in this example) on the system information panel (shown in Figure 7-23 on page 175), you will reach the panel shown in Figure 7-24, where you can get detail information about enclaves service class ZIPHIGH.



```
                      RMF V1R8   Delay Report                    Line 1 of 15
Command ===> _                                                Scroll ===> CSR

Samples: 120     System: S5C   Date: 05/13/06  Time: 14.37.30  Range: 120   Sec

            Service      WFL USG DLY IDL UKN ---- % Delayed for ---- Primary
Name     CX Class     Cr  %   %   %   %   %  PRC DEV STR SUB OPR ENQ Reason

*ZIPHIGH                  99  50   0   0  50   0   0   0   0   0   0
ENC00008 E  ZIPHIGH       98  98   2   0   0   2 N/A   0 N/A N/A N/A N/A
ENC00004 E  ZIPHIGH       99  99   1   0   0   1 N/A   0 N/A N/A N/A N/A
ENC00006 E  ZIPHIGH       99  99   1   0   0   1 N/A   0 N/A N/A N/A N/A
ENC00007 E  ZIPHIGH       99 100   1   0   0   1 N/A   0 N/A N/A N/A N/A
ENC00009 E  ZIPHIGH       99  98   1   0   1   1 N/A   0 N/A N/A N/A N/A
ENC00010 E  ZIPHIGH       99  99   1   0   0   1 N/A   0 N/A N/A N/A N/A
ENC00005 E  ZIPHIGH      100 100   0   0   0   0 N/A   0 N/A N/A N/A N/A
SKZPH024 B  ZIPHIGH            0   0   0 100   0   0   0   0   0   0
SKZPH025 B  ZIPHIGH            0   0   0 100   0   0   0   0   0   0
SKZPH026 B  ZIPHIGH            0   0   0 100   0   0   0   0   0   0
SKZPH027 B  ZIPHIGH            0   0   0 100   0   0   0   0   0   0
SKZPH028 B  ZIPHIGH            0   0   0 100   0   0   0   0   0   0
SKZPH030 B  ZIPHIGH            0   0   0 100   0   0   0   0   0   0
SKZPH014 B  ZIPHIGH            0   0   0 100   0   0   0   0   0   0




                   Report is for service class ZIPHIGH only.
 F1=HELP      F                                                     F6=TOGGLE
```

Figure 7-24   Service class detail information using zIIPs

8

Real Storage Manager (RSM) and System Resource Manager (SRM)

This chapter discusses changes made to the Real Storage Manager (RSM) and System Resource Manager (SRM) in order to support up to 4 TB of central storage in z/OS V1R8. Most of the changes are made to the RSM component.

In this chapter, the following topics are described:

- RSM and SRM roles
  - Real Storage Manager (RSM)
  - System Resource Manager (SRM)
  - Concepts in real storage management
- RSM and SRM algorithms pre z/OS V1R8
  - Page stealing pre-z/OS V1R8
  - UIC calculation pre-z/OS V1R8
  - Pageable storage shortage handling pre z/OS V1R8
  - Swapping pre-z/OS V1R8
- Reasons for changes in algorithms
- Page stealing algorithm in z/OS V1R8
- UIC calculation in z/OS V1R8
- Swap algorithms in z/OS V1R8 - remove of physical swap
- Page frame table changes in z/OS V1R8
- RMF report and SMF record changes
- Messages changed
- Interactions and dependencies
- Migration and coexistence

8.1 Introduction to central storage management

Currently the z/OS operating system limits the amount of central storage that can be configured to a single z/OS image to only 128 GB. As processing power increases, demand for processor memory also increases. Day by day, the cost of a single central storage frame is decreasing. System growth and scalability will be inhibited if central storage continues to be limited to 128 GB.

The operating system needs to remove this limitation in order to take advantage of the greater scalability and larger memory that the new z9 machines provide. The greater than 128 GB support removes this limitation and allows the operating system to support up to 4 terabytes (4 TB) of central storage. The support is totally transparent to problem state applications and authorized programs outside z/OS, but these applications are allowed to benefit due to less system overhead and the greater degree of concurrency that the greater than 128 GB support provides.

Current storage support

Currently, the maximum storage supported by the hardware is as follows:

- z990 - up to 256 GB of central storage
- z9 - up to 512 GB of central storage
- Next generation - some value larger than 512 GB

Prior to z/OS V1R8, RSM and SRM algorithms are not very suitable for supporting this larger amounts of central storage in terms of performance, availability, and quality of service. As the amount of central storage used by one image increases, the impact of today's algorithms is becoming intolerable. Therefore, the method of RSM and SRM handling real storage management is changed in z/OS V1R8.

As larger central storage is put into use, the CPU overhead of paging and swapping this amount of central storage is becoming a significant problem. This support also provides improvements to major consumers of central storage to avoid the CPU overhead of paging and swapping, and it drives greater MIPS growth for z/OS environments.

8.1.1 z/OS V1R8 support of central storage

In Z/OS V1R8, RSM and SRM changes have three main goals:

- Support larger central storage
- Decrease CPU overhead and the risk of problems during paging and swapping in large amounts of central storage
- Improve algorithms to react faster to changing system demands, in order to increase system throughput and availability

These goals, as well as additional internal support, are illustrated in Figure 8-1 on page 179.

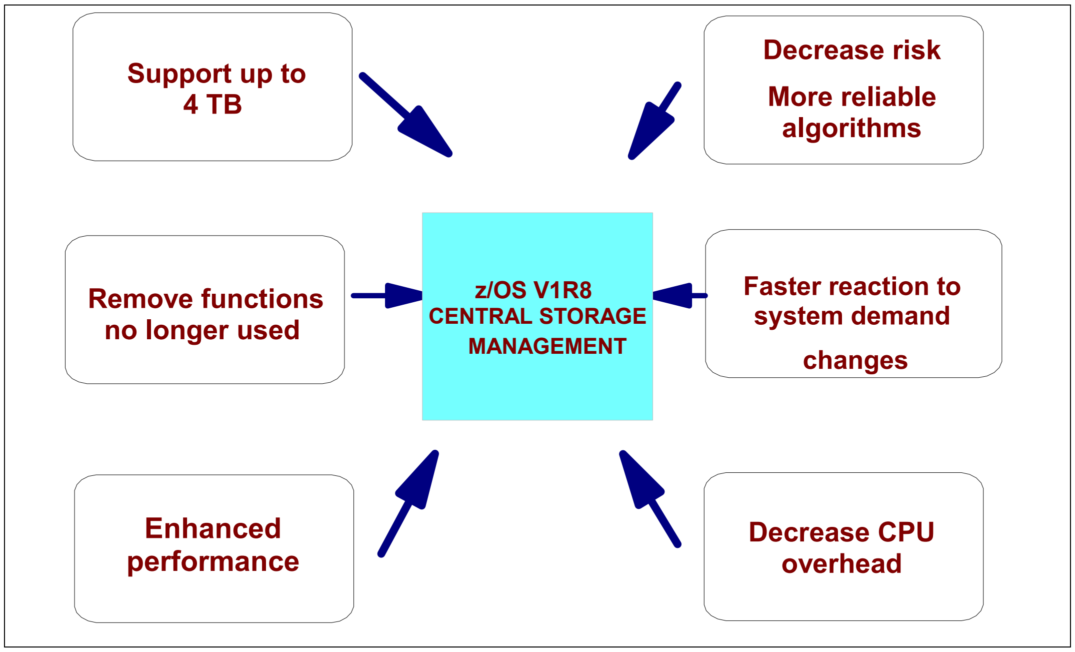

Support up to 4 TB
Decrease risk
More reliable algorithms
Remove functions no longer used
z/OS V1R8 CENTRAL STORAGE MANAGEMENT
Faster reaction to system demand changes
Enhanced performance
Decrease CPU overhead

Figure 8-1 z/OS V1R8 central storage management enhancements

In order to achieve these goals, the following processes are changed:

- Page replacement algorithm
- UIC calculation algorithm
- Swapping algorithm - physical swapping is removed
- Storage shortage handling

8.2 Real Storage Manager (RSM)

Real Storage Manager (RSM) is the component of the base control program that is responsible for allocation of real storage. It works together with SRM in algorithms related to real storage.

RSM is responsible for the following functions:

- Working with SRM in swapping, stealing, and unreferenced interval count (UIC) calculation processes
- Allocating central storage to satisfy GETMAIN requests for SQA and LSQA
- Allocating central storage for page fixing
- Reclaiming the central storage of an address space when it is to be swapped out of central storage
- Building the control blocks that are necessary in order to swap back in an address space
- Allocating central storage for an address space that is to be swapped in

8.3 System Resource Manager (SRM)

System Resource Manager (SRM) is a component of the system control program that is responsible for determining which address spaces should be given access to which system resources, and the rate at which they are allowed to consume these resources.

SRM makes these decisions by considering an installation's defined response and the work priority requirements for the individual address spaces, and it tries to increase system throughput at the same time. It tries to optimize the use of system resources by periodically monitoring and balancing resource utilization.

SRM is responsible for the following functions related to central storage usage:

- Pageable frame stealing
- Swapping
- Prevention of storage shortages
- UIC calculation

8.4 Concepts in real storage management

The following terminology is used to describe real storage and auxiliary storage:

Available frame queue	This queue shows which real storage frames are available to be used by address spaces. It is global to the whole system.
Page stealing	This is the process of moving a used frame of an address space to auxiliary storage by making an address space stop using it, and then placing the stolen frame on the available frame queue in order to give it to another address space.
Swapping	This is a mechanism used by SRM to control distribution of resources and system throughput. Swapping is transferring all of the pages of an address space between central storage and auxiliary storage.
Unreferenced interval count	An unreferenced interval count (UIC) value is related to each pageable frame in central storage that shows how long it has not been referenced. The UIC value is used in page stealing algorithms, and it is also used by SRM in several internal decisions.
Page frame table	This is a table used to contain necessary information related to each frame in all central storage.
Pageable frame queue	Every address space that exists in real storage has its own pageable frame queues. The pageable frame queue of each address space contains information about which frames are eligible for paging.

8.5 Unreferenced interval count (UIC)

In central storage, every in-use pageable frame has an unreferenced interval count (UIC) that signifies the age of unreference for that frame. The calculation of UICs for each pageable frame in central storage is also done by SRM. SRM uses a least recently used algorithm (LRU) during the page stealing process. The LRU algorithm uses the UIC value of each pageable frame during page stealing to page out to auxiliary storage the oldest frames (those with the highest UIC) owned by an address space.

Stealing takes place strictly on a demand basis; that is, there is no periodic stealing of long-unreferenced frames. SRM tells RSM when to and how many frames to steal. SRM uses two constants, AVQLOW and AVQOK, in deciding when to start page stealing and when to stop it.

When the number of frames on the AFQ falls below AVQLOW, SRM calls RSM to replenish the AFQ by removing some of the current page data from real memory. AFQ replenishment continues until the number of frames in AFQ reaches the AVQOK value.

The oldest frames from each address space are stolen first. SRM tells RSM the UIC threshold value. RSM will steal pages older than this UIC threshold value. If RSM cannot steal as many pages as SRM needs, SRM sends another UIC threshold value. This process continues until the AVQOK value is reached. Stealing is done on an address space basis. RSM looks at every address space in the pageable frame queue to check which frames to steal.

8.5.1 UIC calculation pre-z/OS V1R8

A UIC calculation process, as illustrated in Figure 8-2, is necessary for the page stealing process. In pre-z/OS V1R8 systems, SRM starts the UIC update processing every 10 seconds. In addition, the highest value UIC is also necessary for SRM in its decisions.

The page stealing process uses a “least recently used” algorithm and uses information about the frame’s reference times and age, because each frame has a reference bit and a UIC value. The reference bit value shows whether it is referenced since the last check or not. The UIC of a frame shows how long that frame has not been referenced since the last check.

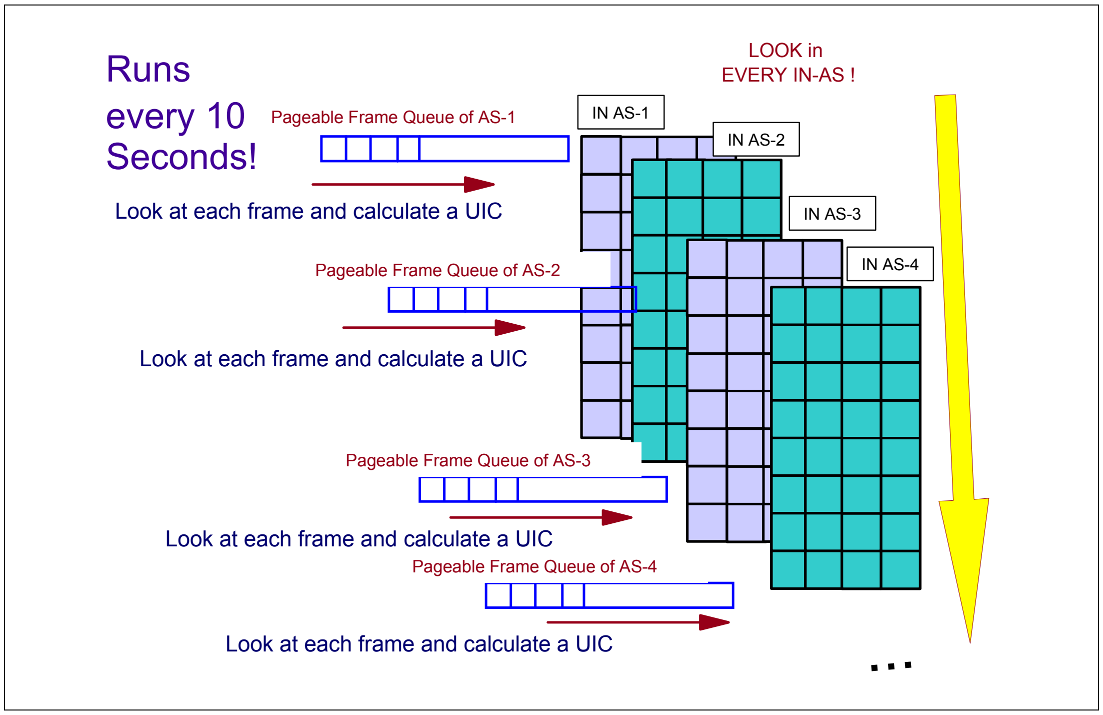

Runs every 10 Seconds!
LOOK in EVERY IN-AS !
Pageable Frame Queue of AS-1
Look at each frame and calculate a UIC
Pageable Frame Queue of AS-2
Look at each frame and calculate a UIC
Pageable Frame Queue of AS-3
Look at each frame and calculate a UIC
Pageable Frame Queue of AS-4
Look at each frame and calculate a UIC
IN AS-1
IN AS-2
IN AS-3
IN AS-4
...

Figure 8-2 UIC calculation in pre-z/OS V1R8 systems

Note: Frames that belong to an address space on the SWAP-IN queue are also called *in-use frames*. Frames that are in the pageable frame queue are referenced as in-use frames.

Frame reference bit

During this process (which occurs every 10 seconds), for each frame that belongs to an address space on the swap-in queue, the UIC value is recalculated and updated using the algorithm shown in Figure 8-3. This checking is done for all in-use frames.

Every frame in use by each address space on the swapped-in queue needs to be processed in order to keep its UIC current. The oldest frame in an address space represents the address space high UIC. The oldest frame of all swapped-in address spaces represents the system high UIC.

In addition to updating the UIC values for each in-use frame in real storage, analyzed frames are counted and, depending on their age, are distributed to buckets.This is called a *frame bucket calculation* and it is done during UIC processing. SRM uses the UIC values of in-use frames in order to decide which frames to steal. The oldest frames from each address space, that is, frames with highest UIC values, are stolen first.

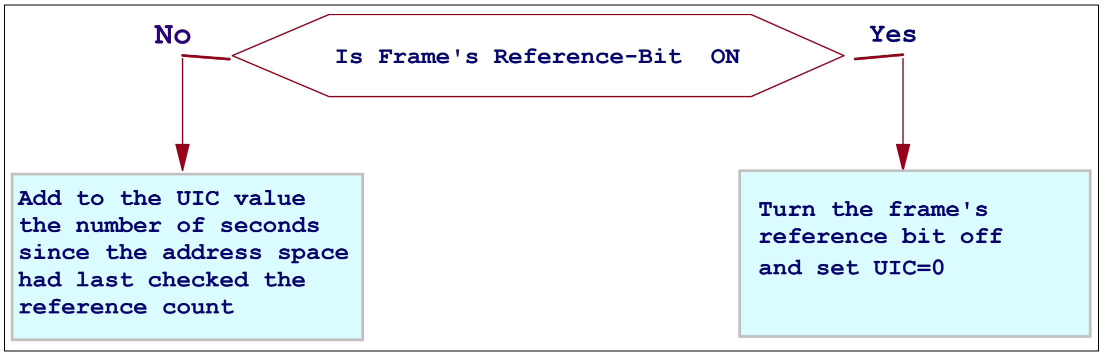

No
Is Frame's Reference-Bit ON
Yes
Add to the UIC value the number of seconds since the address space had last checked the reference count
Turn the frame's reference bit off and set UIC=0

Figure 8-3 UIC calculation

Important: This sort of UIC update processing is totally eliminated in z/OS V1R8. Refer to “8.5.2, “UIC calculation with z/OS V1R8” on page 182” and “New UIC calculation” on page 183 for the new UIC calculation.

8.5.2 UIC calculation with z/OS V1R8

The UIC calculation is totally changed in z/OS V1R8. These changes resulted from the following problems with pre-z/OS V1R8 UIC calculations:

1. The UIC update process becomes more CPU time consuming, and takes longer.

   The UIC update process becomes more CPU time consuming as an address space’s real storage allocations increase. It takes a longer time to finish this process, especially in large real storage allocations.

2. The UIC update process is done even when AFQ has many available frames.

   This UIC update process is done for each in-use frame every 10 seconds even when the available frame queue (AVQ) has a lot of available frames. This is especially true in systems with many large address spaces, and becomes a very CPU time consuming process.

3. The UIC update process holds locks in order to process queues.

   The UIC update works with queues, which requires the use of a serialization mechanism, such as holding address space locks. While the UIC update is running for a particular address space, RSM cannot perform any other functions (such as servicing page faults for that address space). As an address space's real storage allocations increase, it takes much more time to finish this process and the unavailable time for that address space increases.

4. There is disruption of a UIC update.

   Scanning queues presents certain risks. It is dangerous to be interruptible. Disruption of the UIC update can be a problem as the amount of real storage allocated to address spaces increases. The system may even go into a spin loop state while supporting larger real memory usage.

Because there is no UIC value calculated for every in-use frame anymore, the LRU algorithm cannot be used in page stealing. Instead, a global LRU algorithm is implemented in z/OS V1R8. Page stealing looks at an address space's in-frame queues to see which frame's UIC it should check and steal. A lock mechanism is needed to avoid getting a frame out from the queue and putting it into a different queue.

SRM algorithms changes with z/OS V1R8

In z/OS V1R8, although the UIC value is not used in page stealing, the SRM algorithms that depend on UIC values continue to work. These algorithms depend on an average system high UIC and also on the UIC bucket distribution. The average system high UIC is used in several SRM algorithms, as follows:

- MPL adjustment
- Swap decisions
- Storage isolation for address spaces

New UIC calculation

A need still exists for average system high UIC value and UIC bucket distribution information, therefore the UIC calculation is still necessary in z/OS V1R8, but a totally differently algorithm is used. In zOS V1R8, the UIC represents the time for one steal cycle through the whole of storage. The UIC is a good indicator of the demand on central storage. Therefore, the following changes are made:

- The current UIC gets calculated every second.
- The minimum UIC represents the lowest UIC during the last walk through the whole of storage.
- The maximum UIC represents the highest UIC during the last walk through the whole of storage.
- A UIC value of 65535 indicates that there are ample available frames in the system.

Note: The higher the UIC value, the less contention for storage in the system. The lower the UIC value, the more contention for storage in the system. A very low UIC indicates that the system is storage-constrained.

Real storage is split into logical segments, as shown in Figure 8-5 on page 185. The number of logical segments depends on the amount of storage. As discussed in 8.6.1, "Page stealing algorithm in z/OS V1R8" on page 186, there is a cursor that moves on the page frame table when the page stealing SRB is running.

Some calculations are also done on a segment basis during this stealing process, to achieve system UIC values and UIC bucket distribution. Instead of a *frame* having a UIC value, now these logical segments have UIC values.

When the cursor enters and leaves a logical segment, the current time is stored in the logical segment entry. Every second SRM uses these time stamps to recalculate the logical segment UIC value.

A UIC value for each logical segment contains two time values. The first time value represents the time spent in the logical storage segment other than the UIC is calculated for. The second time value represents the time spent in the segment in which the UIC is calculated for.

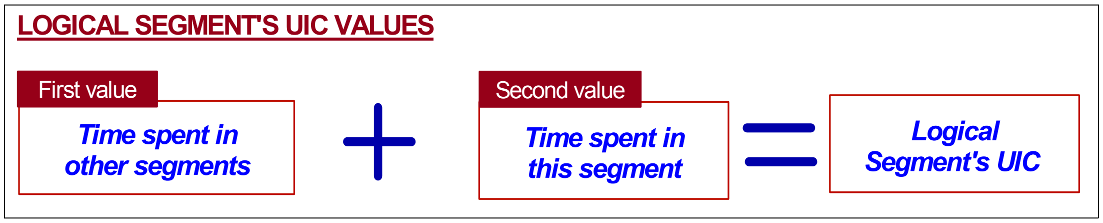

LOGICAL SEGMENT'S UIC VALUES
First value
Time spent in other segments
Second value
Time spent in this segment
Logical Segment's UIC

Figure 8-4 Logical segment's UIC value

New UIC values

The UIC is a good indicator of the demand on central storage. As previously mentioned, in zOS V1R8, the UIC represents the time for one steal cycle though the whole of storage. Every second, SRM recalculates each logical segment's UIC value using the time stamps that the cursor stored in the entry. A logical segment's UIC values are then used to calculate the different systems's UIC for internal management, and to display in external monitor programs like RMF. The calculations are explained in Table 8-1.

Table 8-1 UIC values with z/OS V1R8

System UIC type	Definition
Current UIC	The current UIC gets calculated every second. The current UIC is used for SRM internal management.
Minimum UIC	The Minimum UIC represents the lowest UIC during the last walk through the whole of storage.
Maximum UIC	The Maximum UIC represents the highest UIC during the last walk through the whole of storage.

The current UIC is used for SRM internal management. The new UICs vary between 0 and 65535. A UIC value of 65535 means no cursor movement, and that there are enough frames on the available frame queue in the system. The highest UIC value that is represented in several RMF reports is now replaced by the current system UIC in z/OS V1R8. A UIC value is now a useful indicator of demand on central storage, as follows:

- The higher the UIC value, the less contention for central storage in the system. A UIC value of 65535 indicates there are ample available frames in the system.
- The lower the UIC value, the more contention for storage in the system. A very low UIC indicates that the system is central storage-constrained.

Figure 8-5 on page 185 illustrates the UIC calculation in z/OS V1R8.

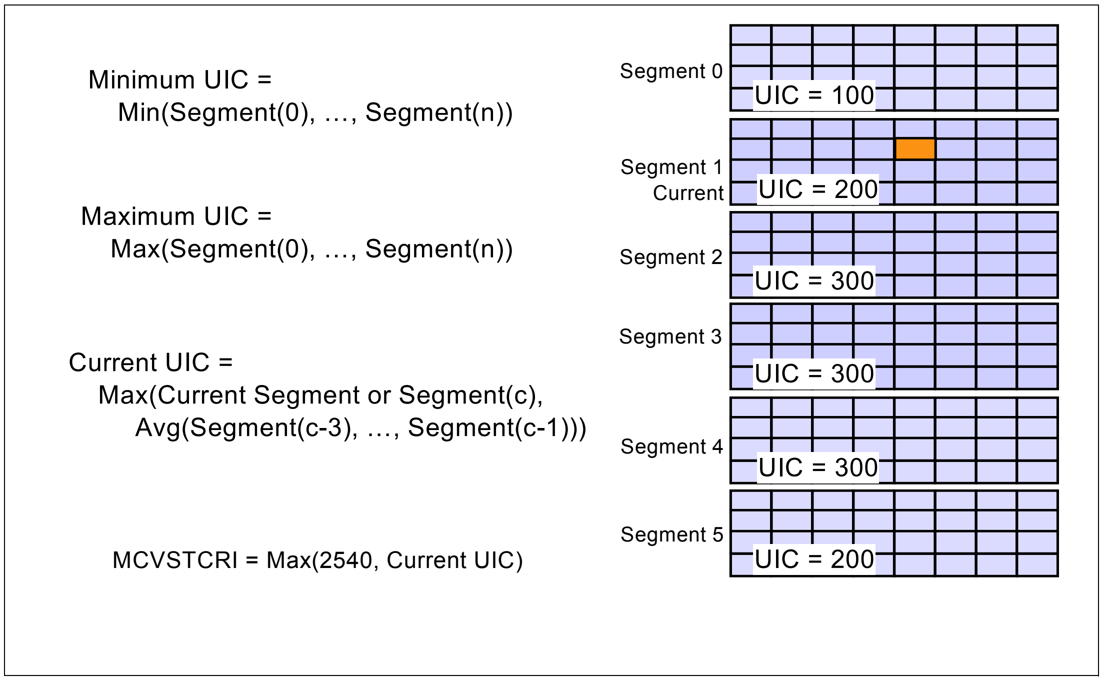

Minimum UIC =
Min(Segment(0), …, Segment(n))
Maximum UIC =
Max(Segment(0), …, Segment(n))
Current UIC =
Max(Current Segment or Segment(c),
Avg(Segment(c-3), …, Segment(c-1)))
MCVSTCRI = Max(2540, Current UIC)
Segment 0
UIC = 100
Segment 1
Current
UIC = 200
Segment 2
UIC = 300
Segment 3
UIC = 300
Segment 4
UIC = 300
Segment 5
UIC = 200

Figure 8-5 UIC calculation in z/OS V1R8

8.6 Page stealing pre-z/OS V1R8

Figure 8-6 on page 186 illustrates the method of page stealing. *Page stealing* (or *page replacement*) is the process of selecting which pages to displace from central storage and paging them out to auxiliary storage. It is accomplished by paging out to auxiliary storage the oldest frames (those with the highest UIC) owned by an address space.

The UIC update process runs periodically to update the UIC of each in use pageable frame owned by an address space. SRM does page stealing to satisfy page faults or to swap in an address space. SRM selects which pages to displace from central storage, and pages them out to auxiliary storage.

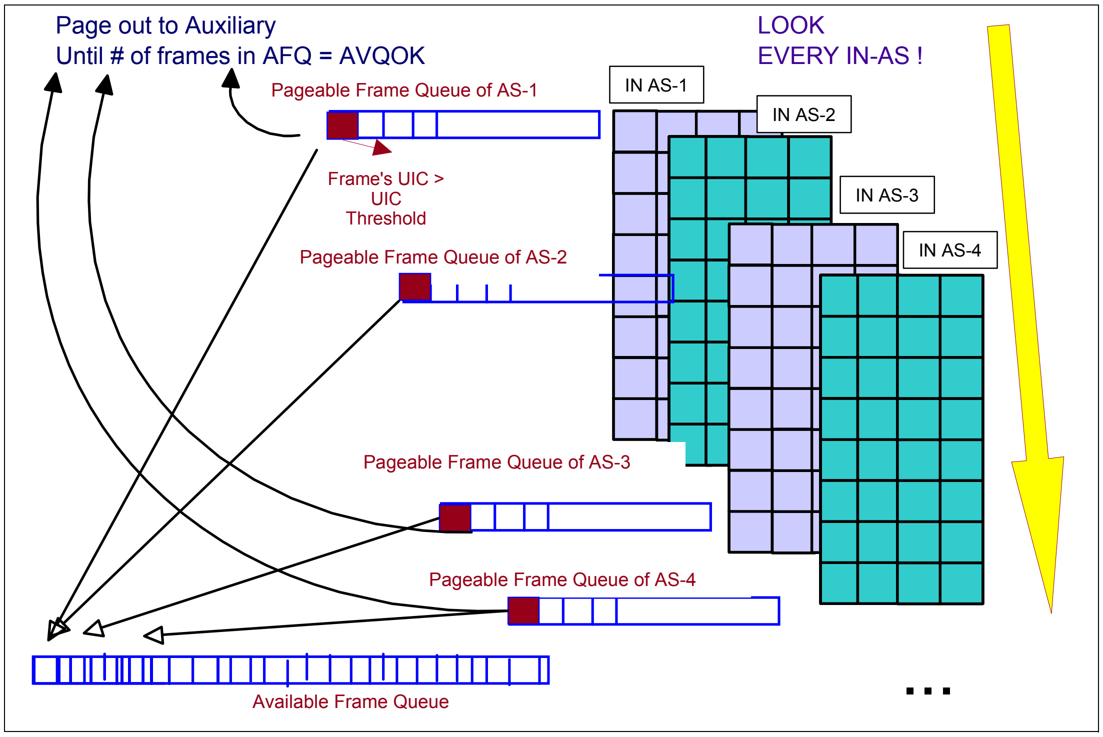

Page out to Auxiliary
Until # of frames in AFQ = AVQOK
LOOK
EVERY IN-AS !
Pageable Frame Queue of AS-1
IN AS-1
IN AS-2
Frame's UIC > UIC Threshold
IN AS-3
Pageable Frame Queue of AS-2
IN AS-4
Pageable Frame Queue of AS-3
Pageable Frame Queue of AS-4
Available Frame Queue
...

Figure 8-6 Page stealing in pre-z/OS V1R8 systems

As previously mentioned, page replacement in z/OS is currently done by a least recently used (LRU) algorithm that keeps an unreferenced interval count (UIC) that signifies the age of unreferenced frames.

RSM maintains a free list of page frames that are available to satisfy page faults; this list is called the *available frame queue* (AFQ). Page frames are returned to the AFQ when storage is freed explicitly or implicitly when a program ends.

In most environments, because the amount of virtual memory that is in use at any given instant may be larger than real memory, the system itself must replenish the AFQ. The AFQ replenishment is driven by the AVQ LOW and AVQOK thresholds managed by the SRM.

When the number of frames on the AFQ falls below the AVQ LOW threshold, SRM calls RSM to replenish the AFQ by removing some of the current page data from real memory. The system stops replenishing the AFQ when the AVQOK threshold is reached. Stealing happens on an address space basis, and the oldest frames from each address space are stolen first.

8.6.1 Page stealing algorithm in z/OS V1R8

The page stealing algorithm that z/OS uses is changed to be more efficient, and to safely process large amounts of real storage. The reasons for the changes (including the potential for high CPU consumption and the risk of scanning queues), and the elimination of UIC update processing, are explained in 8.5.2, "UIC calculation with z/OS V1R8" on page 182. Stealing pages once UICs were calculated was not a CPU-consuming process, but presented risks and therefore also is changed because of the algorithm changes made in the UIC calculation.

In the new UIC calculation algorithm, UIC values of frames that belong to all in-storage address spaces are not calculated periodically anymore; refer to "New UIC calculation" on page 183 for more detail.

Therefore, in order to decide which page to steal, RSM now uses a global LRU algorithm instead of the old LRU. This algorithm is new and totally different from LRU. In z/OS V1R8, stealing works against the whole of storage. Now, only the frames of address spaces that are on the in-storage queue are candidates for steal.

Stealing happens on a global basis. It works against all frames, regardless of which one the address space is on. In contrast to the old LRU algorithm, instead of using only swapped-in address space frames that exist in each address space AFQ, all of the frames in central storage are examined as input to decide which frame to steal.

Global LRU algorithm

The global LRU algorithm technique takes advantage of a referenced bit for each page as an indication of what pages have been recently used (referenced). When a page-stealer routine is called, it cycles through a page frame table, examining each page's referenced bit. If the page was unreferenced and can be stolen (that is, it is not pinned and it meets other page-stealing criteria), it is stolen and placed on the AFQ. Referenced pages may not be stolen, but their reference bit is reset, effectively “aging” the reference so that the page may be stolen the next time a page-stealing algorithm is invoked.

Note: In pre-z/OS V1R8, the page frame table size was 2 GB. Its size is changed in z/OS V1R8 to support up to 4 TB.

SRM and the AFQ

Every SRM second, SRM checks if it is necessary to replenish the AFQ. SRM stealing of pages runs when the AFQ needs to be replenished. SRM analyzes frames and decides to steal it or not. If frame is a candidate to steal, it has the following status:

- Not an offline frame
- Not a fixed frame
- Not an otherwise protected frame
- Not referenced since the last visit

Frames get stolen and are placed on the AFQ. When enough frames have been stolen, the stealing process stops. When the stealing process ends, the cursor stops moving. Until the next time SRM decides to steal frames, the cursor stays where it is.

Referenced bits may not be stolen, but if a frame’s reference bit is ON, this process sets it OFF. By resetting a reference bit, this process is effectively “aging” the reference so that a page may be stolen the next time the page stealing algorithm is invoked.

Stealing frames

During the page stealing process, when the scan runs over some thousands of frames and does not find a certain percentage of frames to steal, a “CURSOR RESET” occurs. This means that the cursor moves to beginning of the page frame table and continues from there. The second pass over the same frames will then find enough frames to steal. Cursor reset occurs mainly when the cursor has not moved for a long time.

With the new algorithm, a new “pre-stealing” mechanism is introduced. The aim of the pre-stealing algorithm is to more quickly steal a frame when it is required. This is done by another cursor, which looks at frames and their conditions. As a result of this analysis, RSM copies (not moves) frames to auxiliary storage when their properties meet certain conditions. This helps RSM to act more quickly when the deciding that pages can be stolen. With this pre-stealing algorithm, there may be more page-outs than page-ins.

The new global LRU algorithm does not look at UIC values. Instead, only reference bits are checked. Stealing is done in two stages, as follows:

- Age stealing

  Age stealing is still performed and therefore frame buckets are still established. UIC calculations are still done, but in a totally different way in which no UIC update is done in only part of real storage frame basis.

- Global LRU stealing

  Instead of scanning queues, page frame table entries are scanned, resulting in no risk and a safer algorithm.

Important: SRM constants AVQLOW and AVQOK are not constants anymore. SRM now raises and lowers the AVQLOW and AVQOK thresholds by sampling low frame events in a measurement interval.

8.7 Swapping pre-z/OS V1R8

Swapping is the primary function used by SRM to exercise control over the distribution of resources and system throughput. *Swapping* is the transferring of pages in an address space between central storage and auxiliary storage.

Note the following points:

- A swapped-in address space is an active one that has pages in central storage and pages in auxiliary storage.
- A swapped-out address space is an inactive one that has all pages in auxiliary storage. A swapped-out address space cannot execute until it is swapped-in.

SRM has the responsibility of deciding which address space to swap-in, which one to swap-out, and when to perform these operations. Using the system status information and feedback related to resource usage that is periodically monitored, SRM determines which address space should have access to system resources.

In general, swapping is used to have control of:

- Domains
- Competition for resources between individual address spaces within domains
- System-wide performance and throughput

Swap triggers, which indicate to SRM that a swap process should be run, can be categorized as two types:

- Domain-related swaps
- System-related swaps

Figure 8-7 on page 189 illustrates the reasons for swap triggers, and the related swap actions.

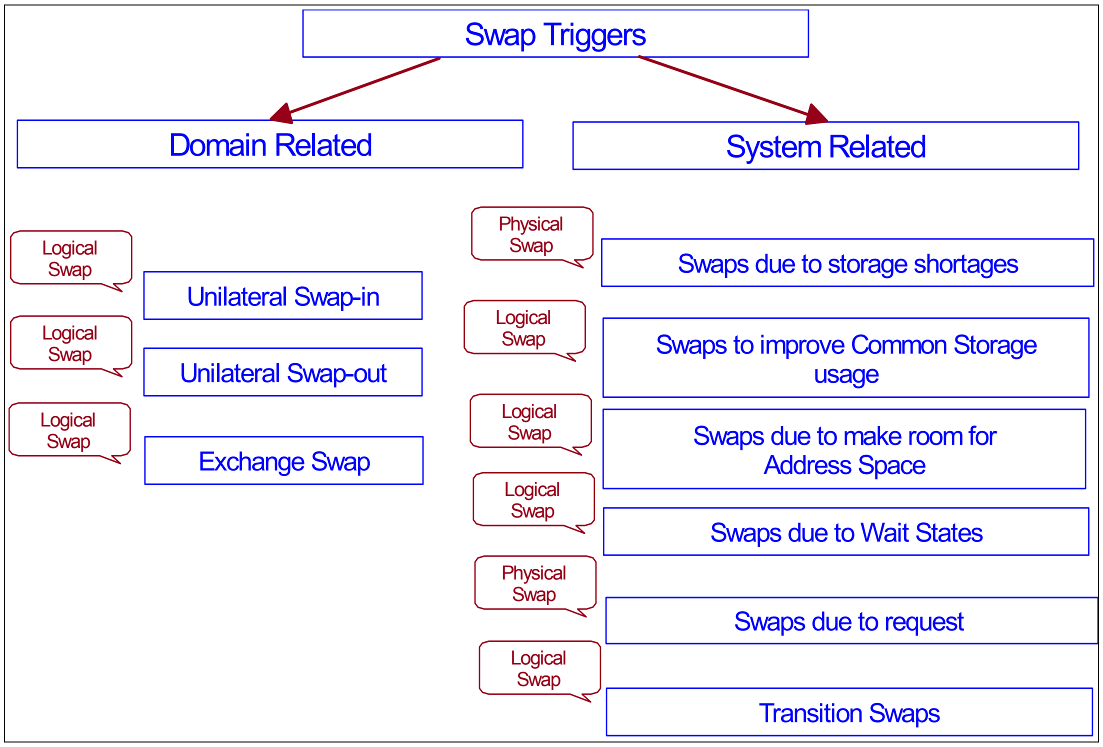


Figure 8-7 Swap triggers

Domain-related swaps

In order to reduce the processor and channel subsystem overhead faced during a physical swap needing to access to auxiliary storage, SRM performs logical swaps where possible. In a logical swap, LSQA fixed frames and recently referenced frames are kept in central storage in contrast to physical swaps, where these frames are moved to auxiliary storage.

Address spaces that are swapped for wait state conditions are the best candidates for logical swaps. Physical swapping is done to resolve page storage shortages. Those swaps are triggered because of pageable storage shortages.

The reasons for domain-related swaps and the related actions are as follows:

Unilateral swap-in Reason: The number of address spaces that are in a multiprogramming set related to one domain is less than the domain's swap-in target number set by SRM.

Action: SRM swaps in additional address spaces related to that domain if possible.

Unilateral swap-out Reason: The number of address spaces that are in a multiprogramming set related to one domain is greater than the domain's swap-out target number set by SRM.

Action: SRM swaps out address spaces from that domain.

Exchange swap Reason: Inside one domain, an address space in a multiprogramming set has exceeded its allotted portion of resources, while another address space of the same domain is waiting to be swapped in.

Action: SRM swaps out address spaces that exceeded resource usage and swaps in address spaces that are waiting to be swapped in.

Important: In pre-z/OS V1R8, physical swapping is used to resolve pageable storage shortages.

System-related swaps

Swaps can be done for the following resource problems:

Storage shortages Two types of shortages cause swaps:

Auxiliary storage shortages - if the number of available auxiliary storage slots is low, SRM will swap out the address space that is acquiring auxiliary storage at the fastest rate.

Pageable frame shortages - for a shortage of pageable frames, if the number of fixed frames is very high, SRM will swap out the address space that acquired the greatest number of fixed frames.

This process continues until the number of available slots rises above a fixed target, or until the number of fixed frames falls below a fixed target.

Improve storage usage The system will swap out an address space when the system determines that the current mix of address spaces is not best utilizing central storage.

The system swaps out address spaces to create a positive effect on system paging and swap costs.

Swap address spaces This occurs when there is a need to swap out an address space to make room for another address space. The system will swap in an address space when the system determines that it has been out longer than its recommendation value would dictate.

Wait states In certain cases, such as a batch job going into a long wait state (LONG option specified on the WAIT SVC; an STIMER wait specification of greater than or equal to 0.5 seconds; an ENQ for a resource held by a swapped out user), the address space will itself signal SRM to be swapped out in order to release storage for the use of other address spaces. Another example would be a time sharing user's address space that is waiting for input from the terminal after a transaction has completed processing.

SRM also detects address spaces in a wait state. That is, address spaces in central storage that are not executable for a fixed interval will be swapped.

System requests The system may request that an address space be swapped out of address spaces that occupy frames in the storage unit to be taken offline.

Address space status A transition swap occurs when the status of an address space changes from swappable to nonswappable. For example, the system performs a transition swap out before a nonswappable program or V=R step gets control.

This special swap prevents the job step from improperly using reconfigurable storage.

8.7.1 Pageable storage shortage handling pre-z/OS V1R8

When the pageable storage available to the system is below the threshold value defined in the IEAOPTxx parmlib member, SRM realizes that pageable storage shortages occurred and starts the process to relieve storage. This process depends on the reason for the shortage. There can be two reasons for pageable storage shortages:

- Too many address spaces are already in storage

  Because SRM performs an MPL adjustment and a logical swap threshold adjustment, this is generally not the reason for the shortage. The methods that SRM uses keeps an adequate amount of fixed storage available to back the address spaces.

- Too much page fixing is taking place

  If the reason for the shortage is that one or more address spaces are using too much fixed storage, SRM selects an address space that uses the most amount of fixed storage and swaps this address space to auxiliary storage. This swap is called a *physical swap*. SRM cannot use logical swap because it needs LSQA and SQA areas that are fixed also to be freed, and only a physical swap can swap these areas out. That is the reason why SRM uses physical swap to resolve pageable storage shortages, as shown in Figure 8-8.

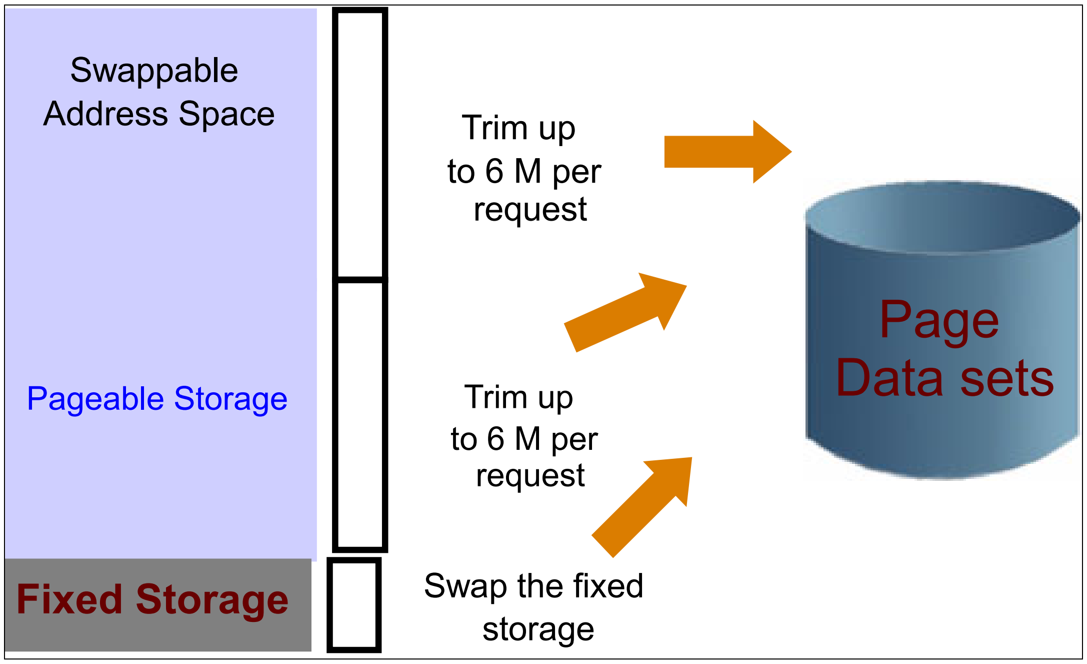


Figure 8-8 Pageable storage shortage handling pre-z/OS V1R8 - physical swap

Note: The fixed storage usage can be done either explicitly or implicitly; either explicitly through requests to fix virtual storage, or implicitly, which is when the address space is using LSQA or SQA that are in defined in fixed areas.

8.7.2 Swap algorithms in z/OS V1R8

In z/OS V1R8, the swapping methods and triggers still function as before. The only change in z/OS V1R8 is that *physical swaps are eliminated*. Also changed is pageable storage shortage handling. In pre-z/OS V1R8 systems, physical swaps are used mainly to resolve pageable storage shortages. The reason for removing physical swapping is because in most installations, physical swaps are now rare occurrences and are not useful anymore.

New swap technique

As z/OS V1R8 no longer swaps address spaces out to auxiliary storage to resolve a pageable storage shortage problem, the frames are now moved higher within central storage. With z/OS V1R8, an address space physical swap to auxiliary storage that used to occur in previous releases is replaced by a different technique, as explained here.

- If the shortage is below 16 MB or between 16 MB and 2 GB, as shown in Figure 8-9, such a shortage gets resolved as follows:
  - Search for the best candidate address space to resolve such a shortage
  - Put the address space in an “IN-Real Swap” state by quiescing the address space so that nothing is running in the address space
  - Move the fixed frames of this address space to other storage locations. Below 16 MB frames are moved above 16 MB, either above 16 MB or preferably above 2 GB. Fixed frames between 16 MB and 2 GB can only be moved into storage locations above 2 GB.

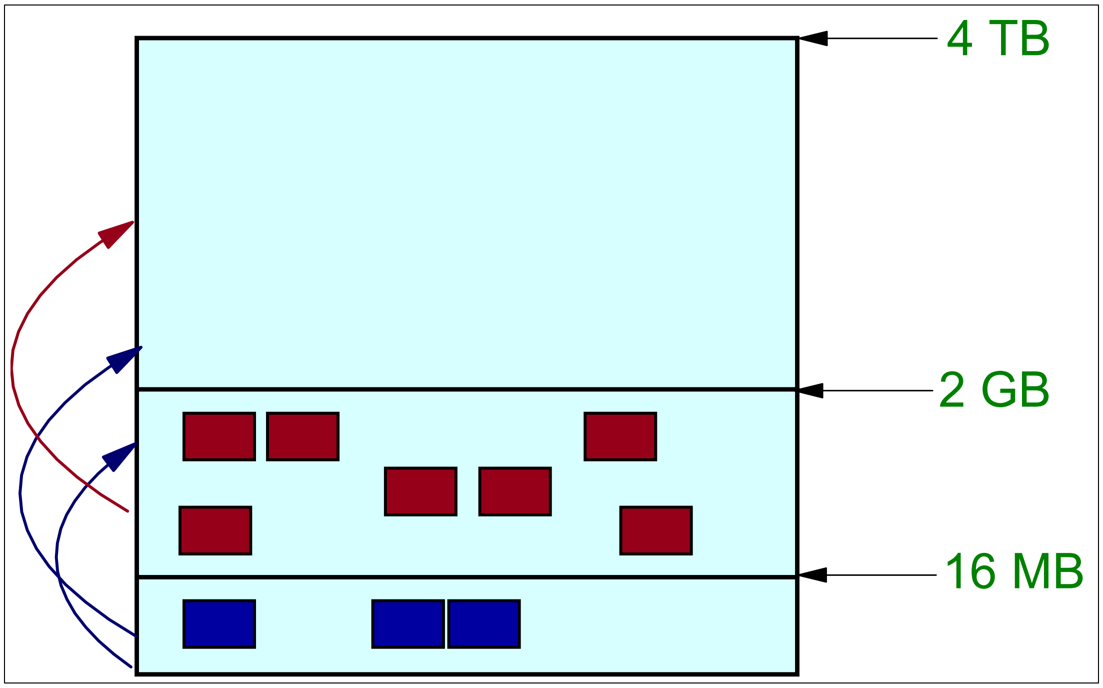


Figure 8-9 Pageable frame shortage handling in z/OS V1R8

Pageable storage shortage in the whole system

If the pageable storage shortage is above 2 GB, then there are too many fixed frames in the system, which is an indication that additional central storage is needed in order to run your workload.

When the system runs in a storage shortage, the system issues these messages:

- Message IRA400E, indicating there is a pageable storage shortage.
- Message IRA401E, indicating there is a critical pageable storage shortage; this message is issued after message IRA400E has been issued.
- Message IEA404I, listing the five largest users of fixed frames in the shortage area.

Note: It is highly recommended that you automate these messages by using an automation product.

Page frame table changes in z/OS V1R8

The page frame table size was 2 GB, which is needed to support a 128 GB central storage. The size is changed in z/OS V1R8 to support up to 4 TB. This table is now used in the page stealing algorithm. It contains one entry for every frame in the whole of central storage.

8.8 RMF report and SMF record changes

To support the changes in the amount of central storage and the changes to RSM and SRM, there are also many changes in RMF reports in z/OS V1R8. For example, RMF replaces the highest system UIC by the current system UIC in the following reports:

- Header area of any Monitor II report
- Monitor II SRCS – Central Storage/Processor/SRM report
- Monitor II SPAG – Paging Activity report
- Monitor III STORS – Storage Delay Summary report
- Monitor III STORR – Storage Resource Delays report
- Postprocessor Paging report – Central Storage section

UIC report changes

The Monitor II and Monitor III reports displaying the UIC are changed to handle a 5-digit value. The minimum UIC, maximum UIC, and current system UIC values observed during the reporting interval are saved in the paging data section of the RMF reports.

The RMF Overview Conditions report columns, AVGHUIC and MAXHUIC, are now based on the new current system UIC.

SMF changes

In SMF record type 71, the UIC value is saved in the paging data section (Paging Activity). For compatibility purposes, RMF keeps the old record fields SMF71LIC (minimum high UIC during interval), SMF71HIC (maximum high UIC during interval), and SMF71ACA (average high UIC during interval). These fields are still based on the “old” UIC value in MCVSTCRI, and have a value of 0 to 2540.

In addition, the new current system UIC fields SMF71UAC, SMF71UHC, and SMF71ULC are used as follows in RMF:

- For the minimum system UIC (MCTMinSystemUIC), Monitor I saves the lowest value observed during the interval and the average value during the interval.
- For the current system UIC (MCTCurSystemUIC), Monitor I saves the lowest and highest value observed during the interval and the average value during the interval.
- For the maximum system UIC (MCTMaxSystemUIC), Monitor I saves the highest value observed during the interval and the average value during the interval.

Table 8-2 on page 193 lists the SMF record type 71 enhancements.

Table 8-2 SMF record type 71 enhancements

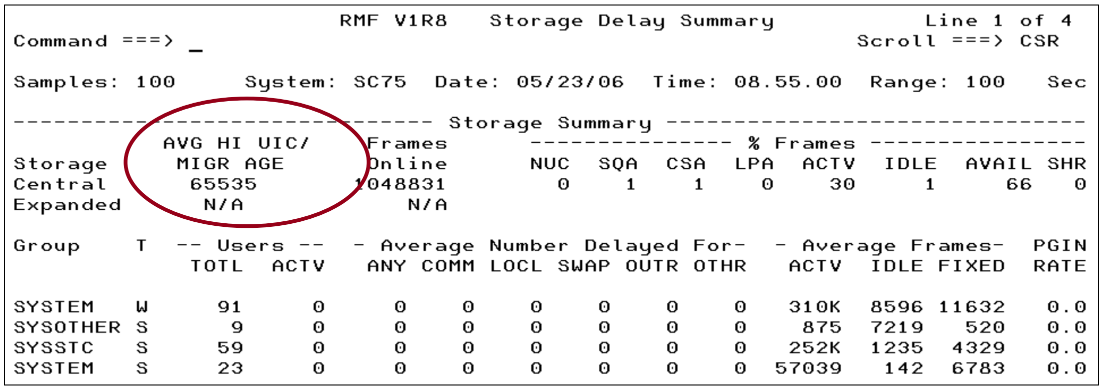
Offsets	Name	Len	Format	Description
1088 x440	SMF71ULM	4	binary	Lowest minimum system UIC during the interval (from MCTMinSystemUIC)

Offsets	Name	Len	Format	Description
1092 x444	SMF71ULC	4	binary	Lowest current system UIC during the interval (from MCTCurSystemUIC)
1096 x448	SMF71UHC	4	binary	Highest current system UIC during the interval (from MCTCurSystemUIC)
1100 x44C	SMF71UHX	4	binary	Highest maximum system UIC during the interval (from MCTMaxSystemUIC)
1104 x450	SMF71UAM	4	binary	Average minimum system UIC during the interval (from MCTMinSystemUIC)
1108 x 454	SMF71UAC	4	binary	Average current system UIC during the interval (from MCTCurSystemUIC)
1112 x458	SMF71UAX	4	binary	Average maximum system UIC during the interval (from MCTMaxSystemUIC)

SMF record type 79 subtype 3 (storage and processor data) and subtype 4 (paging activity data) is changed.

Monitor III changes

Monitor III displays UIC values up to 65535, as shown in Figure 8-10.

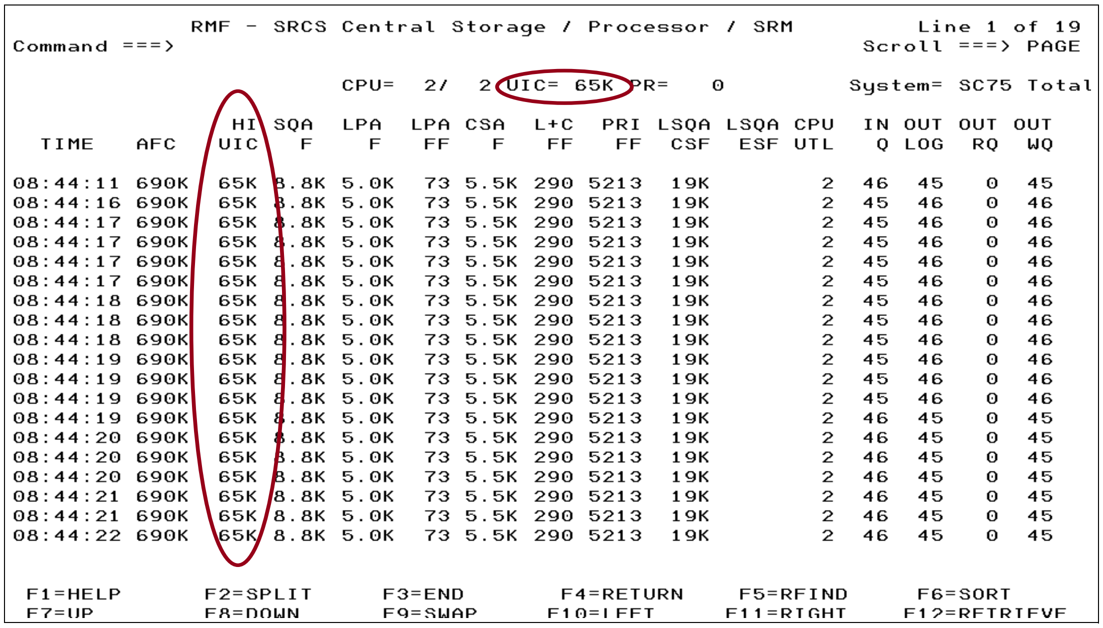


```
                          RMF V1R8   Storage Delay Summary          Line 1 of 4
Command ===> _                                                  Scroll ===> CSR

Samples: 100     System: SC75  Date: 05/23/06  Time: 08.55.00  Range: 100   Sec

------------------------------------ Storage Summary -------------------------------
           AVG HI UIC/        Frames      --------------- % Frames ----------------
Storage     MIGR AGE          Online       NUC  SQA  CSA  LPA  ACTV  IDLE  AVAIL SHR
Central      65535           1048831         0    1    1    0    30     1     66   0
Expanded      N/A                N/A

Group     T   -- Users --   - Average Number Delayed For-  - Average Frames-  PGIN
               TOTL  ACTV    ANY COMM LOCL SWAP OUTR OTHR    ACTV  IDLE FIXED  RATE

SYSTEM    W     91     0      0    0    0    0    0    0    310K  8596 11632   0.0
SYSOTHER  S      9     0      0    0    0    0    0    0     875  7219   520   0.0
SYSSTC    S     59     0      0    0    0    0    0    0    252K  1235  4329   0.0
SYSTEM    S     23     0      0    0    0    0    0    0   57039   142  6783   0.0
```

Figure 8-10 RMF Monitor III Storage Delay Summary UIC value display

Monitor II changes

Monitor II can display UIC value up to 4 digits (UIC= nnnn). Values greater than 9999 are divided by 1000, rounded, and are displayed as UIC=nnK, as shown in Figure 8-11 on page 195. The width of the HI UIC column is also limited to four digits. Any UIC values greater than 9999 are divided by 1000, rounded, and displayed with scaling character K (kilo).

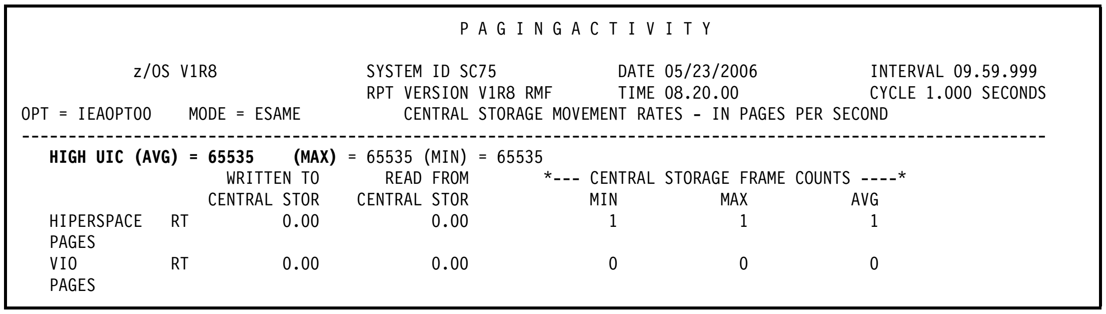


```
             RMF - SRCS Central Storage / Processor / SRM         Line 1 of 19
Command ===>                                                    Scroll ===> PAGE

                        CPU=  2/  2 UIC= 65K PR=   0          System= SC75 Total

                   HI SQA  LPA  LPA CSA  L+C  PRI LSQA LSQA CPU  IN OUT OUT OUT
  TIME    AFC     UIC   F    F   FF   F   FF   FF  CSF  ESF UTL   Q LOG  RQ  WQ

08:44:11 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45
08:44:16 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:17 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:17 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:17 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:17 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:18 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:18 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:18 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:19 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:19 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:19 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  45  46   0  46
08:44:19 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45
08:44:20 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45
08:44:20 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45
08:44:20 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45
08:44:21 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45
08:44:21 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45
08:44:22 690K  65K 8.8K 5.0K   73 5.5K 290 5213  19K           2  46  45   0  45

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=SORT
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 8-11 Monitor II UIC value display

RMF postprocessor report

The RMF postprocessor only formats the current UIC fields (MCTCurSystemUIC). The MCTMinSystemUIC and MCTMaxSsytemUIC based fields in the SMF 71 record are not formatted in the report. The highest values possible change from 2540 to 65535, as shown in Figure 8-12.



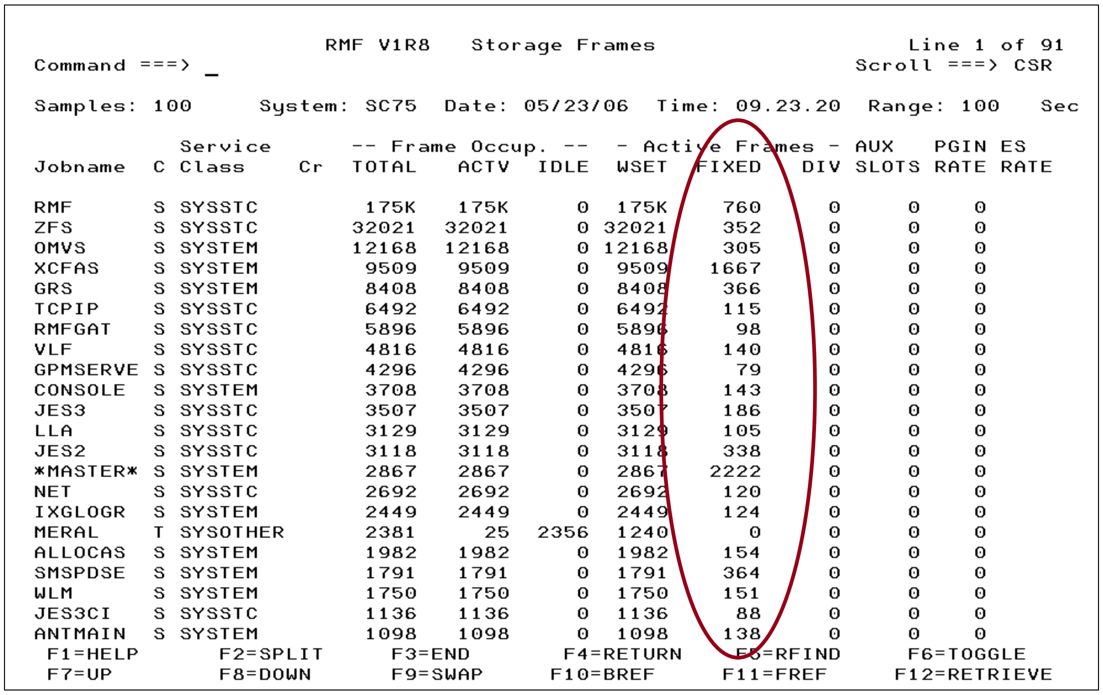
```
                                        P A G I N G A C T I V I T Y

            z/OS V1R8              SYSTEM ID SC75           DATE 05/23/2006           INTERVAL 09.59.999
                                   RPT VERSION V1R8 RMF     TIME 08.20.00             CYCLE 1.000 SECONDS
OPT = IEAOPT00    MODE = ESAME          CENTRAL STORAGE MOVEMENT RATES - IN PAGES PER SECOND
-------------------------------------------------------------------------------------------------------------
   HIGH UIC (AVG) = 65535    (MAX) = 65535 (MIN) = 65535
                     WRITTEN TO      READ FROM        *--- CENTRAL STORAGE FRAME COUNTS ----*
                   CENTRAL STOR    CENTRAL STOR             MIN           MAX           AVG
   HIPERSPACE   RT         0.00            0.00               1             1             1
   PAGES
   VIO          RT         0.00            0.00               0             0             0
   PAGES
```

Figure 8-12 RMF Postprocessor Paging Activity report showing UIC value

8.8.1 RMF report changes

You can use the RMF STORF report to determine the number of frames each address space owns that are fixed. This is storage that can never move to auxiliary anymore in z/OS V1R8. Find out which address spaces that are swapped-IN use how many fixed frames, and which address spaces were physically swapped.

If you have a swappable address space that owns a considerable amount of fixed that is being swapped out to auxiliary storage on your current system, you should consider increasing the amount of central storage; refer to Figure 8-13 on page 196 for an example.



```
                         RMF V1R8   Storage Frames                    Line 1 of 91
Command ===> _                                                        Scroll ===> CSR

Samples: 100     System: SC75  Date: 05/23/06  Time: 09.23.20  Range: 100    Sec

           Service       -- Frame Occup. --  - Active Frames - AUX   PGIN ES
Jobname  C Class    Cr  TOTAL   ACTV  IDLE  WSET  FIXED   DIV SLOTS RATE RATE

RMF      S SYSSTC        175K   175K     0  175K    760     0     0    0
ZFS      S SYSSTC       32021  32021     0 32021    352     0     0    0
OMVS     S SYSTEM       12168  12168     0 12168    305     0     0    0
XCFAS    S SYSTEM        9509   9509     0  9509   1667     0     0    0
GRS      S SYSTEM        8408   8408     0  8408    366     0     0    0
TCPIP    S SYSSTC        6492   6492     0  6492    115     0     0    0
RMFGAT   S SYSSTC        5896   5896     0  5896     98     0     0    0
VLF      S SYSSTC        4816   4816     0  4816    140     0     0    0
GPMSERVE S SYSSTC        4296   4296     0  4296     79     0     0    0
CONSOLE  S SYSTEM        3708   3708     0  3708    143     0     0    0
JES3     S SYSSTC        3507   3507     0  3507    186     0     0    0
LLA      S SYSSTC        3129   3129     0  3129    105     0     0    0
JES2     S SYSSTC        3118   3118     0  3118    338     0     0    0
*MASTER* S SYSTEM        2867   2867     0  2867   2222     0     0    0
NET      S SYSSTC        2692   2692     0  2692    120     0     0    0
IXGLOGR  S SYSTEM        2449   2449     0  2449    124     0     0    0
MERAL    T SYSOTHER      2381     25  2356  1240      0     0     0    0
ALLOCAS  S SYSTEM        1982   1982     0  1982    154     0     0    0
SMSPDSE  S SYSTEM        1791   1791     0  1791    364     0     0    0
WLM      S SYSTEM        1750   1750     0  1750    151     0     0    0
JES3CI   S SYSSTC        1136   1136     0  1136     88     0     0    0
ANTMAIN  S SYSTEM        1098   1098     0  1098    138     0     0    0
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN     F5=RFIND     F6=TOGGLE
 F7=UP        F8=DOWN      F9=SWAP     F10=BREF      F11=FREF     F12=RETRIEVE
```

Figure 8-13 RMF Monitor III STORF report that shows fixed storage usage for each address space

8.9 Page data set enhancements

A new message is introduced which informs the operator that 50% of the auxiliary storage slots are in use. This message can also be used as a trigger for automation products to add additional page data sets to the system. The message is repeated every two hours as long as more than 50% of the auxiliary storage slots are allocated.

```
IRA205I 50% AUXILIARY STORAGE ALLOCATED
```

Explanation: The system has allocated 50% of all available slots in the auxiliary storage paging space.

System action: Processing continues.

System programmer response: Consider adding additional page data sets to your system. When utilization exceeds 30%, the slot allocation algorithms become less efficient, and may degrade I/O performance. Refer to *z/OS MVS Initialization and Tuning Guide*, SA22-7591, for more information about auxiliary storage management.

8.10 Migration and coexistence

The decision to eliminate physical swaps in z/OS V1R8 system was based on the analysis of many user environments: RMF reports showed that physical swaps are a rare event in most system configurations. The operating system on pre-z/OS V1R8 systems uses physical swap mainly to resolve pageable storage shortages. With the exception of pageable storage shortages, physical swaps only occur in rare situations.

Using RMF reports

Use the RMF reports to find out how many physical swaps occur in your particular configuration. Look at the OUT Ready and OUT Wait fields, as shown in Figure 8-14. If physical swaps are a frequent event in your configuration, then you should consider increasing the amount of central storage on your system. Otherwise, you might run the risk of performance degradation or even outages.

The SMF record type 71 change is handled consistently within RMF. Additional fields are added at the end of the paging data section. The Postprocessor Paging Activity report formats the AVG, MIN, and MAX HIGH UIC values based on the new Current System UIC values SMF71UAC, SMF71ULC, and SMF71UHC instead of on the “old” UIC fields SMF71ACA, SMF71LIC, and SMF71HIC. The highest possible value changes from 2540 to 65535. If SMF 71 records from a previous release are used as input, the old UIC fields will continue to be formatted.

```
                                               C P U  A C T I V I T Y

            z/OS V1R8               SYSTEM ID SC75            DATE 05/22/2006           INTERVAL 10.00.001
                                    RPT VERSION V1R8 RMF      TIME 18.20.00             CYCLE 1.000 SECONDS
CPU  2094   MODEL  712   H/W MODEL  S18
---CPU---  ONLINE TIME   LPAR BUSY       MVS BUSY      CPU SERIAL  I/O TOTAL          % I/O INTERRUPTS
NUM  TYPE  PERCENTAGE    TIME PERC       TIME PERC     NUMBER      INTERRUPT RATE     HANDLED VIA TPI
 0   CP    100.00         2.32            2.21         11991E       7.69               0.30
 1   CP    100.00         2.36            2.25         11991E       7.62               0.26
CP   TOTAL/AVERAGE        2.34            2.23                     15.32               0.28

SYSTEM ADDRESS SPACE ANALYSIS            SAMPLES =    600
NUMBER OF ADDRESS SPACES
         -------------------- QUEUE TYPES --------------------   --------- ADDRESS SPACE TYPES ----------
                     IN      OUT       OUT    LOGICAL  LOGICAL       BATCH     STC     TSO    ASCH    OMVS
              IN     READY    READY     WAIT    OUT RDY  OUT WAIT
MIN           44        1        0        0         0        43         0      82       4       0       3
MAX           47        6        0        0         0        45         0      83       4       0       3
AVG        44.3     1.0     0.0     0.0       0.0     44.7       0.0    82.0    4.0     0.0    3.0
```

Figure 8-14 RMF CPU Activity report to analyze physical swap occurrences

Important: With the usage of pre-stealing, customers may see more page-outs than page-in values in their systems. This is done to help the page stealing algorithm to steal frames much more quickly.

9 Miscellaneous BCP changes

In this chapter we describe some small enhancements to components of the base control program (BCP), as follows:

- MVS device allocation enhancements
  - The VARY ON command for DASD and tape devices
  - Allocation device group limits
- RRS enhancements
  - Resource managers with RRS
  - Command to shut down RRS normally
  - RRS display commands

9.1 VARY ONLINE command

Currently, when a VARY ONLINE command is entered, the range of devices is processed sequentially. For the newest Enterprise Storage Server® (ESS) and tape library devices, this can mean waiting minutes to bring a new hardware device online. In z/OS V1R7, VARY OFFLINE command processing was parallelized, resulting in a great reduction in the amount of time to take devices offline.

In addition to being processed sequentially, it is synchronous to the caller. That is, the caller cannot see the results until the entire command is processed. This is different than VARY OFFLINE, which has always allowed a device to go “pending offline” and be returned to the caller as soon as the command was attempted (versus completed).

For an operator, who might be waiting to enter another VARY command, this means waiting for several minutes to enter the next command in a series.

With z/OS V1R8, devices can be made online in parallel, processing up to 32 devices at a time, while continuing to maintain the same externals required by operators. The messages issued by this process, upon which automation and vendor programs depend, are now the same as in past releases.

In pre-z/OS V1R8 systems, a VARY ONLINE command gets a SYSIEFSD.VARYDEV enqueue exclusively and does not dequeue it until all devices are online, as shown in Figure 9-1. Since it is done sequentially, no one can own this ENQ shared attribute until all devices are online.

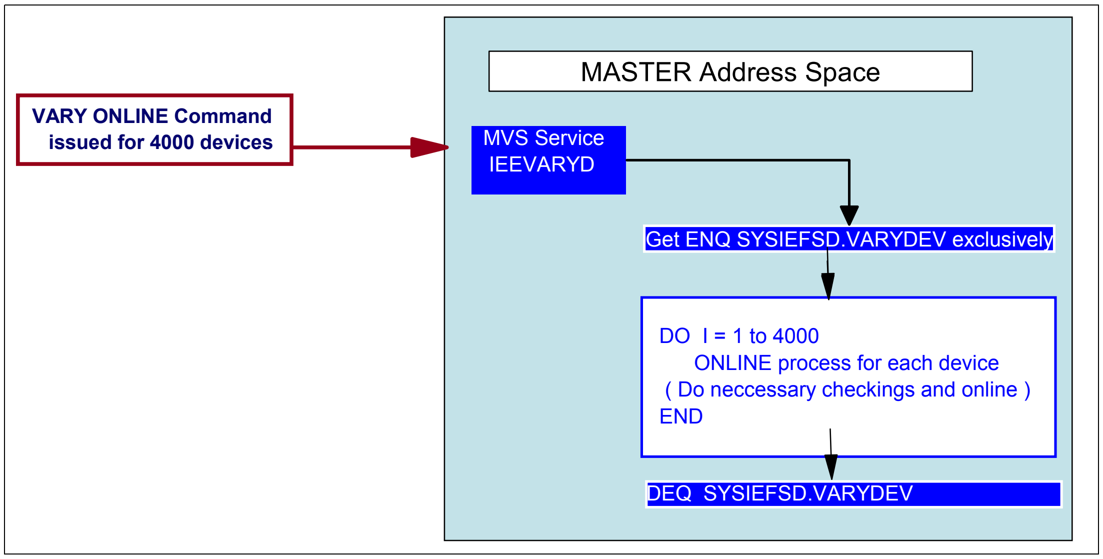

VARY ONLINE Command issued for 4000 devices
MASTER Address Space
MVS Service IEEVARYD
Get ENQ SYSIEFSD.VARYDEV exclusively
DO I = 1 to 4000
ONLINE process for each device
(Do neccessary checkings and online)
END
DEQ SYSIEFSD.VARYDEV

Figure 9-1 VARY ONLINE process pre-z/OS V1R8

9.1.1 VARY ONLINE processing with z/OS V1R8

In Figure 9-2 on page 201 a new flow for VARY ONLINE processing is shown. The SYSIEFSD.VARYDEV enqueue is not held until all devices are online as in previous releases. It is held just for the duration of necessary checking related to the online process. Then requests are sent to the ALLOCAS address space with the data gathered. The ALLOCAS main task sends each device online request to a separate task. It can send 32 requests at one time. Each task does what is shown in Figure 9-2 on page 201, that is, each task issues a SYSIEFSD.VARYDEV enqueue shared (not exclusive) and then the task gets a new

SYSIEFSD.VARYDEV_dev enqueue exclusively, where dev is the device number. This enqueue is held for each device. Then each task does online processing and sends the result back to the master address space. 32 tasks can work in parallel, each having a new exclusive enqueue.

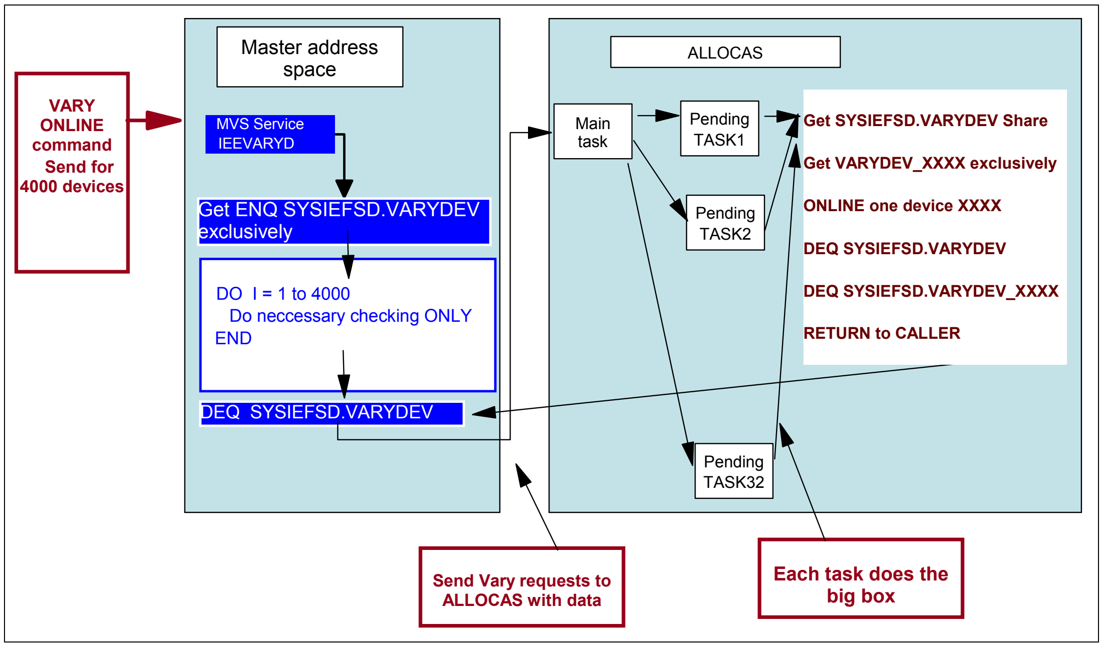

VARY ONLINE command Send for 4000 devices
Master address space
MVS Service IEEVARYD
Get ENQ SYSIEFSD.VARYDEV exclusively
DO I = 1 to 4000
Do neccessary checking ONLY
END
DEQ SYSIEFSD.VARYDEV
ALLOCAS
Main task
Pending TASK1
Pending TASK2
Pending TASK32
Get SYSIEFSD.VARYDEV Share
Get VARYDEV_XXXX exclusively
ONLINE one device XXXX
DEQ SYSIEFSD.VARYDEV
DEQ SYSIEFSD.VARYDEV_XXXX
RETURN to CALLER
Send Vary requests to ALLOCAS with data
Each task does the big box

Figure 9-2 VARY ONLINE processing in z/OS V1R8

Note: Almost all VARY ONLINE paths exploit these changes, with the exception of the IOS commands CONFIG CHP and VARY PATH. These commands utilize IOS multitasking protocols instead of the changed IEEVARYD interface.

New processing advantages

Devices now come online more quickly. In the example shown in Figure 9-3, you see the output of a VARY ONLINE command for 645 volumes in one command. This command is issued on a z/OS V1R7 system.

```
15:40:27.24 LUTZ     00000290  V (D000-D560),ONLINE
15:40:33.51 LUTZ     00000090  IEF502I UNIT D20D NOT VARIED. DUPLICATE VOLUME #
15:40:34.93          00000290  IEF196I IEF237I 8603 ALLOCATED TO SYS00136
15:40:34.94          00000290  IEF196I IEF285I   SYS1.LINKLIB
15:40:34.94          00000290  IEF196I IEF285I   VOL SER NOS= Z17RC1.
15:40:43.88 LUTZ     00000290  D U,,,D000,00646,L=LUTZ-Z     VARY RANGE DISPLAY
15:40:43.88 LUTZ     00000090  IEE457I 15.40.43 UNIT STATUS 509
                 509 00000090  UNIT TYPE STATUS        VOLSER     VOLSTATE
                 509 00000090  D000 3390 O             Z18RZ1     PRIV/RSDNT
```

Figure 9-3 VARY ON LINE command on a z/OS V1R7 system

To compare the times, the same command is entered on a z/OS V1R8 system that has access to the same volumes. Figure 9-4 on page 202 shows the output of the command.

```
11:02:04.29 LUTZ     00000290  V (D000-D560),ONLINE
11:02:08.51 LUTZ     00000090  IEF502I UNIT D20D NOT VARIED. DUPLICATE VOLUME #
11:02:15.34 LUTZ     00000290  D U,,,D000,00646,L=LUTZ-Z     VARY RANGE DISPLAY
11:02:15.34 LUTZ     00000090  IEE457I 11.02.15 UNIT STATUS 681
                 681 00000090  UNIT TYPE STATUS        VOLSER     VOLSTATE
                 681 00000090  D000 3390 O             Z18RZ1     PRIV/RSDNT
```

Figure 9-4   VARY ON command on a z/OS V1R8 system

As you can see, the time improvement is about 30% compared to earlier releases of z/OS.

Note: z/OS V1R8 can utilize IODFs built with more devices and esoterics defined. However, older levels of z/OS cannot. If you are concerned, you may want to have separate IODFs for z/OS V1R8 and previous levels. Installing APAR OA02983 will help older systems until they migrate to z/OS V1R8. With this APAR, MVS allocation has been changed to correctly handle up to 64,000 (64K) devices.

Installing APAR OA15349 will help ensure that an IODF that is usable by z/OS V1R8 is not successfully activated on a downlevel system. Over the past several years, previous releases have seen errors when customers inadvertently defined groups or devices that exceed device allocation limits.

Because of the structure changes of some internal control blocks, you should contact your software vendor if you have programs or subsystems that influence device allocation.

9.2 Allocation device group limits

For allocation devices there is an internal mechanism for grouping devices, so they can be serialized for device selection during MVS allocation. Originally, MVS allocation was written to only support 4096 devices.

Esoterics are used as a method of grouping devices for usability, and also have some limitations. These limits are now being reached by installations with many esoterics or many devices per esoteric. APAR OA02893 increased the limits compatibly but perhaps there are still some limits that could be reached. These limits, when exceeded, cause ABENDs in MVS allocation.

With z/OS V1R8, to improve the device group limits, 2-byte fields are increased to 4-byte fields. This means the address range is now increased to 2 GB. This required changes to the EDT and EDL control blocks. The SSTA, an SSI exit parameter list, also changed; this is an external interface for vendor programs.

9.2.1 IPCS and device groups

Using IPCS, it is possible to see the number of groups defined only after activating an IODF with those limits.

While device groups are internal to MVS allocation, IPCS can be used to see the numbers of groups defined. This information can be displayed with the IPCS LISTEDT HEADER command. As shown in Figure 9-5 on page 203, the number of entries in the device Group Pointer Table and the Group Section are well below the device group limit of 65535, which is FFFF in hex.

```
                         ******************************
                         *         EDT HEADER         *
                         ******************************



     Version = z/OS 01.08.00
     ID      = IEFEDT01
     Date    = 05/08/2006
     Time    = 15:06

                                                   Number of   First Entry/
                                          Offset   Entries     Entry Length
                                          -------- ---------   ------------

     Look-Up-Value Section                0FD57D90 0000001E    0001
     Generic Section                      0FD591C0 00000020
     Group Pointer Table                  0FD5EC90 00000FC4
     Group Section                        105BB658 00000E27
     Device Number Section                105C98D8 0000333C
     Group Mask Table                     1001B548 0000001E    000001C5
     Group Mask Conversion Table          0FBC64E8 00000E27    000001C5
     Preference Table                     1249A538 0000000D    0001
     Tape Max Eligible Table              105F8EC8 00000001
```

Figure 9-5 Example of output from an IPCS LISTEDT HEADER command

Note: As the fields Group Pointer Table and Group Section, shown in Figure 9-5, approach FFFF, further updates to esoterics that exceed that number may cause problems later.

9.3 RRS enhancements

A resource manager (RM) 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 events, takes actions that commit or back out changes to a resource it manages. There are three types of resource managers:

- **Data Resource Manager:** A resource manager 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 syncpoint event, a data resource manager would take actions such as committing or backing out changes to the data it manages.
- **Communications Resource Manager:** A resource manager that controls access to distributed resources and acts as an extension to the syncpoint manager. 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 syncpoint manager by allowing the local syncpoint manager to communicate with other syncpoint 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 syncpoint event, a communication resource manager communicates the event to the distributed syncpoint managers.

- **Work Manager:** 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 syncpoint event, a work manager might ensure that the application is in the correct environment to allow the syncpoint processing to continue.

9.3.1 z/OS V1R8 resource manager enhancements

The resource manager name is kept in the RRS RM data log and is never deleted. Therefore, when the name changes, a new RM is defined to RRS leaving the old RM data in the RRS log. Since there is currently no mechanism provided by RRS to delete the old RMs that are no longer needed, the old, abandoned RM names continue to exist and are returned on ATRQUERY macro invocations and RRS ISPF panels. These abandoned RM names can complicate problem determination and potentially create confusion when using the ATRQUERY and the RRS ISPF panels.

ATRSRV macro

With RRS in z/OS V1R8, an RM delete capability is added to the RRS ISPF panels and to the ATRSRV service. The delete request can be performed programmatically via new support in the ATRSRV service. The command syntax and new return and reason codes are provided to indicate successful deletion or inability to delete, similar to the panels. The new macro service parameters are:

```
REQUEST(REMOVRM)
RMNAME(xrmname)]
```

Note: The ATRQUERY macro allows callers to retrieve information about resource managers that are involved with RRS processing and retrieve information about units of recovery (URs). The ATRSRV macro allows authorized callers to remove a resource manager's interest in a UR, resolve a UR state from In-doubt to In-Commit, and resolve a UR state from In-doubt to In-Backout.

RM example

For example, WebSphere distributed z/OS 5.0 changes server identities when a base application server joins a cell. On z/OS, when WebSphere registers with RRS as a resource manager, the resource manager name is based on the server identities. WebSphere z/OS requires RRS to provide a mechanism to allow WebSphere to delete these abandoned RMs at the point in time when a new server joins a cell. Similar RM delete support should be provided via the RRS ISPF panels.

RRS ISPF panels

RRS provides Interactive System Productivity Facility (ISPF) panels to allow installation personnel, such as database administrators or system programmers, to work with RRS. In addition, you can write a batch job to take a snapshot of the contents of RRS's logs. The ATRBATCH member of SYS1.SAMPLIB contains sample JCL to do this. When you use the panels, you can view the following information:

- RRS logs
- UR information
- Resource manager information

Figure 9-6 shows a sample RM named LOGGING1_RM1E and a `d` action character to delete the RM entered on the panel.

```
                            RRS Resource Manager List          ROW 1 TO 1 OF1
Command ===>                                             Scroll ===> PAGE

Commands: v-View Details u-View URs r-Remove Interest d-Delete RM

S   RM Name                 State            System     Logging Group
d   LOGGING1_RM1E           Reset            SY1        PLEX1
```

Figure 9-6 RRS ISPF application panel to delete a resource manager name

A delete request from the ISPF panel is followed by the confirmation panel shown in Figure 9-7. You must confirm to continue with the delete.

```
                        RRS Delete RM Confirmation
Command ===>

You are requesting that this resource manager be deleted from all systems
within the logging group and the RRS Resource Manager Data log.
RM name  . . . . . : LOGGING1_RM1E
Logging Group  . : PLEX1

Press ENTER to continue, PF3 to cancel
```

Figure 9-7 RM Delete panel for confirmation of the delete

When you confirm the request and assuming the RM is valid for delete, the RM is deleted and the new message ATR529I is displayed to indicate successful deletion. The RM is removed from the list of RMs on the ISPF panel.

In some cases, the RM may not be valid for delete. If so, appropriate messages are displayed on the panel after entering the delete request, as follows:

```
ATR530I RM LOGGING1_RM1E cannot be deleted since it is still active.
ATR528I RM LOGGING1_RM1E cannot be deleted since it still has
outstanding interests in one or more URs
```

```
                         RRS Resource Manager List
Command ===>                                              Scroll ===> PAGE

 Commands: v-View Details u-View URs r-Remove Interest d-Delete RM

 S   RM Name                          State           System   Logging Group
******************************* Bottom of data *****************************

 ATR529I RM LOGGING1_RM1E was deleted successfully.
```

Figure 9-8 RM successfully deleted

Migration and coexistence considerations

Note: This support can be installed with the coexistence APAR OA15144 on all z/OS V1R4 to z/OS V1R7 systems.

9.3.2 Command to shut down RRS normally

Resource Recovery Services (RRS) is currently stopped by using a cancel command. When issued, the stopping of RRS results in one or more abends during the cancel processing. Some customers, by preference or IT policy, do not want to experience abends that the operator must understand are normal for this processing. The changes in z/OSV1R8 eliminate these abends and provide for a cleaner shutdown of RRS.

Attention: This new service is also available for z/OS V1R4 to z/OS V1R7 systems with APAR OA15144.

New command

The new command to be used for shutting down RRS cleanly is the SETRRS SHUTDOWN command. Figure 9-9 shows an example of the clean shutdown of RRS.

```
SETRRS SHUTDOWN
ATR104I SHUTDOWN REQUEST WAS RECEIVED FOR RRS.
IEF196I IEF142I RRS RRS - STEP WAS EXECUTED - COND CODE 0000
IEF196I IEF373I STEP/RRS     /START 2006130.0946
IEF196I IEF374I STEP/RRS     /STOP  2006130.0957 CPU    0MIN 00.
IEF196I SRB    0MIN 00.01SEC VIRT    36K SYS   160K EXT    3684K
IEF196I 28468K
IEF196I IEF375I  JOB/RRS     /START 2006130.0946
IEF196I IEF376I  JOB/RRS     /STOP  2006130.0957 CPU    0MIN 00.
IEF196I SRB    0MIN 00.01SEC
IEF352I ADDRESS SPACE UNAVAILABLE
ASA2960I RRS SUBSYSTEM FUNCTIONS DISABLED. COMPONENT ID=SCRRS
ATR167I RRS RESMGR PROCESSING COMPLETED.
```

Figure 9-9 New command for RRS shutdown

In order to notify RRS resource managers that RRS is terminating, all the currently active resource managers are *unset*. After the unset processing is completed, the RRS jobstep task and all of its subtasks are normally terminated to clean up the address space. In addition to the RRS infrastructure tasks, there are also timed process tasks and server tasks running in the RRS address space. These tasks are shut down normally as well.

New messages

When issued, the command is acknowledged by a new ATR104I message. If a shutdown has already been issued, the new request is rejected and a new ATR105I message is issued. If shutdown cannot complete cleanly, RRS is stopped using the existing cancel support and a new ATR106I message is issued.

```
ATR104I SHUTDOWN REQUEST WAS RECEIVED FOR RRS.

ATR105I RRS SHUTDOWN REJECTED, RRS SHUTDOWN IS ALREADY IN PROGRESS.

ATR106I AN UNEXPECTED ERROR OCCURRED DURING RRS SHUTDOWN PROCESSING. RRS CANCEL
COMMAND IS ISSUED.
```

Tip: If the SETRRS SHUTDOWN command does not work, you can use the SETRRS CANCEL command to terminate RRS.

9.3.3 RRS display commands

RRS has good support for displaying RRS information via ISPF panels. This same information cannot currently be displayed using an operator command and the output is not routed to a log for capture. This information can be especially useful in problem determination.

The display command can be used for either UR information or RM information. For both types of information, either a summary or detailed display can be requested. In many cases, you may want a summary display and then use the information from that display to get more detailed information.

The DISPLAY RRS command shown in Figure 9-10 is an extension of the existing display support and utilizes existing parameter checking and messages. New display output is provided.

```
D RRS[,UR[ ,SUMMARY|,SUM|,S|,DETAILED|,D]uroptions]
            [,RM[ ,SUMMARY|,SUM|,S|,DETAILED|,D]rmoptions]
            [,L=a|name|name-a]

uroptions:
[,URID=ur-identifier]
[,STATE=FLT|SCK|PRP|DBT|CMT|BAK|END|OLA|CMP|FGT]
[,SYSNAME=system-name]
[,GNAME=logging-group-name]

rmoptions:
[,RMNAME=resource-manager-name]
[,SYSNAME=system-name]
[,GNAME=logging-group-name]
```

Figure 9-10 Syntax of the DISPLAY RRS command

Command examples

Figure 9-11 shows a display of RM information at a summary level.

```
d rrs,rm,summary
ATR602I   15.39.21  RRS RM SUMMARY 495
RM NAME                          STATE            SYSTEM   GNAME
AONEWAY_WMNAME_SY1               Set              SY1      PLEX1
AONEWAYA_RMNAME_SY1              Run              SY1      PLEX1
BONEWAY_WMNAME_SY1               Set              SY1      PLEX1
BONEWAYB_RMNAME_SY1              Run              SY1      PLEX1
CONEWAY_WMNAME_SY1               Set              SY1      PLEX1
CONEWAYC_RMNAME_SY1              Run              SY1      PLEX1
```

Figure 9-11 RM information at the summary level

Figure 9-12 shows RM detail for a particular RM.

```
d rrs,rm,detailed,rmname=d
ATR604I   15.40.43  RRS RM DETAIL 498
RM NAME:  AONEWAY_WMNAME_SY1
RM TOKEN: 01000002022BF7000000001300000001
RM STATE: Run
SYSTEM:   SY1
GNAME:    PLEX1

URID                             SYSTEM   GNAME    ST  TYPE COMMENTS
BE7915D87E1FC2880000000001010000 SY1      PLEX1    END Prot *
```

Figure 9-12 RM detail for rmname of d

Figure 9-13 shows a display of UR information at a summary level.

```
d rrs,ur,summary
ATR601I   15.30.55  RRS UR SUMMARY 454
URID                             SYSTEM   GNAME    ST  TYPE COMMENTS
BE7915D87E1FA0000000008401010000 SY1      PLEX1    FLT Prot
BE7915D87E1FA3740000004401010000 SY1      PLEX1    SCK Prot *
BE7915D87E1FA6E80000003401010000 SY1      PLEX1    PRP Prot *X
BE7915D87E1FAA5C0000003201010000 SY1      PLEX1    OLA Prot *
BE7915D87E1FADD00000002E01010000 SY1      PLEX1    CMT Prot *X
BE7915D87E1FB1440000002601010000 SY1      PLEX1    FGT Prot X
BE7915D87E1FB4B80000002201010000 SY1      PLEX1    BAK Prot *
BE7915D87E1FB82C0000001001010000 SY1      PLEX1    DBT Prot *X
BE7915D87E1FBBA00000000A01010000 SY1      PLEX1    DBT Prot *X
BE7915D87E1FBF140000000401010000 SY1      PLEX1    DBT Prot *X
BE7915D87E1FC2880000000001010000 SY1      PLEX1    END Prot *
BE7915D87E1FC5FC0000000001010000 SY1      PLEX1    CMP Prot *
```

Figure 9-13 UR display at the summary level

Figure 9-14 on page 209 shows a display of UR information at a detail level, with a request for detail about a UR with the highlighted urid shown in the command in Figure 9-13.

```
SY1  d rrs,ur,detailed,urid=BE7915D87E1FAA5C0000003201010000
SY1  ATR603I   15.36.52  RRS UR DETAIL 485
URID: BE7915D87E1FAA5C0000003201010000
SURID:N/A
LUWID
  NETID.LUNAME: NETWORK2.LUNAME01
  TPINSTANCE  : 999999999999
  SEQNUM      : 0001
EID
  TID : 000000000002  (decimal)
  GTID: 00-0F C7E3C9C4 F0F14040 40404040 40404040 |GTID01          |
        10-1F 40404040 40404040 40404040 40404040 |                |
        20-27 40404040 40404040                   |                |
XID
  FORMATID : 003654931682 (decimal) D9D9D4E2 (hexadecimal)
  GTRID: 00-0F E2C9D4C4 C5E5F0F0 F1C9C2D4 D7D2D9D9 |SIMDEV001IBMPKRR|
         10-1F E2F1F2F4 40404040 4040BE79 15D87CD9 |S124      . .Q@R|
         20-21 5EF8                                |;8              |
  BQUAL: 00-0F D9D9D4E2 4BBE7915 D87CD963 38E2C9D4 |RRMS.. .Q@R..SIM|
         10-1F C4C5E5F0 F0F1C9C2 D4D7D2D9 D9E2F1F2 |DEV001IBMPKRRS12|
         20-26 F4404040 404040                     |4               |
SYSTEM: SY1        GNAME: PLEX1      STATE: OLA   TYPE: Prot
URI TOKEN: 7E0F420000000000000055000755555555
  RM NAME: ONEWAYB_RM2NAME_SY1
  TYPE: Prot   STATUS: DEFERRED     ROLE: Participant   STATE: OLA
ST   STATUS     DURATION
  BAK: Uncalled
  CMP: Uncalled
  CMT: Uncalled
  DBT: Uncalled
  END: Uncalled
  EFE: Uncalled
  OLA: 00000030   >00:45:43.4744
  PRP: Uncalled
  SCK:  Uncalled
SYSTEM: SY1        GNAME: PLEX1      STATE: OLA   TYPE: Prot
URI TOKEN: 7E0F420000000000000055000755555555
  RM NAME: ONEWAYB_RM2NAME_SY1
  TYPE: Prot   STATUS: DEFERRED     ROLE: Participant   STATE: OLA

  ST   STATUS     DURATION
  BAK: Uncalled
  CMP: Uncalled
  CMT: Uncalled
  DBT: Uncalled
  END: Uncalled
  EFE: Uncalled
  OLA: 00000030   >00:45:43.4744
  PRP: Uncalled
  SCK:  Uncalled
```

Figure 9-14 Display detail for a urid

10

z/OS UNIX System Services

In this chapter we introduce the new functions for z/OS UNIX System Services:

- Kernel Support for IBM 5.0 JVM
  - Overview of the feature
  - Implementation
  - Dependencies
- USS prerouter route selection support
  - USS CINET prerouter route selection
  - USS CINET prerouter route modification
- Preventing mounts during file system shutdown
  - Reasons for changes to file system shutdown
  - Functional content and benefit
  - Filtering capability added to the shutdown process
- Raising the descriptor limit
- File system display filter
- New functions in BPXBATCH
- z/OS message integration
- Recovery of BRLM
- z/OS UNIX shell and utilities
- Latch contention analysis
- New inittab support

10.1 Kernel support for IBM 5.0 JVM

The IBM 5.0 JVM version of the JVM is designed to exploit UNIX shared memory. In releases prior to z/OS V1R8, UNIX shared memory could only be exploited from environments that ran PSW Key 8. The JVM must now be able to run in the following environments:

- A WebSphere Application Server environment which runs PSW Key 2
- A CICS environment which runs PSW Key 9

z/OS UNIX support for shared memory from Key 2 and Key 9 is now provided with z/OS V1R8. Additionally, to allow for more efficient sharing in an address space such as CICS that can run IBM 5.0 JVM in multiple processes, support for enhanced single address space sharing is provided.

Note: Following are the IBM 5.0 JVM versions:

- IBM 31-bit SDK for z/OS, Java 2 Technology Edition, Version 5, product 5655-I98
- IBM 64-bit SDK for z/OS, Java 2 Technology Edition, Version 5, product 5655-I99

10.1.1 z/OS V1R8 support

With the introduction of the kernel IBM 5.0 JVM support, Java applications are able to run and exploit the latest features provided by the IBM 5.0 JVM in all z/OS supported environments, which now include critical environments such as WebSphere Application Server and CICS. The z/OS V1R8 support provides the following enhancements:

- Ability to use UNIX shared memory services for WebSphere Application Server and CICS environments
- Enhanced sharing capabilities for address spaces running multiple processes

Using kernel support for IBM 5.0 JVM, applications are able to:

- Get the benefits of the enhanced IBM 5.0 JVM in customer critical environments such as WebSphere Application Server and CICS
- Run Java applications that exploit the latest features provided by the IBM 5.0 JVM in all z/OS supported environments, such as:

  `shmgt`, `shmat`, `shmdt`, and `shmctl` from Key 9 and Key 2 environments

Coexistence support

For z/OS V1R6 and z/OS V1R7, this feature can also be exploited by installing APAR OA11519 and APAR PK05350.

Note: It is strongly recommended that z/OS APAR OA11519, available for z/OS R1V6 and onwards, is applied to any z/OS system where shared classes are used. This APAR ensures that multiple **`shmat`** requests for the same shared segment will map to the same virtual address across multiple VMs.

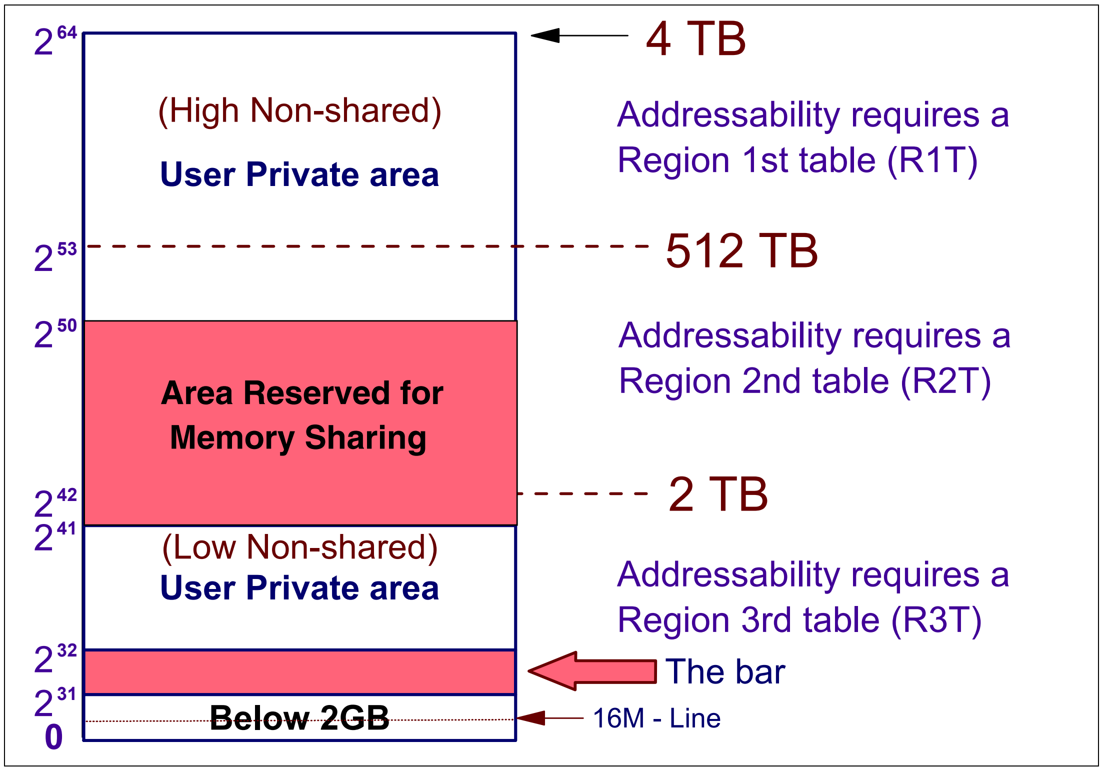
Without this APAR, there is a problem with using shared memory when multiple |VMs reside in a single address space. Each **`shmat`** call consumes a separate virtual address range. This is not acceptable because shared classes will run out of shared memory pages prematurely.

10.1.2 Implementation for IBM 5.0 JVM support

The user address space storage request for a shared memory segment, the area shown in Figure 10-1 for 64-bit callers, is normally obtained in storage key 8. In the following special circumstances, the storage key does not have to be 8:

- The caller creating a shared memory segment is running PSW Key 9 or 2 in 64 -bit AMODE and the segment is not of type IPC_MEGA and IPC_BELOWBAR.
- The caller creating a shared memory segment is running PSW Key 9 or 2 in 31-bit AMODE and the shared memory segment is of type IPC_MEGA.

Note: In these circumstances, the user address space storage is obtained in the PSW key of the caller (key 9 or key 2). It is important to note that any subsequent usage of the segment from any address space causes the user address space storage to be obtained in the key the segment was initially created in. This is true regardless of the PSW key the caller is running in at the time of a subsequent attach.



2^64
2^53
2^50
2^42
2^41
2^32
2^31
0
(High Non-shared)
User Private area
Area Reserved for Memory Sharing
(Low Non-shared)
User Private area
Below 2GB
4 TB
Addressability requires a Region 1st table (R1T)
512 TB
Addressability requires a Region 2nd table (R2T)
2 TB
Addressability requires a Region 3rd table (R3T)
The bar
16M - Line

Figure 10-1 Map of virtual address space range showing memory sharing area of storage

New shared address space option

z/OS V1R8 provides a new shared address space option on **`shmgt`** to allow for more efficient sharing of storage in multiple process address spaces.

Applications running in PSW Key 2 and PSW Key 9 can now exploit shared memory services similarly to the way they could be exploited by typical key 8 applications. For 31-bit callers, the shared memory must be of type IPC_MEGA to be sharable by Key 2 and Key 9 callers. The caller must operate against shared memory in the same key that the shared memory was created in, with the exception of Key 9 shared memory, which can be operated against by Key 8 callers, as follows:

- **shmgt()** callers running in PSW Key 2 or PSW Key 9 can now create a shared memory segment in an associated key.

  The request must be for IPC_MEGA type of shared memory segment for 31-bit AMODE callers or the caller must be running in 64 bit AMODE.
- **shmat(), shmdt(), shmctl()** calls against a segment must be from the same PSW key in which the segment was initially created.

  An exception is that PSW Key 8 callers can operate against Key 9 segments.

A new shared address space option, IPC_SHAREAS, allows for more efficient usage of storage in a single address, because the same storage segment can be shared between multiple processes in the address space, without requiring a separate segment of private storage to be allocated for each process, such as:

- **shmgt** assembler callers can now specify IPC_SHAREAS flag to allow a shared memory segment to be attached at the same address from multiple processes in the same address space.

  C/C++ callers can specify this via the __IPC_SHAREAS flag.
- **shmat** for IPC_SHAREAS segment that is already attached by another process in the address space causes "reattach" to the same storage.
- **shmdt** for IPC_SHAREAS segment that is still attached by another process in the address space causes no cleanup of the segment storage.

Note: In AMODE31, **shmat()** allows the caller to specify an input address which is the address the caller's storage will be obtained at. In AMODE64, **shmat()** only allows a value of 0 or the address already obtained in the **shmgt()** call for the input address. This is because the **shmgt()** already obtained the storage for the attach and a new address cannot be specified. If a **shmat()** is done in AMODE64, attaching to a segment that was obtained via **shmgt()** with IPC_BELOWBAR or IPC_MEGA, then the caller should still be able to specify an address on the **shmat()** call. Although the caller is running AMODE64, the storage is below the bar.

10.2 USS prerouter route selection support

The CINET support enables an installation to connect up to 32 transport drivers to z/OS UNIX. The user of the sockets library does not need to change any code to take advantage of the multiple transports connected to the kernel services.

Supporting multiple transports and providing a single AF_INET or AF_INET6 image to the user means that CINET must perform a set of management and distribution functions that govern how a socket behaves with multiple transports. A fundamental requirement for distributing work across multiple transports is the need to understand the IP configurations of each. The IP configurations are needed to determine which transport should handle a bind(), a connect(), or a sendto() to a particular Internet Protocol (IP) address. An IPv4 address is a 32-bit address defined by the Internet protocols, and an IPv6 address is a 128-bit address defined by the Internet protocols.

Each transport connected to kernel services must provide CINET with a copy of its internal IP routing table. The CINET function queries the routing tables of the transports connected to the kernel services. After the CINET prerouter function has successfully retrieved and stored routing information from a particular transport, the following message BPXF206I is issued:

```
BPXF206I ROUTING INFORMATION FOR TRANSPORT DRIVER tdname HAS BEEN
```

```
INITIALIZED OR UPDATED.
```

10.2.1 New and changed behavior

In pre-z/OS V1R8 versions, the USS CINET prerouter used to employ a hop count metric to select a route to a destination IP address, which is inconsistent with the way IBM TCP/IP selects a route to a destination IP address. However, IBM TCP/IP uses both route type and route metric when selecting the best route. With the enhanced USS prerouter route selection, you can configure your multi-stack networks with predictable results, because now USS prerouter and IBM TCP/IP use the same methodology to select the best routes.

IBM TCP/IP processes IOCTLs; therefore there is a dependency on z/OS Communications Server to support these changes. In order for route selection based on route precedence to completely work, all TCP/IP stacks connected to the prerouter must report the route type on above IOCTLs.

Route selection

If there is more than one route to a destination, the route with the best route precedence value is selected. If there is more than one route to a destination with the best route precedence value, the route with the best route precedence and the best route metric is selected.

Note: Route "type" values and route "precedence" are architected values, which are defined by IBM TCP/IP. The route "precedence values" are defined by USS prerouter.

10.2.2 USS prerouter implementation

There are two new fields to identify the route type, as follows:

- For IPv4 - Iocn_RtMsgRteType
- For IPv6 - IPV6FlgRouteType

The new fields are added to the route entries that are received by the USS prerouter via the following IOCTLs:

- SIOCGRTTABLE, SIOCGRT6TABLE, SIOCMSADDRT, SIOCMSADDRT6, SIOCMSDELRT, SIOCMSDELRT6, SIOMSMETRIC1RT, and SIOCMSCHGRT6METRIC
- In order for route selection based on route precedence to completely work, all TCP/IP stacks connected to the prerouter must report the route type on the IOCTLs.
- TCP/IP processes those IOCTLS; therefore, there is a dependency on z/OS Communications Server to support these changes.

USS prerouter

The USS prerouter assigns a route precedence value based on the route type for each route that is received via the IOCTLs. The higher the route precedence value, the better the route. If there is more than one route to a destination with the best route precedence value, the route with the best route precedence and the best route metric is selected.

Note: Route type values and route precedence are architected values that are defined by TCP/IP.

Figure 10-2 on page 216 shows route precedences, from best to worst, based on route types shown in Figure 10-3 on page 216.

```
ROUTE TYPE                           VALUE         PRECEDENCE VALUE
NON-REPLACEABLE STATIC (NO GATEWAY)    2           BEST ROUTE  127
ICMP (NO GATEWAY)                      4                       124
OSPF (NO GATEWAY)                     13                       121
RIP (NO GATEWAY)                       8                       118
OTHER (NO GATEWAY)                     1                       116
ROUTER ADVERTISED (NO GATEWAY)       129                       115
REPLACEABLE STATIC (NO GATEWAY)      130                       112
NON-REPLACEABLE STATIC (GATEWAY)       2                       109
ICMP (GATEWAY)                         4                       106
OSPF (GATEWAY)                        13                       103
RIP (GATEWAY)                          8                       100
OTHER (GATEWAY)                        1                        98
ROUTER ADVERTISED (GATEWAY)          129                        97
REPLACEABLE STATIC (GATEWAY)         130           WORST ROUTE  94
```

Figure 10-2 Route precedence based on route type

```
Non-Replaceable Static Route - STAT
ICMP Route                   -  ICMP
OSPF Route                   -  OSPF
RIP Route                    -  RIP
Router Advertised Route      -  RADV
Replaceable Static Route     -  REPL
Other Route (connection)     - CONN
No Route Type                -  N/A
```

Figure 10-3 Route types

Command to display route type information

The D OMVS,CINET command is enhanced to include the route type information for IPv4 and IPv6 as shown in Figure 10-4 on page 217.

```
D OMVS,CINET
BPXO047I 13.47.47 DISPLAY OMVS 059
OMVS     000E ACTIVE                                   OMVS=(00,FS)
INADDRANYPORT: 10000  INADDRANYCOUNT: 2000
IPV4 HOME INTERFACE INFORMATION
 TP NAME  HOME ADDRESS       FLAGS
  TCPIP     127.000.000.001    DRS
  TCPIP     010.001.040.004    DRS
  TCPIP     010.001.040.005    DRS
IPV4 HOST ROUTE INFORMATION
 TP NAME  HOST DESTINATION     TYPE     METRIC
  TCPIP     127.000.000.001    STAT          0
  TCPIP     010.001.040.004    STAT          0
  TCPIP     010.001.040.005    STAT          0
IPV4 NETWORK ROUTE INFORMATION
 TP NAME   NET DESTINATION     NET MASK           TYPE       METRIC
  TCPIP     010.001.040.000    255.255.255.000    STAT            0
IPV6 NETWORK ROUTE INFORMATION
   TP NAME  NET DESTINATION                               TYPE   METRIC
    TCPCS2   50C9:C2D4:0000:000A:0000:0000:0000:0000/064  RADV        0
    TCPCS2   50C9:C2D4:0000:0000:0000:0000:0000:0000/064  STAT        0
```

Figure 10-4 Command response for D OMVS,CINET

Prerouter messages

The prerouter only issues the message BPXF239I one time for each unknown route type reported to the prerouter by the stack, as follows:

```
BPXF239I ROUTE SELECTION BASED ON ROUTE PRECEDENCE MAY NOT BE ENABLED.
TRANSPORT DRIVER  <name> REPORTED AN UNKNOWN ROUTE TYPE: <Route Type>
```

The prerouter only issues the message BPXF238I during TD initialization, as follows:

```
 BPXF238I ROUTE SELECTION BASED ON ROUTE PRECEDENCE MAY NOT BE ENABLED.
TRANSPORT DRIVER <name> DOES NOT SUPPORT ROUTE PRECEDENCE
```

10.2.3 Migration and coexistence issues and concerns

In a mixed TCP/IP stack environment, two situations can occur:

- When there are multiple routes to the same destination, and at least one of the routes to the destination is from a TCP/IP stack that does not supply route types, then the best route will be selected by comparing route metric only.
- When a non-supporting TCP/IP stack is in the mix, but it doesn't have a route to the destination and the precedences supporting TCP/IP stacks do have a route to it, then route selection is based on precedence.

If none of the stacks support route precedence, then all route selection will be solely by metric.

10.2.4 USS CINET prerouter route modification

In the previous versions of USS CINET, the granularity does not exist to target certain sets of addresses (network or home) for change or modification, which means that a rebuild of the entire USS prerouter routing tables is necessary to change only a few routes.

With z/OS V1R8, a new route modification IOCTL is implemented to allow the TCP/IP stack to modify the routes in the USS prerouter routing tables without having to do an entire rebuild. With this enhancement, the TCP/IP stack can do the following:

- Dynamically activate the home interface and all of the routes associated with that interface.
- Dynamically deactivate the home interface and all of the routes associated with that interface

The current set of IOCTLs is used by the TCP/IP stack to inform the USS prerouter of changes made to the stack's route table, but does not provide the stack the ability to inform the prerouter that a set of routes has become active or inactive due to a change in state for its associated interface.

New IOCTL for route implementation

These two new interfaces allow the TCP/IP stack to activate or deactivate an IPv4 or IPv6 home interface, which in turn will activate or deactivate all the routes associated with that interface.

- For IPv4 - SIOCMSMODHOMEIF Ioctl
- For IPv6 - SIOCMSMODHOMEIF6 Ioctl

These two new interfaces allow the TCP/IP stack to:

- Improve performance during interface state changes
- Efficiently add and remove home interfaces and associated routes

Note: This modification for IOCTL functionality is disabled if a vendor's TCP/IP product does not support the new interface. The USS prerouter will then work as it does today without this new function.

The USS CINET prerouter's route selection algorithm is now consistent with TCP/IP's selection method, which makes it more predictable and efficient. The USS CINET prerouter's new route modification IOCTL allows the TCP/IP stack to modify the routes in the USS prerouter's routing tables dynamically without having to do an entire rebuild.

10.3 Preventing mounts during file system shutdown

Prior to z/OS V1R8, as part of a planned shutdown, you needed to prepare the file systems before shutting down z/OS UNIX by issuing one of these commands:

```
F BPXOINIT,SHUTDOWN=FILEOWNER
F BPXOINIT,SHUTDOWN=FILESYS
```

Issuing one of these commands synchronizes data to the file systems and possibly unmounts or moves ownership of the file systems. If you use SHUTDOWN=FILEOWNER, the system is disabled as a future file system owner via move or recovery operations until z/OS UNIX has been restarted.

In the previous versions of z/OS UNIX, mounts were permitted to occur during shutdown processing and after file system shutdown was completed. One problem that exists today is that there are file systems still owned by the system that is being shut down.

10.3.1 Functional changes in z/OS V1R8

z/OS V1R8 provides the following solutions for this problem:

- Automounted file systems will not be mounted during or after the file system shutdown processing. Using the following command, which limits additional file systems from being mounted on the system where the command is executed, changes to the output of the command summarize the number of file systems that are still owned by the system:

  ```
  F BPXOINIT,SHUTDOWN=fileowner

  BPXM048I  BPXOINIT FILESYSTEM SHUTDOWN INCOMPLETE.
   3 FILESYSTEM(S) ARE STILL OWNED BY THIS SYSTEM.
   2 FILESYSTEM(S) WERE MOUNTED DURING THE SHUTDOWN PROCESS.
  ```

  After the shutdown=fileowner has been issued, automount-managed file systems will no longer be mounted. File systems can only be mounted on this system by an explicit **mount** command.

  The file system display is easier to use by providing only the specific information that is needed to prepare for a file system shutdown.

- A filtering capability has been added to provide a limit to the message responses of the following command:

  ```
  D OMVS,F
  ```

Command change benefits

You can benefit from these changes to the operator commands as follows:

- Preventing unanticipated automount-managed mounts
- Displaying pertinent file system information using the D OMVS,F command and providing only the specific information to prepare for a file system shutdown, as shown in Figure 10-5 on page 220
- Limiting additional file systems from being mounted on the system where the command was executed
- Filtering the output display to a particular file system

After the shutdown=fileowner has been issued, file systems can only be mounted on this system by an explicit **mount** command.

10.3.2 New filtering capability added to the shutdown process

There are no messages issued when using the F BPXOINIT,SHUTDOWN=fileowner command if an automount was attempted. However, an appropriate error return and reason code will be issued for the mount request. Explicit mounts are still accepted during and after the file owner shutdown processing.

Behavior for the F BPXOINIT,SHUTDOWN=filesys command is not affected by this change. Automount-managed file systems can be mounted during and after the shutdown process. However, the resulting message, BPXM048I, uses the new format with the additional line that summarizes new mounts that occurred during the shutdown process.

When either shutdown commands are accepted, a time stamp is taken. If the system that is being shut down becomes the owner of a newly mounted file system or one that was acquired due to a move operation (only for shutdown=filesys) before the command completes, a console message is issued that includes the number of file systems that were acquired during this timeframe.

New command examples

z/OS V1R8 provides new filtering syntax for the D OMVS,F command, as shown in the following examples:

- D OMVS,F,name=nn or D OMVS,F,n=nn

  Where nn is the name of a file system and the output displays information about the specified file system(s). One wildcard (*) is permitted in the name specification and displays all file systems whose names match, as shown in Figure 10-5.

```
D OMVS,F,NAME=BECKER.ZFS
BPXO045I 12.51.23 DISPLAY OMVS 380
OMVS     000E ACTIVE                                  OMVS=(00,FS)
TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
ZFS            14 ACTIVE                      RDWR  05/22/2006  L=30
  NAME=BECKER.ZFS                                   12.35.23    Q=0
  PATH=/u/becker
  AGGREGATE NAME=BECKER.ZFS
```

Figure 10-5 D OMVS,F,NAME=BECKER.ZFS command

- D OMVS,F,name=* or D OMVS,F,n=*

  Using the wildcard, *, this displays all file systems, as shown in Figure 10-6.

```
D OMVS,F,NAME=*
BPXO045I 12.53.15 DISPLAY OMVS 382
OMVS     000E ACTIVE                                  OMVS=(00,FS)
TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
AUTOMNT         6 ACTIVE                      RDWR  05/19/2006  L=20
  NAME=*AMD/u                                       11.43.27    Q=0
  PATH=/u
TFS             5 ACTIVE                      RDWR  05/19/2006  L=17
  NAME=/TMP                                         11.43.25    Q=0
  PATH=/SYSTEM/tmp
  MOUNT PARM=-s 100
TFS             3 ACTIVE                      RDWR  05/19/2006  L=15
  NAME=/DEV                                         11.43.25    Q=0
  PATH=/SYSTEM/dev
  MOUNT PARM=-s 10
HFS             4 ACTIVE                      RDWR  05/19/2006  L=16
  NAME=OMVS.SC75.VAR                                11.43.25    Q=0
  PATH=/var
HFS             2 ACTIVE                      RDWR  05/19/2006  L=14
  NAME=OMVS.SC75.ETC                                11.43.25    Q=0
  PATH=/etc
```

Figure 10-6 D OMVS,F,NAME= command*

- D OMVS,F,name=*.ETC would display file systems, as shown in Figure 10-7 on page 221.

```
D OMVS,F,N=*.ETC
BPXO045I 13.04.42 DISPLAY OMVS 399
OMVS     000E ACTIVE                                 OMVS=(00,FS)
TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
HFS             2 ACTIVE                      RDWR  05/19/2006  L=14
  NAME=OMVS.SC75.ETC                                11.43.25    Q=0
  PATH=/etc
```

Figure 10-7   D OMVS,F,N=*.ETC command

Displays all file systems a system owns

This command displays all file systems that are owned by the system name where the command is entered:

```
D OMVS,F,OWNER or D OMVS,F,O
```

Display all file systems that are owned by the specified system:

```
D OMVS,F,OWNER=sc75 or D OMVS,F,O=sc75
```

```
D OMVS,F,O=SC75
BPXO045I 12.57.04 DISPLAY OMVS 388
OMVS     000E ACTIVE                                 OMVS=(00,FS)
TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
AUTOMNT         6 ACTIVE                      RDWR  05/19/2006  L=20
  NAME=*AMD/u                                       11.43.27    Q=0
  PATH=/u
TFS             5 ACTIVE                      RDWR  05/19/2006  L=17
  NAME=/TMP                                         11.43.25    Q=0
  PATH=/SYSTEM/tmp
  MOUNT PARM=-s 100
TFS             3 ACTIVE                      RDWR  05/19/2006  L=15
  NAME=/DEV                                         11.43.25    Q=0
  PATH=/SYSTEM/dev
  MOUNT PARM=-s 10
HFS             4 ACTIVE                      RDWR  05/19/2006  L=16
  NAME=OMVS.SC75.VAR                                11.43.25    Q=0
  PATH=/var
HFS             2 ACTIVE                      RDWR  05/19/2006  L=14
  NAME=OMVS.SC75.ETC                                11.43.25    Q=0
  PATH=/etc
```

Figure 10-8   D OMVS,F,O=SC75 command

Displays all file systems in an exception state

This command displays all file systems in an exception state, for example, quiesced, unowned, or in recovery:

```
D OMVS,F,exception or D OMVS,F,e
```

Useful for the daily business is also the D OMVS,F,E command to see any exceptions in the file systems. Figure 10-9 on page 222 shows the display of an exception in the file systems on the particular system. We tried to unmount a file system that still has a file system mounted on it.

```
D OMVS,F,E
BPXO045I 11.22.25 DISPLAY OMVS 832
OMVS     000E ACTIVE                                 OMVS=(00,FS)
TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
ZFS            9 DRAIN UNMOUNT                RDWR  05/15/2006  L=25
  NAME=LUTZ.ZFS                                     10.52.37    Q=0
  PATH=/u/lutz
  AGGREGATE NAME=LUTZ.ZFS
```

Figure 10-9 Example of the D OMVS,F,E command

Display file system types

Display all file systems of the type xx, where xx is a PFS type (either HFS or ZFS):

```
D OMVS,F,type=xx or D OMVS,F,t=xx
```

```
D OMVS,F,T=HFS
BPXO045I 13.01.48 DISPLAY OMVS 395
OMVS     000E ACTIVE                                 OMVS=(00,FS)
TYPENAME   DEVICE ----------STATUS----------- MODE  MOUNTED    LATCHES
HFS            4 ACTIVE                       RDWR  05/19/2006  L=16
  NAME=OMVS.SC75.VAR                                11.43.25    Q=0
  PATH=/var
HFS            2 ACTIVE                       RDWR  05/19/2006  L=14
  NAME=OMVS.SC75.ETC                                11.43.25    Q=0
  PATH=/etc
HFS            1 ACTIVE                       RDWR  05/19/2006  L=13
  NAME=OMVS.ZOSR18.Z18RA1.ROOT                      11.43.24    Q=0
  PATH=/
```

Figure 10-10 D OMVS,F,T=HFS command

The format resulting from the output of these new filtering commands has not changed from the current display. The output will still be the BPXO045I display contents, only the amount of the display shown will be dependant on the filter option that was used.

10.4 Raising the descriptor limit

The file descriptor limit is the maximum number of files per process for a z/OS UNIX user. In some z/OS UNIX environments with a very large number of clients, the file descriptor limits are too small.

In z/OS V1R6, the limit was increased from 64 KB up to 128 KB. This limit was increased again in z/OS V1R8 to 512 KB. Also, the performance was improved in this release.

Use MAXFILEPROC to set the maximum number of file descriptors that a single process can have open concurrently, such as all open files, directories, sockets, and pipes. By limiting the number of open files that a process can have, you limit the amount of system resources a single process can use at one time. The limit that was increased is the limit per user, not the system as a whole. The system itself has no limit.

There are some ways to define and modify this limit in your system. The default is set in the BPXPRMxx parmlib member with MAXFILEPROC parameter. In Figure 10-11 you see a sample entry from the BPXPRMxx parmlib member.

New limits

The default process descriptor limit is set at IPL or on an OMVS restart with the MAXFILEPROC() keyword of the BPXPRMxx parmlib member. New limits are as follows:

- The default in z/OS V1R8 for MAXFILEPROC is now 64000.
- The new limit is 524287.

```
MAXFILEPROC(300000)            /* Allow at most 300000 open files per user
```

Figure 10-11 Sample of MAXFILEPROC parameter

Modifying the MAXFILEPROC limit

There are three ways to modify the system limit. The first two choices will dynamically activate a whole BPXPRMxx parmlib member in your system, which means all statements you place into a BPXPRMxx member specified by the xx in the command will be activated.

1. Use the SET OMVS=xx command with BPXPRMxx MAXFILEPROC(nnnnnn).
2. Use the SETOMVS RESET=(xx) command with BPXPRMxx MAXFILEPROC(nnnnnn).
3. Use the SETOMVS MAXFILEPROC=nnnnnn command.

The third choice just modifies the MAXFILEPROC limit in your system and nothing else. In Figure 10-12 you see the successful activation of a new descriptor limit.

```
 SETOMVS MAXFILEPROC=10000
 BPXO015I THE SETOMVS COMMAND WAS SUCCESSFUL.
```

Figure 10-12 Console command to modify MAXFILEPROC

When you enter a wrong value to modify the descriptor limit, you get the error message shown in Figure 10-13.

```
SETOMVS MAXFILEPROC=1000000
BPXO006I ERROR IN SETOMVS COMMAND. THE MAXFILEPROC 059
PARAMETER VALUE IS OUT OF THE ALLOWED RANGE OF 3 TO 524287.
BPXO012I ERRORS OCCURRED IN THE PROCESSING OF THE 060
SETOMVS COMMAND; NO VALUES WERE SET.
```

Figure 10-13 Wrong size to modify MAXFILEPROC

Set a limit for a process

If you want to change the descriptor limit for an individual process, this can be done with the SETOMVS system command. Figure 10-14 shows the z/OS system command to modify the descriptor limit for a selected process by PID.

```
SETOMVS PID=33555255,MAXFILEPROC=3000
BPXO015I THE SETOMVS COMMAND WAS SUCCESSFUL.
```

Figure 10-14 Modify MAXFILEPROC with the SETOMVS command

Set file descriptor maximum for a single user

These methods modify the system limit:

1. Use the FILEPROCMAX parameter in the user's OMVS segment.
2. Use scalable services or system APIs.
3. Use the SETOMVS system command.

You can set the limit for individual users by modifying their RACF OMVS segment. The parameter that you can change calls different from the parmlib. Figure 10-15 shows the RACF command to modify the descriptor limit for individual users. In this example we increase the limit for the user ID LUTZ to 10000 open objects.

```
ALU LUTZ OMVS(FILEPROCMAX(10000))
```

Figure 10-15 Sample command for modifying the limit

Figure 10-16 shows the modified OMVS segment for user ID LUTZ.

```
LU LUTZ NORACF OMVS
OMVS INFORMATION
----------------
UID= 0000001014
HOME= /u/lutz
PROGRAM= /bin/sh
CPUTIMEMAX= NONE
ASSIZEMAX= NONE
FILEPROCMAX= 00010000
PROCUSERMAX= NONE
THREADSMAX= NONE
MMAPAREAMAX= NONE
```

Figure 10-16 Sample listing of the OMVS segment

10.4.1 Using APIs to modify FILEPROCMAX

You also have the possibility, if you are permitted, to modify the descriptor limit by using system APIs. Figure 10-17 on page 225 shows a sample REXX procedure to modify your own descriptor limits.

```
/*REXX==========================================================*/
/*                                                              */
/* title  : use of the setrlimit callable service               */
/*                                                              */
/* author : ITSO                                                */
/*                                                              */
/* purpose: for use in Redbook SG247265 z/OS V1R8 Implementation */
/*                                                              */
/*==============================================================*/
 if syscalls('ON') > 3 then exit(12)

 smfp.2=5000
 smfp.1=1000

 address syscall 'setrlimit' RLIMIT_NOFILE smfp.

 address syscall 'getrlimit' RLIMIT_NOFILE gmfp.

 say 'maximum open file desciptor are:' gmfp.2
 say 'current open file desciptor are:' gmfp.1

exit
```

Figure 10-17 Sample of using REXX callable services

10.5 New functions in BPXBATCH

BPXBATCH is an MVS utility that you can use to run shell commands or shell scripts and to run executable files through the MVS batch environment. You can invoke BPXBATCH from a JCL job or from TSO/E (as a command, through a CALL command, or from a CLIST or REXX EXEC).

BPXBATCH has logic in it to detect when it is run from JCL. If the BPXBATCH program is running as the only program on the job step task level, it sets up STDIN, STDOUT, and STDERR and executes the requested program. If BPXBATCH is not running as the only program at the job step task level, the requested program will run as the second step of a JES batch address space from JCL in batch. If BPXBATCH is run from any other environment, the requested program will run in a WLM initiator in the OMVS subsys category.

BPXBATCH limitations

Currently, limitations on the input and output capabilities of BPXBATCH jobs make it difficult to use and maintain. For example, allowing only 100 characters of input parameter data to be specified makes it difficult to invoke some z/OS UNIX applications that require long pathnames or input data. Dealing with output from the jobs in z/OS UNIX files requires special processing to be done that is not required for normal batch jobs.

BPXBATCH enhancements in z/OS V1R8

These problems are solved by enhancing BPXBATCH to support greater parameter input data and to allow output to be directed to traditional MVS data sets.

In z/OS V1R8 you are now able to:

- Pass up to 65,536 characters of parameter data supplied as input to BPXBATCH

- Allow both STDOUT and STDERR to specify traditional MVS data sets to be the target of the output of the BPXBATCH jobs

10.5.1 BPXBATCH increase of parameter data

Currently, BPXBATCH supports only up to 100 characters of parameter data when invoked from JCL and 500 characters when invoked from TSO. With z/OS V1R8, BPXBATCH now supports the following:

- Up to 65,536 characters of parameter data

  This is supported via a new STDPARM DD that points to an MVS data set or UNIX file that contains the parameter data. The MVS data sets that can be used include SYSIN, sequential, or PDS/PDSE members. Specification of the STDPARM DD overrides the usage of PARM= on the EXEC PGM= JCL statement.
- For the BPXBATCH TSO command, up to 32,754 characters are supported in the parameter string specified on the command invocation.

Note: The support is also available for z/OS V1R5, V1R6, and V1R7 systems that have installed the PTF for APAR OA11699.

10.5.2 MVS data sets for STDERR and STDOUT

The STDOUT and STDERR DDs must currently be represented by z/OS UNIX files. This limitation is changed so that these DDs can be represented by MVS data sets.

Previously, STDENV could only be represented by a SYSIN, sequential, or PDS data set. This limitation is now changed to allow PDSEs.

Implementation rules

If you define an MVS data set for STDOUT or STDERR, it must follow the following rules:

- It must be a basic format sequential data set, a partitioned data set (PDS) member, a partitioned data set extended (PDSE) member, or SYSOUT.

  Extended format and large format sequential data sets are not supported.
- The data set must have a non-zero logical record length (LRECL) and a defined record format (RECFM); otherwise, BPXBATCH fails with an error message BPXM012I indicating an open failure for the affected ddname.
- If the LRECL of the target STDOUT or STDERR data set is not large enough to hold a line of output, the data is truncated and message BPXM080I is issued. This can happen for both fixed and variable blocked data sets. For variable blocked data sets, the first four bytes of each record, record segment, or block make up a descriptor word containing control information. You must allow for these additional 4 bytes in the specified LRECL if you intend to avoid truncation of the output to the STDOUT and STDERR DDs.
- If you use two members of the same partitioned data set for the STDOUT and STDERR ddnames, then you must use a PDSE (not a PDS). Using a PDS instead of a PDSE can result in a 213 ABEND (and, if running in a batch job, an abnormal end for the job step) or the output does not appear in the members as expected.
- When you specify an MVS data set for either the STDOUT or STDERR DDnames, a child process is created to run the target z/OS UNIX program. In some cases, the child process runs in a separate address space from the BPXBATCH job. In such cases, the job log messages for the child do not appear in the job log of the BPXBATCH job. To capture the child's job log messages, set the _BPXK_JOBLOG=STDERR environment variable. This

will cause the child’s job log messages to be written to the STDERR data set specified in the BPXBATCH job.

10.5.3 Implementing BPXBATCH enhancements

In the following examples with the BPXBATCH enhancements, you can allocate the MVS standard files STDOUT, STDERR, or STDENV as MVS data sets or z/OS UNIX text files. The STDERR and STDOUT files for saving job output can be allocated as SYSOUT, PDSE, PDS, or sequential data sets. If you do not allocate them, STDOUT and STDERR default to /dev/null.

Allocating STDPARM with JCL

The parameter data resides in the z/OS UNIX file /u/bpx/abc.parms:

```
//STDPARM DD PATH=‘/u/bpx/abc.parms’,PATHOPTS=ORDONLY
```

The BPXBATCH parameter data resides in member BPXPARM of the MVS PDSE BPX.PARM.LIBRARY:

```
//STDPARM DD DSN=BPX.PARM.LIBRARY(BPXPARM),DISP=SHR
```

Allocating STDPARM with TSO

In this example, STDPARM is allocated to the data set BPX.ABC.PARMS with DISP set to share:

```
ALLOCATE DDNAME(STDPARM) DSN(‘BPX.ABC.PARMS’) SHR
```

The contents for STDPARM might look something like this:

```
SH echo “Hello, world”
```

Using STDOUT and STDERR with TSO

Here are two separate TSO allocation commands that specify STDOUT and STDERR, respectively, to the same new data set BPX.MYOUTPUT:

```
ALLOCATE DDNAME(STDOUT) DSNAME(‘BPX.MYOUTPUT’) VOLUME(volser) DSORG(PS)
SPACE(10) TRACKS RECFM(F,B) LRECL(512) NEW KEEP

ALLOCATE DDNAME(STDERR) DSNAME(‘BPX.MYOUTPUT’) VOLUME(volser) DSORG(PS)
SPACE(10) TRACKS RECFM(F,B) LRECL(512) NEW KEEP
```

Using STDOUT and STDERR with JCL

The following example allocates member STD of a new PDSE BPX.MYOUTPUT.LIBRARY to STDOUT and directs STDERR output to SYSOUT:

```
//STDOUT DD DSNAME=TURBO.MYOUTPUT.LIBRARY(STD),
//  DISP(NEW,KEEP),SPACE=(TRK(5,1,1)),
//  UNIT=3390,VOL=SER=volser,RECFM=FB,LRECL=80
//STDERR DD SYSOUT=*
```

Figure 10-18 on page 228 shows STDOUT and STDERR going to SYSOUT, and also allocations with already defined data sets.

```
//LUTZBPX JOB (ITSO,TXX,T870270),LUTZ,MSGCLASS=X,
//            MSGLEVEL=(1,1),
//            NOTIFY=&SYSUID,CLASS=D
//*=============================================================
//* STDOUT and STDERR pointing to SYSOUT
//*=============================================================
//BPXBATCH EXEC PGM=BPXBATCH,PARM='SH ls -la /'
//STDOUT   DD   SYSOUT=D,DCB=(LRECL=133,RECFM=F,DSORG=PS)
//STDERR   DD   SYSOUT=D,DCB=(LRECL=133,RECFM=F,DSORG=PS)
//*=============================================================
//* STDOUT and STDERR pointing to conventional MVS data sets
//*=============================================================
//BPXBATCH EXEC PGM=BPXBATCH,PARM='SH ls -la /'
//STDOUT   DD   DISP=SHR,DSN=LUTZ.STDOUT.LIST
//STDERR   DD   DISP=SHR,DSN=LUTZ.STDERR.LIST
//
```

Figure 10-18 Sample BPXBATCH with MVS data sets

10.5.4 New authorized interfaces

Two new APF-authorized interfaces are introduced to allow the invocation of APF-authorized z/OS UNIX programs in the same address space as the original job.

Prior support for BPXBATCH

BPXBATCH has logic in it to detect when it is running from JCL. By default, BPXBATCH sets up the stdin, stdout, and stderr file descriptors and then calls the exec callable service to run the requested program. The exec service ends the current job step and creates a new job step to run the target program. Therefore, the target program does not run in the same job step as the BPXBATCH program; it runs in the new job step created by the exec service. BPXBATCH follows this default behavior and uses the exec service to run the target program only when all of the following are true:

- The BPXBATCH program is the only program running on the job step task level.
- The _BPX_BATCH_SPAWN=YES environment variable is not specified.
- The STDOUT and STDERR ddnames are not allocated as MVS data sets.

If BPXBATCH does not follow the default behavior, the target program runs either in the same job step as the BPXBATCH program or in a WLM initiator in the OMVS subsys category. Where the target program runs depends on the environment variable settings specified in the STDENV file and on the attributes of the target program.

z/OS V1R8 enhancements

BPXBATA2 and BPXBATA8 are provided as APF-authorized alternatives to BPXBATSL. BPXBATA2 and BPXBATA8 provide the capability for a target APF-authorized z/OS UNIX program to run in the same address space as the originating job, allowing it to share the same allocations and the joblog. BPXBATA2 is specifically intended to provide the capability for a PSW Key 2 APF-authorized z/OS UNIX program to be started. To insure the target program receives control with PSW Key 2, a PPT entry for BPXBATA2 must be set up that specifies that BPXBATA2 starts up in PSW Key 2. The same restrictions that apply to BPXBATSL apply to BPXBATA2 and BPXBATA8, in addition to the following:

- The PGM keyword is the only invocation type supported. The SH keyword is not supported.
- The interfaces can only be used from started task address spaces.

- The z/OS UNIX program that is the target of the BPXBATA2 and BPXBATA8 job must be marked as an APF-authorized z/OS UNIX executable file.

10.6 File format support for extattr

Previous to this release, there are no supported commands to set the file format from the OMVS shell. The **`extattr`** command is enhanced to accept a **`–F`** option with values consistent with the **`cp`** command to indicate the format of the file.

The new syntax is shown in Figure 10-19. The file format flag is used by the **`cp`** command.

```
extattr [+alps] [-alps] [-F] file
```

Figure 10-19 New syntax for the extattr command

Specifying the file format option

The format option can be specified in lowercase, uppercase, or in mixed case. The format option can also be specified with a space or no space after the file format flag (-F). For example, specifying -F with line feed (LF) and carriage return (cr):

```
extattr -FLFcr filename
```

The file format flag (-F) can be used with other extattr flags (+alps/-alps), but it must be separated by a space or tab. For example:

```
extattr +aps -F BIN filename           (is a valid entry)
extattr -apsF NA filename              (is not a valid entry)
```

So the new option -F can have the values shown in Table 10-1.

Table 10-1 Possible -F options for extattr

Value	Format options
NA	Not specified
BIN	Binary data file
Value	**Text data delimeters**
NL	New line
CR	Carriage return
LF	Line feed
CRLF	Carriage Return followed by Line Feed
LFCR	Line Feed followed by Carriage Return
CRNL	Carriage Return followed by New Line

Note: The **`extattr`** command only sets the file format for the file. The display for the file format is not added to the **`extattr`** command. The **`ls–H`** command already has the ability to display the file format.

The setting of a file format flag has no impact on the contents of the file. You can easily set it as shown in Figure 10-20. In this example, the file extattr.test is set as a text file that contains carriage returns and file feed, which is also known as a regular text file.

```
extattr -F CRLF extattr.test
```

Figure 10-20 Setting the file format flag

Figure 10-21 shows the output of the **ls** command using the **-H** option. This option was introduced in prior releases of z/OS and allows you to see this new file format flag.

```
ROGERS @ SC75:/u/rogers>extattr -F CRLF maxcore
ROGERS @ SC75:/u/rogers>ls -H
total 160
-rwxr-xr-x  crlf   1 SYSPROG  SYS1       73728 Apr 21 16:08 maxcore
ROGERS @ SC75:/u/rogers>
```

Figure 10-21 Sample output of ls -H

The output of the command in Figure 10-22 now shows two sets of **extattr** values for files, with the value s showing the shared library value.

```
ROGERS @ SC75:/u/rogers>ls -alHE
total 184
drwxr-x---             7 SYSPROG  SYS1        544 Sep 13 14:15 .
dr-xr-xr-x             7 SYSPROG  TTY           0 Sep 13 13:54 ..
-rw-------  --s- ----  1 SYSPROG  SYS1        638 Sep 15 13:35 .sh_history
drwxr-xr-x             2 SYSPROG  SYS1        256 Sep 13 14:15 00000100
drwxr-xr-x             5 SYSPROG  OMVSGRP    1120 Jul 26 21:57 ITSO-link
drwxr-xr-x             2 SYSPROG  SYS1        256 Sep 13 14:06 db2
drwxr-xr-x             2 SYSPROG  SYS1        256 Sep 13 14:05 echo
drwxr-xr-x             2 SYSPROG  SYS1        256 Sep 13 14:15 ims
-rwx------  --s- ----  1 SYSPROG  SYS1          0 Aug 28 09:31 inetd-stderr
-rwx------  --s- ----  1 SYSPROG  SYS1          0 Aug 28 09:31 inetd-stdout
-rwxr-xr-x  --s- crlf  1 SYSPROG  SYS1      73728 Apr 21 16:08 maxcore
```

Figure 10-22 Sample output showing the -HE values

The -s option of the extattr command

To have the c89 and TSO utilities not run in an address space shared with other processes, issue the following command:

```
extattr -s /bin/c89 /bin/tso
```

Restriction: The **extattr** command ignores the file format setting for directories and symlinks. It only sets for regular files.

Implementing file format support

Setting the file format flag on a file does not modify the data in the file. These format options can be specified in lower case, upper case, or in mixed cases. These format option can be specified with space or no space in-between file format flag(-F).

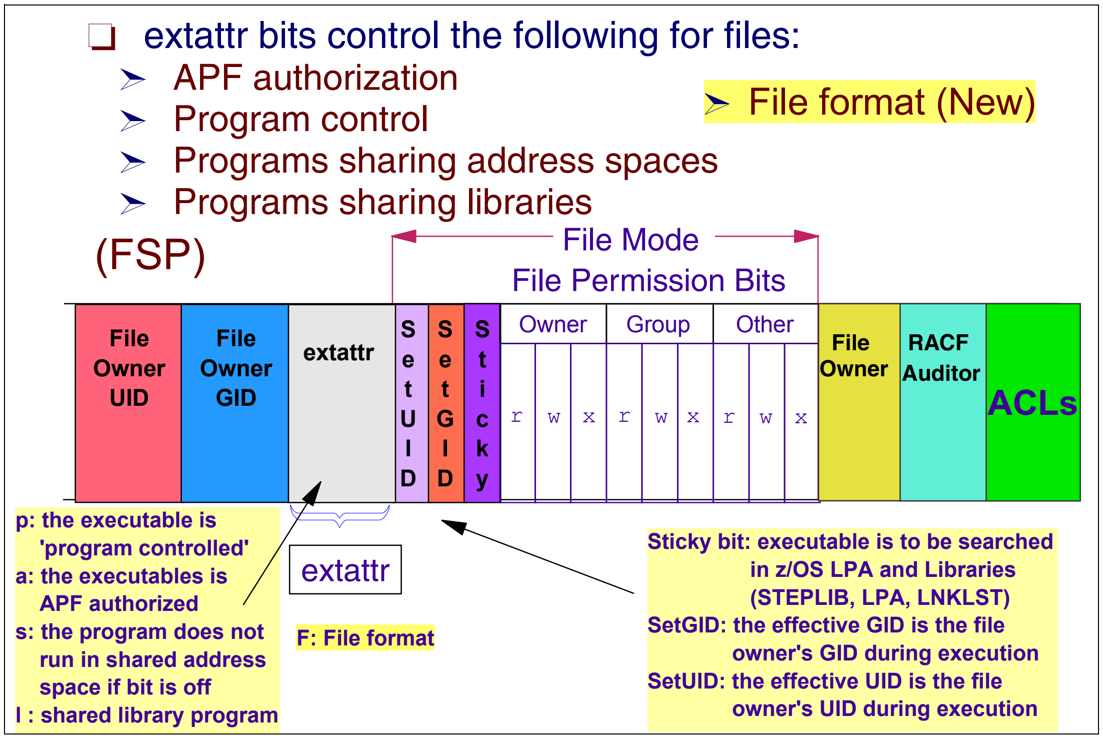
The file format flag (-F) can be used with the other **extattr** flags, (+apsl or -apsl), but needs to be separated by a space or tab.

File security packet (FSP)

The FSP is created by an SAF call from the physical file system (PFS) when a file is created. Some of the information is taken from the current security environment, and some of it is passed as parameters.

When a security decision is needed, the file system calls RACF and supplies the FSP (and ACL, if one exists). RACF makes the decision, does any auditing, and returns control to the file system. RACF does not provide commands to maintain the FSP (and ACL), but it does provide SAF services that do the FSP (and ACL) maintenance. z/OS UNIX provides commands that invoke these SAF services.

Figure 10-23 shows all the fields of the FSP, which now include the file format.



extattr bits control the following for files:
APF authorization
Program control
Programs sharing address spaces
Programs sharing libraries
File format (New)
(FSP)
File Mode
File Permission Bits
File Owner UID
File Owner GID
extattr
SetUID
SetGID
Sticky
Owner
Group
Other
r w x r w x r w x
File Owner
RACF Auditor
ACLs
p: the executable is 'program controlled'
a: the executables is APF authorized
s: the program does not run in shared address space if bit is off
l : shared library program
extattr
F: File format
Sticky bit: executable is to be searched in z/OS LPA and Libraries (STEPLIB, LPA, LNKLST)
SetGID: the effective GID is the file owner's GID during execution
SetUID: the effective UID is the file owner's UID during execution

Figure 10-23 Description of the format of the FSP

10.7 z/OS message integration

In the current z/OS releases, the syslog daemon (syslogd) is a server process that must be started as one of the first processes in your z/OS UNIX environment. Commications Server server applications and components use syslogd for logging purposes and can also send trace information to syslogd. Servers on the local system use AF_UNIX sockets to communicate with syslogd; remote servers use the AF_INET socket.

The syslogd daemon reads and logs system messages to the MVS console, log files, other machines, or users, as specified by the configuration file /etc/syslog.conf. A sample is provided in /usr/lpp/tcpip/samples/syslog.conf.

If the syslog daemon is not started, the application log may appear on the MVS console. The syslog daemon must have a user ID; for example, SYSLOGD defined in RACF with UID=0. The syslogd daemon uses the following files:

/dev/console	Operator console

/etc/syslog.pid	Location of the process ID
/etc/syslog.conf	Default configuration file
/dev/log	Default log path for z/OS UNIX datagram socket
/usr/sbin/syslogd	Syslog server

Syslogd can only be started by a superuser. It can be terminated using the SIGTERM signal. If you want syslogd to receive log data from or send log data to remote syslogd servers, reserve UDP port 514 for the syslogd job in your PROFILE.TCP/IP data set and enter the syslog service for UDP port 514 in the services file or data set (for example, /etc/services).

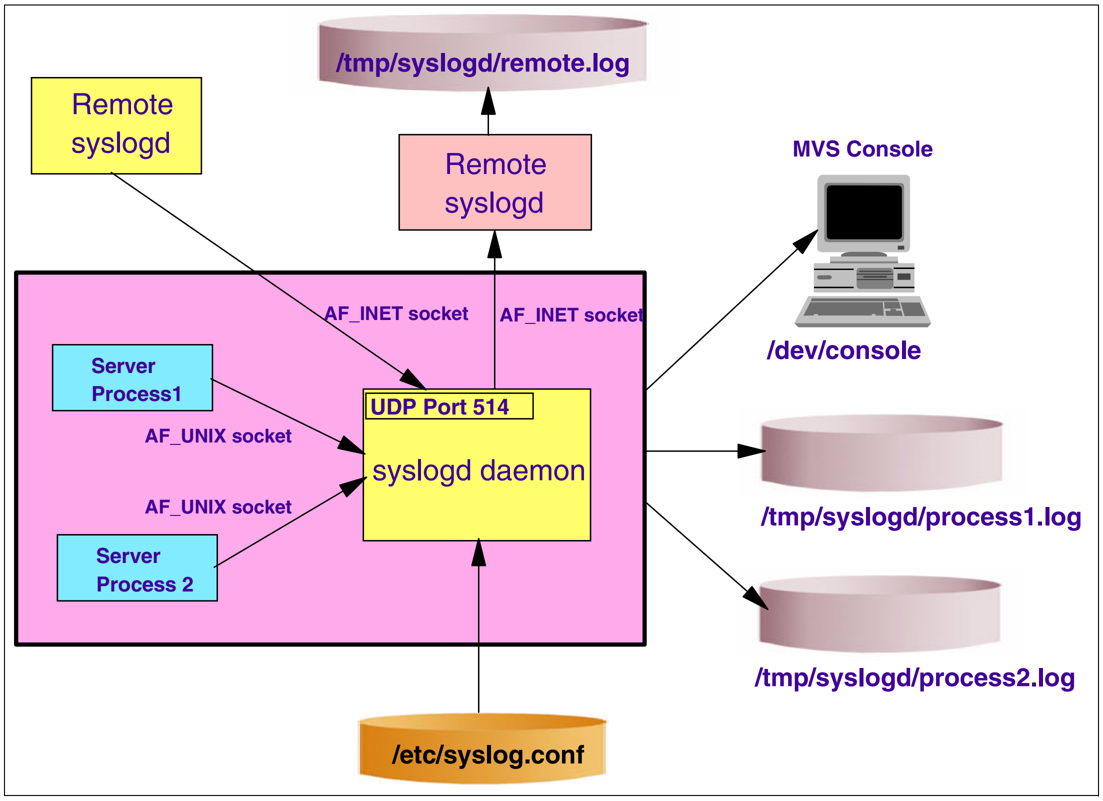


Figure 10-24 Current syslogd support

z/OS V1R8 enhancements

With z/OS V1R8, the Communication Server's syslog daemon now supports the following changes:

- Introduce a new filter Hostname/IP address for the syslog daemon. This allows administrators to limit what remote messages are logged and where.
- Issue a WTO indicating that OPERLOG is not active if OPERLOG is inactive.
- Create a new file, /etc/syslog_net.pid.

The /dev/operlog support provides a much lower path length to /dev/console, and a WTO is then issued for each message in that file. The resulting messages from /dev/operlog do not appear on a console or the SYSLOG.

Note: Linux messages generally do not contain message IDs, making automation of these messages difficult, and requiring parsing of every Linux message. Thus, /dev/operlog does not support automation of the messages.

10.7.1 Message integration support

Figure 10-25 displays an overview of how the new message routing works. You can use the message routing from every platform that supports the SYSLOG standards. Shown are the following:

- Three Linux partitions and z/OS running on the network.

  zSeries customers are deploying functions and workloads on Linux on zSeries. Some of these functions and workloads may interact with z/OS images or perform functions on behalf of z/OS. For example:

  – Middle-tier servers such as Web servers or Application servers
  – Communications Controller for Linux (CCL)

  In these scenarios it may be desirable to be able to integrate certain key messages from these Linux hosts into the z/OS environment and allow the z/OS operator to monitor key events that occur on the Linux hosts that may have an effect on z/OS operations as well.

- For messages from the Linux systems and some of the local z/OS messages, the new message integration support allows a common logging in z/OS for messages from multiple systems and provides a quick alternative way to log messages to the OPERLOG. Normally you would need to go through the system log of each system separately.

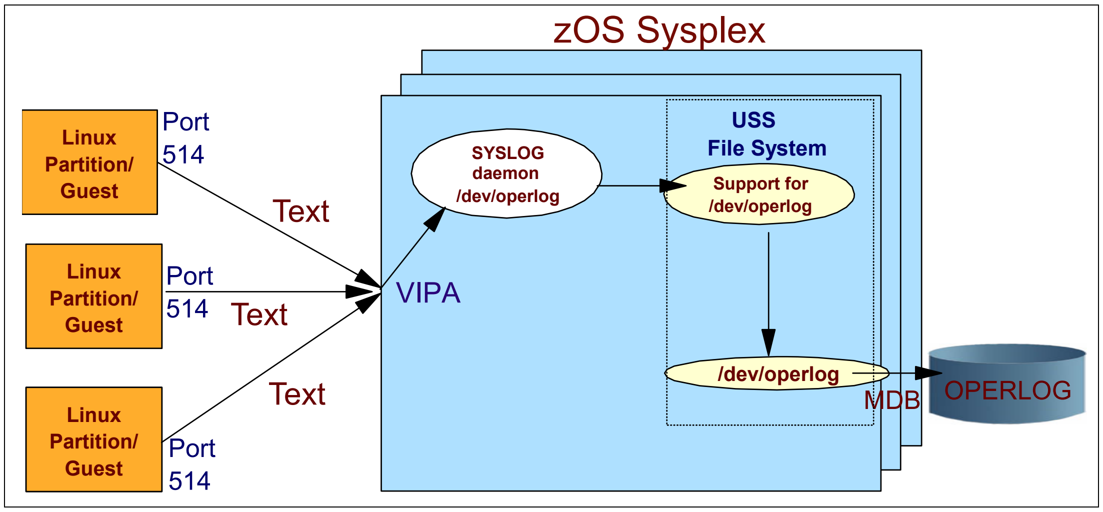


Figure 10-25 Message routing to z/OS

New syslogd support

The ability to receive messages from remote syslogd instances can be very useful in providing a consolidated log of messages from multiple hosts into a consolidated z/OS message log. For example, in scenarios where you are running Linux hosts on zSeries processors, and these Linux hosts are performing processing in cooperation with or on behalf of z/OS systems, you might want to have certain important messages generated on the Linux hosts visible from a z/OS system. This capability would enable z/OS operators to be alerted of specific conditions on the Linux hosts that might require actions to be taken locally or on the Linux hosts.

If you decide that this remote logging capability is useful in your environment, there are several configuration considerations that should be examined prior to enabling this function. It is also important to note that the syslogd remote logging capability can work in both directions, as follows:

- The z/OS syslogd can forward some of its messages to another remote syslogd instance.

- The z/OS syslogd instance can be the receiver of remote syslogd messages.

Following are the changes to the new syslogd support:

- The first character of a message indicates whether the message is from a local (z/OS) or a remote system.
- To be able to log messages to OPERLOG, /dev/operlog builds a data block in a 4k storage area (around 40 lines), as shown in Figure 10-25 on page 233.
- Seventy characters can be written on a line. If in any of those 70 characters a new line character is found, the message is split there. Otherwise, if a blank can be found in the last 10 characters, the message is split on the blank. If neither a new-line character nor an appropriate blank is found, all 70 characters are displayed on the line, and the message is split on a character.
- If the input data is more than can fit in a 4k storage area, the remaining data is discarded and the last line of the message indicates:

  ```
  xxxx BYTES OF INPUT DATA HAS BEEN TRUNCATED
  Where xxxx is the total amount of data (in Decimal) that has been discarded.
  ```

10.7.2 Setting up and customizing syslogd

To set up the syslogd in your system, do the following:

1. Customize the parameter file for the syslogd client and server.
2. Set up the z/OS start procedures.

It is now possible to start two instances of syslogd, shown in Figure 10-26 on page 235, as follows:

- One instance in local only mode (-i option)
- One instance in network only mode (-n option)

Starting syslogd

The syslogd command has some new options with z/OS V1R8, as follows:

```
syslogd [-f conffile ] [-m markinterval ] [-p logpath ] [-c ] [-d ] [-i ] [-n ]
[-x ] [-u ] [-? ]
```

Where:

-i	This option starts syslogd in local only mode (old option).
-n	This option starts syslogd in network only mode.
-x	This option causes syslogd to avoid resolver calls for converting IP addresses to hostnames.

Using both local and network logging, it is recommended to use two instances of syslogd, as shown in Figure 10-26 on page 235. This ensures that local syslogd logging is not adversely affected by the amount of remote messages being forwarded to z/OS.

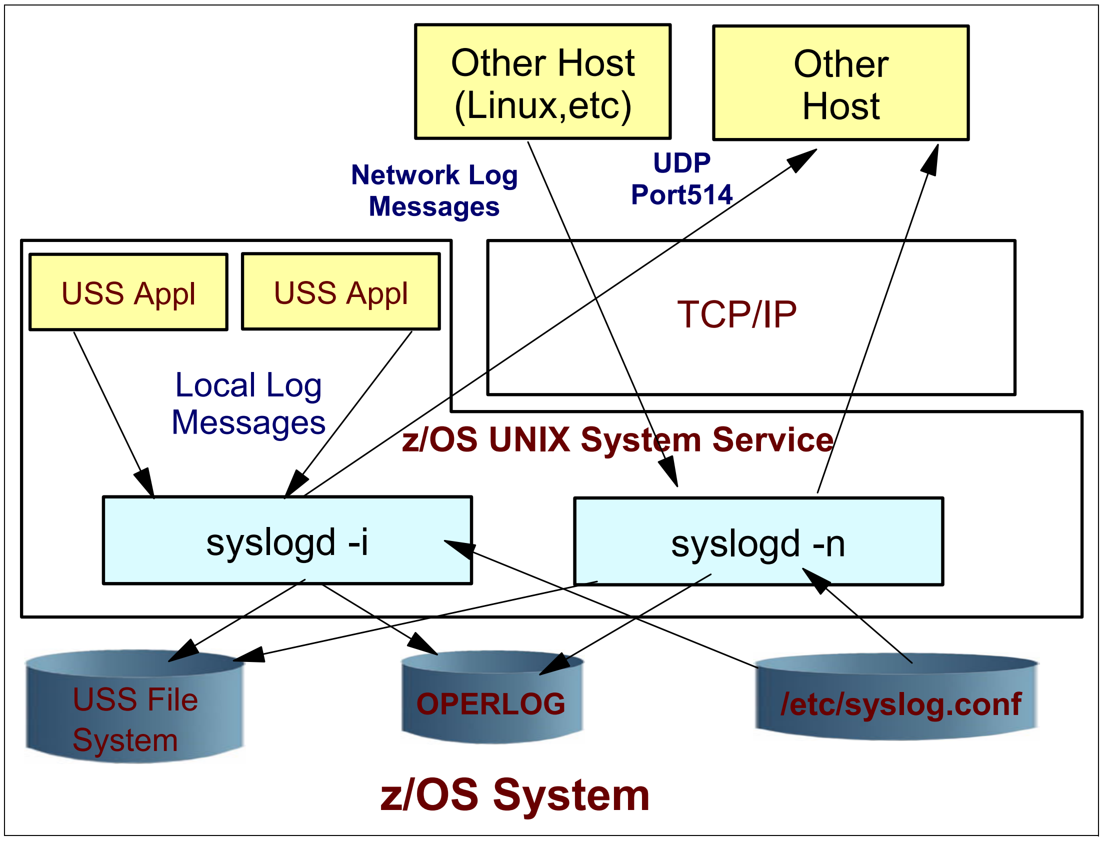


Figure 10-26 Relationship of syslogd to applications for which it logs information

Customize the syslog.conf file

With z/OS V1R8, the format of the syslog.conf entry is extended. With this change filtering of messages for specified hosts or networks is possible. This allows administrators to limit what remote messages are logged and where. You can identify which messages the z/OS syslogd should process by identifying the known remote syslogd instances from which you expect to receive messages by defining them in the syslog.conf configuration file, as follows:

- Write all messages with priority crit and higher that arrive from host 192.168.0.6 to the OPERLOG log stream:

```
(192.168.0.6).*.crit /dev/operlog
```

- Write all messages with priority crit and higher that arrive from any host with IP address in the range 192.168.0.0 to 192.168.0.255 to the OPERLOG log stream:

```
(192.168.0.6/24).*.crit /dev/operlog
```

Note: If using IP addresses, as shown in the previous examples, the IP address can be followed by an optional forward slash and a number representing the number of significant bits of the address. This is called the prefix length. The prefix length provides a means to indicate that a condition applies to all IP addresses that have the bit pattern for the specified number of bits.

The device file /dev/operlog is reserved for syslog daemon use only. However, there is no way to prevent someone from writing messages directly to /dev/operlog.

Customize the parameter file for the syslogd client

The configuration for the syslogd is similar to the server. We used a z/OS V1R7 system to act as a client in our test scenario. Figure 10-27 shows an example of the client configuration for syslogd. The meaning of this file is that all messages are routed to the z/OS V1R8 system. At the z/OS V1R8 system the messages will appear on the system console.

```
*.*                  @10.1.40.4
```

Figure 10-27 Example of syslog.conf for the client

10.7.3 Setting up the z/OS start procedure for the client and server

There are several ways to activate the syslog daemon. We recommend that you establish a z/OS started task. The started task is more flexible to use in z/OS. Figure 10-28 shows an example of a syslogd started task.

```
//TCPLOG    PROC
//TCPSYS  EXEC PGM=SYSLOGD,REGION=30M,TIME=NOLIMIT,
//  PARM=('POSIX(ON) ALL31(ON)',
//  '/-f /etc/operlog.server ')
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD SYSOUT=*
//SYSERR    DD SYSOUT=*
//SYSOUT    DD SYSOUT=*
//
```

Figure 10-28 Sample started task for syslogd

After starting the started task you see on the z/OS V1R8 system, the new BPX060I message appears on the system console shown in Figure 10-29.

```
BPXF060I LOGGED BY SYSLOGD FROM A LOCAL SOURCE.... 000
May 11 14:13:01 WTSC75 syslogd: FSUM1220 syslogd: restart..
BPXF060I LOGGED BY SYSLOGD FROM A LOCAL SOURCE.... 000
May 11 14:13:01 WTSC75 syslogd: FSUM1237 Job TCPLOG1  running in
local-only mode..
BPXF060I LOGGED BY SYSLOGD FROM A LOCAL SOURCE.... 000
May 11 14:13:01 WTSC75 syslogd: FSUM1232 syslogd: running non-swappable..
```

Figure 10-29 Example messages from the local system

After starting the client syslogd, all messages are routed to the server system and appear on its system console, as shown in Figure 10-30.

```
BPXF060I LOGGED BY SYSLOGD FROM A REMOTE SOURCE.... 000
May 15 20:06:04 10.1.40.30 FSUM1220 syslogd: restart..
BPXF060I LOGGED BY SYSLOGD FROM A REMOTE SOURCE.... 000
May 15 20:06:04 10.1.40.30 FSUM1232 syslogd: running non-swappable..
```

Figure 10-30 Sample messages from the client system

At this point, message routing from a remote client is in place and usable. You can also send individual messages to the system console using the SYSLOG interfaces.

10.7.4 Availability considerations for syslogd

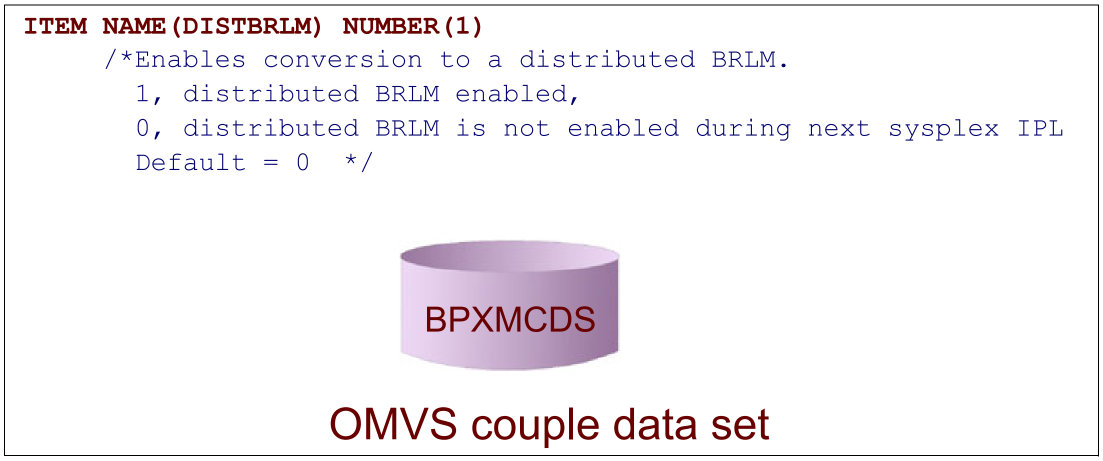
Syslogd availability is important to the logging of both local and remote messages. The following configuration considerations can help improve the availability of the z/OS syslogd remote logging services.

- Ensure that there is a process in place to restart a z/OS syslogd after a failure that results in the termination of syslogd. This can be accomplished by placing syslogd in the AUTOLOG list in the TCP/IP profile. This enables TCP/IP to initially start the syslogd instance as a started task, and also enables TCP/IP to monitor whether syslogd has a socket bound to the syslogd port used for remote message receipt (UDP port 514). This approach works well for single-stack INET configurations. If multiple TCP/IP stacks are deployed on a single system (that is, a CINET configuration), you should not use the AUTOLOG list. Alternatively, an installation can use any available automated operations software package to automate the start and restart of syslogd. For more information about the AUTOLOG statement, refer to *z/OS Communications Server: IP Configuration Reference*, SC31-8776.
- On MVS systems that are not part of a sysplex and that have TCP/IP stacks with multiple network interfaces, using a static VIPA address can provide for better availability characteristics should a specific network interface or network experience an outage. This involves configuring a static VIPA in the TCP/IP profile or reusing an existing one. No special configuration of the local syslogd is needed. However, you should configure the remote syslogd instances to use the static VIPA address as the destination address when forwarding messages. This can typically be accomplished by either specifying the static VIPA as the destination address, or specifying a host name that maps to the static VIPA in the remote syslogd configuration.
- When the recipient syslogd instance is running in a sysplex environment, additional availability features are available that should be explored. For example, you can use a multiple application-instance dynamic VIPA (DVIPA) to represent the syslogd instance on a given system, and the remote syslogd instances are configured to use that DVIPA address as the destination IP address for the messages they forward. In this configuration, if a failure to the MVS system or TCP/IP stack on which the syslogd instance is running occurs, the DVIPA is automatically moved to a predefined backup system that is currently active. This enables the syslogd instance on that backup system to begin processing these remote messages in a transparent manner. In this configuration, using the MVS operations log (OPERLOG) as the destination provides additional benefits, because the operations log is implemented as a coupling facility log stream that any system in the sysplex can access.

10.8 Recovery of BRLM locks

Management of locks in a z/OS UNIX environment began with OS/390 V2R9 using central BRLM in a shared HFS sysplex that included support for one BRLM. This implementation became a single point of failure. z/OS V1R4 offered distributed BRLM that provided one BRLM per system in a sysplex where lock commands are routed to the file system owner. In z/OS V1R6, distributed BRLM with moveable locks was introduced and the locks moved when the file system moved. Distributed BRLM then became the default in a z/OS V1R6 sysplex.

To activate distributed BRLM in a sysplex with z/OS V1R6 systems, you must redefine the sysplex couple data sets for BPXMCDS using the DISTBRLM(1) option, as shown in Figure 10-31. In a sysplex with all participants on at a z/OS V1R6 level or higher, distributed BRLM is the default.



```
ITEM NAME(DISTBRLM) NUMBER(1)
     /*Enables conversion to a distributed BRLM.
       1, distributed BRLM enabled,
       0, distributed BRLM is not enabled during next sysplex IPL
       Default = 0  */
```

Figure 10-31 Definition for distributed BRLM in the OMVS couple data set

Attention: See

```
http://www-03.ibm.com/servers/eserver/zseries/zos/unix/bpxa1ty2.html
```

for the rangelks REXX tool to identify currently running lock holders.

BRLM status

The MVS console command F BPXOINIT,FILESYS=DISPLAY,GLOBAL identifies whether USS is using central BRLM or distributed BRLM in a sysplex. In Figure 10-32, the command displays whether distributed BRLM is active or not. The problem in the past was, that when a USS application locked a remote file, the lock was lost when the remote system failed.

```
F BPXOINIT,FILESYS=DISPLAY,GLOBAL
BPXM027I COMMAND ACCEPTED.
 BPXF041I 2006/09/06 14.58.42 MODIFY BPXOINIT,FILESYS=DISPLAY,GLOBAL
 SYSTEM   LFS VERSION ---STATUS--------------------- RECOMMENDED ACTION
 SC74       1.  8.  0 VERIFIED                       NONE
 SC75       1.  8.  0 VERIFIED                       NONE
 CDS VERSION=  2        MIN LFS VERSION=  1.  8.  0
 BRLM SERVER=N/A        DEVICE NUMBER OF LAST MOUNT=     126
 MAXIMUM MOUNT ENTRIES=     500   MOUNT ENTRIES IN USE=      69
 MAXIMUM AMTRULES=           50   AMTRULES IN USE=            2
 DISTBRLM ENABLED=N/A             DISTBRLM ACTIVE=YES
 MAXSYSTEM=                   4
 BPXF040I MODIFY BPXOINIT,FILESYS PROCESSING IS COMPLETE.
```

Figure 10-32 Display the BRLM status

z/OS V1R8 support

In z/OS V1R8, the ability to recover file locks when the remote system fails is implemented. The order for a remote file lock occurs in the following way:

1. An application issues a lock for a file owned by another system.
2. USS forwards the lock to that system's BRLM.
3. USS forwards the lock to the owner system's BRLM.
4. The file owner system fails.

5. USS recovers the file system and declares a new owning system.
6. USS re-issues the lock to the new owner.

Note: When the file system containing the file is owned remotely, the lock will also be remote.

10.9 z/OS UNIX expands POSIX standards

z/OS UNIX has always been focused on providing the infrastructure needed to port applications from other UNIX platforms or to develop applications that are portable. z/OS UNIX relies on industry standards to define the interfaces that are standard across UNIX platforms. The current specifications published by X/Open are called SUSv3 or UNIX03.

Single UNIX Specification Version 3 (SUSv3) expands on the POSIX standards and provides interfaces for portable UNIX applications such as shell scripts. The problem in the past was that UNIX applications and shell scripts could not always easily be moved to z/OS. A pax archive creating a sequential data set had to use the default allocation parms.

In the following topics we introduce changes to the following functions:

- How the new environment variable _UNIX03 works
- The new `cp` command options
- The new option for the pax utility

10.9.1 Single UNIX Specification Version 3 (SUSv3) - UNIX03

The original z/OS UNIX design chose z/OS-specific options that conflict with new options added by UNIX03. Also, the UNIX03 specification is not fully compatible with prior UNIX specifications. A new _UNIX03 variable only affects paths where options and behavior conflict with the existing z/OS implementation.

Where new options and behavior do not conflict with existing externals, the _UNIX03 variable is not needed. When specifying this option in z/OS V1R8, only an uppercase YES is recognized, as follows:

```
_UNIX03=YES
```

New SUSv3 options flags

Some conflict with existing z/OS UNIX-specific options is now possible, but the following changes have been implemented:

- SUSv3 behavior
  - Changes the pax -o keyword parsing
- _UNIX03=YES
  - Dictates to use the SUSv3 behavior
  - A new -W option to handle the z/OS specific options
  - Can be set in /etc/profile:
    ```
    $HOME/.profile
    command-line
    setenv()
    ```
- _UNIX03 unset or not YES maintains the existing behavior

10.9.2 Enhancement to the cp and mv commands

There are new options provided by z/OS V1R8 for the **cp** and **mv** UNIX commands. You are now able to copy or move files, including their symbolic links, using the new options shown in Table 10-2. The **mv** command only uses the -W option.

Table 10-2 New options for the cp and mv commands

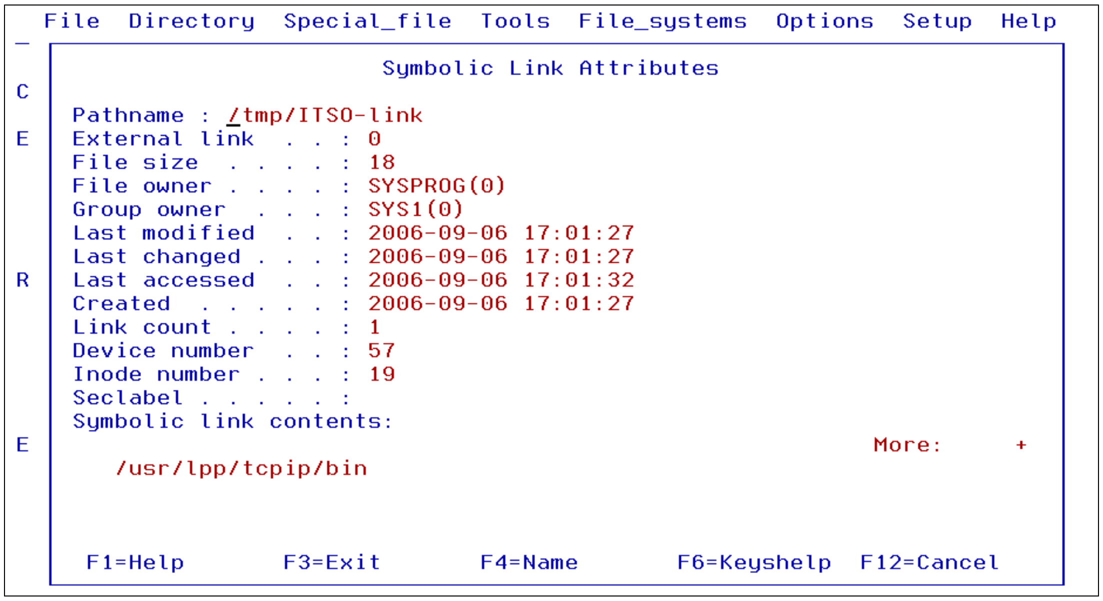
Option	Meaning
-H	Follows symbolic links specified as source operand on the command line. Symbolic links encountered in the tree traversal are not followed. This option can only be used with the -R or -r option. This is the default behavior when -R or -r option is specified but none of the options -H, -L, or -P are specified.
-L	Follows symbolic links specified as source operand on the command line and those encountered in the tree traverse. This option can only be used with the -R or -r option.
-P	This parameter has two different meanings depending on whether the _UNIX03 environment variable is set to YES or not. If _UNIX02 is set to YES, and does not follow any symbolic links, neither those specified as source operand on the command line nor those encountered in the tree traverse. In the other case the -P option will be treated as specifying parameters needed to create a new sequential data set if one does not already exist. You can specify the RECFM, LRECL, BLKSIZE, and SPACE in the format SPACE=(units,(primary,secondary) where the following values are supported for units: ► Any positive integer indicating BLKSIZE ► CYL (mixed case) ► TRK (mixed case)
-W	Specifies the parameters needed to create a sequential data set if one does not already exist. You can specify RECFM, LRECL, BLKSIZE, and SPACE. The format is exactly the same as for the -P option.

cp command changes

In prior releases of z/OS, it was possible to use the **cp** command with the **-P** option to pass MVS data set allocation parameters. Unfortunately, z/OS UNIX implementation chose z/OS-specific options that conflict with the new options added by UNIX03. Also, the UNIX03 specifications are fully compatible with prior UNIX specifications. The _UNIX03 variable only affects paths where options conflict with the existing z/OS implementation. Where new options do not conflict with existing externals, the _UNIX03 variable is not needed.

Copying symbolic links

In Figure 10-33 on page 241 a symbolic link is defined pointing to a directory in the HFS. Next, a copy of the /tmp directory is going to be made that includes the contents of the symbolic link called ITSO-Link.



```
   File  Directory  Special_file  Tools  File_systems  Options  Setup  Help
_
                        Symbolic Link Attributes
C
   Pathname : /tmp/ITSO-link
E  External link  . . . : 0
   File size  . . . . . : 18
   File owner . . . . . : SYSPROG(0)
   Group owner  . . . . : SYS1(0)
   Last modified  . . . : 2006-09-06 17:01:27
   Last changed . . . . : 2006-09-06 17:01:27
R  Last accessed  . . . : 2006-09-06 17:01:32
   Created  . . . . . . : 2006-09-06 17:01:27
   Link count . . . . . : 1
   Device number  . . . : 57
   Inode number . . . . : 19
   Seclabel . . . . . . :
   Symbolic link contents:
E                                                          More:     +
      /usr/lpp/tcpip/bin


    F1=Help       F3=Exit       F4=Name       F6=Keyshelp  F12=Cancel
```

Figure 10-33 Define a symlink

To copy the whole directory structure, /tmp, including the symbolic links, the **cp** command is used as shown in Figure 10-34.

```
cp -Lrp /tmp/* /u/rogers
```

Figure 10-34 Sample cp command

After the **copy** command we have an identical copy of the source tree. The symbolic links will be replaced by a directory that contains the content of the resolved symbolic links in the source tree.

```
ROGERS @ SC75:/u/rogers>ls -al
total 176
drwxr-x---   3 SYSPROG  SYS1         416 Sep  6 17:09 .
dr-xr-xr-x   6 SYSPROG  TTY            0 Sep  5 16:11 ..
-rw-------   1 SYSPROG  SYS1         574 Sep  6 18:13 .sh_history
drwxr-xr-x   5 SYSPROG  OMVSGRP     1120 Jul 26 21:57 ITSO-link
-rwx------   1 SYSPROG  SYS1           0 Aug 28 09:31 inetd-stderr
-rwx------   1 SYSPROG  SYS1           0 Aug 28 09:31 inetd-stdout
-rwxr-xr-x   1 SYSPROG  SYS1       73728 Apr 21 16:08 maxcore
ROGERS @ SC75:/u/rogers>
```

Figure 10-35 Sample directory structure after copy

10.9.3 New pax options

There are new options provided by z/OS V1R8 on the **pax** command. You can now handle files including their symbolic links using the new options you see in Table 10-3 on page 242.

_UNIX03 effects

As a result of the new SUSv3, the options shown in Table 10-3 have the following effect:

- pax -o parsing of multiple keywords
  - When _UNIX03=YES - keywords are separated by a comma.
  - When _UNIX03 is not YES - keywords are separated by a comma or blank.
- pax -r restore of access permission bits, when neither -p p nor -p e is specified
  - When _UNIX03=YES - files are restored with permissions 0666 modified by umask.
  - When _UNIX03 is not YES - files are restored with the saved permissions modified by the umask.

Note: _UNIX03=YES affects the parsing of -o keywords and restored permissions.

Table 10-3 New SUSv3 options for the pax command

Options	Meaning
-H	Follows symbolic links specified as source operand on the command line. Symbolic links encountered in the tree traversal are not followed. This option can only be used with the -R or -r option. This is the default behavior when the -R or -r option is specified but none of the options -H, -L, or -P are specified.
-L	Follows symbolic links specified as source operand on the command line and those encountered in the tree traverse. This option can only be used with the -R or -r option.
-x pax	Standing for the pax Interchange format which, like the os390 format(-x os390) and extended USTAR (-o saveext), saves or restores file attributes that cannot be stored in the USTAR header format, such as ACLs, external links, long link names, long path names, file tags, and extended attributes. Additionally, the pax Interchange format can save/restore file attributes that cannot be handled by any other format, such as: files greater than 8 GB in size, uid and gid values greater than 2097151, and z/OS-specific attributes such as user audit and auditor audit flags and file format. To restore certain information, the user must also have the appropriate privileges and have specified the corresponding options. See -p for options for restoring file attributes.
-o	Provides information for modifying the algorithm for writing and extracting files.

The -H and -L options work the same way as was introduced in the description of the **cp** command in 10.9.2, “Enhancement to the cp and mv commands” on page 240.

Note: On a z/OS system, superuser privileges or read access to the appropriate FACILITY class resources are required to create character special files, restore user and group names, and to set certain extended attributes (read access to the corresponding FACILITY class resources).

10.9.4 z/OS V1R8 pax enhancements

The following enhancements to the pax utility are implemented in z/OS V1R8:

- pax Interchange Format
- A new -x pax option

New pax Interchange Format

The pax Interchange Format (-x pax), which is a standard UNIX format, stores all file attributes that extended USTAR (-o saveeext) or os390 format (-x os390) store, and additionally can save and restore file attributes that cannot be handled by any other formats, such as:

- Files greater than 8 GB in size
- uid and gid values greater than 2097151
- z/OS-specific attributes such as user audit and auditor audit flags and file format

The pax Interchange Format is supported on z/OS release 8 and above. pax Interchange Format archives can be extracted on older systems; however, there will be loss of information for archived files that have attributes which cannot be stored in USTAR format. When creating archives that may be extracted on older z/OS systems, it is recommended that USTAR (default), extended USTAR(-o saveext), or os390 (-x os390) format be used. When creating archives that will be extracted on z/OS release 8 systems and above, the pax format (-x pax) is the recommended format. See the -x pax option for more information about preserving extended attributes with the pax format.

The new pax Interchange Format also allows you to:

- Archive and copy z/OS specific attributes:
  - ACLs
  - Tag information
  - File format
  - Extended attributes
  - Audit flags
- Archive and copy files with pathnames > 255 chars
- Override file attributes on the command line

10.10 New inittab support

In z/OS V1R8, the /etc/inittab file is the same as used on other UNIX platforms. You have the support for /etc/initab to allow an installation to identify system processes that can be started at system initialization; and to identify processes that can be restarted automatically when they unexpectedly end.

The **/etc/inittab** file lists system processes such as commands and shell scripts that are to be started when z/OS UNIX is initialized. Copy the IBM-supplied /samples/inittab file, shown in Figure 10-36 to /etc/inittab and add additional entries to it. For example, you can add daemons such as syslogd and cron that are normally started from /etc/rc to take advantage of the respawn capability. The /etc/inittab file is processed only once during initialization. If a command entry in the file needs to be run again, it must be manually restarted (for example, from the z/OS UNIX shell or via BPXBATCH). To restore the respawn attribute, the command can be started using the _BPXK_INITTAB_RESPAWN environment variable.

```
etcrc::wait:/etc/rc > /dev/console 2>&1
inetd::respfrk:/usr/sbin/inetd /etc/inetd.conf
msgend::once:/bin/echo Done processing /etc/inittab > /dev/console
:end of file
```

Figure 10-36 New inittab file in /samples/inittab

Rules for coding /etc/inittab

Each entry is delimited by a newline character. A backslash (\) preceding a newline character indicates the continuation of an entry. There are no limits (other than maximum entry size) on the number of entries in the /etc/inittab file. The maximum entry size is 1024 characters. Each entry field is only limited by the maximum entry size. The entry fields are:

Identifier Identifies the inittab entry. The identifier for a given entry can be up to 7 mixed-case characters and must be unique in the file. Because the identifier is used as the job name for the command to be run, it must be syntactically acceptable as a job name. The lowercase characters are converted to uppercase to be used in the job name

RunLevel Not supported for z/OS UNIX. Identifies the run levels in which this entry can be processed. Even though the RunLevel field is not supported for z/OS UNIX, it is in the inittab entry for compatibility with other UNIX implementations The run level field can be up to 32 alphanumerical characters.

Action Specifies how to handle the process that is specified in the command field. The supported actions are:

- **once** Starts the process and does not wait for it to end. When it ends, the process is not restarted when it ends.

- **respawn** Starts the process as specified in the entry and does not wait for it to end. Instead, it continues scanning the /etc/inittab file. The process is restarted when it ends. When a process is spawned again, it is restarted with the same file descriptors and environment variables that it was started with originally. If a process ends due to a shutdown of all fork activity, the process is not restarted until fork activity is re-enabled. If a respawnable process ends and then ends again after being restarted within 15 minutes of its first ending, then message BPXI082D is displayed. The message identifies the process entry and asks whether to retry or ignore the error. A process identified for respawn will not be able to register as a permanent process that can survive a shutdown and restart cycle because the /etc/inittab file will be processed again during restart. **Tip:** To avoid excessive consumption of common storage, limit the number of processes started with the respawn attribute to 100 or fewer.

- **respfrk** Starts the process as specified in the entry. Do not wait for the process to end; in other words, continue scanning the /etc/inittab file.

  - If the process never issues a fork command, then this action behaves the same way as the respawn action.
  - If the process issues a fork, then the respawn attribute is transferred to the forked child process and the original process is respawned when the child process ends.
  - If the original process issues any additional forks, the respawn attribute is not transferred. It is only transferred the first time the fork is issued.

    Tip: This option is intended for a program that forks itself to create a child process which then continues running while the parent process ends. For example, the cron and inetd daemons are written this way. This option is not found on any other UNIX platforms.

wait	Starts the process and waits for it to end. The process is not restarted when it ends.

Important: If /etc/inittab exists in your system, it is used instead of the /etc/rc file.

z/OS UNIX initialization

During OMVS initialization the system checks for the existence of the /etc/inittab file. The /etc/inittab file is only processed during z/OS UNIX initialization and only if the new SPAWN_PROCESS_INITTAB bit is set. This effectively limits the processing of /etc/inittab to the init process.

_BPXK_INITTAB_RESPAWN

This variable specifies whether a process is to be dynamically started with the respawn attribute. _BPXK_INITTAB_RESPAWN=YES specifies that a process is to be started with the respawn attribute. Setting the YES attribute after the process has started does not affect the setting of the respawn attribute.

If a process is started by a spawn with _BPXK_INITTAB_RESPAWN=YES (set by an export shell command, for example), the shell invokes the target program. The program will be automatically restarted when it ends, even if it was not originally started from the /etc/inittab file. Following are the rules for using this option:

- To set the variable to YES, you must have superuser authority. A specification of NO disables the respawn capability of the process.
- The NO setting must be set by an application (by a putenv call, for example), while it is running. You can choose the NO setting to allow for a problem to be fixed if one forced the application to end. Doing so prevents the application from being restarted automatically again with the same problem.

11 zFS enhancements

This chapter describes the enhancements to zSeries File System (zFS) in z/OS V1R8:

- zFS RAS enhancements
  - New stop ZFS support
  - Multi-file system aggregates - deny mounts
  - zFS hang detection
- zFS bpxmtext enhancements
- zFS mount performance
- zFS migrations from HFS

11.1 zFS RAS improvements

The enhancements in zFS in the area of RAS consist of three sub-items, as follows:

- Replacement of the current **`stop zfs`** command with a new command
- Removal of support for mounting a zFS file system that is contained in a multi-file system aggregate in a z/OS V1R8 system
- Console messages when zFS detects that there is a potential hang condition

11.1.1 LFS support for zFS shutdown

In prior releases of z/OS there was no way to stop the ZFS address space in a normal way. The only way to stop the zFS PFS was the P ZFS command or to cancel the address space. This caused problems after remounting the zFS file systems. A file system will not be dismounted correctly and applications may become unstable. In z/OS V1R8, a new command is introduced to stop the ZFS address space:

```
F OMVS,STOPPFS=ZFS
```

The new operator command allows z/OS UNIX to move file system ownership (nondisruptively) to another system before the PFS is terminated. This allows applications that are accessing file systems owned on the system where the zFS PFS is being terminated to continue to proceed without I/O errors in a shared file system environment.

In Figure 11-1 you can see a sample output of stopping the zFS file system. After receiving the stop command the system invokes a sync to all locally mounted zFS file systems. Afterward, all zFS file systems are unmounted in a single system. When you are in a shared file system environment, all AUTOMOVE-defined file systems will be moved to another system. Before the file system move starts, a new message, BPXI068D, appears to confirm the file system shutdown.

```
 F OMVS,STOPPFS=ZFS
*010 BPXI078D STOP OF ZFS REQUESTED. REPLY 'Y' TO PROCEED. ANY OTHER REPLY WILL
 CANCEL THIS STOP
 R 10,Y
 IEE600I REPLY TO 010 IS;Y
 IOEZ00048I Detaching aggregate LUTZ.ZFS
 IOEZ00048I Detaching aggregate PELEG.ZFS
 IOEZ00048I Detaching aggregate DAURCES.ZFS
 IOEZ00048I Detaching aggregate TEMEL.ZFS
 IOEZ00048I Detaching aggregate VAINI.ZFS
 IOEZ00050I zFS kernel: Stop command received.
 IOEZ00057I zFS kernel program IOEFSCM is ending
 IEF352I ADDRESS SPACE UNAVAILABLE
 IEA989I SLIP TRAP ID=X33E MATCHED.  JOBNAME=*UNAVAIL, ASID=0057.
*011 BPXF032D FILESYSTYPE ZFS TERMINATED.  REPLY 'R' WHEN READY TO RESTART.
```

Figure 11-1 Example of stopping a zFS file system

Restriction: The P ZFS command no longer functions in z/OS V1R8. Otherwise you see the message:

```
IOEZ00523I zFS no longer supports the stop command. Please issue f
omvs,stoppfs=zfs
```

Remember to remove the command from all procedures used for automated shutdown. If the address space is already down or you give a wrong file system type, the new message BPXI077I appears, as shown in Figure 11-2.

```
F OMVS,STOPPFS=HFS
BPXI077I THE PFS NAME IS INVALID OR THE PFS DOES NOT SUPPORT STOPPFS OR IS
ALREADY STOPPED.
```

Figure 11-2 Wrong file system name that does not support shutdown

11.1.2 Deny mount of multi-file system aggregate

zFS file systems contained in multi-file system aggregates, can not be mounted in a shared file system environment on a z/OS V1R8 system as of the z/OS V1R8 release. Any data in those file systems must be copied into a compatibility mode aggregate.

Important: This must be done using a system prior to z/OS V1R8 or a non-shared file system environment.

In a shared file system environment, with z/OS V1R8 and z/OS V1R7 (or earlier) systems, a file system contained in a multi-file system aggregate can be mounted on a z/OS V1R7 (or earlier) system and the z/OS V1R8 system can access that file system via z/OS UNIX function shipping, as shown in Figure 11-3. However, the file system cannot be mounted on the z/OS V1R8 system by any of the following ways:

- Via AUTOMOVE on system failure
- Explicit move (**chmount** command)
- Automount
- System shutdown

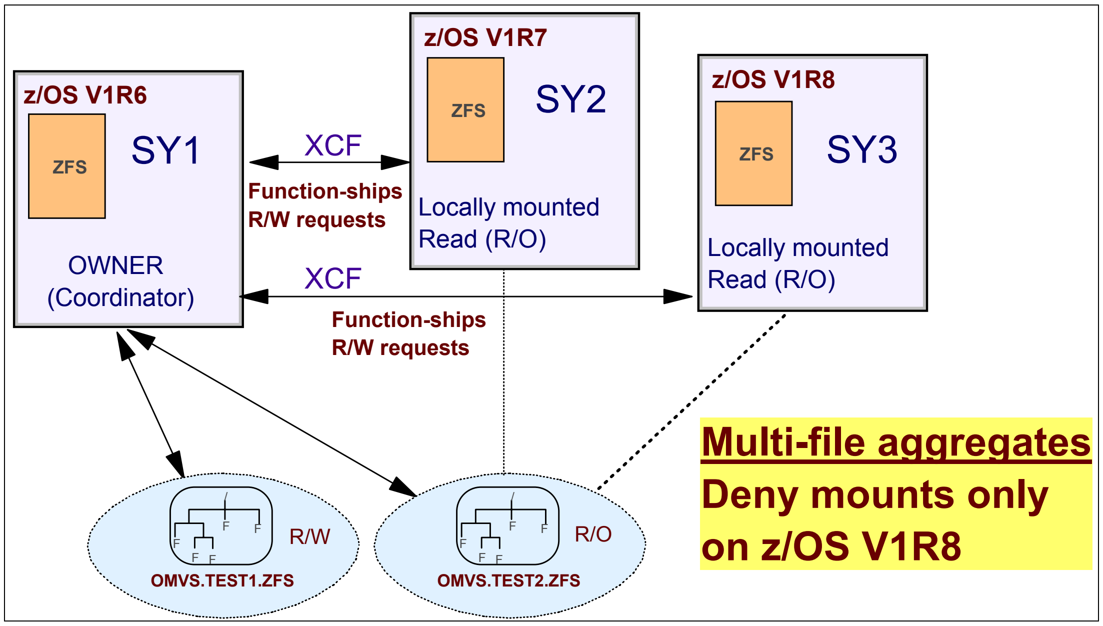


Figure 11-3 Three-system sysplex in a shared file system environment

Deny mount message

When you try to mount a multi-file system on a z/OS V1R8 system in a shared environment you see the following error message:

```
IOEZ00522I Filesystem LUTZ.MULTI.ZFS is not in a compat aggregate.  It cannot
be mounted
```

Important: Before you migrate to z/OS V1R8, it is highly recommended to migrate all multi-file aggregates to a compatibility mode aggregate.

Aggregates in a non-shared file system environment

You can still create and format a multi-file system but you will see a warning message like the one shown in Figure 11-4.

```
SYSPROG @ SC75:/u/lutz>zfsadm format -aggregate LUTZ.MULTI.ZFS
IOEZ00552I Multi-file system aggregates are restricted and support will be
removed; plan to migrate.
Done.  LUTZ.MULTI.ZFS is now a zFS aggregate.
```

Figure 11-4 Warning message for a multi-file system aggregate with z/OS V1R8

If you are in a non-shared environment, you can still mount a multi-file aggregate.

11.1.3 zFS hang detection

Determining the cause of a hang in zFS can be difficult. Usually, the hang is not noticed right away. This means that one or more applications are not making progress and that the information needed to diagnose the problem may be lost.

Hang detection support in z/OS V1R8

zFS has added support to try to detect when a hang has occurred in the ZFS address space. The zFS hang detector, intended for use by IBM Service, monitors the current location of the various tasks processing in zFS. At a set interval, the hang detector thread wakes up and scans the current user requests that have been called into zFS. The hang detector processes this list of tasks and notes various pieces of information that allow it to determine the location of the task. When the hang detector determines that a task has remained in the same location for a predefined period of time, it flags the task as a potential hang and generates message IOEZ00524I or IOEZ00547I to the console. If on a subsequent iteration the hang detector recognizes that this task has finally progressed, it will DOM the message (remove it from the console). If the message is removed by zFS, it means that there was no hang.

If zFS finds that a user thread is at the same (wait instruction) for 3 consecutive minutes, then zFS displays a message to the operator console indicating that there may be a hang in zFS, as shown in Figure 11-5.

```
IOEZ00524I zFS has a potentially hanging thread.
```

Figure 11-5 New hang detection message

Find hanging thread

To find a hanging thread, the operator can use the F ZFS,QUERY,THREADS command to determine if a thread is hanging, as shown in Figure 11-6 on page 251.

```
F ZFS,QUERY,THREADS
IOEZ00438I Starting Query Command THREADS.
            zFS and USS Tasks
            ---------------------
Recov     TCB       ASID  Stack     Routine               State
--------  --------  ----  --------  --------              --------
79D4A218  008EE650  0020  7993E6A0  comm_daemon           RUNNING
       since  Sep 8 17:03:01 2006


xcf_stg_pool: 0
---------------
>>size 4084 num subpools 1
    subpool 0 total 2147483647 available 2147483647
>>size 36852 num subpools 1
    subpool 0 total 2147483647 available 2147483647
>>size 92148 num subpools 1
    subpool 0 total 2147483647 available 2147483647

xcf_stg_pool: 1
---------------
. . . . (text omitted)
--------------
xcf thread pool: 0  size 10
---------------------------
  all threads in this pool are idle

xcf thread pool: 1  size 1
---------------------------
  all threads in this pool are idle

xcf thread pool: 2  size 4
---------------------------
  all threads in this pool are idle
. . . . (text omitted)
IOEZ00025I zFS kernel: MODIFY command - QUERY,THREADS completed successfully.
```

Figure 11-6 Command to find a hanging thread

Command to break hangs

If hangs are detected, the operator can use the F ZFS,HANGBREAK command to attempt to break the hang condition.

This command posts any threads that zFS suspects to be in a hang with an error and can cause abends and dumps to occur, which you can ignore. When you issue the F ZFS,HANGBREAK command, the hang message can remain on the console for up to one minute. If you question the hang condition or if the command does not seem to have resolved the situation, contact IBM Support and provide the dump and SYSLOG information.

11.2 zFS bpxmtext enhancements

zFS reason codes are returned to APIs and are sometimes displayed in messages. Users do not always know where to find the meaning of the reason code. z/OS UNIX provides a

bpxmtext command that can be issued from a shell session that displays the meaning of z/OS UNIX reason codes.

Previously, the **bpxmtext** command could not display the meaning of zFS reason codes. With z/OS V1R8, support is added to allow the **bpxmtext** command to do this. What is displayed is the same as that contained in the *z/OS Distributed File Service Messages and Codes*, SC24-5917. Following is a message for a zFS error code in which you can see the new functionality to display the meaning of zFS reason codes:

```
ROGERS @ SC75:/u/rogers>bpxmtext EF096800
zFS Fri Jul 28 10:36:15 EDT 2006
Description: Mount for file system contain in multi-file system aggregate is
not allowed

Action: Using a release of z/OS prior to z/OS V1R8, attach the aggregate,
mount the file system and copy the file system data to a compatibility mode
aggregate.
```

11.3 zFS mount performance improvements

In prior releases of z/OS the zFS mount of a zFS file system could take a long time if it contained many millions of files and directories. This is particularly problematic during dead system recovery when a file system needs to be automoved due to a system failure in a sysplex. This means that the file system is unavailable until the file system mount is complete on the system that file system is being moved to.

11.3.1 zFS on-disk format

In z/OS V1R8, the on-disk of the zFS file system is now completely integrated into the zFS code. In previous releases this support is available with APARs. Also included, is additional information to improve the zFS mount. This allows recovery and the mount process to proceed more quickly, thereby improving availability when a system failure occurs.

Mount changes on-disk

A new on-disk format is created during the zFS mount, as follows:

- The design of a zFS aggregate requires zFS to read the allocation bitmap to determine the number of blocks thaty are free.
- For each file system, zFS must read the volume table to determine the number of free inode slots per block.
- Finally, for each file system, zFS must read the volume table to determine any zero link count files.

For the new fast mount support, this information is maintained on disk.

Note: When creating archives that will be extracted on z/OS V1R8 systems and above, using the pax format (-x pax) is recommended.

Coexistence support for fast mount

Toleration or coexistence APAR OA11573 is provided to allow the new zFS on-disk format to be handled by prior releases. APAR OA11573 must be installed on prior releases before the first IPL of z/OS V1R8. You must install this APAR on prior releases before IPLing z/OS V1R7. The PTFs needed are:

- z/OS V1R5 - UA20482
- z/OS V1R6 - UA20483
- z/OS V1R7 - UA20484

The zFS file format is now called Version 1.4. At the first time of mounting a zFS file system from a prior release in read/write mode, it is converted automatically to Version 1.4.

APAR OA11573

This APAR allows the prior releases to correctly access the new Version 1.4 for zFS aggregates. If you do not install this APAR on prior releases, prior releases will not be able to correctly access the new structure.

Converting back to version 1.3

Use in case the 1.4 aggregate is to be used on a system that does not have APAR OA11573 applied. To convert a zFS aggregate back to a Version 1.3 structure, use the zFS IOEAGSLV (salvager) utility. A new option, -converttov3, is provided to convert a version 1.4 zFS aggregate back to a version 1.3 zFS aggregate. The IOEAGSLV utility with the new -converttov3 option is provided with z/OS V1R7. Figure 11-7 shows the output after converting a zFS file system to Version 3 level.

```
Converting LUTZ.ZFS2
Done.  LUTZ.ZFS2 converted to version 3.
```

Figure 11-7 Output of converting job

IOEAGSLV utility example

IOEAGSLV is installed in the MIGLIB PDS. IOEAGSLV can be executed from any supported release by STEPLIBing to MIGLIB. Figure 11-8 shows a sample job.

```
//LUTZZFS  JOB (999,POK),'LK',CLASS=A,MSGCLASS=T,NOTIFY=&SYSUID,
// REGION=0M
//SALVAGE  EXEC PGM=IOEAGSLV,REGION=0M,
// PARM=('-aggregate LUTZ.ZFS2 -converttov3')
//STEPLIB  DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//STDERR   DD  SYSOUT=*
//CEEDUMP  DD  SYSOUT=*
//
```

Figure 11-8 Sample converting job

Mounting a version 1.3 aggregate

You should mount the Version 1.3 aggregate NOAUTOMOVE until the toleration APAR is applied to all systems in the sysplex or you could choose to AUTOMOVE(EXCLUDE) all z/OS V1R7 systems. Otherwise, the aggregate is automatically converted to a Version 1.4 aggregate if it is moved to a z/OS V1R7 system.

During the first mount process of a zFS file system on a z/OS V1R8 system, it is automatically converted to the 1.4 level, as shown in Figure 11-9 with the messages that appear during mount processing of a zFS file system at the 1.3 level on a z/OS V1R8 system.

```
IOEZ00500I Converting LUTZ.ZFS2 for fast mount processing
IOEZ00518I Converting filesystem LUTZ.ZFS2 to allow for fast mount
```

Figure 11-9 Messages during converting

11.4 LFS support for HFS to zFS migration

The logical file system (LFS), shown in Figure 11-10 on page 255, plays a large role in the conversion of HFS to zFS data sets. The migration to zFS file systems needs to be well planned since it will take significant effort to migrate all of the data sets from HFS file systems to zFS file systems. No capability exists to convert a file system in place to zFS or to use an HFS data set with zFS. From this, it follows that file systems that are used in a read/write mode will have to be made unavailable while being migrated to zFS and that two data sets will exist at least during the migration.

The unavailability of the data for read/write must be planned for, as well as ensuring the availability of adequate space for two file systems that are about the same size.

Given that there are now two data sets, you may want to have different naming standards for the zFS data sets and their HFS data set counterparts, or you may prefer to keep the names just as they are. In general, if the zFS data set is named to be the same as the HFS data set, no changes are needed in this area.

z/OS V1R7 provides a migration utility that is invoked via a REXX exec called BPXWH2Z. This is a tool that may help you migrate your HFS file systems to zFS file systems. It can migrate one file system at a time, or build a list and do many. This utility must run from ISPF to set up the file system migrations, but the actual migration work can be run in a UNIX background as well as foreground.

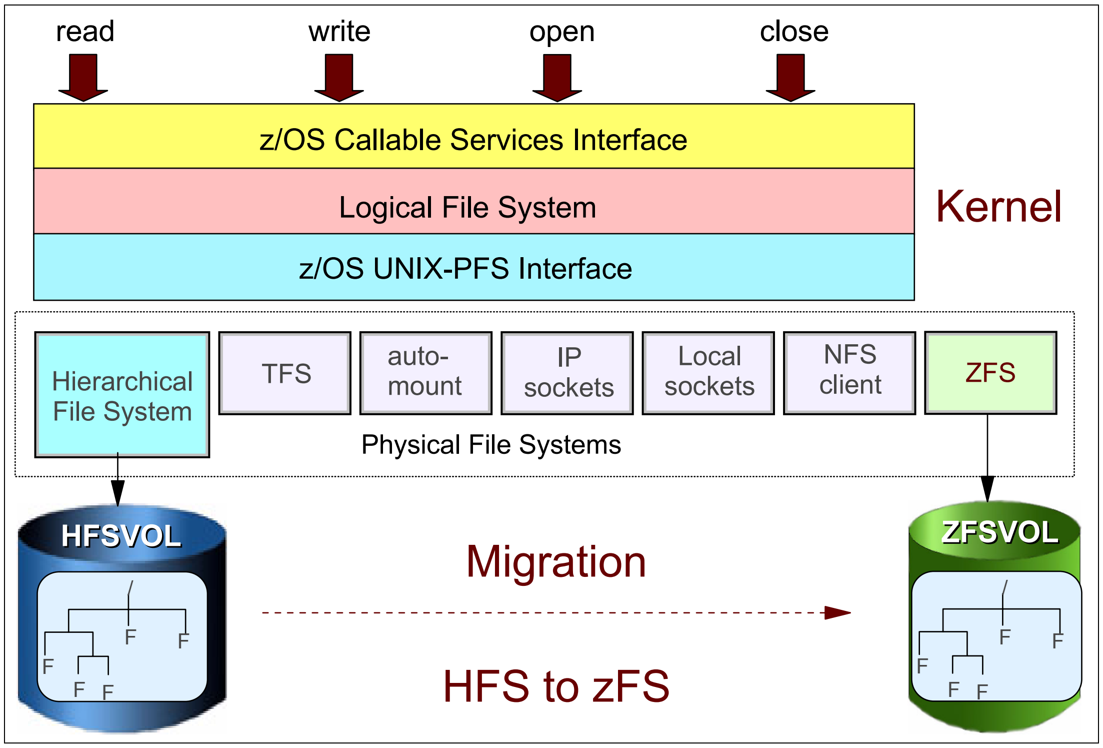


Figure 11-10 z/OS UNIX communication between LFS and PFS

11.4.1 Migrations steps for HFS to zFS

The following steps are recommended to migrate an HFS file system to zFS. They can be done manually, or the migration tool can be used since it will perform all of the following steps.

- If the HFS file system is mounted, either unmount it or switch it to read-only mode.

  The migration tool switches it to R/O.

- If the file system is not mounted, make a directory for it and mount it in R/O mode.

  The migration tool creates a temporary directory in /tmp for the mountpoint and deletes it when the migration is complete.

- Create a temporary directory for the zFS file system.

  The migration tool creates a temporary directory in /tmp for the mountpoint and deletes it when the migration is complete.

- Check the allocation attributes of the HFS file system.
- Define a new zFS file system with the appropriate allocation attributes.

  The tool defaults these to the same attributes as the HFS file system if its utilization is below about 75 percent; otherwise, it adds about 10 percent (below 90 percent) or 20 percent (above 90 percent). It is difficult to determine the exact size necessary for a zFS file system to contain all of the data in an HFS file system. Depending on the number of files, directories, ACLs, symlinks, and file sizes, zFS can consume either more or less space than HFS. zFS also has a log in the aggregate. For general purpose file systems, it appears that they consume about the same amount of space.

- Mount the zFS file system on its temporary mountpoint (a preexisting zFS cannot already be mounted).

- To use the pax utility to copy the HFS contents to the zFS, enter the shell, change directory to the HFS mountpoint, and run the pax utility. The **pax** command will look something like this:

  ```
  pax -rw -X -E . /tmp/zfsmountpoint
  ```

- If the HFS contains active mountpoints, do an **mkdir** for each of these in the zFS file system and set directory attributes as required.
- Set attributes for the zFS root based on the HFS root that was copied.

  Note: This must include all extended attributes, ACLs, owner, group, code page, mode bits, or SECLABEL.

- Unmount the zFS file system.
- Unmount the HFS file system.

  Note: If it contains active mountpoints, those file systems and any hierarchy below them must be unmounted first.

- Remove any temporary directories used as mountpoints.
- Rename the data sets as appropriate and ensure all mount scripts and policies have been updated as needed.

  The migration tool does not modify or search for any type of mount scripts or policies.
- If the HFS file system was originally mounted, mount the zFS file system in that same location, along with any hierarchy that also had to be unmounted to unmount the HFS.

11.4.2 Migration of the root

If you have not migrated the root file system during a ServerPac install, you can use the migration tool to copy the root from HFS to zFS, as follows:

- Set attributes for the zFS root based on the HFS root that was copied.

  Note: This must include all extended attributes, ACLs, owner, group, code page, mode bits, and SECLABELs.

- Unmount the zFS file system.
- Unmount the HFS file system.

  Note: If it contains active mountpoints, those file systems and any hierarchy below them must be unmounted first.

- Remove any temporary directories used as mountpoints.
- Rename the data sets as appropriate and ensure all mount scripts and policies have been updated as needed.

  The migration tool does not modify or search for any type of mount scripts or policies.
- If the HFS file system was originally mounted, mount the zFS file system in that same location, along with any hierarchy that also had to be unmounted to unmount that HFS.

11.4.3 Migration and coexistence considerations

Currently, OMVS initialization suspends until the recalls are complete for HFS data sets and mounts complete asynchronously to the processing for zFS data sets. Parmlib mounts in z/OS V1R7 are now changed to suspend for zFS mounts and fail HFS mounts for migrated data sets.

If multi-file system aggregates are in use, it is necessary to ensure that there are no HFS data sets cataloged with the same name as any file system in a multi-file system aggregate. The mount direction locates the HFS data set in the catalog and directs or redirects the mount to HFS.

Non-parmlib zFS mounts are done asynchronously. There is an accommodation in automount for HFS so that an HSM-migrated data set is recalled prior to entering the mount flow.

File system names

If the zFS data set names are to be the same as the HFS data set names, it is likely that scripts or policies that are used to mount file systems would not need to be changed. Automount policies that specify allocany or allocuser will continue to allocate file systems based on the file type specified. So, policies will need to change if new file systems are to be zFS file systems.

Since this is a migration scenario, mount processing first checks for the ZFS file system type along with the assumption that if that data set exists, the migration has been done. In order to revert back to using HFS, that data set must be renamed or removed.

Mount processing (HFS or zFS)

HFS and zFS are now generic file system types. They can be used for either HFS or ZFS. Mount processing first searches for a data set matching the file system name, as follows:

- If the data set is not an HFS data set and zFS has been started, the file system type is changed to ZFS and the mount proceeds to zFS.
- If the data set is not found, the mount proceeds to zFS (assuming zFS is started). If zFS is not started, the type is not changed and the mount proceeds to HFS.

A new place holder, ///, is created and represents the string HFS or ZFS as appropriate if the file system type is HFS, as shown in the following mount statement and MOUNT command:

```
MOUNT    FILESYSTEM('ZOSR17.MAN.///')
   TYPE(HFS)
   MODE(READ)
   MOUNTPOINT('/usr/man')
```

Mount processing first substitutes ZFS for the place holder, then checks if the data set exists. If it does not, HFS is used in the placeholder and again a check is run to determine if the data set exists. The MOUNT is directed either to zFS or HFS depending on which data set was found.

11.4.4 Using the BPXWH2Z migration tool

If you use the BPXWH2Z migration tool with no arguments, it prompts for an HFS file system name. You can also specify one or more file system names as an argument. The names can be simple names or patterns that you might use for the ISPF data set list panel. When you invoke BPXWH2Z, the ISPF panel shown in Figure 11-11 on page 259 is displayed.

Using BPXWH2Z

From the ISPF Option panel, specify `BPXWH2Z`. When you do this, you can also specify option flags. The option flags are preceded by a dash (-), and then followed immediately by the flags. The available flags and their meanings are:

-v	Additional messages are issued at various points in processing.
-c	Summary information goes to a file when background (`bg`) is selected once you are in the application. If this is not specified, summary information goes to the console.

Command line examples from ISPF Option 6 are:

```
bpxwh2z -v
bpxwh2z -c
```

In addition to the option flags, you can specify a command argument that is a character substitution string with the BPXWH2Z command.

You can preallocate the zFS file system or specify a character substitution string. This causes the zFS data set to not be renamed by the migration tool. To rename the zFS file system, use a substitution string as the first argument on the command line, as follows:

```
bpxwh2z /fromstring/tostring/ datasetname
```

For example, if your data set is OMVS.DB2.HFS and you want your zFS data set to be named OMVS.DB2.ZFS, you can specify the following command argument:

```
    bpxwh2z /hfs/zfs/ omvs.db2.hfs
```

Be careful; all strings matching `fromstring` in the data set name are replaced.

BPXWH2Z panels

By default, the panels are primed such that your HFS data set is renamed with a .SAV suffix and the zFS data set is renamed to the original HFS name. The documentation about these panels is available online by pressing PF1.

Use the **`End`** or **`Cancel`** commands to exit from this tool. No changes or allocations will have been done.

This migration process makes use of /tmp. The file system containing /tmp should not be migrated with this utility. In general, if you are using HFS as your /tmp, it would be best to start with a clean zFS. Also, you should not migrate your /dev file system. Start with a clean zFS. If you are migrating your root file system you should do this in the foreground. As part of migration, all file systems will be unmounted, including /tmp and /dev, which will keep you from seeing the status of the migration if run in the background.

This tool can display many different messages. These messages are documented only in the online help file.

To begin the migrations, enter either the FG or BG command to run the migration in foreground or background, respectively.

Data set migration considerations

Consider the following options when migrating data sets from HFS to zFS:

- After an HFS file system is migrated to zFS, the HFS data set will be renamed to the name in the “Save HFS as” field, and the zFS data set will be renamed to the old HFS data set name. If you do not want data sets to be renamed, set this field to the HFS data set name or clear the field and the tool will set it to the HFS name.

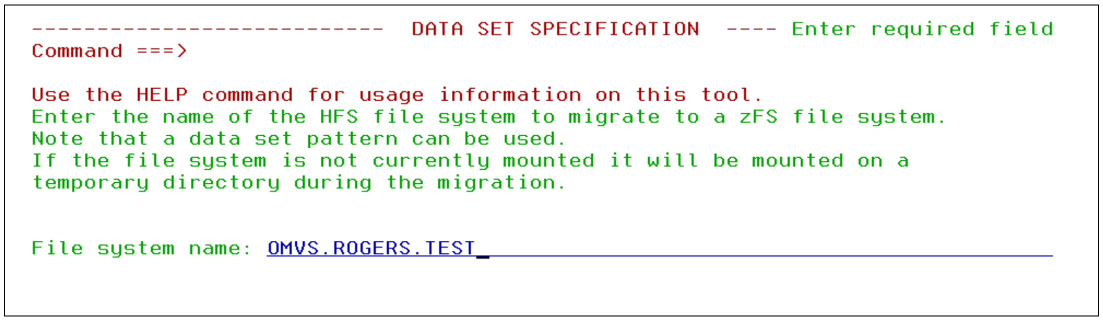
- If your zFS file system is not preallocated, it is allocated based on the attributes for the HFS data set. You will have an opportunity to alter any of these attributes.

  The size is set to the same size as your HFS if current utilization is below about 75 percent. If your HFS is multivolume, you must preallocate your zFS in order to have similar attributes and multivolumes.

- This utility best handles file systems that are not currently mounted or file systems that are mounted but don't have other file systems mounted below them.
  - If the file system is mounted, it is unmounted and the new zFS put in its place. If the file system does contain active mount points, this attempt to unmount that subtree to replace the HFS with the zFS and put back everything it needed to unmount.
  - Mounted file systems are switched to read-only during the migration. They should not be in use for update when you are doing this migration. The file system table display will indicate file systems that are mounted.

11.4.5 Data set migration examples

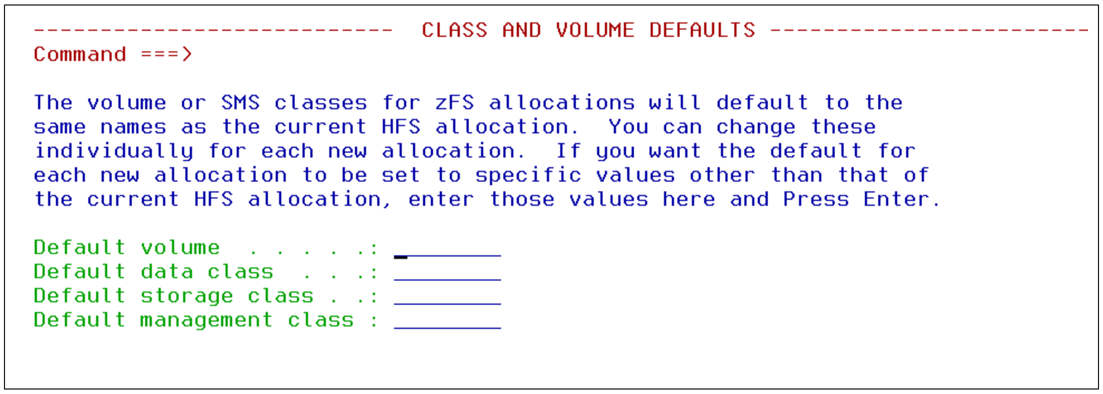
To migrate an HFS data set to zFS, type in the data set name as shown in Figure 11-11. In this example, the HFS file system to be migrated is mounted.



```
--------------------------- DATA SET SPECIFICATION ---- Enter required field
Command ===>

Use the HELP command for usage information on this tool.
Enter the name of the HFS file system to migrate to a zFS file system.
Note that a data set pattern can be used.
If the file system is not currently mounted it will be mounted on a
temporary directory during the migration.


File system name: OMVS.ROGERS.TEST_
```

Figure 11-11 ISPF migration tool primary panel

Press Enter; the screen shown in Figure 11-12 is displayed. As shown in the figure, you can change the SMS classes and volume information for the creation of the zFS data set.



```
--------------------------- CLASS AND VOLUME DEFAULTS ------------------------
Command ===>

The volume or SMS classes for zFS allocations will default to the
same names as the current HFS allocation.  You can change these
individually for each new allocation.  If you want the default for
each new allocation to be set to specific values other than that of
the current HFS allocation, enter those values here and Press Enter.

Default volume  . . . . .: ________
Default data class  . . .: ________
Default storage class . .: ________
Default management class : ________
```

Figure 11-12 Panel to change data set attributes when creating the zFS data set

Press Enter; the screen shown in Figure 11-13 on page 260 is displayed. Here you see the defaults for saving the HFS and the initial zFS data sets. The next step is to create the new zFS data set. There are two options for doing this to begin the migrations: enter the command FG or BG to run the migration in either the foreground or background, respectively.

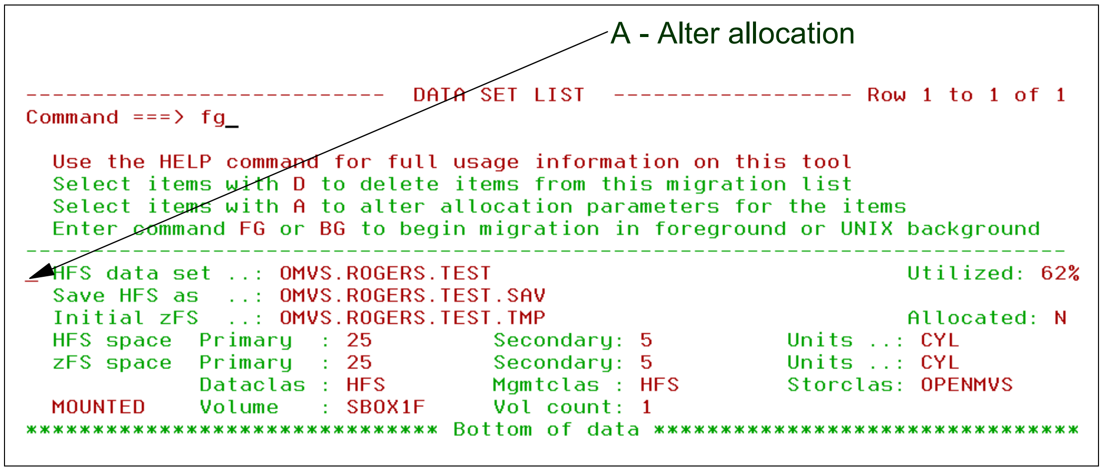
Note: Depending on the size of the HFS data set, the copy process may take a long time. If the migration is run in the background, the tool keeps a standard output log and also a summary log. The prefix for the pathnames is displayed.

Use the **End** or **Cancel** command to exit from this tool if no changes or allocations have been done. Note that the copy process may take a long time. If this is run in the background the tool will keep a standard output log and also a summary log. The prefix for the pathnames will be displayed.

Use of /tmp

This migration process makes use of /tmp. Make sure the following considerations are evaluated:

- The file system containing /tmp should not be migrated with this utility. In general, if you are using HFS as your /tmp, it would be best to start with a clean zFS.
- You should not migrate your /dev file system. Start with a clean zFS.
- If you are migrating your root file system, you should do this in the foreground. As part of migration all file systems will be unmounted, including /tmp and /dev, which will keep you from seeing the status of the migration if run in background.



```
---------------------------  DATA SET LIST  ------------------ Row 1 to 1 of 1
Command ===> fg_

  Use the HELP command for full usage information on this tool
  Select items with D to delete items from this migration list
  Select items with A to alter allocation parameters for the items
  Enter command FG or BG to begin migration in foreground or UNIX background
------------------------------------------------------------------------------
_ HFS data set ..: OMVS.ROGERS.TEST                          Utilized: 62%
  Save HFS as  ..: OMVS.ROGERS.TEST.SAV
  Initial zFS  ..: OMVS.ROGERS.TEST.TMP                      Allocated: N
  HFS space  Primary  : 25        Secondary: 5         Units ..: CYL
  zFS space  Primary  : 25        Secondary: 5         Units ..: CYL
             Dataclas : HFS       Mgmtclas : HFS       Storclas: OPENMVS
  MOUNTED    Volume   : SBOX1F    Vol count: 1
******************************* Bottom of data ********************************
```

Figure 11-13 Create the zFS data set in the foreground

The following messages are seen on the screen after the creation of the zFS data set:

```
 Migrating OMVS.ROGERS.TEST
creating zFS OMVS.ROGERS.TEST.TMP
copying OMVS.ROGERS.TEST to OMVS.ROGERS.TEST.TMP Blocks to copy: 2832
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IDC0531I ENTRY OMVS.ROGERS.TEST.TMP ALTERED
IDC0531I ENTRY OMVS.ROGERS.TEST.TMP.DATA ALTERED
mount /u/rogers/tmp OMVS.ROGERS.TEST ZFS 1
***
```

Note: If the HFS file system was mounted, it will be unmounted and the new zFS put in its place. If the file system does contain active mount points, this will attempt to unmount that subtree to replace the HFS with the zFS and put back everything it needed to unmount. Mounted file systems are switched to read-only during the migration. They should not be in use for update when you are doing this migration.

Altering the allocation parameters

As shown in Figure 11-13 on page 260, if you use the action character A, you can alter the space requirements for the creation of the zFS data set. The new allocation changed the zFS primary space to 35 from 25. The following messages are then seen on the screen after the creation of the zFS data set when, as in this example, the HFS file system was not mounted when migration was started:

```
Migrating OMVS.ROGERS.TEST
 creating zFS OMVS.ROGERS.TEST.TMP
 copying OMVS.ROGERS.TEST to OMVS.ROGERS.TEST.TMP Blocks to copy: 2832
 IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
 IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
 IDC0531I ENTRY OMVS.ROGERS.TEST.TMP ALTERED
 IDC0531I ENTRY OMVS.ROGERS.TEST.TMP.DATA ALTERED
 ***
```

The data set display from ISPF Option 3.4 shows the following, where you can see that the zFS data set after migration (OMVS.ROGERS.TEST.DATA) is larger than the original HFS data set:

```
OMVS.ROGERS.TEST
OMVS.ROGERS.TEST.DATA                           525    ?   1  3390
OMVS.ROGERS.TEST.SAV                            375   62   1  3390
```

Specifying a data set list for migration

You can specify a list of data sets to be migrated. After getting the list, this utility obtains data set information on each data set. The list is examined and it determines that it cannot migrate data sets if they meet any of the following conditions:

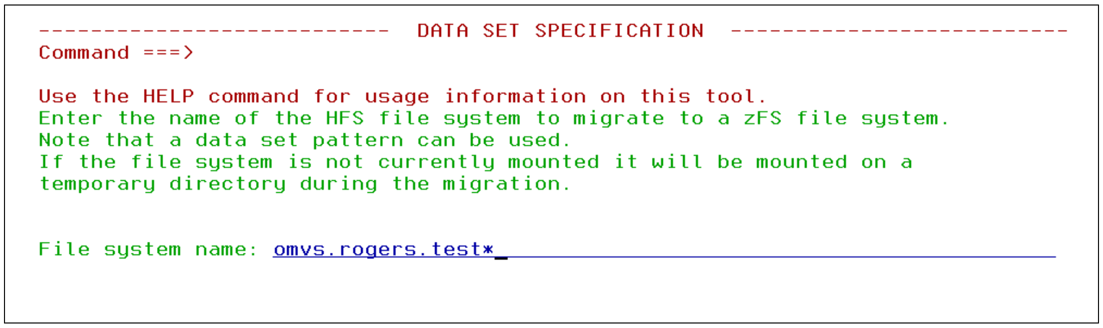
- They do not exist.
- They are HSM-migrated.
- They are not HFS data sets.

Data set list example

In this example, the HFS data sets are:

```
OMVS.ROGERS.TEST1
OMVS.ROGERS.TEST2
OMVS.ROGERS.TEST3
OMVS.ROGERS.TEST4
OMVS.ROGERS.TEST5
OMVS.ROGERS.TEST6
OMVS.ROGERS.TEST7
```

To specify this list to the migration tool, see Figure 11-14.



```
---------------------------  DATA SET SPECIFICATION  --------------------------
Command ===>

Use the HELP command for usage information on this tool.
Enter the name of the HFS file system to migrate to a zFS file system.
Note that a data set pattern can be used.
If the file system is not currently mounted it will be mounted on a
temporary directory during the migration.


File system name: omvs.rogers.test*_
```

Figure 11-14 Migrating a list of data sets

The resulting data set list is presented in a table, shown in Figure 11-15. Use your normal ISPF Up and Down to scroll through the list. Each table entry shows the following:

- The data set names that are used
- Current HFS space utilization and space allocation
- Allocation parameters that are used to create the zFS file system

In Figure 11-15, only the first 3 data sets of the 7 data sets are shown. At the top of the list of data sets are the three options to continue the migration. They are:

- Select items with D to delete items from this migration list.
- Select items with A to alter allocation parameters for the items.
- Enter command FG or BG to begin migration in the foreground or UNIX background, respectively.

```
---------------------------  DATA SET LIST------------------------------ Row 1 to 3 of 8
 Command ===> fg
Use the HELP command for full usage information on this tool
   Select items with D to delete items from this migration list
   Select items with A to alter allocation parameters for the items
   Enter command FG or BG to begin migration in foreground or UNIX background
 ------------------------------------------------------------------------------
 _ HFS data set ..: OMVS.ROGERS.TEST                              Utilized: 62%
   Save HFS as  ..: OMVS.ROGERS.TEST.SAV
   Initial zFS  ..: OMVS.ROGERS.TEST.TMP                          Allocated: N
   HFS space  Primary  : 25        Secondary: 5         Units ..: CYL
   zFS space  Primary  : 25        Secondary: 5         Units ..: CYL
              Dataclas : HFS       Mgmtclas : HFS       Storclas: OPENMVS
   MOUNTED    Volume   : SBOX1F    Vol count: 1
 ------------------------------------------------------------------------------
 _ HFS data set ..: OMVS.ROGERS.TEST1                             Utilized: 62%
   Save HFS as  ..: OMVS.ROGERS.TEST1.SAV
   Initial zFS  ..: OMVS.ROGERS.TEST1.TMP                         Allocated: N
   HFS space  Primary  : 25        Secondary: 5         Units ..: CYL
   zFS space  Primary  : 25        Secondary: 5         Units ..: CYL
              Dataclas : HFS       Mgmtclas : HFS       Storclas: OPENMVS
              Volume   : SBOX1E    Vol count: 1
 ------------------------------------------------------------------------------
 _ HFS data set ..: OMVS.ROGERS.TEST2                             Utilized: 62%
   Save HFS as  ..: OMVS.ROGERS.TEST2.SAV
   Initial zFS  ..: OMVS.ROGERS.TEST2.TMP                         Allocated: N
   HFS space  Primary  : 25        Secondary: 5         Units ..: CYL
   zFS space  Primary  : 25        Secondary: 5         Units ..: CYL
  Dataclas : HFS       Mgmtclas : HFS       Storclas: OPENMVS
             Volume   : SBOX1H    Vol count: 1
------------------------------------------------------------------------------
_ HFS data set ..: OMVS.ROGERS.TEST3                             Utilized: 62%
  Save HFS as  ..: OMVS.ROGERS.TEST3.SAV
  Initial zFS  ..: OMVS.ROGERS.TEST3.TMP                         Allocated: N
  HFS space  Primary  : 25        Secondary: 5         Units ..: CYL
  zFS space  Primary  : 25        Secondary: 5         Units ..: CYL
             Dataclas : HFS       Mgmtclas : HFS       Storclas: OPENMVS
             Volume   : SBOX1D    Vol count: 1
------------------------------------------------------------------------------
```

Figure 11-15 List of data sets to be migrated

Using the **fg** option for migration, shown in Figure 11-15, the migration begins; the resulting messages displayed on the screen are shown in Figure 11-16 on page 264.

```
Migrating OMVS.ROGERS.TEST
creating zFS OMVS.ROGERS.TEST.TMP
copying OMVS.ROGERS.TEST to OMVS.ROGERS.TEST.TMP Blocks to copy: 2832
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IDC0531I ENTRY OMVS.ROGERS.TEST.TMP ALTERED
IDC0531I ENTRY OMVS.ROGERS.TEST.TMP.DATA ALTERED
mount /u/rogers/tmp OMVS.ROGERS.TEST ZFS 0
Migrating OMVS.ROGERS.TEST1
creating zFS OMVS.ROGERS.TEST1.TMP
copying OMVS.ROGERS.TEST1 to OMVS.ROGERS.TEST1.TMP Blocks to copy: 2832
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IDC0531I ENTRY OMVS.ROGERS.TEST1.TMP ALTERED
IDC0531I ENTRY OMVS.ROGERS.TEST1.TMP.DATA ALTERED
Migrating OMVS.ROGERS.TEST2
creating zFS OMVS.ROGERS.TEST2.TMP
copying OMVS.ROGERS.TEST2 to OMVS.ROGERS.TEST2.TMP Blocks to copy: 2832
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IDC0531I ENTRY OMVS.ROGERS.TEST2.TMP ALTERED
IDC0531I ENTRY OMVS.ROGERS.TEST2.TMP.DATA ALTERED
Migrating OMVS.ROGERS.TEST3
creating zFS OMVS.ROGERS.TEST3.TMP
copying OMVS.ROGERS.TEST3 to OMVS.ROGERS.TEST3.TMP Blocks to copy: 2832
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IGD01009I MC ACS GETS CONTROL &ACSENVIR=RENAME
IDC0531I ENTRY OMVS.ROGERS.TEST3.TMP ALTERED
IDC0531I ENTRY OMVS.ROGERS.TEST3.TMP.DATA ALTERED
--------------------------
***
```

Figure 11-16 Messages issued by the migration tool during the migration

Preallocated zFS data sets

In this list of data sets, you can change the allocation attributes for any of the data sets. Since the migration tool allows the preallocation of the zFS data set, this is shown in the following example of a data set list entry in Figure 11-17.

The allocation attributes for this new zFS file system can be changed. If the file system is already allocated, enter `Y` for preallocated (replacing the `N`) and the name of the data set as the temp name. The attributes will not be used since you have preallocated the zFS data set.

```
 File system name . . . OMVS.HFS.ROGERS
 Save HFS as  . . . . . OMVS.HFS.ROGERS.SAV
 Temp name for zFS. . . OMVS.HFS.ROGERS.TMP
 Preallocated zFs . . . N

 Primary allocation . . 120
 Secondary allocation   0
 Allocation units . . . CYL        (CYL or TRK)
 Data class . . . . . . HFS
 Management class . . . IMS
 Storage class  . . . . STANDARD
 Volume . . . . . . . . SBOX1F
```

Figure 11-17 Data set entry for a preallocated zFS data set

11.4.6 Migration summary considerations

Decide whether to have a different naming standard for your zFS data sets than for your HFS data sets. You may decide to keep the names as they are. If the zFS data set names are to be the same as the HFS data set names, it is likely that scripts or policies that cause it to be mounted would not need to change.

If you decide to have a new naming convention, then you must change mount policies and mount scripts (including from the BPXPRMxx parmlib) to mount the correct file system name. You will have to look at your automount policies very carefully. You may need to modify the generic entry, migrate all file systems at one time, or add specific entries for each migrated file system.

Two data sets will exist during the migration. Plan for the availability of space for two file systems of about the same size. During the migration, the data will not be available for read or write. To prevent loss of updates during this time, change the mount mode to read only or do the migration when updates are not being done.

Automount policy migration example

Consider a generic entry that contains a file system parameter similar to the following:

```
OMVS.HFS
```

You can change it to use a pattern that will replace the HFS string with ZFS when appropriate:

```
OMVS.///
```

zFS file systems

zFS file systems that will span volumes must be preallocated prior to invoking the tool.

zFS file systems that are greater than 4 GB (about 4825 cylinders of a 3390) must be defined with a data class that includes extended addressability. zFS file systems cannot be striped.

12

JES2 z/OS V1R8 enhancements

This chapter describes the changes that the system programmer needs to implement in JES2 in z/OS V1R8, as follows:

- Easier diagnostic action during $P JES2 processing
  - How to detail the specific reason(s) that would cause a $P JES2 command to hang
  - Implementation of a new $D JES2 command
  - SCAN updates
  - Migration considerations for the message (HASP003)
  - Migration considerations for the message (HASP607)
  - Implementation of a new message (HASP608)
  - Implementation of a new message (HASP715)
- WLM support
  - WLM batch support
  - Granular controls for a planned IPL
  - Duplicate job name processing
  - Changes in commands $D JOBCLASS(x) and $T JOBCLASS(X)
  - Migration considerations for message HASP637
  - Implementation of a new message (HASP889)
- Migration and coexistence considerations

12.1 $P JES2 processing

For many years, the problems in terminating JES2 cleanly with a $PJES2 command have been requested for resolution. Either the $PJES2 command is rejected or after being accepted it could cause a hang condition, as follows:

- Before z/OS V1R8, if the return jobid was never performed, JES2 would hang in termination processing without any notification to the operator.
- With z/OS V1R8, JES2 now indicates what is found during the termination process.

For instance, when the command is rejected, a minimal amount of information is given for the reason. Prior to z/OS V1R8, the only assistance was the HASP607 message, which provided only high level information, such as PCEs or address spaces. The operator was then left with the job of finding out what systems and address spaces were still active. The operator had to know what commands to issue to find the appropriate information. This could become a long process for the operator if he really needed to cleanly shut down JES2.

12.1.1 z/OS V1R8 enhancements

With JES2 in z/OS V1R8, a new HASP608 message and related $D JES2 command now detail the reasons for a $P JES2 command being rejected. When the $P JES2 command is accepted, there is a chance for JES2 termination processing to hang without any notification. Address spaces that have done a request jobid must perform a return jobid before JES2 can successfully terminate. The new HASP715 message will list address spaces where JES2 is waiting for a return jobid to be performed.

When a $P JES2 command is rejected, a new message, HASP608, is issued as a command response to supplement the existing HASP607 message. In a table format, this message lists specific items causing JES2 to reject the command. This makes the information more easily readable and details the specific reason(s) so that the operator need not issue more commands to figure out what is causing the $P JES2 command to be rejected. The new message listing the reasons is shown in Figure 12-1.

```
$pjes2
$HASP608 $PJES2
$HASP608 ACTIVE PROCESSORS (PCES)
$HASP608 NAME                              COUNT JOBID
$HASP608 -------------------------------- ------ --------
$HASP608 JCL CONVERTER                         1 STC00016
$HASP608 PRT1                                  1 STC00004
$HASP608 ACTIVE ADDRESS SPACES
$HASP608 ASID     JOBNAME  JOBID
$HASP608 -------- -------- --------
$HASP608 001D     ZFS      STC00003
$HASP608 0022     BPXAS    STC00009
$HASP608 0027     IBMUSER  TSU00015
$HASP608 PROCESSOR(S) ENDED BY PRIOR RECOVERY
         *$HASP623 MEMBER DRAINING
         *$HASP607 JES2 NOT DORMANT -- MEMBER DRAINING,
           RC=03 ACTIVE PROCESSORS
           RC=08 PROCESSOR(S) ENDED BY PRIOR RECOVERY.
             $PJES2,ABEND IS REQUIRED TO TERMINATE JES2
           RC=10 ACTIVE ADDRESS SPACES
```

Figure 12-1 $P JES2 command showing shutdown messages

Note: Unlike the existing HASP607 message when the $P JES2 command is rejected, the HASP608 message is *not* refreshed with updated information.

12.1.2 $D JES2 command

A new display command to list all reasons why a $P JES2 command would be rejected is very beneficial. As a result of doing the $P JES2 work, a new command ($D JES2) was implemented to refresh the message HASP608. This command displays the same information as the HASP608 message resulting from a $P JES2 command. This allows the operator to determine what might cause the $P JES2 command to be rejected before trying a $P JES2 command. If there is no activity to report, an alternate form of the HASP608 is displayed with the text:

```
$HASP608 ALL AVAILABLE FUNCTIONS COMPLETE.
```

Figure 12-2 shows that there are various PCEs and address spaces active. When applicable, the PCE's current job is displayed. Also, in this display it can be seen that networking is active on this member, which is using a LINE. There is clarification of the reason when JES2 will not cleanly terminate and avoidance of problems with running processes that are hanging JES2.

```
$djes2
 $HASP608 $DJES2
 $HASP608 ACTIVE PROCESSORS (PCES)
 $HASP608 NAME                             COUNT JOBID
 $HASP608 -------------------------------- ------ --------
 $HASP608 PRT1                                 1 STC00004
 $HASP608 ACTIVE ADDRESS SPACES
 $HASP608 ASID     JOBNAME  JOBID
 $HASP608 -------- -------- --------
 $HASP608 001D     ZFS      STC00003
 $HASP608 0022     BPXAS    STC00009
 $HASP608 0025     TCAS     STC00014
 $HASP608 0026     VTAM     STC00013
 $HASP608 0027     IBMUSER  TSU00015
 $HASP608 0050     BECKER1  JOB02468
 $HASP608 0086     BECKER   TSU02467
 $HASP608 ACTIVE NETWORKING DEVICES
 $HASP608 NAME                             STATUS
 $HASP608 -------------------------------- --------
 $HASP608 LINE15                           ACTIVE
 $HASP608 LINE16                           ACTIVE
 $HASP608 NETSRV1                          ACTIVE
 $HASP608 OUTSTANDING CROSS MEMBER REQUESTS
 $HASP608 ASID     JOBNAME  JOBID     COUNT
 $HASP608 -------- -------- -------- ------
 $HASP608 0048     PELEG    JOB04039     37
```

Figure 12-2 $D JES2 command showing active functions

$D JES2 display information

The $D JES2 command displays the current activity in the JES2 address space. The information displayed corresponds to the specific reasons provided in the $HASP607 message when a $PJES2 command is rejected. The following is the list of reasons:

- Outstanding I/O
- Active processors

- Active address spaces
- Active networking devices
- Outstanding held processors
- Allocated internal readers
- Outstanding cross member requests
- Outstanding PSO, SAPI, or end of memory activity

Successful $P JES2

When there is no activity to report, the message returned is shown in Figure 12-3.

```
$P JES2
$HASP608 $PJES2 COMMAND ACCEPTED
$HASP314 INIT 1    DRAINED  ******** C=ABCDEFG12
$HASP314 INIT 2    DRAINED  ******** C=AB
......
$HASP395 INIT     ENDED
$HASP395 INIT     ENDED
$HASP9085 JES2 MONITOR ADDRESS SPACE STOPPED FOR JES2
$HASP085 JES2 TERMINATION COMPLETE
```

Figure 12-3 $P JES2 command that is successful

$P JES2 hangs and the HASP715 message

Whenever something hangs without any external notifications, this can be very upsetting for the operator, as in previous releases. However, no external notification is given to the operator when JES2 finds one or more such address spaces that do not perform a return jobid in a reasonable period of time.

With z/OS V1R8, a new HASP715 message has been created to list up to 5 such address spaces. The message is refreshed regularly until all address spaces have done their return jobid. At that point, JES2 termination processing continues.

The process after a $P JES2 command is as follows:

- All request jobid address spaces must perform a return jobid before JES2 termination may proceed.
- If a return jobid is not done, then JES2 termination will hang.
- A HASP715 is issued with a list of request jobid address spaces that still need to perform a return jobid.
- The HASP715 message lists in table format up to 5 such address spaces with ASID and JOBNAME, and this message is refreshed periodically until no more request jobid address spaces remain, allowing JES2 termination processing to proceed.

Figure 12-4 on page 271 shows the command response to a $P JES2 command when there are jobid address spaces remaining. In this example, there are two address spaces waiting for a return jobid when the $P JES2 command is issued. After one of these address spaces does the return jobid, you can see the next update of the HASP715 message displays only one address space. After this address space does the return jobid, JES2 termination continues and completes. Also, note that the HASP608 message indicating the $P JES2 command was accepted.

```
$P JES2
 $HASP608 $PJES2 COMMAND ACCEPTED
 $HASP314 INIT 1    DRAINED  ******** C=A
 $HASP395 INIT     ENDED
*$HASP715 ADDRESS SPACES WAITING FOR RETURN JOBID
 ASID     JOBNAME
 -------- --------
 0022     RUNIT
 0023     RUNIT
 9 SUPPRESSED
 IEE600I REPLY TO 09 IS;SUPPRESSED
*$HASP715 ADDRESS SPACES WAITING FOR RETURN JOBID
 ASID     JOBNAME
 -------- --------
 0023     RUNIT
 14 SUPPRESSED
 IEE600I REPLY TO 14 IS;SUPPRESSED
 $HASP085 JES2 TERMINATION COMPLETE
```

Figure 12-4 Notice that the HASP715 message is being refreshed periodically

12.1.3 SCAN updates

To provide the data in the HASP608 and HASP715 messages in table format, SCAN processing was changed significantly. Line type capability was added to the out-of-line areas, and the concept of width for the output display was added to column alignment. An unrelated change was also made to add the command verb to the HASP003 message.

SCAN has the following changes and enhancements:

- There are related SCAN updates to display the data in the new messages in table format.
- There are five possible values for Line types:
  - C – control line
  - L – label line
  - D – data line
  - DE – data and end line
  - E – end line
- On $SCANTAB, line type is the second positional for CRLF. Only valid with CRLF=YES.
- On $SCAND, line type is the second positional for BRKOPT. Only valid with BRKOPT=CRLF.
- Width (for output display)
  - To preserve column alignment, a WIDTH keyword has been added to $SCANTAB.
  - Accepts a value of 1-32.
  - Indicates the width of the output display for the value being displayed by the $SCANTAB.
    - CONV=HEX (left justified)
    - CONV=NUM (right justified)
    - CONV=CHAR (either)

Migration considerations for message HASP003

As a result of the SCAN updates, the HASP003 message is changed to add the command verb to it, as shown in Figure 12-5 on page 272 where the command verb is added to the

HASP003 message as command response to invalid commands. This allows for easier determination of which command had the error.

```
$ADD CONNECT,FJB
$HASP003 RC=(03),ADD CONNECT FJB  - INVALID PARAMETER STATEMENT

$D JES2,FJB
$HASP003 RC=(03),D JES2 FJB  - INVALID PARAMETER STATEMENT
```

Figure 12-5 Command response to an invalid parameter

Migration considerations for message HASP607

The existing HASP607 message has been changed slightly as a result of the HASP608 message. This was done to coincide with the sections of the HASP608 message since PCEs, address spaces, and networking devices needed different data to be displayed.

The RC 03 ACTIVE PROCESSORS OR ADDRESS SPACES reason code has been separated into 3 different reason codes. Automation that examined the message HASP607 may need updating, because its return codes were being changed significantly as follows:

RC 03 – ACTIVE PROCESSORS OR ADDRESS SPACES is now split out into the following reason codes:

- RC 03 – ACTIVE PROCESSORS
- RC 10 – ACTIVE ADDRESS SPACES
- RC 11 – ACTIVE NETWORKING DEVICES

Also changing is the RC 06 – OUTSTANDING CROSS MEMBER REQUESTS. This reason code actually did something similar to the goal of the HASP608 message. It displayed address space information about these requests. This information has been deleted from the HASP607 message and is now contained in the HASP608 message. RC 06 – OUTSTANDING CROSS MEMBER REQUESTS text has been updated as follows:

The second line detailing ASID and JOBNAME has been deleted (information now provided in HASP608 message. Figure 12-6 shows the command response to a $P JES2 command when there are several reasons for an JES2 shutdown process hang.

```
$P JES2
HASP608 $PJES2 785
HASP608 ACTIVE NETWORKING DEVICES
HASP608 NAME                           STATUS
HASP608 ------------------------------ --------
HASP608 LINE6                          ACTIVE
HASP608 LINE11                         ACTIVE
HASP608 NETSRV1                        ACTIVE
HASP623 MEMBER DRAINING
HASP607 JES2 NOT DORMANT -- MEMBER DRAINING, 786
          RC=11 ACTIVE NETWORKING DEVICES
....
HASP607 JES2 NOT DORMANT -- MEMBER DRAINING, 800
          RC=02 OUTSTANDING WTO ACTIVITY
          RC=11 ACTIVE NETWORKING DEVICES
....
HASP607 JES2 NOT DORMANT -- MEMBER DRAINING, 810
          RC=11 ACTIVE NETWORKING DEVICES
```

Figure 12-6 Notice that the HASP607 message is being refreshed periodically

12.2 WLM support

The objective of this enhancement is to defer job selection for newly arriving work until it can be determined which member is most in need of work in terms of idle initiators. Basically, the algorithm determines how many WLM-managed batch jobs could be running (this means already executing jobs and jobs awaiting execution) and thus how many initiators would be required. If more initiators are available than jobs to run, then the percentage of busy initiators is computed. That percentage is applied to the initiators available on each system to determine the goal for that member. There is no change to WLM policies or algorithms.

z/OS V1R8 enhancements

The following changes are implemented in this release:

- An internal algorithm is implemented to balance initiators across the JESplex on a percentage basis.

  In this release, there is a new specification available on the JOBCLASS statement specifications call QAFF. QAFF stands for Queue AFFinity. Members now select from a given jobclass if and only if that member is part of the QAFF affinity mask. If the installation wants to no longer select work from JOBCLASS X on member SC75, the operator command would be $T JOBCLASS(X),QAFF=-SC75.

  In addition, JOBCLASSes can be assigned a maximum execution value on a member by member basis. For example if the maximum number of jobs that can run on member SC75 in JOBCLASS Z is to be 3, then $T JOBCLASS(Z),XEQMEMBER(SC75)=MAX=3. QAFF overrides this maximum count. If member SC75 is removed from the QAFF for class Z, then no class Z jobs are selected regardless of the max value.

  Service classes can also be controlled via QAFF. If service class LONG is no longer to be selected on members SC75 and SC74, $T SRVCLASS(LONG),QAFF=(-SC75,-SC74) is entered.

- Handling planned outages with WLM managed workloads.

  Planned outages can be managed better if the installation can remove certain kinds of workloads from a system long before the planned outage time. For example, if service class “large” consists of jobs which normally take three hours to run, it would be nice to not select work in the “large” service class three hours before the IPL, but to have no effect upon other work loads on the system.

- Duplicate job name processing.

  The SMF 26 record indicator that says “job held at least once during its life because of duplicate job processing” will always be on if the job was a duplicate sometime in its life.

  In prior releases, the indicator was set only if the job selection process rejected the job because it duplicated an already running job.

  Jobs that are duplicate jobs will be counted as “held” whenever a member of the “family” is executing, not just when selection failed. WLM queue delay time will be lower, which might in extreme cases reduce the number of initiators WLM believes are necessary.

To implement these changes, the following are introduced:

- Commands $T JOBCLASS(X) and $D JOBCLASS(x) were changed and the message (HASP867) has additional information.
- Five new commands ($D/$T/$ADD/$S/$P SRVCLASS(x)) and a new message (HASP889) were implemented.

Notice that these changes have the following prerequisites:

- None of these problems are addressed in a mixed-MAS with JES2 levels that are not all z/OS V1R8).
- The supporting code is disabled until the MAS becomes pure z/OS V1R8.
- The support is enabled as long as all members are z/OS V1R8 (or later), but is immediately disabled again when the MAS becomes “impure.”

There are no cold starts or any other kind of special procedures needed to implement this support.

12.2.1 The WLM batch support

The new support is to defer job selection for newly arriving work until it can be determined which member is most in need of work in terms of idle initiators. The algorithm determines how many WLM-managed batch jobs could be running (this means already executing jobs and jobs awaiting execution) and thus how many initiators would be required.

If more initiators are available than jobs to run, then the percentage of busy initiators is computed. That percentage is applied to the initiators available on each system to determine the “goal.”

There is no change to WLM policies or algorithms.

12.2.2 Granular controls for a planned IPL

There is a new specification available on the JOBCLASS initialization statement called QAFF. QAFF stands for Queue AFFinity. Members will select from a given jobclass if and only if that member is part of the QAFF affinity mask, as follows:

- To select work only from JOBCLASS X on member SYSA, the operator command would be $T JOBCLASS(X),QAFF=-SYSA.
- JOBCLASSes can be assigned a maximum execution value on a member by member basis such as:

  - $T JOBCLASS(Z),XEQMEMBER(SYSA)=MAX=3 limits to 3 the maximum number of jobs that can run on member SYSA in JOBCLASS Z.
  - QAFF overrides this maximum count and if member SYSA is removed from the QAFF for class Z, then no class Z jobs will be selected regardless of the max value.

Service classes can also be controlled via QAFF, as follows:

- To no longer select service class BATCHL on members SYSB and SYSD, a $T SRVCLASS(BATCHL),QAFF=(-SYSB,-SYSD) must be entered.

12.2.3 The duplicate job name processing

In the previous versions of JES2 the technique used was as follows:

- To mark a job as a duplicate whenever it was tentatively selected to execute if it was determined at that time that a job by the same name was already executing.
- When a job finished executing, it was determined whether other jobs with the same name were awaiting execution and if so, they were released.
- And the whole process was done over when the next duplicate name situation was discovered.
- If there are many jobs with the same name, a lot of time was spent marking and unmarking jobs.

z/OS V1R8 enhancements

z/OS V1R8 provides new techniques to solve problems, as follows:

- Duplicate jobname jobs are made a member of a family, one family for each duplicated jobname.
- As jobs enter and leave execution, the family control is updated, but no job is altered. This eliminates queue searching and removes all but a small amount of checkpoint activity.
- Jobs that are duplicates always have the indicator in the SMF 26 record (SMF26JDL) set even if the job was never selected and then rejected for execution.
- Hold times because of being a duplicate name are accumulated even if the job was never selected and then rejected for execution. The only effect of this change is that WLM will now have a more accurate picture of “queue hold” time.
- Jobs are identified as duplicates and grouped into a duplicate job family when originally added to the queue.
- A family attribute is modified as a family member begins and ends execution. No jobs are directly modified.

This form of duplicate jobname support does not take effect until the JESplex is all z/OS V1R8 or later, and the good news is:

- Superior performance when an installation has a large number of duplicate job names
- Better accounting of hold time for WLM goal processing
- Dollars saved in some IBM pricing schemes
- More efficient system shutdown

12.2.4 Commands related to WLM support

The JOBCLASS management is via the $T JOBCLASS and $D JOBCLASS commands, and via the JOBCLASS initialization statement. There are two new keywords:

XEQMEMBER	Used to set a maximum number of jobs that can execute in the job class on a member-by-member basis
QAFF	Used to specify the members of the JESplex that have affinity to the queue, the members that will select from the jobclass.

Figure 12-7 shows the use of the two new keywords:

- $D JOBCLASS(X), that shows the status of the JOBCLASS
- $T JOBCLASS(X),QAFF=-SYSA, that changes its affinity

```
$D JOBCLASS(X)
$HASP837 JOBCLASS(X) 930
$HASP837 JOBCLASS(X)         MODE=JES,QAFF=(ANY),QHELD=NO,
$HASP837                     SCHENV=,XEQCOUNT=(MAXIMUM=*,
$HASP837                     CURRENT=0),
$HASP837                     XEQMEMBER(SYSA)=(MAXIMUM=*,
$HASP837                     CURRENT=0)
....
$T JOBCLASS(X),QAFF=-SYSA
$HASP837 JOBCLASS(X) 933
$HASP837 JOBCLASS(X)         MODE=JES,
$HASP837                     QAFF=(ONLY UNDEFINED MEMBERS),
$HASP837                     QHELD=NO,SCHENV=,
$HASP837                     XEQCOUNT=(MAXIMUM=*,CURRENT=0),
$HASP837                     XEQMEMBER(SYSA)=(MAXIMUM=*,
$HASP837                     CURRENT=0)
```

Figure 12-7 Command responses to $D JOBCLASS and $T JOBCLASS

$D JOBCLASS enhancements

The command $D JOBCLASS(x) output—HASP837—has additional information. As shown in Figure 12-8, the command $D JOBCLASS shows the QAFF value as well as the MAX and CURRENT number of jobs running in the jobclass on each member.

```
$D JOBCLASS(X)
$HASP837 JOBCLASS(X) 924
$HASP837 JOBCLASS(X)         MODE=JES,QAFF=(ANY),QHELD=NO,
$HASP837                     SCHENV=,XEQCOUNT=(MAXIMUM=*,
$HASP837                     CURRENT=0),
$HASP837                     XEQMEMBER(SYSA)=(MAXIMUM=*,
$HASP837                     CURRENT=0)
```

Figure 12-8 Command responses to $D JOBCLASS

12.2.5 New commands for WLM support

There are some new commands to create a JES2 status for service classes. Also, a new message, HASP889, is introduced with the new commands. JES2 maintains two types of service classes, as follows:

- Permanent service classes

  Permanent service classes are created with either the $ADD or $T SRVCLASS command. JES2 permanent service classes are not deleted unless the corresponding WLM service class is deleted.

- Dynamic service classes

  Dynamic service classes are created when a job enters the system and the service class assigned to that job is not currently known. Dynamic service classes are deleted when there are no longer any pre-execution jobs in that class and a timer has expired.

$ADD SRVCLASS command

The $ADD SRVCLASS command adds a new permanent service class element that corresponds to a WLM service class. The class specified cannot match the class name of any existing JES2 SRVCLASS statement, no matter whether permanent or dynamic. The $ADD SRVCLASS command, as shown in Figure 12-9, adds a new permanent service class element.

```
$ADD SRVCLASS(SYSOTHER)
$HASP889 SRVCLASS(SYSOTHER) 014
$HASP889 SRVCLASS(SYSOTHER)  QAFF=(ANY),TYPE=PERMANENT,
$HASP889                     MASCOUNT=(INITS=0,ACTIVE=0)
```

Figure 12-9 Command responses to $ADD SRVCLASS

$D SRVCLASS command

The $D SRVCLASS command displays JES2 information for the specified service class, as shown in Figure 12-10.

```
$D SRVCLASS(SYSOTHER)
$HASP889 SRVCLASS(SYSOTHER) 020
$HASP889 SRVCLASS(SYSOTHER)  QAFF=(ANY),TYPE=PERMANENT,
$HASP889                     MASCOUNT=(INITS=0,ACTIVE=0)
```

Figure 12-10 Shows the command responses to $D SRVCLASS

If the keyword LONG is specified on the command, as shown in Figure 12-11, it shows, in addition, the internal active job goal. The counts and goals are on a member-by-member basis as well as JESplex values for initiators and batch job counts.

```
$D SRVCLASS(SYSOTHER),LONG
$HASP889 SRVCLASS(SYSOTHER) 023
$HASP889 SRVCLASS(SYSOTHER)  QAFF=(ANY),TYPE=PERMANENT,
$HASP889                     COUNTS(SC75)=(INITS=0,ACTIVE=0,
$HASP889                     TARGET=*),MASCOUNT=(INITS=0,
$HASP889                     ACTIVE=0)
```

Figure 12-11 $D SRVCLASS when LONG was specified

$P SRVCLASS command

The $P SRVCLASS command drains JES2 processing for a given service class on the member where the command is issued. This command is equivalent to $T SRVCLASS(class),QAFF=(-*). The command is shown in Figure 12-12.

```
$P SRVCLASS(SYSOTHER)
$HASP889 SRVCLASS(SYSOTHER)  QAFF=(NONE),TYPE=PERMANENT
```

Figure 12-12 Shows the command responses to $P SRVCLASS

$S SRVCLASS command

The $S SRVCLASS command starts JES2 processing for a given service class on the member where the command is issued. The command is equivalent to the $T SRVCLASS(class),QAFF=(+*), as shown in Figure 12-13 on page 278.

```
$S SRVCLASS(SYSOTHER)
$HASP889 SRVCLASS(SYSOTHER)  QAFF=(SC75),TYPE=PERMANENT
```

Figure 12-13 Shows the command responses to $S SRVCLASS

$T SRVCLASS command

The $T SRVCLASS command modifies an existing permanent or dynamic JES2 service class element, as shown in Figure 12-14.

```
$T SRVCLASS(SYSOTHER),QAFF=-SC75
$HASP889 SRVCLASS(SYSOTHER) 058
$HASP889 SRVCLASS(SYSOTHER)  QAFF=(ONLY UNDEFINED MEMBERS),
$HASP889                     TYPE=PERMANENT,MASCOUNT=(INITS=0,
$HASP889                     ACTIVE=0)
....
$T SRVCLASS(SYSOTHER),QAFF=SC75
$HASP889 SRVCLASS(SYSOTHER) 060
$HASP889 SRVCLASS(SYSOTHER)  QAFF=(SC75),TYPE=PERMANENT,
$HASP889                     MASCOUNT=(INITS=0,ACTIVE=0)
```

Figure 12-14 Command responses to $T SRVCLASS

$D JOBCLASS command

Additional information is displayed on the existing $D JOBCLASS command, as shown in Figure 12-16 on page 279 and Figure 12-15. The QAFF value as well as the MAX and CURRENT number of jobs running in the jobclass on each member are new keywords.

```
$D JOBCLASS(X)
$HASP837 JOBCLASS(X) 071
$HASP837 JOBCLASS(X)         MODE=JES,
$HASP837                     QAFF=(ONLY UNDEFINED MEMBERS),
$HASP837                     QHELD=NO,SCHENV=,
$HASP837                     XEQCOUNT=(MAXIMUM=*,CURRENT=0),
$HASP837                     XEQMEMBER(SC75)=(MAXIMUM=*,
$HASP837                     CURRENT=0)
```

Figure 12-15 Command responses to $D JOBCLASS

```
$D JOBCLASS(X),LONG
$HASP837 JOBCLASS(X) 073
$HASP837 JOBCLASS(X)        ACCT=NO,AUTH=(ALL),BLP=YES,
$HASP837                    COMMAND=DISPLAY,COPY=NO,
$HASP837                    DUPL_JOB=DELAY,HOLD=NO,IEFUJP=YES,
$HASP837                    IEFUSO=YES,JESLOG=(NOSPIN),
$HASP837                    JOURNAL=YES,LOG=YES,MODE=JES,
$HASP837                    MSGLEVEL=(1,1),OUTDISP=(,),
$HASP837                    OUTPUT=YES,PERFORM=000,PGMRNAME=NO,
$HASP837                    PROCLIB=00,
$HASP837                    QAFF=(ONLY UNDEFINED MEMBERS),
$HASP837                    QHELD=NO,RESTART=YES,REGION=0002M,
$HASP837                    SCAN=NO,SCHENV=,SWA=ABOVE,
$HASP837                    TIME=(000450,00),TYPE26=YES,
$HASP837                    TYPE6=YES,XBM=,XEQCOUNT=(MAXIMUM=*,
$HASP837                    CURRENT=0),
$HASP837                    XEQMEMBER(SC75)=(MAXIMUM=*,
$HASP837                    CURRENT=0)
```

Figure 12-16 Command responses to $D JOBCLASS (LONG)

12.2.6 Migration and coexistence considerations

These enhancements can coexist in a MAS with HJE7707, HJE7708, and HJE7720, as follows:

- APAR OA12922 required on down-level members.
- The PTF numbers are:
  - HJE7707 – UA24982
  - HJE7708 – UA24983
  - HJE7720 – UA24984

13

JES3 z/OS V1R8 enhancements

This chapter describes JES3 enhancements introduced in z/OS V1R8. The following subjects are covered:

- TCP/IP protocol support for NJE nodes.
- Support for read-only SYSOUT application program interface (SAPI) calls.

13.1 Network job entry (NJE) over TCP/IP

Prior to z/OS V1R8, the network job entry (NJE) function in JES3 was limited to the binary synchronous (BSC) and system networking architecture (SNA) protocols. Both these communication protocols require dependent hardware which is, or soon will be, out of service. There are several solutions that provide SNA connectivity over TCP/IP (such as Enterprise Extender). However, to compare to a pure TCP/IP network, these solutions suffer from performance and interoperability problems, and are usually harder to maintain.

Starting with z/OS V1R8, JES3 provides support for NJE over TCP/IP. JES2 has been providing NJE over TCP/IP support since z/OS V1R7. VM (RSCS), VSE, and AS/400® have been providing NJE over TCP/IP support for several releases. JES3 on z/OS V1R8 is now able to communicate with JES2, VM, VSE, AS/400, or any other applicable component, using NJE over TCP/IP.

13.1.1 NJE basics

NJE provides the installation with the ability to:

- Send a job to another node for execution.
- Receive a job from another node either to execute or to send to yet another node for execution.
- Send or receive SYSOUT data produced from a job.
- Reroute jobs or SYSOUT data to another node.
- Monitor data sent to or from your node, using installation exit routines.

To make your JES3 complex part of an NJE environment, you must define the network using JES3 initialization statements.

Nodes

Each participant in the JES3 job entry network is called a *node*. A node is called either a *home node* or a *remote node*. What you call a particular node depends on your point of reference. The view from any given node is:

- That node is the home node.
- Any other node is a remote node.

There are two kinds of remote nodes: directly-connected and indirectly-connected. A directly-connected remote node is adjacent to your node, and is connected through direct links. An indirectly-connected remote node is not adjacent to your node, but is connected to your node through one or more other nodes.

Figure 13-1 on page 283 shows a simple NJE environment consisting of 3 nodes. The point of view of a system programmer at Node A is that:

- Node A is the home node.
- Node B is a directly-connected remote node.
- Node C is an indirectly-connected remote node.

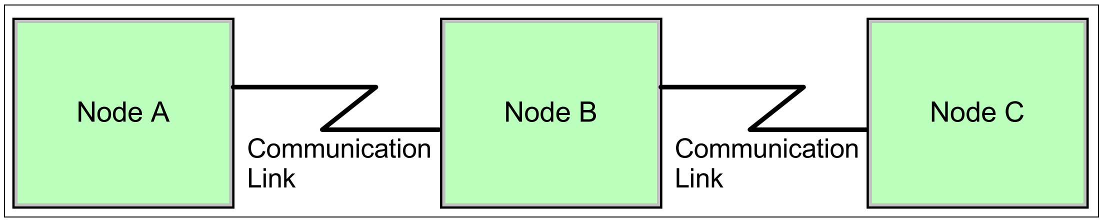


Figure 13-1 A simple NJE networking environment

Networking protocols

When defining a node to JES3, you must specify the type of networking protocol that JES3 is to use for this node. JES3 supports three networking protocols:

- Binary synchronous (BSC)

  BSC protocol allows data flow between nodes over physical links, which are typically channel-to-channel (CTC) adapters, or leased or dial-up telephone lines. BSC is a form of line control that uses a standard set of control characters and control character sequences for transmission of binary coded data between systems in a network. A BSC node uses remote terminal access method (RTAM), emulation program (EP) or partitioned emulation program (PEP) running in the communications controller to control the communication with other nodes.

- System networking architecture (SNA)

  SNA protocol provides a networking capability for JES3 that works in combination with MVS/Bulk Data Transfer (MVS/BDT) Version 2. Networking is established between nodes through MVS/BDT "sessions." Sessions can be established between CTC adapters, telephone lines, microwave links, or by satellite, and are controlled by VTAM. SNA is the description of logical structures, formats, protocols, and operational sequences, for transmitting information units through, and controlling the configuration of, networks. An SNA node uses VTAM, network control program (NCP) or partitioned emulation program (PEP) extension running in the communications controller to control the sessions with other nodes.

- Transmission control protocol/internet protocol (TCP/IP)

  This is a new function of JES3 provided with z/OS V1R8. It is discussed in detail in 13.1.2, "TCP/IP support for NJE nodes" on page 284.

Note: All three protocols can coexist in the same NJE networking environment. A user submitting an NJE job is not aware of whether JES3 is using BSC, SNA, or TCP/IP.

Using NJE to send a SYSOUT to a remote node

Figure 13-2 on page 284 shows an example of how to use NJE to send a SYSOUT to a remote node. When job PELEGJ1 completes its execution, the data in SYSUT2 is sent to node WTSC75.

```
//PELEGJ1  JOB ACCNT#,MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*
//STEP1    EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD *
 THIS DATA WILL BE SENT TO NODE WTSC75
/*
//SYSUT2   DD SYSOUT=*,DEST=WTSC75
//
```

Figure 13-2 Using NJE to send a SYSOUT to a remote node

Using NJE to submit a job for execution on a remote node

Figure 13-3 on page 284 shows an example of how to use NJE to submit a job for execution on a remote node. When job PELEGJ2 is submitted, JES3 recognizes the ROUTE XEQ statement and submits the job PELEGJ3 for execution on node WTSC75.

```
//PELEGJ2  JOB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*ROUTE XEQ WTSC75
//PELEGJ3  NJB ACCNT#,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//STEP1    EXEC PGM=IEFBR14
//
```

Figure 13-3 Using NJE to submit a job for execution on a remote node

13.1.2 TCP/IP support for NJE nodes

BSC and SNA nodes are dependent on hardware equipment such as the ESCON® CTC adapters or the IBM 3705 communications controller. TCP/IP uses a layered stack structure and therefore has the advantage of being hardware independent. This allows the installation to implement an NJE networking environment over existing TCP/IP networks using hardware protocols such as ethernet and token ring. NJE over TCP/IP also supports IPv6 and TLS.

NJE over TCP/IP can be thought of as a hybrid between BSC and SNA. NJE over TCP/IP uses the same record structure and data streams as BSC NJE. But like SNA NJE, the NJE communication is driven through a separate NETSERV address space that is analogous, although not functionally identical, to JES3 BDT.

The NETSERV address space

NETSERV is a common JES component used by both JES2 and JES3. The role of the NETSERV address space is to make all the communication interface calls to TCP/IP on behalf of JES3. NETSERV listens for connections, incoming and outgoing data and cooperates with JES3 to read data from the spool or write data to the spool.

The NETSERV address space can run either on the global or a local. There is no limit to the number of NETSERV address spaces that can run in the JES3 complex at a single time. You specify the system on which a NETSERV is to run when you define it to JES3.

A NETSERV is a started task that requires authority to communicate with JES in order to spool and despool jobs, and data sets. NETSERV also uses the common NJE component, IAZNJTCP, which uses TCP/IP services, and then it must operate in the z/OS UNIX System

Services environment. Therefore, NETSERV must have the proper authority to function. The required authority consists of:

- A definition in the STARTED class to associate the started task with a user ID.

  It is suggested that you associate NETSERV started tasks with the same userid that you associate with JES3. Although you can pick a different id with no impact on security, it will be simpler for you to use the JES3 userid.

  ```
  RDEFINE STARTED JES3*.* STDATA( USER(JES3ID) GROUP(SYS1) TRUSTED(YES))
  ```

- An OMVS segment in the security definition for the user ID associated with the started task.

  ```
  ALTUSER JES3ID OMVS(UID(0) HOME('/') PROGRAM('/'))
  ```

Figure 13-4 on page 285 shows the relationship between JES3, NETSERV, and TCP/IP.

Note: The NETSERV address space must be defined to the security product STARTED class. To minimize the number of STARTED profiles, it is suggested that you define all of your NETSERVs with a common name pattern so that you can cover them all with one generic profile.

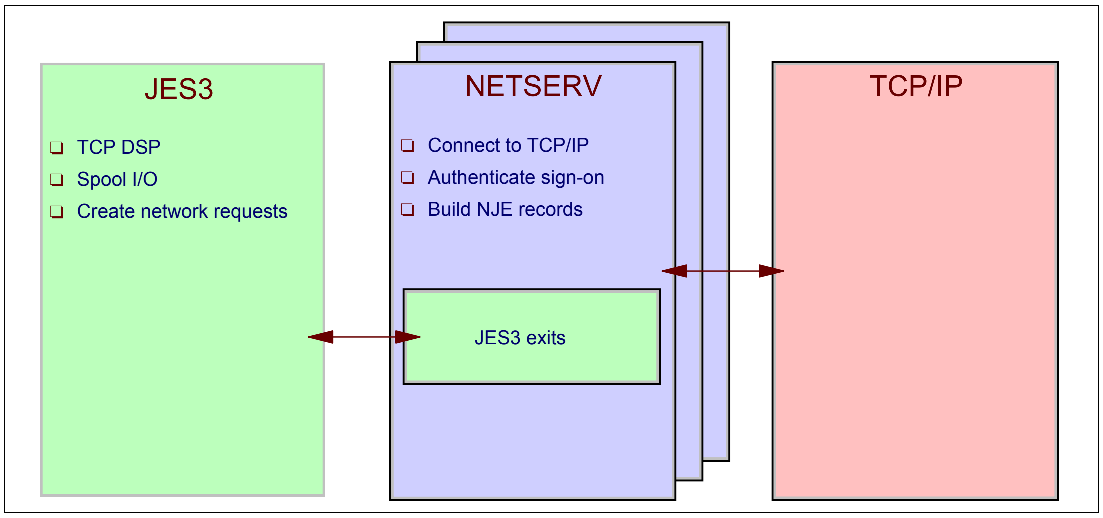


Figure 13-4 The relationship between JES3, NETSERV, and TCP/IP

The name of the NETSERV address space can be any address space name allowed in z/OS. It is your responsibility to make sure you have the proper RACF STARTED class definitions in place for this address space name to run in the system.

Sockets

Under TCP/IP, information is communicated under a logical entity known as a *socket*. To JES3, a socket describes a task under NETSERV that communicates with either another NETSERV (if the remote node is JES2 or JES3) or a VM (RSCS) or VSE socket. The socket is described by the NETSERV it runs under, the node it connects to and the hostname and port that it communicates with.

In order for two nodes to communicate using the TCP/IP protocol, a socket must be started on one node. This node is referred to as a client. The corresponding node is referred to as a server. If JES3 is a server node, a server socket will be defined automatically. If JES3 is a client node, a socket must be define using the SOCKET initialization statement or JES3 commands.

13.1.3 Prerequisites for TCP/IP networking

In order for a node to use the TCP/IP protocol, the following software requirements must be met:

- The home node must be z/OS V1R8 JES3 (level HJS7730) on the global. In addition, any local processor on which a NETSERV is to run must also be at level HJS7730.
- The remote node must be NJE over TCP/IP-capable. For a JES3 remote node, JES3 must be at level HJS7730.
- Any JES3 processor running a NETSERV address space must also run:
  - Language Environment
  - z/OS UNIX System Services
  - VTAM
  - TCP/IP stack
- TCP/IP must be configured on both the home node and the remote node to allow TCP/IP communications between them. JES3 NJE over TCP/IP definitions make the assumption that the TCP/IP connections are in place and correct.
- If TLS is used to secure the communications between two nodes, the proper TCP/IP and key ring definitions must also be in place. For further information on TLS definitions under TCP/IP see *z/OS Communications Server: IP Configuration Guide*, SC31-8775.

13.1.4 JES3 initialization statements

To support NJE over TCP/IP, JES3 introduces changes to the NJERMT initialization statement and adds two new initialization statements: NETSERV and SOCKET. All parameters of the changed and new initialization statements can be added or modified using cold start, warm start, or hot start with refresh. However, the *F CONFIG command does not support the addition of any of these statements. Instead, new commands to modify NETSERV and SOCKET are introduced. The new commands are discussed in "JES3 commands for TCP/IP NJE" on page 290.

Important: Since the NJERMT statement is an existing statement, it will not use the parameter validation procedures introduced in z/OS V1R4. Therefore, if the syntax rules for NJERMT keywords are violated, initialization will fail, as opposed to the NETSERV and SOCKET statements, where initialization will continue with defaults applied.

Changes to the NJERMT statement

The TYPE=TCPIP parameter defines the node as an NJE over TCP/IP node. These new parameters on this statement are used as follows:

- JOBTRANS, OUTTRANS, JOBRECV, and OUTRECV define the number of transmitters and receivers for the node. Each of these is a number between 1 to 7. According to the NJE protocol, the sum of the transmitters can be a maximum of 8, and likewise for the sum of receivers.
- SECSIGNON=YES indicates that nodes signing on to each other verify their identity using session keys defined to the security product's APPCLU class. It is equivalent to JES2's SIGNON=SECURE definition. Unlike JES2, JES3 allows SECSIGNON=YES only for TYPE=TCPIP nodes.

- TLS=YES indicates that the TCP/IP Transport Layer Secure (TLS) facility will be used to encrypt transmissions.

The format of the new NJERMT statement parameters is shown in Figure 13-5.

```
NJERMT,
    TYPE=TCPIP,
    JOBTRANS=jt,
    OUTTRANS=ot,
    JOBRECV=jr,
    OUTRECV=or,
    SECSIGNON=YES/NO,
    TLS=YES/NO
```

Figure 13-5 Format of the new NJERMT statement parameters

Note: Nodes must define the SECSIGNON and TLS parameters consistently or else they cannot sign on to each other.

The NETSERV statement

The NETSERV statement describes the NETSERV address space. Its characteristics are the address space name, the TCP/IP stack it uses, the system where it runs, and various tracing information. You can also specify the hostname and port which the NETSERV uses to listen for TCP/IP connections. The default is that NETSERV listens over any host that TCP/IP can use for this system. The port can be any number from 1 to 65535. The default NJE port is 175. A special value of 0 is also allowed (and is the default) and it causes the port assigned to be 175.

The format of the NETSERV statement is shown in Figure 13-6.

```
NETSERV,
    NAME=nsvname,
    STACK=stackname,
    SYSTEM=sysname,
    HOSTNAME=hostname,
    PORT=port,
    JTRACE=YES/NO,
    VTRACE=YES/NO,
    ITRACE=YES/NO
```

Figure 13-6 Format of the NETSERV statement

Where:

NAME=	Specifies a 1-8 character name that uniquely identifies this NETSERV and is also used to start the NETSERV address space. The name must contain only A-Z, @, $, #, or 0-9 and cannot start with a digit.
HOSTNAME=	Specifies the IP host name or IP address that the NETSERV uses to listen over TCP/IP for incoming data. This name can be omitted to use the default, which tells TCP/IP to listen over any IP address that is defined for this processor. Multiple NETSERVs can listen over the same host name or use the default, if different ports are used. The host name can consist of up to 60

characters and must be a legal TCP/IP address or host name defined to the TCP/IP resolver. The host name can be in IPV4 or IPV6 format.

PORT= Specifies a port number that the NETSERV will use, in combination with the host name, to listen over TCP/IP for incoming data. The port must be a number from 0 to 65,535. Port 0 indicates that when the NETSERV starts a standard service name of VMNET. VMNET will be used instead of a numeric port. The service of VMNET corresponds to port 175. Although there is a corresponding port for the service, NJENET-SSL, it is suggested that you use the TLS=YES parameter on the NJERMT statement or a *F NJE command instead.

SYSTEM= Specifies the name of the system that the NETSERV will run on. If the SYSTEM= parameter is omitted, the NETSERV will run on the global. After a DSI, an active NETSERV that defaulted to the global will remain running on the old global. However, the NETSERV will run on the new global if it is subsequently brought down and back up.

STACK= Specifies the name of the stack that TCP/IP uses to get its definitions. If the STACK= parameter is omitted, TCP/IP will use its default stack. The stack must contain only A-Z, @, $, #, or 0-9 and cannot start with a digit except when removing an existing assignment.

ITRACE= Indicates that internal tracing in the NETSERV address space will be active.

JTRACE= Indicates that JES tracing will be active. JES tracing occurs during JES specific exit points in the NETSERV when data records, headers, and trailers are transmitted and received.

The SOCKET statement

A socket is described by the NETSERV address space it runs under, the node it connects to, the hostname and port that it communicates with, and various trace information. The format of the SOCKET statement is shown in Figure 13-7.

```
SOCKET,
    NAME=socket,
    NETSERV=nsvname,
    NODE=nodename,
    HOSTNAME=hostname,
    PORT=port,
    JTRACE=YES/NO,
    VTRACE=YES/NO,
    ITRACE=YES/NO
```

Figure 13-7 Format of the SOCKET statement

13.1.5 Networking node definition example

Here we show a scenario where two nodes, one in Boston and one in New York, wish to communicate using NJE over TCP/IP. Each node defines itself as HOME=YES and the remote node as TYPE=TCPIP. Each node also defines a NETSERV address space. Note that both nodes do not specify the SYSTEM parameter on the NETSERV statement, this defaults to the global processor. The Boston node also defines a SOCKET under its NETSERV address space to communicate over with the New York node.

The HOSTNAME and PORT parameters on the NETSERV statement are optional. The default is that the NETSERV listens to TCP/IP over any host and port that it can for the node.

The HOSTNAME and PORT parameters shown in Figure 13-8 on page 289 illustrates the relationship between the NETSERV definition on one node and the SOCKET definition on the other.

There is an optional parameter, SYSTEM, on the NETSERV definition that is not shown here. It defines which system the NETSERV runs on. The default is the global. If a DSI is done, the default moves to the new global, but only when the NETSERV is inactive. If a NETSERV defined to default to the global is active while a DSI is done, it will remain up where it is, but if it is brought down and back up, it will come up on the new global.

The definitions on the Boston node are shown in Figure 13-8.

```
NJERMT,NAME=BOSTON,HOME=YES
NJERMT,NAME=NEWYORK,TYPE=TCPIP
NETSERV,NAME=SERVB,HOSTNAME=BOSTON.COM,PORT=175
SOCKET,NAME=NY,HOSTNAME=NEWYORK.COM,PORT=175,
   NETSERV=SERVB,NODE=NEWYORK
```

Figure 13-8 Boston node NJE over TCP/IP definitions

And the definitions on the New York node are shown in Figure 13-9.

```
NJERMT,NAME=NEWYORK,HOME=YES
NJERMT,NAME=BOSTON,TYPE=TCPIP
NETSERV,NAME=SERVN,HOSTNAME=NEWYORK.COM,PORT=175
```

Figure 13-9 New York node NJE over TCP/IP definitions

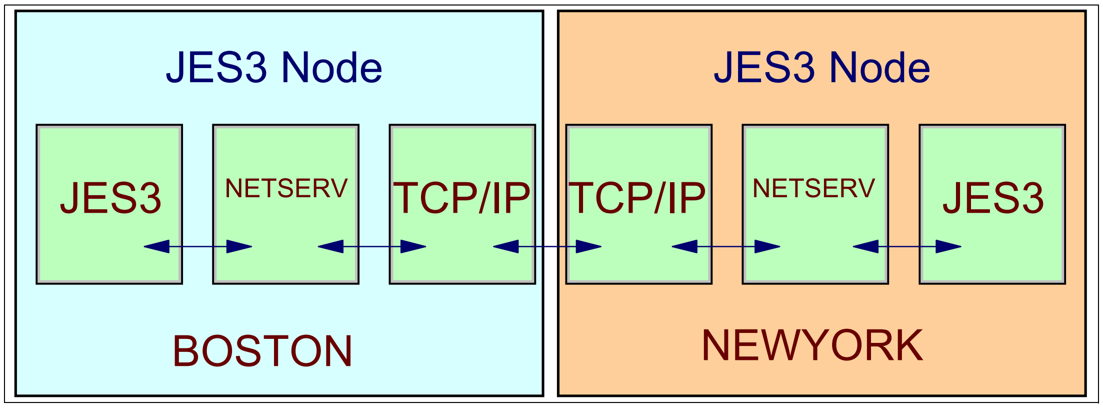


Figure 13-10 Two JES3 nodes using TCP/IP networking

Note: A socket statement for NEWYORK to BOSTON is optional and will cause two connections to be made. Only one socket statement is necessary. A “server” socket @0000001 is dynamically created to BOSTON. If more sockets are needed, the next socket is @0000002; if @0000001 is still active, then @0000003, and so forth.

13.1.6 JES3 commands for TCP/IP NJE

As mentioned earlier in the chapter, new commands are added to JES3 to dynamically control TCP/IP nodes, NETSERV address spaces, and sockets. Table 13-1 describes the new commands.

Table 13-1 JES3 commands to control NJE over TCP/IP nodes

Command	Action
`*X,TCP,NETSERV=netserv`	Starts the NETSERV address space. All the sockets under this NETSERV are also activated.
`*S,TCP,SOCKET=socket`	Starts communication with the specified socket.
`*C,TCP,SOCKET=socket`	Cancels the connection to the specified socket. This command waits for the socket to complete all activity before canceling.
`*C,TCP,SOCKET=socket,I`	Immediately cancels the connection to the specified socket.
`*S,TCP,NODE=nodename`	Starts all sockets for the specified node. The CONCMD DSP issues a *S,TCP,SOCKET=socket for every socket defined to this node.
`*R,netserv`	Resets all active socket connections under this NETSERV address space.
`*C,netserv`	Cancels the specified NETSERV address space.
`*F,NJE,ADD=nodename,TYPE=TCPIP`	Dynamically adds a new TCP/IP node.
`*F,NJE,NAME=nodename, TYPE=TCPIP/SNA/BSC`	Modifies the protocol type used by the specified node.
`*F,NJE,NAME=nodename,SS=YES/NO, TLS=YES/NO,8 JT=jtno, OT=otno, JR=jrno, OR=orno`	Modifies the parameters of a TYPE=TCPIP node.
`*I,NJE,NAME=nodename`	If this is a TYPE=TCPIP node, displays the new parameters added in support to this type of nodes.
`*I,NETSERV=name`	Displays information about the specified NETSERV and the sockets running under it.
`*F,NETSERV,ADD=name`	Dynamically adds a new NETSERV.
`*F,NETSERV=name,HOSTNAME=hostname, PORT=portname,STACK=stakname, JTRACE=YES/NO,ITRACE=YES/NO, VTRACE=YES/NO,SYSTEM=sysname`	Modifies the parameters of the specified NETSERV.
`*F,NETSERV,DELETE=name`	Dynamically deletes the specified NETSERV. The NETSERV address space must not be active.
`*I,SOCKET=socket`	Displays information about the specified socket.
`*F,SOCKET,ADD=name`	Dynamically adds a new socket.

Command	Action
`*F,SOCKET=socket, HOSTNAME=hostname,PORT=portno, NETSERV=netserv,JTRACE=YES/NO, ITRACE=YES/NO,VTRACE=YES/NO, NODE=nodename`	Modifies the parameters of the specified socket.
`*F,SOCKET,DELETE=name`	Dynamically deletes the specified socket. The socket must not be active.

Defining TCP/IP NJE definitions with commands

In Figure 13-11, we show how to use JES3 commands to dynamically define an NJE over TCP/IP connection between a JES3 node called WTSC75J3 and a JES2 node called WTSC75. The JES3 node runs on system SC75 and its IP address is 10.1.40.4. We begin with defining a NETSERV address space called JES3TCP, and then define a socket named J2SC75 under it.

```
*F NETSERV,ADD=JES3TCP
IAT8162 ADD    COMPLETE FOR NETSERV JES3TCP
*F NETSERV=JES3TCP,SYSTEM=SC75
IAT8162 MODIFY COMPLETE FOR NETSERV JES3TCP
*F NETSERV=JES3TCP,HOSTNAME=10.1.40.4
IAT8162 MODIFY COMPLETE FOR NETSERV JES3TCP
*F NETSERV=JES3TCP,PORT=175
IAT8162 MODIFY COMPLETE FOR NETSERV JES3TCP
*F NETSERV=JES3TCP,STACK=TCPIP
IAT8162 MODIFY COMPLETE FOR NETSERV JES3TCP
*I NETSERV=JES3TCP
IAT8707 NETSERV INQUIRY RESPONSE
INFORMATION FOR NETSERV JES3TCP
  SYSTEM=SC75, HOST=10.1.40.4, PORT=  175, STACK=TCPIP, JTRACE=NO,
  VTRACE=NO, ITRACE=NO, ACTIVE=NO
  SOCKETS DEFINED IN THIS NETSERV
  NONE
END OF NETSERV INQUIRY RESPONSE
*F SOCKET,ADD=J2SC75
IAT8160 ADD    COMPLETE FOR SOCKET J2SC75
*F SOCKET=J2SC75,NETSERV=JES3TCP
IAT8160 MODIFY COMPLETE FOR SOCKET J2SC75
*F SOCKET=J2SC75,HOSTNAME=10.1.40.4
IAT8160 MODIFY COMPLETE FOR SOCKET J2SC75
*F SOCKET=J2SC75,PORT=2345
IAT8160 MODIFY COMPLETE FOR SOCKET J2SC75
*F SOCKET=J2SC75,NODE=WTSC75
IAT8160 MODIFY COMPLETE FOR SOCKET J2SC75
*I SOCKET=J2SC75
IAT8709 SOCKET INQUIRY RESPONSE
INFORMATION FOR SOCKET J2SC75
  NETSERV=JES3TCP, HOST=10.1.40.4, PORT= 2345, NODE=WTSC75,
  JTRACE=NO, VTRACE=NO, ITRACE=NO, ACTIVE=NO, SERVER=NO
END OF SOCKET INQUIRY RESPONSE
```

Figure 13-11 Commands to dynamically define an NJE over TCP/IP node

Commands to start TCP/IP NJE

Now that the node NETSERV and socket are defined to JES3, we need to start them. This is shown in Figure 13-12.

```
 *X TCP,NETSERV=JES3TCP
 IAT6306 JOB (JOB04055) IS TCP     , CALLED BY PELEG
 IRR812I PROFILE JES3*.* (G) IN THE STARTED CLASS WAS USED 328
         TO START JES3TCP WITH JOBNAME JES3TCP.
 IAT6100 ( DEMSEL ) JOB IEESYSAS (JOB04056), PRTY=15, ID=JES2
 ICH70001I JES2     LAST ACCESS AT 10:19:53 ON WEDNESDAY, MAY 10, 2006
 IAT9301 TCP START SUCCESSFUL FOR SERVER JES3TCP
 IAZ0542I JES3TCP IAZNJTCP for HBB7730 compiled Mar 26 2006 23:31:05
*IAZ0537I JES3TCP NJETCP SERVER WAITING FOR WORK
 *S TCP,SOCKET=J2SC75
*IAZ0537I JES3TCP NJETCP SERVER WAITING FOR WORK
 IAZ0543I JES3TCP TCP/IP connection with IP Addr: 10.1.40.4 Port: 2345
 Initiated
*IAZ0537I NETSRV1 NJETCP SERVER WAITING FOR WORK
 IAZ0543I JES3TCP TCP/IP connection with IP Addr: 10.1.40.4 Port: 2345
 Successful
 IEF196I IGD103I SMS ALLOCATED TO DDNAME SYS00025
 IEF196I IGD104I HFS FILE WAS RETAINED, DDNAME IS (SYS00025)
 IEF196I FILENAME IS (/etc/resolv.conf)
 IEF196I IGD103I SMS ALLOCATED TO DDNAME SYS00028
 IEF196I IGD104I HFS FILE WAS RETAINED, DDNAME IS (SYS00028)
 IEF196I FILENAME IS (/etc/resolv.conf)
 IEF196I IGD103I SMS ALLOCATED TO DDNAME SYS00029
 IEF196I IGD104I HFS FILE WAS RETAINED, DDNAME IS (SYS00029)
 IEF196I FILENAME IS (/etc/hosts)
*IAZ0537I NETSRV1 NJETCP SERVER WAITING FOR WORK
 IAZ0543I NETSRV1 TCP/IP connection with IP Addr: wtsc75.itsot2.ibm.com Port:
 1036 Successful
 IAZ0544I NETSRV1 LINE5 NJE connection with IP Addr: wtsc75.itsot2.ibm.com
 Port: 1036 Successful
 $HASP200 WTSC75J3 STARTED ON LNE5     NODE WTSC75   BFSZ=32768
 IAT9305 NODE WTSC75  SIGNED ON  NETSERV JES3TCP  SOCKET J2SC75
 IAZ0544I JES3TCP J2SC75 NJE connection with IP Addr: wtsc75.itsot2.ibm.com
 Port: 2345 Successful
 *I NETSERV=JES3TCP
 IAT8707 NETSERV INQUIRY RESPONSE 733
 INFORMATION FOR NETSERV JES3TCP
   SYSTEM=SC75, HOST=10.1.40.4, PORT=  175, STACK=TCPIP, JTRACE=NO,
   VTRACE=NO, ITRACE=NO, ACTIVE=YES
   SOCKETS DEFINED IN THIS NETSERV
   SOCKET   ACTIVE   NODE     SERVER
   J2SC75   YES      WTSC75   NO
 END OF NETSERV INQUIRY RESPONSE
```

Figure 13-12 Commands to start the NETSERV address space and socket

13.1.7 Restarting a node

Networking jobs or SYSOUT that are being sent to another node should not be lost if either system fails. Because JES3 runs in a separate address space than the NETSERV address space, the failure of one system does not impact processing in the other. JES3 and the NETSERV address space automatically take certain actions according to the type of restart that is required. If JES3 fails, the NETSERV address space must wait to access NJE over TCP/IP work. If the NETSERV address space fails, the TCP DSP that JES3 uses to control that NETSERV address space will also end. It can be started again using JES3 *CALL command. Until this is done, JES3 will not send the failing NETSERV address space any NJE over TCP/IP work.

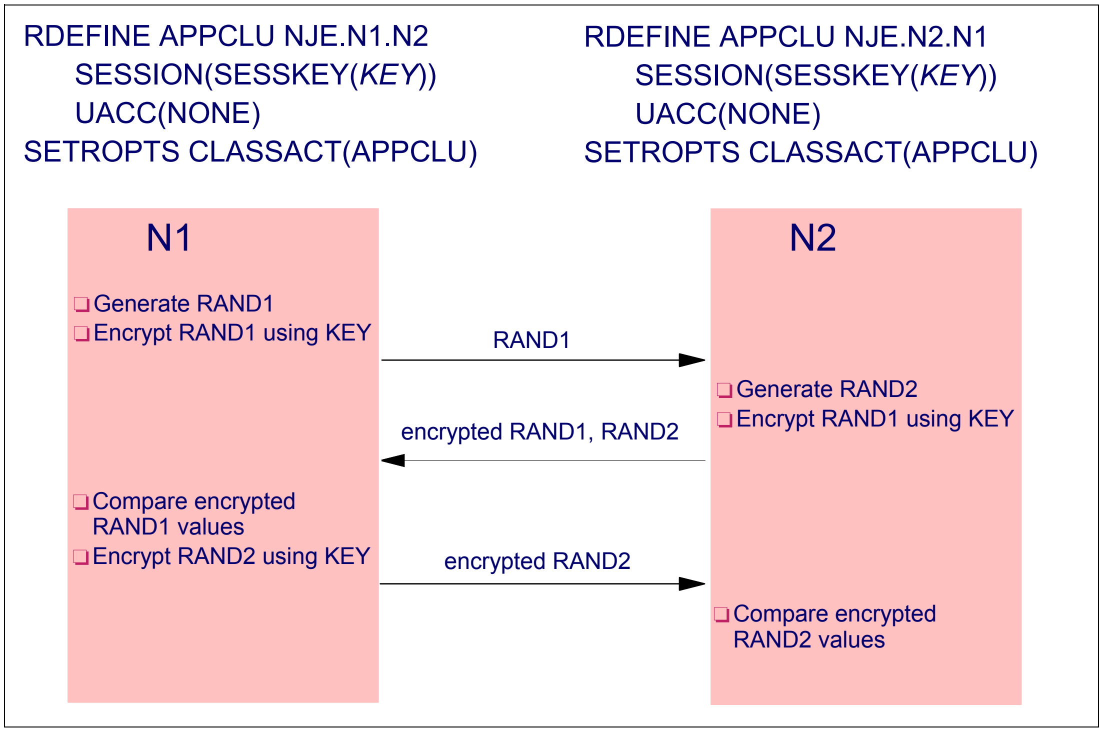
13.1.8 NJE security enhancements

With TCP/IP, there is a greater potential for a hacker to pretend to be an NJE over TCP/IP node than with BSC or SNA nodes. It is much harder for someone to hack into BSC or SNA and pretend to be an NJE node. Generally, this can happen if hardware equipment is connected the wrong way. To cope with this potential risk, JES3 provides new mechanisms to secure the NJE over TCP/IP communications. These are secure signon and the transport layer secure (TLS) protocol.

Secure signon

Secure signon is selected by using the SECSIGNON=YES/NO parameter on the NJERMT statement. It uses profiles in the RACF APPCLU class to provide an encryption key. The RACF profile name should be NJE.homenode.remotenode.

A node participating in secure signon supplies a random string to its NJE partner. The other node encrypts the string and sends the result back to the first node. At the same time, it supplies its own random string. The first node must then encrypt that string and send the result back. If both nodes have the same encryption key defined to the RACF APPCLU class, the encryptions will yield the same result and the nodes are allowed to sign on. Otherwise the signon fails. This is a means for two nodes to authenticate one another. Figure 13-13 illustrates the process of secure signon.



```
RDEFINE APPCLU NJE.N1.N2
    SESSION(SESSKEY(KEY))
    UACC(NONE)
SETROPTS CLASSACT(APPCLU)
```

```
RDEFINE APPCLU NJE.N2.N1
    SESSION(SESSKEY(KEY))
    UACC(NONE)
SETROPTS CLASSACT(APPCLU)
```

N1
❑ Generate RAND1
❑ Encrypt RAND1 using KEY
❑ Compare encrypted RAND1 values
❑ Encrypt RAND2 using KEY
RAND1
encrypted RAND1, RAND2
encrypted RAND2
N2
❑ Generate RAND2
❑ Encrypt RAND1 using KEY
❑ Compare encrypted RAND2 values

Figure 13-13   The process of secure signon

Transport layer secure (TLS)

The TLS protocol provides communications security over TCP/IP. The protocol uses encryption to allow two applications to communicate safely. To indicate TLS is desired, use the TLS=YES parameter of the NJERMT statement or the *MODIFY,NJE command.

TLS support is transparent to JES3 and is provided by z/OS Communications Server. For further information on TLS see *z/OS Communications Server: IP Configuration Guide*, SC31-8775.

13.1.9 Checkpointing NETSERV, socket, and TCP/IP node definitions

One reason checkpointing is done is so that when an operator adds, modifies, or deletes a NETSERV, socket, or TCP/IP node the change is preserved across a hot start. This is different from BSC or SNA where modifications, including additions, must be redone after every JES3 restart until the definition is made permanent with a hot start with refresh.

A second purpose, which is actually a special case of the first one but externally appears different, is to remember server sockets. A server socket is created by an internally issued *F SOCKET command. If a NETSERV has a connection to another node using a server socket, it is necessary to keep that definition if JES3 should happen to be restarted on that node. Without checkpointing, this would not happen.

A third purpose for checkpointing is to remember a NETSERV or socket's active state during a hot start with refresh and disallow any attempt to delete something that is active. There is a similar situation with active FSAs under an FSS, but whereas initialization fails if an attempt is made to delete active FSS or FSAs, with TCP/IP NJE definitions, a message is issued and retains the definition that the installation attempted to delete, and initialization continues as long as there are no other more severe errors.

NETSERV, socket, and NJERMT definitions

These definitions with TYPE=TCPIP are checkpointed. For the other JES3 networking nodes, if a node is modified from BSC or SNA to TCP/IP, the modification is treated as if the BSC or SNA node was deleted and a TCP/IP node of the same name was added. So, if a restart is done, the TCP/IP definition prevails because its information is restored from the checkpointed information.

If a node is modified from TCP/IP to BSC or SNA, this is treated as a deletion of the TCP/IP node and an addition of the BSC or SNA node. Over a restart, the BSC or SNA definition does not survive, and the TCP/IP node does not survive either because its deletion is checkpointed and the node is re-deleted when the checkpointed information is restored. So the node gets deleted.

This is a consequence of the fact that BSC and SNA definitions are not checkpointed, and cannot easily be checkpointed without major compatibility issues with previous releases. Since there is growing interest in TCP/IP and less interest in BSC and SNA, it is not considered worth pursuing.

13.1.10 Compatibility considerations

The IATXNTS macro is used to search the networking node table to verify the validity of a destination. Since JES3 z/OS V1R8 adds a new TCP/IP type of node, a new keyword is required for IATXNTS to support TYPE=TCPIP nodes. The new keyword is TYPETCP. If you have a user modification or installation exit that calls IATXNTS and uses the TYPEBSC and the TYPESNA keywords, you must also specify the TYPETCP keyword. The three keywords must all be specified or all omitted.

This change must be made before starting JES3 level HJS7730. The change is required even if you do not plan to make use of NJE over TCP/IP support. If you do not have any intention of using the NJE over TCP/IP support yet, or if you do not know what TCP/IP action to take for a node, you can specify the same label on TYPETCP as you do for TYPESNA.

Falling back to a previous release

When you use the new NJERMT, NETSERV, and SOCKET definitions on JES3 level HJS7730, the new definitions are saved on the spool. After falling back to a lower level release of JES3 (IPL the global on a lower level release or perform a DSI to a lower level local processor), any NJERMT definition previously made with TYPE=TCPIP appears to be TYPE=BSC. The node is not usable. The SOCKET and NETSERV statements are ignored. When you return to JES3 release HJS7730, the definitions are reinstated.

During fallback to a lower release, the following messages will be issued:

```
IAT4131 SPOOL RECORD ERROR DETECTED (JCT ) FOR JOB TCP (JOBxxxxx)
IAT4174 CONFIRM DELETION OF JOB TCP (JOBxxxxx) DUE TO SPOOL RECORD ERROR(S)
(CONTINUE, SNAP(,ALL) OR TERMINATE)
```

When returning to a JES3 level HJS7730, the following messages will be issued:

```
IAT4159 ERROR RESTORING TCP/IP CHECKPOINT
IAT4103 ERROR(S) ENCOUNTERED RECOVERING JES3 STATUS CHECKPOINT DATA (CONTINUE
OR CANCEL)
```

When you enter the *X DSI command on the global, if any NETSERV address spaces are active, the following message will be issued:

```
IAT0921 DSI - WARNING: SYSTEM sysname HAS nnnnn ACTIVE NETSERVS
```

13.2 SYSOUT application program interface (SAPI) changes

The SYSOUT application program interface (SAPI) allows JES to function as a server for applications needing to process SYSOUT data sets residing on the spool. Using SAPI, an application can access SYSOUT data sets for reading or writing independently from the normal JES-provided functions, such as print or network.

The users of SAPI are application programs running in address spaces external to JES. SAPI supports multiple, concurrent requests from the applications' address space. For example, FTP uses SAPI to provide users with access to SYSOUT data sets through FTP commands, and the Infoprint server uses SAPI to allow users to transform and print SYSOUT data sets on their printers.

13.2.1 Using SAPI to process SYSOUT data sets

SAPI is part of the subsystem interface (SSI). Therefore, in order to issue SAPI requests you have to use the IEFSSREQ macro. Each issuer of the IEFSSREQ macro is referred to as an *application thread*. You use the *subsystem option block (SSOB)* to identify the function requested from the subsystem and the *subsystem identification block (SSIB)* to identify the particular subsystem to which the request is being directed. To request SAPI processing from JES3, you specify function code 79 on the SSOB and identify the subsystem as JES3 in the SSIB.

SAPI requests

SAPI supports three types of requests:

- SSS2PUGE - indicates a SAPI PUT/GET request

  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, specify 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. The actual processing of the data set depends on the application.

- SSS2COUN - indicates a SAPI COUNT request

  Returns the count of scheduler elements (OSEs, in JES3's case) matching the input criteria to the application thread, without returning a particular data set.

- SSS2BULK - indicates a SAPI BULK MODIFY request

  With the BULK MODIFY request, the application thread can select SYSOUT data sets to modify their attributes. Some of the attributes that can be modified are the data set's output class, the data set's destination, release a held data set or delete a data set.

To indicate the desired type of SAPI request, you use the *SSOB extension for SAPI (SSS2)* control block field SSS2TYPE, which is mapped by the IAZSSS2 macro.

Fields in the SSS2 are differentiated into input, output, and disposition fields. An application thread sets the input fields upon each call to IEFSSREQ. In response, JES updates the output fields. An application thread sets the disposition fields according to its specific needs.

Figure 13-14 illustrates a SAPI PUT/GET request issued by an application thread.

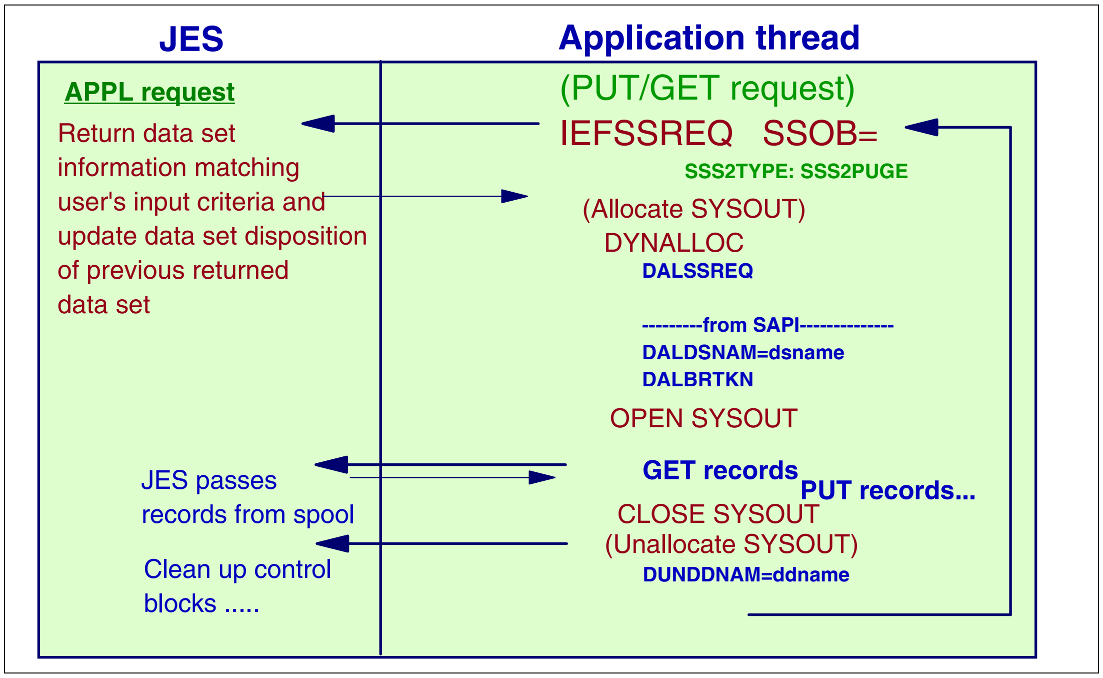


Figure 13-14 SAPI PUT/GET request

For more information about SAPI, see *z/OS MVS Using the Subsystem Interface*, SA22-7642.

13.2.2 JES3 SAPI read-only access

Prior to z/OS V1R8, JES3 required any caller to the SAPI SSI to have update access to the JESSPOOL resource class. Update access was required even if the caller only intended to read from the spool. This behavior forced security administrators to permit an application for update access even if only read access was required, which resulted in excess permissions and possible security breaches. For example, an FTP user who should only read certain SYSOUT data sets from the spool (using the **`get`** FTP command) was forced by JES3 to be permitted for update access. This allowed the FTP user not only to read spool data sets but to modify them as well.

Starting with z/OS V1R8, JES3 changes the way it handles SAPI PUT/GET requests to provide a mechanism for read-only access to SYSOUT data sets. JES2 provided this support with z/OS V1R7, and with z/OS V1R8, JES3 does so as well. This will allow security administrators to apply tighter controls over JESSPOOL resources by permitting users to READ access instead of UPDATE in cases where read-only access is desired. The new JES3 mechanism allows the application thread can indicate JES3 that only read access is desired, and that does not intend to make any updates to SYSOUT data sets.

The SAPI read-only access is invoked by setting the flag SSS2SRON in the flag byte SSS2SEL5 of SSS2 when making a SAPI PUT/GET call to get a data set. Setting this flag indicates to JES3 that the application thread will not to do anything that requires higher access than READ. When JES3 receives the SAPI request with the SSS2SRON flag set, it will call the security product to clear the application thread for READ access instead of UPDATE access.

If the application thread does anything that would require more than READ access, the subsequent SAPI PUT call for the data set will fail with a return code (SSS2BDIS) and reason code (SSS2RRON) that means it has violated its read-only access indication. After setting the SSS2SRON flag, the application thread cannot make updates, even if it turns out that it has UPDATE access. JES3 will not call the security product again to clear the application thread for UPDATE access. If you intend to change your application to make use of the SSS2SRON flag, you should also add code to check for the SSS2BDIS return code and the SSS2RRON reason code and handle it as desired.

14

DFSMS enhancements

DFSMS is a key component of z/OS that automatically manages data from creation to expiration. DFSMS consists of five elements: DFSMSdfp, DFSMSdss, DFSMShsm, DFSMSrmm, and DFSMStvs. DFSMS provides allocation control for availability and performance, backup and recovery, disaster recovery services, space management, tape management, reporting, and simulation for performance and configuration tuning.

This chapter describes the z/OS V1R8 DFSMS changes and enhancements and covers the following topics:

- DFSMSdfp enhancements
- DFSMShsm enhancements
- DFSMSrmm enhancements
- DFSMS OAM enhancements

14.1 DFSMSdfp enhancements

The DFSMSdfp (Data Facility Product) component of DFSMS provides the storage, program, data, and device management functions of z/OS. The storage management subsystem (SMS) component of DFSMSdfp is fundamental to providing these functions. Therefore DFSMSdfp is a base element of z/OS.

z/OS V1R8 DFSMSdfp has been enhanced in the following areas:

- AMS LISTCAT command
- Catalog reliability, availability, and serviceability enhancements as follows:
  - Large page data set support
  - Dynamic service task count
- SMS enhancements as follows:
  - SMS fast path volume selection performance
  - New SETSMS COPYSCDS command
- DFSMS recovery
- New DFSMS diagnostic tools as follows:
  - New diagnostic command for VSAM Record Level Sharing
  - New trace points
  - VSAM code modernization
- PDSE enhancements as follows:
  - PDSE 64-bit virtual storage
  - Dynamic change of hiperspace storage for SMSPDSE1 address space
  - Retain buffers after PDSE close
- Rapid VTOC index rebuild

14.1.1 Access Method Services LISTCAT command

Currently the Access Method Services (AMS) LISTCAT command runs too slow. This happens especially when you process multiple objects in your jobs.

With z/OS V1R8, the **LISTCAT** command is enhanced to present the results of that commands much faster. This enhancement relates to those commands, that process the following types of commands:

```
LISTCAT LEVEL(level) CATALOG(catalogname)
LISTCAT ALL CATALOG(catalogname)
```

There are no changes to parameters of the **LISTCAT** command. When you look at the output in SDSF, you can see differences when you have multiple catalogs involved. In that case, you can see the entries sorted within a particular catalog.

You can see differences in elapsed time very clearly, when you perform a **LISTCAT ALL** on a system running z/OS V1R8 against a system running a z/OS level below z/OS V1R8.

14.1.2 Large page data set support

Prior to z/OS V1R8, it was not possible to allocate page data sets larger than 4 GB. Starting with z/OS V1R8, this is no longer a constraint. This support gives you an improved scalability when you have to use page data sets.

When you plan to use this large page data set support, the following conditions apply:

- Use disk devices types larger than 4 GB, where you are able to allocate these page data sets on a 3390-9 or 3390-27.
- Use Access Method Services (AMS) to define a page data set with the following specifications:
  - Allocate it on an SMS managed volume (using storage class and storage group attributes).
  - Specify extended format (data set name type **EXT** in data class attributes).
  - Specify extended addressability (data class attribute for data sets > 4 GB).

Note: You must install APAR OA07281 and APAR OA10493 when you operate releases prior to z/OS V1R8 to avoid problems if a page data set greater than 4 GB is accessed by that prior release.

14.1.3 Dynamic service task count

The dynamic service task count is an enhancement to DFSMSdfp. Starting with z/OS V1R8, you are now able to set the maximum number of catalog requests to 999 at a time. The default now is set to 200. Before z/OS V1R8 this maximum count was 180. That might have led to performance bottlenecks, when there were more than 180 concurrent catalog requests.

Increasing the service task count

You have to do the following actions when you plan to increase the limit to a number greater than 200:

- Create a new SYSCATxx member in SYS1.NUCLEUS with the appropriate number of concurrent tasks (maximum 999).

  You must implement it this way, because there is no space left in the SYSCAT statement in the LOADxx member, which is limited to 72 characters per line.
- Remove the SYSCAT statement from LOADxx member of SYS%.IPLPARM.

  During IPL, this leads to an operator prompt, where you have to specify the appropriate SYSCATxx member of SYS1.NUCLEUS.

SYSCAT from IPLPARM

Following is a LOADxx specification for SYSCAT:

```
         10        20        30        40        50        60        70
----+----+----+----+----+----+----+----+----+----+----+----+----+----+--
SYSCAT   MXACAT123CCATALOG.MXAICFM.VMXACAT                         SMS
```

The mapping of the SYSCAT in the LOADxx member is shown in Figure 14-1 on page 302.

```
Column   Contents
1-6      SYSCAT
10-15    The volume serial of the device that contains the master catalog.
16       The character "1", unless SYS% to SYS1 conversion is active, in which
         case this will be a "2".
17       Alias name level of qualification.
         Value Range: 1 - 4
         Default: 1
18-19    CAS service task lower limit.
         Value Range: X'18' - X'B4'
         Default: X'3C'
20-63    The 44-byte data set name of the master catalog.
64-71    The 1 to 8 character high level qualifier of the tape volume catalog.
         Default: SYS1
72       Specify Y to enable AUTOADD when the catalog address space (CAS)
         makes the first connection to the coupling facility. The AUTOADD
         function enables coupling facility support of enhanced catalog
         sharing (ECS) for eligible catalogs.
```

Figure 14-1 Mapping of the SYSCAT LOADxx member

If you want to specify the CAS service task lower limit, specify the value with EBCDIC characters, for instance, hexadecimal B4 is specified as C'B4' or X'C2F4'.

Note: Since the new limit is now 999, this value can not be placed in the LOADxx member because there is only a 2-byte field for the count. Therefore, the recommendation to remove the SYSCAT statement specification and use SYSCATxx in SYS1.NUCLEUS.

SYSCATxx in SYS1.NUCLEUS

The system data sets SYS1.NUCLEUS and SYS1.PARMLIB contain members used to define portions of your catalog configuration. When you IPL your z/OS system, you must identify the master catalog to z/OS. This can be done either through the SYSCATxx member of SYS1.NUCLEUS or the LOADxx member of SYS1.PARMLIB. If you use the SYSCATxx member, z/OS issues a message asking you to identify which member you are using. This is done during the nucleus initialization program (NIP) time. You respond with the last two characters in the SYSCATxx member name (the xx value):

```
IEA347A SPECIFY MASTER CATALOG PARAMETER
```

If you enter a blank line in response to this message, z/OS uses “LG” as the parameter, identifying SYSCATLG which is the default. If you use the LOADxx member to identify the master catalog, the system uses the master catalog specified in the member, and the master catalog message is not issued.

The mapping is shown in Figure 14-2 on page 303.

```
1-6      VOLSER
7        Catalog Type
8        Alias Level
9-10     CAS task low limit
11-54    Catalog name
55-62    Tape vol catalog HLQ
63       AUTOADD indicator

z/OS R1R8 adds to SYSCATxx:

65-67    Maximum number of concurrent catalog requests. If no value is
         supplied, the default is 200. Any value less than 200 is rounded up to
         200. The maximum value is 999.
```

Figure 14-2 Mapping of the SYSCATxx in SYS1.NUCLEUS

A job step that can be used to create SYS1.NUCLEUS(SYSCATLG) is shown in

```
//SYSCAT   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT2   DD DSN=SYS1.NUCLEUS(SYSCATLG),DISP=SHR,DCB=(RECFM=U)
//SYSUT1   DD *
MXACAT123CCATALOG.MXAICFM.VMXACAT                    SMS       999
/*
```

Figure 14-3 Job step to create the SYSCATxx member in SYS1.NUCLEUS

Catalog performance report

The main factors affecting catalog performance are the amount of I/O required for the catalog and the subsequent amount of time it takes to perform the I/O. These factors can be reduced by caching catalogs in special caches used only by catalogs.

If the master catalog only contains entries for catalogs, catalog aliases, and system data sets, the entire master catalog is read into main storage during system initialization. Because the master catalog, if properly used, is rarely updated, the performance of the master catalog is not appreciably affected by I/O requirements.

The **`F CATALOG,REPORT,PERFORMANCE`** command can be used to examine certain events that occur in the catalog address space. These events represent points at which catalog code calls some function outside of the catalog component, such as enqueues, I/O, or allocation. All such events are tracked, except for lock manager requests and GETMAIN/FREEMAIN activity. An example of the output from this command is shown in Figure 14-4 on page 304.

```
000290  F CATALOG,REPORT,PERFORMANCE
000090  IEC351I CATALOG ADDRESS SPACE MODIFY COMMAND ACTIVE
000090  IEC359I CATALOG PERFORMANCE REPORT 385
000090  *CAS***************************************************
000090  *  Statistics since 21:53:07.69 on  06/05/2006        *
000090  *  -----CATALOG EVENT----   --COUNT--  ---AVERAGE---  *
000090  *  Entries to Catalog        47,637      3.306 MSEC  *
000090  *  BCS ENQ Shr Sys           49,419      0.811 MSEC  *
000090  *  BCS ENQ Excl Sys             203      0.720 MSEC  *
000090  *  BCS DEQ                   91,840      0.126 MSEC  *
000090  *  VVDS RESERVE CI              557      0.330 MSEC  *
000090  *  VVDS DEQ CI                  557      0.063 MSEC  *
000090  *  VVDS RESERVE Shr          70,652      0.528 MSEC  *
000090  *  VVDS RESERVE Excl             11      0.649 MSEC  *
000090  *  VVDS DEQ                  70,663      0.072 MSEC  *
000090  *  SPHERE ENQ Excl Sys          228      0.616 MSEC  *
000090  *  SPHERE DEQ                   228      0.028 MSEC  *
000090  *  CAXWA ENQ Shr                  2      0.032 MSEC  *
000090  *  CAXWA DEQ                      2      0.020 MSEC  *
000090  *  VDSPM ENQ                 49,920      0.006 MSEC  *
000090  *  VDSPM DEQ                 49,920      0.003 MSEC  *
000090  *  RPL ENQ                        8      0.726 MSEC  *
000090  *  RPL DEQ                        8      0.053 MSEC  *
000090  *  BCS Get                   61,708      0.063 MSEC  *
000090  *  BCS Put                       94      0.658 MSEC  *
000090  *  BCS Erase                     55      0.566 MSEC  *
000090  *  VVDS I/O                  71,231      0.528 MSEC  *
000090  *  VLF Delete Minor               5      0.006 MSEC  *
000090  *  VLF Define Major               1      0.244 MSEC  *
000090  *  VLF Identify               6,248      0.000 MSEC  *
000090  *  RMM Tape Exit                 71      0.000 MSEC  *
000090  *  OEM Tape Exit                 71      0.000 MSEC  *
000090  *  BCS Allocate                 657      3.566 MSEC  *
000090  *  SMF Write                    193      0.019 MSEC  *
000090  *  IXLCONN                        2    378.896 MSEC  *
000090  *  IXLCACHE Read                  6      1.632 MSEC  *
000090  *  MVS Allocate                 642      3.622 MSEC  *
000090  *  Capture UCB                    2      0.003 MSEC  *
000090  *  SMS Active Config              1      0.075 MSEC  *
000090  *  RACROUTE Auth                804      0.369 MSEC  *
000090  *  RACROUTE Define               70      0.070 MSEC  *
000090  *  DADSM Allocate SMS             3     18.230 MSEC  *
000090  *  ENQ SYSZPCCB               2,586      0.004 MSEC  *
000090  *  DEQ SYSZPCCB               2,586      0.002 MSEC  *
000090  *  Capture to Actual             28      0.003 MSEC  *
000090  *CAS***************************************************
000090  IEC352I CATALOG ADDRESS SPACE MODIFY COMMAND COMPLETED
```

Figure 14-4 Output from console command F CATALOG,REPORT,PERFORMANCE

Catalog report

You can verify maximum number of tasks by issuing the following operator command after IPL by issuing the following command and the output is shown in Figure 14-5 on page 305.

```
F CATALOG,REPORT
```

```
F CATALOG,REPORT
IEC351I CATALOG ADDRESS SPACE MODIFY COMMAND ACTIVE
IEC359I CATALOG REPORT OUTPUT
*CAS*************************************************************
*  CATALOG COMPONENT LEVEL   = HDZ1180                          *
*  CATALOG ADDRESS SPACE ASN = 0025                             *
*  SERVICE TASK UPPER LIMIT  =  180                             *
*  SERVICE TASK LOWER LIMIT  =   60                             *
*  HIGHEST # SERVICE TASKS   =   18                             *
*  CURRENT # SERVICE TASKS   =   18                             *
*  MAXIMUM # OPEN CATALOGS   = 1,024                            *
*  ALIAS TABLE AVAILABLE     = YES                              *
*  ALIAS LEVELS SPECIFIED    = 2                                *
*  SYS% TO SYS1 CONVERSION   = OFF                              *
*  CAS MOTHER TASK           = 008FF510                         *
*  CAS MODIFY TASK           = 0089FE88                         *
*  CAS ANALYSIS TASK         = 0089FA28                         *
*  CAS ALLOCATION TASK       = 0089FC58                         *
*  VOLCAT HI-LEVEL QUALIFIER = SYS1                             *
*  NOTIFY EXTENT             =   80%                            *
*  DEFAULT VVDS SPACE        = ( 10, 10) TRKS                   *
*  ENABLED FEATURES          = DSNCHECK DELFORCEWNG SYMREC      *
*  ENABLED FEATURES          = UPDTFAIL AUTOTUNING              *
*  DISABLED FEATURES         = VVRCHECK                         *
*  INTERCEPTS                = (NONE)                           *
*CAS*************************************************************
IEC352I CATALOG ADDRESS SPACE MODIFY COMMAND COMPLETED
```

Figure 14-5 Command output from F CATALOG,REPORT command

14.1.4 SMS fast path volume selection performance

As the number of volumes in z/OS installations is increasing rapidly, there might be situations in large installations, where the SMS volume selection may take very long time due to the large number of possible candidate volumes. Situations, where those problems occur, can be:

- Available space on candidate volumes is a constraint
- An allocation requires a large amount of DASD space
- Large SMS managed data sets are restored by DFSMSdss or recalled by DFSMShsm

In these situations, SMS does not fail an allocation until all available eligible candidate volumes have been tried for allocation and then have been rejected.

z/OS V1R8 fast volume selection

The solution in z/OS V1R8 is a user selectable approach to speed up SMS volume selection. SMS does the volume selection through a number of steps depending on different conditions, that might occur:

- SMS excludes volumes from its selection when the requested space exceeds the volume size for non-best-fit allocations. If none of the eligible candidate volumes have enough space, SMS looks if a data class with the “Space Constraint Relief” parameter fits and does the allocation. Otherwise the allocation fails.
- SMS limits the number of retries for both best-fit and non-best-fit allocations.

You can activate the fast volume selection in two ways:

- Specify the new parameter FAST_VOLSEL(ON) IGDSMSxx parmlib member.

  If you follow this approach, fast volume selection is activated during IPL
- While a system or sysplex is active, type the operator command **D SMS,OPTIONS**, to view the SMS status. If fast volume selection is OFF, activate it dynamically with the operator command, as shown in Figure 14-6 on page 306:

  SETSMS FAST_VOLSEL(ON)

```
SETSMS FAST_VOLSEL(ON)
IGD002I 11:50:14 DISPLAY SMS 998
ACDS     = SYS1.STPPLEX.ACDS
COMMDS   = SYS1.STPPLEX.COMMDS
INTERVAL = 15          DINTERVAL = 150
SMF_TIME = YES         CACHETIME = 3600
CF_TIME = 3600         PDSE_RESTARTABLE_AS = NO
PDSE_BMFTIME = 3600    PDSE1_BMFTIME = 3600
PDSE_LRUTIME = 60      PDSE1_LRUTIME = 60
PDSE_LRUCYCLES = 15   PDSE1_LRUCYCLES = 15
LOCAL_DEADLOCK = 15    GLOBAL_DEADLOCK = 4
REVERIFY = NO          DSNTYPE = PDS
ACSDEFAULTS = NO       PDSESHARING = NORMAL
OVRD_EXPDT = NO        SYSTEMS = 8
PDSE_HSP_SIZE = 0MB   PDSE1_HSP_SIZE   = 0MB
USE_RESOWNER = YES     RLS_MAX_POOL_SIZE = 100MB
RLSINIT = NO           RLSTMOUT = 0
CICSVR_INIT = NO       COMPRESS = GENERIC
CICSVR_DSNAME_PREFIX = DWW.
CICSVR_RCDS_PREFIX = DWW
CICSVR_GRPNAME_SUFFIX = PROD
CICSVR_ZZVALUE_PARM =
CICSVR_UNDOLOG_CONTROL =
CICSVR_UNDOLOG_PREFIX = DWW
CICSVR_BACKOUT_CONTROL =
CICSVR_GENERAL_CONTROL =
Rls_MaxCfFeatureLevel = Z
RlsAboveTheBarMaxPoolSize = 0
RlsFixedPoolSize = 0
PDSE_MONITOR = (YES,0,0)  PDSE1_MONITOR = (YES,0,0)
PDSE_DIRECTORY_STORAGE = 2000M
PDSE1_DIRECTORY_STORAGE = 2000M
PDSE_BUFFER_BEYOND_CLOSE = NO
PDSE1_BUFFER_BEYOND_CLOSE = NO
GDS_RECLAIM = YES      DSSTIMEOUT = 0
BLOCKTOKENSIZE = NOREQUIRE       FAST_VOLSEL = ON
IGD002I 11:50:14 DISPLAY SMS
```

Figure 14-6 Activating and displaying SMS fast volume selection

14.1.5 SETSMS COPYSCDS command

Prior to z/OS V1R8, it was not possible to create an SMS ACDS (active control data set) from an existing SCDS (source control data set) without activating that SCDS. This was considered to be a constraint when preparing an SMS configuration for the deployment in a disaster recovery location.

With z/OS V1R8 you are now able to create such an ACDS directly from an SCDS with the following operator command:

```
SETSMS COPYSCDS(scds_dsn,acds_dsn)
```

You have to ensure, that the SCDS is a validated SCDS. You do this by selecting option 8.5 (Validate the SCDS) within the ISMF panels. Before you type that command, you must also preallocate the future ACDS with IDCAMS DEFINE. The space of the ACDS must be greater or equal than the space of the input SCDS.

14.1.6 New DFSMS diagnostic tools

z/OS V1R8 DFSMS supplies a number of new diagnostic tools to provide you with better serviceability, as follows:

- A new SMS trace event, DEBUG, has been introduced to allow trace points to be added temporarily by IBM support personnel to aid in problem determination. This approach significantly reduces the number of trace entries, while providing specific data that the normal trace entries do not provide. To invoke this support, one or more modules are modified, depending on the problem scenario, to add specific trace points for the DEBUG trace event. After the modified modules are installed, this specific trace can be turned on by issuing the SMS trace command, **SETSMS**, with 'DEBUG' as the selected trace option, as follows:

  ```
  SETSMS DESELECT(ALL) SETSMS TRACE(ON) TYPE(ALL) SIZE(8M) SELECT(DEBUG)
  ```

  The DEBUG trace event may be specified along with other events. See *z/OS DFSMSdfp Diagnosis*, GY27-7618 for more information.
- VSAM record management now automatically generates a dump when certain unexpected errors, such as an invalid CI or a physical I/O error, are detected. No action is necessary by the user that authorized these dumps.
- In addition to the automatic dumps previously described, you can also optionally specify whether VSAM record management dynamic dumping is to occur for specific events identified by a combination of a return code, error (reason) code, problem determination function (PDF or FUNC) code, and component code returned in the request parameter list (RPL) feedback area. You can request a user directed dump (or change the parameters for an existing dump request) with the following command:

  ```
  F CATALOG,VDUMPON
  ```

  You can turn off the dump request with the **F CATALOG,VDUMPOFF** command. See *z/OS MVS System Commands*, SA22-7627 and *z/OS DFSMSdfp Diagnosis*, GY27-7618 for a complete description of these commands.
- If you suspect that VSAM RLS latch contention is causing a hang or deadlock, use the following operator command:

  ```
  D SMS,SMSVSAM,DIAG(CONTENTION)
  ```

  This command produces a console message that either:

  – Displays all latch contention active on the system (if any), or

  – Indicates that there is no latch contention on this system.

  For more information on the **D SMS,SMSVSAM,DIAG(CONTENTION)** command, refer to *z/OS MVS System Commands*, SA22-7627 and *z/OS DFSMSdfp Diagnosis*, GY27-7618.

14.1.7 PDSE enhancements

z/OS V1R8 provides two enhancements for the handling of partitioned data set extended (PDSE). Those enhancements deal with scalability and performance.

PDSE 64-bit virtual storage

Prior to z/OS V1R8, the SMSPDSE and SMSPDSE1 address spaces were 31-bit implementations. This limited the number of concurrently opened PDSE members that could exist in a running system to approximately one million.

With z/OS V1R8, both address spaces are implemented in 64-bit mode. Now you are allowed to use 16 GB as a maximum, which gives you the possibility to have a much larger number of PDSE members open.

You can increase the amount of virtual storage in two ways:

- Specify the parameters PDSE_DIRECTORY_STORAGE(nnn) and PDSE1_DIRECTORY_STORAGE(nnn) in the IGDSMSxx member of SYS1.PARMLIB. nnn is a number between 64M and 16G.

  If you follow this approach, this option is activated during IPL.

  While a system or sysplex is active, type the **`D SMS,OPTIONS`**, command to view the SMS status. Dynamically increase the directory buffer size to a higher value with the following operator commands:

```
SETSMS PDSE_DIRECTORY_STORAGE(nnn)
SETSMS PDSE1_DIRECTORY_STORAGE(nnn)
```

14.1.8 Dynamic change of hiperspace storage for SMSPDSE1 address space

Another enhancement introduced in z/OS V1R8 is the ability to dynamically change the size of the hiperspace storage for the SMSPDSE1 address space. Now you can change the value with the **`SETSMS PDSE1_HSP_SIZE(nnn)`**command. The size nnn stands for a value up to 2,047MB. To activate this changed number, you must restart the SMSPDSE1 address space.

14.1.9 Retain buffers beyond PDSE close

This enhancement in DFSMS gives you better performance for those PDSE members that are opened and closed very often. Before z/OS V1R8, when a PDSE was closed, the buffers for directory and member data was purged. To give you improved performance, this data can be retained in storage now.

You can invoke the buffer beyond PDSE close option in two ways:

- Specify the parameters PDSE_BUFFER_BEYOND_CLOSE(YES) and PDSE1_BUFFER_BEYOND_CLOSE(YES**)** in member IGDSMSxx of SYS1.PARMLIB.

  If you follow this approach, this option is activated during IPL.

- While a system or sysplex is active, use the **`D SMS,OPTIONS`** command to view the SMS status. If the parameter is set to NO, dynamically change it with the operator command as follows:

```
SETSMS PDSE_BUFFER_BEYOND_CLOSE(YES)
SETSMS PDSE1_BUFFER_BEYOND_CLOSE(YES)
```

14.1.10 Rapid VTOC index rebuild

Another performance enhancement in z/OS V1R8 DFSMSdfp deals with building or rebuilding the VTOC index. Having a VTOC index on a DASD volume, gives you better performance in accessing the data set control blocks (DSCB) within the volume table of contents (VTOC). Today, you normally have VTOC indexes on your volume, but sometimes you have to do defragmentation on your volumes, even in today's times of RAID disk subsystems. When you plan to defragment a volume, you first have to deactivate the VTOC index of that volume.

In z/OS V1R8, the DADSM conversion routine which rebuilds the FORMAT-5 DSCB chains (free space DSCBs) during the ICKDSF BUILDIX process is moved to the z/OS DEVMAN address space. With this technique, the conversion routine uses a Data Space, and reads the whole content of the appropriate VTOC into that Data Space. This operation requires only one channel program.

Note: When you use Device Support Facility (ICKDSF) R17 to convert a VTOC, make sure you have applied the appropriate PTF for APAR PK19625.

14.2 DFSMShsm enhancements

DFSMShsm (Hierarchical Storage Manager) provides availability management, and space management functions.

z/OS V1R8 DFSMShsm has been enhanced to provide:

- Error handling on alternate duplex tapes
- Recycle SYNCDEV at intervals
- Migration scratch queue for non-VSAM data set
- Individual data set restore for ARECOVER processing
- Fast replication tape support
- Fast replication data set recovery

14.2.1 Error handling on alternate duplex tapes

Duplex tapes in DFSMShsm backup and migration processes are normally used when you have different locations and plan to store each copy in one of those locations. Data set copies on duplex tapes are generated at the same time.

Prior to z/OS V1R8, when DFSMShsm detected an error on the alternate tape in the migration process, this tape is demounted and discarded, but the write process continues on the original tape. In such a case, DFSMShsm does a `TAPECOPY` to create the alternate tape after a write has completed on the original tape. With todays high capacity tapes, this leads to a long period of time without having a valid duplex copy.

New error handling

In z/OS V1R8, DFSMShsm error handling on alternate duplex tape for migration is enhanced to provide you an option that insures that the original and alternate tapes are filled up with the same content rather than creating an alternate tape at a later time. When an error situation occurs on the alternate tape, both, the original and the alternate tape are marked full. This error handling applies to initial migration processing and in the case of recycling migration tapes. It insures that original and alternate tape sets are in sync.

SETSYS command enhancement

Handing of errors on alternate duplex tapes is invoked by a new option to the SETSYS DUPLEX(MIGRATION(Y)) specification, as follows:

```
ERRORALTERNATE(CONTINUE | MARKFULL)
```

Note: ERRORALTERNATE(MARKFULL) requires that a SYNCDEV be issued for each data set on the original and alternate tape during migration processing and at least at certain intervals during RECYCLE processing.

Issue the following **SETSYS** command when an error occurs writing to an alternate tape during initial migration and when a migration tape is recycled during data set migration, volume migration, primary space management, and interval migration.

```
SETSYS DUPLEX(MIGRATION(Y ERRORALTERNATE(MARKFULL)))
```

Note: If you normally suppress SYNCDEV processing by patching DFSMShsm with the following command:

```
PATCH .MCVT.+196 BITS(..1.....)
```

this kind of patching is suppressed. Beginning in V1R8, this patch is no longer beneficial and no longer recommended.

Types of tape errors

The following types of errors deal with the ERRORALTERNATE processing:

- Tape data set allocation error
- Mount error
- DFSMShsm shutdown request while waiting for alternate volume mount
- Tape data set open error
- Tape data set close error
- Tape data set write error
- End-of-volume (EOV) error
- SYNCDEV error
- Control block error

If the MARKFULL option of the ERRORALTERNATE parameter is specified, then instead of demounting the alternate and continuing to write to the original when an error occurs on the alternate volume, the error handling will be the same as if the error occurred on the original tape.

If ERRORALTERNATE(CONTINUE) is specified, then errors that occur on the duplex alternate tape will have the effect of duplexing working as it does currently. The alternate tape is demounted and discarded, DFSMShsm then continues to write to the original tape. When DFSMShsm has completed writing the original tape, it marks the tape full and schedules a TAPECOPY to create a new alternate tape. The ERRORALTERNATE parameter is only valid when the following command is specified:

```
SETSYS DUPLEX(MIGRATION(Y))
```

If ERRORALTERNATE is specified when **SETSYS DUPLEX(MIGRATION(N))** is specified, then the command fails.

Depending on the error there are two possible actions which might occur:

- Both the original and alternate tapes are demounted and the original tape is marked full with its associated alternate tape. In this case the failed data set will be retried on a new set of original and alternate tape volumes.
- Both the original and alternate tapes are demounted and the migration or recycle process will fail with the corresponding return code.

Identifying tape library tapes

Because DFSMShsm functions that process tapes in a tape library require the complete connected set of tapes to be in the library, it is helpful to be able to identify both individual tapes and connected sets in a tape library. Enhancements to the DFSMShsm **LIST** command allow you to identify the status of individual tapes and connected sets.

Output from the **LIST** commands indicates whether:

- Connected sets contain data sets that span multiple tapes
- Tapes are partially full, full, or empty
- Backup and migration tapes are managed in an SMS-managed tape library
- Dump tapes are managed in an SMS-managed tape library
- A tape is marked full and has an alternate volume

To determine which tapes in your installation are marked full by DFSMShsm, use the following command:

```
LIST TTOC SELECT(FULL) or LIST ML2(TAPE) SELECT(FULL)
```

14.2.2 Recycle SYNCDEV at intervals

Currently a SYNCDEV operation on the original and alternate tape is performed after each data set is written to tape during the RECYCLE process. When you have a large number of data sets this may lead to long processing times.

With z/OS V1R8, the RECYCLE process is changed to allow a SYNCDEV to be used more efficiently by doing it in intervals. The SYNCDEV on the original and alternate tapes is now accomplished after ‘x’ number of data sets have been written to the original and alternate tapes. Both SYNCDEVs are issued asynchronously, for instance the processing flow is as follows:

- Issue an asynchronous SYNCDEV on the original tape
- Issue an asynchronous SYNCDEV on the alternate tape
- WAIT for the completion of both

Since the SYNCDEVs are only performed at specific intervals on both the original and alternate tapes, this should significantly improve RECYCLE performance even when duplexing

Tape errors during processing

Errors that occur during RECYCLE are handled as follows:

- A tape error occurs during simplex RECYCLE

  The output tape is marked full and a new output tape is selected to continue processing with the first data set after the last successful SYNCDEV.
- Recycling a duplexed tape and an error occurs on the original volume

Both the original and alternate output tapes are marked full and a new set of output tapes is selected.

- Recycling a duplexed backup volume and an error occurs on the alternate volume

  The alternate tape is discarded and the original tape is written to. After completing the output to the original tape, it marks the tape full and schedules a TAPECOPY to create a new alternate tape.

14.2.3 Migration scratch queue for non-VSAM data sets

Prior to z/OS V1R8, a significant amount of time was spent during migration of non-VSAM data sets for scratching those data sets on the original level 0 volume. If the scratch process failed the user data set had to be restored to its original condition.

Starting with z/OS V1R8, a new migration scratch queue is implemented to handle those scratch requests asynchronously from other migration steps. This technique gives you an improved performance during the migration process.

During migration of non-VSAM data sets, the time spent doing the scratch of the data set from the Level 0 volume is significant. If the scratch fails, the user data set is restored to its pre-migration condition. A new task is created when migrating to tape, that works on a queue of scratch requests and overlaps that work with other steps of migration. This includes when the volume is being released for use by another task or when releasing the volume when tape processing is completed.

This processing is only applied when DFSMShsm is migrating a volume as follows:

- Command volume migration
- Primary space management
- Interval migration

It is not applied when migrating a single data set. This is also not applicable when migrating an already migrated data set such as during L1 to L2 migration during secondary space management.

14.2.4 Individual data set restore for ARECOVER processing

Prior to z/OS V1R8, it is possible to restore an individual data set or a group of data sets with the same logical attributes from an aggregate. There is currently no way to restore an individual data set from the backup copy of an aggregate group.

With z/OS V1R8, this processing is changed. The DFSMShsm **ARECOVER** command is enhanced to allow a single data set or a select group of data sets to be recovered from an aggregate. New parameters are added to the **ARECOVER** command to allow recovery of a single fully qualified data set name or a select group of fully qualified names listed in a data set.

Note: A LISTOFNAMES data set must be allocated as DSORG=PS, RECFM=FB, LRECL=80, and the data set must be cataloged.

You have two choices how to restore data sets with this technique, with TSO commands and with the ISMF panels.

- TSO command examples:

  ```
  ARECOVER ONLYDATASET(NAME(fully qualified dsname)
  ```

```
ARECOVER LISTOFNAMES(fully qualified dataset name)
```

If a data set in the list is not on the tape for the aggregate, the ARECOVER command recovers those data sets on the tape and fails those data sets that are not on the tape and then issues a non-zero return code.

In the examples shown in Figure 14-7 on page 313, you can see the syntax how to restore data sets. The second example shows you the option to restore a group of data sets. In this case each of the data sets to be restored are listed in a separate data set, called MY.PERSONAL.DATASET.LIST.

```
ARECOVER ONLYDATASET(NAME(MY.PERSONAL.DATASET))
ARECOVER ONLYDATASET(LISTOFNAMES(MY.PERSONAL.DATASET.LIST))
```

Figure 14-7 ARECOVER command examples

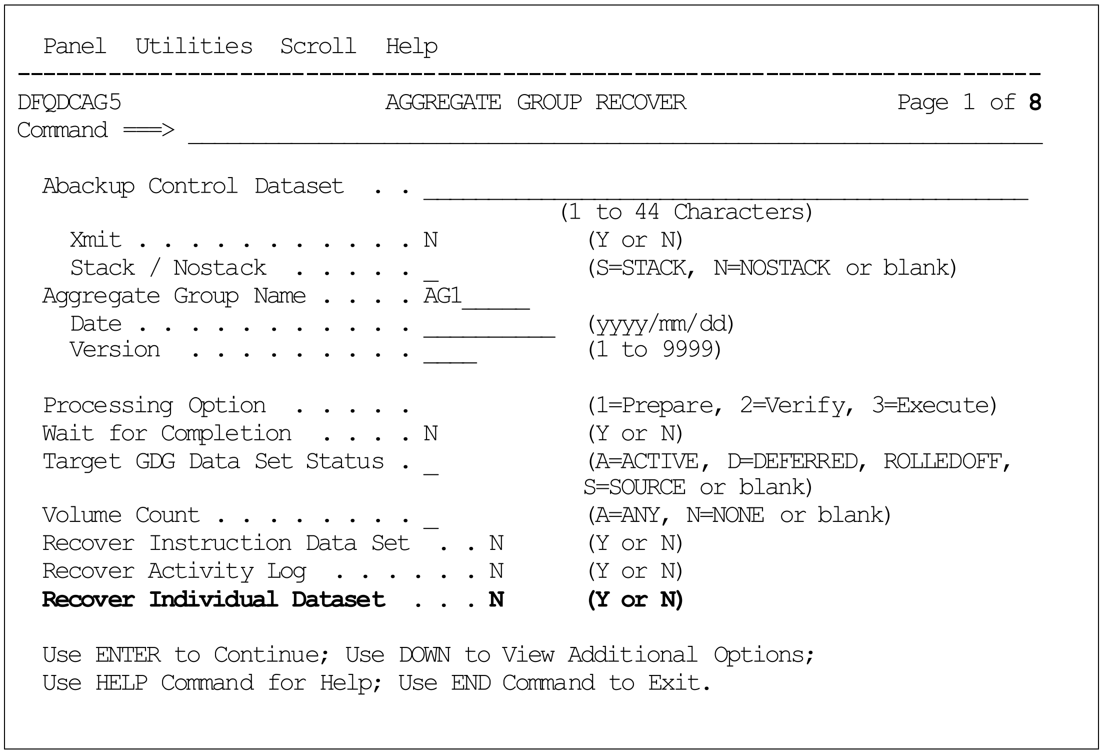
This enhancement gives you improved flexibility and availability when you work with aggregates.

- ISMF panel

  Using the following ISMF panel shown in Figure 14-8 on page 313, you can select the option to recover individual data sets by specifying the data sets name with this option. You can use ISMF to:

  – Restore one data set

  – Restore a list of data sets from the list data set



```
  Panel  Utilities  Scroll  Help
------------------------------------------------------------------------------
DFQDCAG5                    AGGREGATE GROUP RECOVER                Page 1 of 8
Command ===> __________________________________________________________________

  Abackup Control Dataset  . . ____________________________________________
                                          (1 to 44 Characters)
    Xmit . . . . . . . . . . . N            (Y or N)
    Stack / Nostack  . . . . . _            (S=STACK, N=NOSTACK or blank)
  Aggregate Group Name . . . . AG1_____
    Date . . . . . . . . . . . __________   (yyyy/mm/dd)
    Version  . . . . . . . . . ____         (1 to 9999)

  Processing Option  . . . . .              (1=Prepare, 2=Verify, 3=Execute)
  Wait for Completion  . . . . N            (Y or N)
  Target GDG Data Set Status . _            (A=ACTIVE, D=DEFERRED, ROLLEDOFF,
                                           S=SOURCE or blank)
  Volume Count . . . . . . . . _            (A=ANY, N=NONE or blank)
  Recover Instruction Data Set  . . N       (Y or N)
  Recover Activity Log  . . . . . . N       (Y or N)
  Recover Individual Dataset  . . . N       (Y or N)

  Use ENTER to Continue; Use DOWN to View Additional Options;
  Use HELP Command for Help; Use END Command to Exit.
```

Figure 14-8 ISMF Aggregate Group Recover panel to recover data sets

14.2.5 Fast replication tape support

DFSMShsm introduced fast replication backup in z/OS V1R5. At that time the new SMS copy pool construct was implemented. This fast replication technique is used to copy source volumes to target volumes in a copy pool. Prior to z/OS V1R8 it took you two steps to dump volumes from a copy pool to tape:

1. Fast replication backup (FRBACKUP)
2. A DFSMSdss job to dump the copy pool volumes

Starting with z/OS V1R8 you are able to perform this process with one single command.

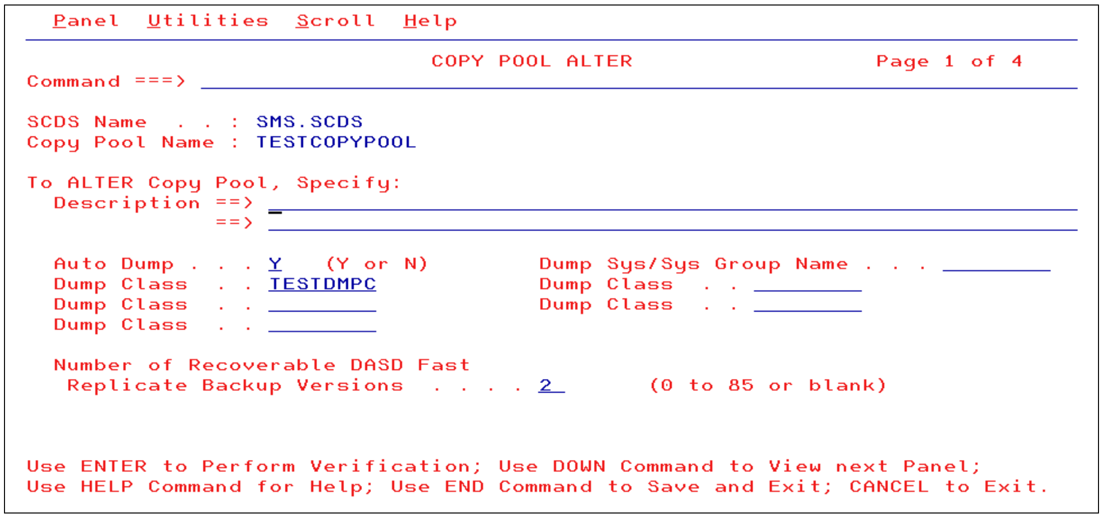
Now, using DFSMShsm fast replication, tape support in z/OS V1R8, you can use DFSMShsm to dump the fast replication backup copies to tape. You can create these dumps on tape through the **FRBACKUP** command or through automatic dump processing. You can recover fast replication backups at the volume level from dumps or DASD copies, or at the copy pool level from DASD copies.

In DFSMS, a *copy pool* is a defined set of pool storage groups that contains data that DFSMShsm can backup and recover collectively, using fast replication. DFSMShsm manages the use of volume-level fast replication functions, such as FlashCopy which is used with the IBM TotalStorage® DS8000™, DS6000™, and Enterprise Storage Server series. These functions provide point-in-time copy services that can quickly copy data from a source location to a target location.

Preparations for fast replication tape support

There are several steps you must check and do before you can perform fast replication backup to tape, as follows:

1. Install toleration PTFs to all the systems in your HSMplex based on APAR OA13177 to insure that all participating systems are at the supported level of DFSMShsm.
2. Take a backup of your current SMS SCDS prior to modifying it on the higher release level.
3. Identify potential SMS storage groups for which DFSMShsm will manage the fast replication backup.
4. Determine which storage group should be in the same copy pool, based on step 3.
5. Modify and validate the SMS SCDS on system running z/OS V1R8:
   a. Create new SMS copy pool constructs in ISMF to configure your copy pools, as shown in Figure 14-9 on page 314.



```
  Panel  Utilities  Scroll  Help
 ------------------------------------------------------------------------------
                              COPY POOL ALTER                  Page 1 of 4
 Command ===> ___________________________________________________________

 SCDS Name   . . : SMS.SCDS
 Copy Pool Name  : TESTCOPYPOOL

 To ALTER Copy Pool, Specify:
   Description ==> _
               ==>

   Auto Dump . . . Y   (Y or N)        Dump Sys/Sys Group Name . . . ________
   Dump Class  . . TESTDMPC            Dump Class  . . ________
   Dump Class  . . ________            Dump Class  . . ________
   Dump Class  . . ________

   Number of Recoverable DASD Fast
    Replicate Backup Versions . . . . 2        (0 to 85 or blank)


 Use ENTER to Perform Verification; Use DOWN Command to View next Panel;
 Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 14-9 Definition of an SMS copy pool, part 1

b. Enable auto dump at the copy pool level to dump the copy pool as an aggregate, if this is desired, as shown in Figure 14-9 on page 314.

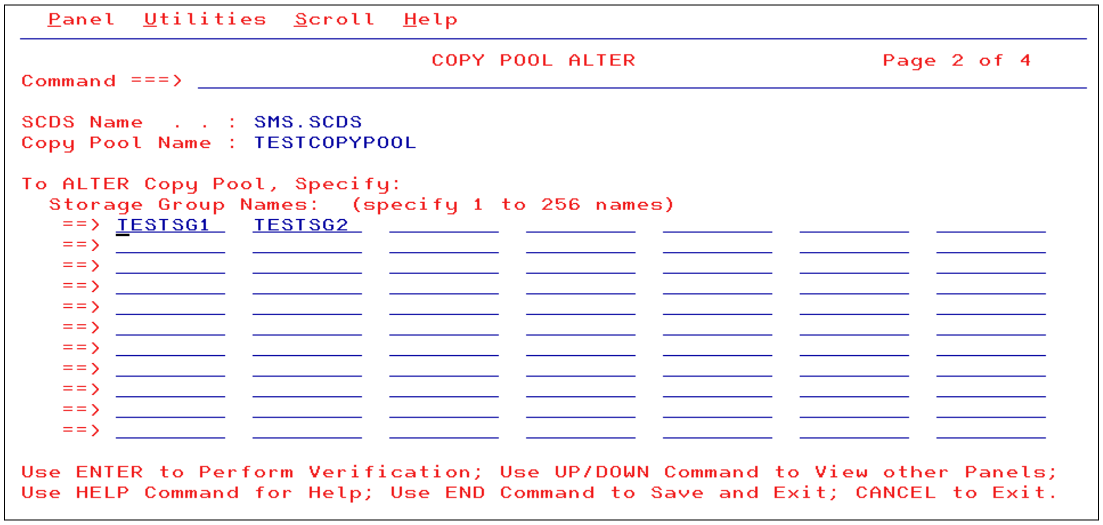
Note: Disable auto dump in the copy pool source storage groups unless you have a good reason having the same storage groups dumped at the copy pool and the storage group level. Dumping the same volumes twice with different techniques requires extra time, which you might not have in your environment.

A reason why you might dump volumes twice can be if there is a system with a lower level than z/OS V1R8 than might need to restore from volume dumps.

c. Decide how many recoverable DASD copies your installation needs, as shown in Figure 14-9 on page 314. If only dump copies are needed, enter a value of 0.

d. Enter the name(s) of the dump class(es) you will use for dumps, as shown in Figure 14-9 on page 314. You can define up to five dump classes into a copy pool definition.

e. Enter the name(s) of the storage group(s) you will use in the copy pool, as shown in Figure 14-10 on page 315. You can add up to 256 storage groups into one copy pool definition.

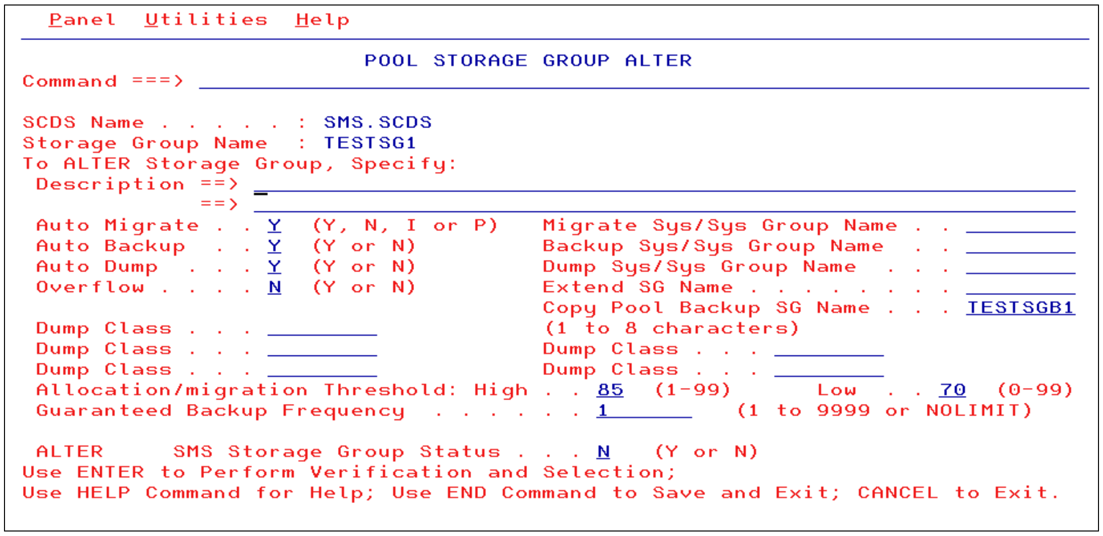


```
  Panel  Utilities  Scroll  Help
------------------------------------------------------------------------------
                              COPY POOL ALTER                  Page 2 of 4
Command ===> _________________________________________________________________

SCDS Name  . . : SMS.SCDS
Copy Pool Name : TESTCOPYPOOL

To ALTER Copy Pool, Specify:
  Storage Group Names:  (specify 1 to 256 names)
   ==> TESTSG1   TESTSG2   ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________
   ==> ________  ________  ________  ________  ________  ________  ________

Use ENTER to Perform Verification; Use UP/DOWN Command to View other Panels;
Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 14-10 Definition of an SMS copy pool part 2

f. Establish the copy pool backup storage groups in the ISMF storage group panels. Insure that a sufficient number of eligible target volumes are available.

g. Update the source storage group definitions with the name of the copy pool backup storage groups as shown in Figure 14-11 on page 316:



```
  Panel  Utilities  Help
 ------------------------------------------------------------------------------
                          POOL STORAGE GROUP ALTER
Command ===> ___________________________________________________________________

SCDS Name . . . . . : SMS.SCDS
Storage Group Name  : TESTSG1
To ALTER Storage Group, Specify:
 Description ==> _____________________________________________________________
             ==> _____________________________________________________________
 Auto Migrate . . Y  (Y, N, I or P)    Migrate Sys/Sys Group Name . . ________
 Auto Backup  . . Y  (Y or N)          Backup Sys/Sys Group Name  . . ________
 Auto Dump  . . . Y  (Y or N)          Dump Sys/Sys Group Name  . . . ________
 Overflow . . . . N  (Y or N)          Extend SG Name . . . . . . . . ________
                                       Copy Pool Backup SG Name . . . TESTSGB1
 Dump Class . . . ________             (1 to 8 characters)
 Dump Class . . . ________             Dump Class . . . ________
 Dump Class . . . ________             Dump Class . . . ________
 Allocation/migration Threshold: High . . 85  (1-99)     Low  . . 70  (0-99)
 Guaranteed Backup Frequency  . . . . . . 1_______  (1 to 9999 or NOLIMIT)

 ALTER     SMS Storage Group Status . . . N    (Y or N)
Use ENTER to Perform Verification and Selection;
Use HELP Command for Help; Use END Command to Save and Exit; CANCEL to Exit.
```

Figure 14-11 Define storage specifications including copy pool backup storage group name

6. Validate and activate the SMS policy.
7. Adjust the auto dump window in the ARCCMDxx parmlib member if auto dump was enabled for some or even all copy pools. The auto dump function creates workloads that need additional time.
8. Perform the DFSMShsm the following command for each copy pool:

```
FRBACKUP COPYPOOL(copy_pool_name) PREPARE
```

Note: This command verifies that a valid environment exists.

- If the number of DASD copies requested is 0, a pre-assignment of target volumes will not be performed. A check will be made for sufficient target volumes.
- If the number of DASD copies requested is greater or equal 1, this command will assign target volumes to each source volumes.

Using fast replication

If you replace of your dump processing with fast replication, all of your standard dumps will expire. You must look into your backup control data set (BCDS) to verify this. If there are dumps that are not expired, do a manual expiration with the DFSMShsm **DELVOL** command.

If you use both forms of dumps, keep their contents on different tapes, using different dump classes for each type of dump. Note that one tape can store dumps of only one dump class.

Note: Systems lower than z/OS V1R8 are not able to recover from a copy pool which is defined on a z/OS V1R8 system. Consider the following:

- Once you process a copy pool that has been created on a lower level system with your z/OS V1R8 system, the copy pool is converted to a z/OS V1R8 copy pool. This behavior has impacts on the next point.
- With a system lower than z/OS V1R8 you are not able to delete, withdraw, create, or dump a z/OS V1R8 copy pool. Furthermore are you not able to do data set recover operations on a z/OS V1R8 copy pool from lower level systems.

14.2.6 Fast replication data set recovery

In z/OS V1R8 you can now recover individual data sets from copy pool backup versions. Prior to this release you were only able to recover backups at the volume or copy pool level.

You use the DFSMShsm `FFRECOV` command to recover individual data sets from copy pool backups. The `FFRECOV` command is expanded to allow you to specify fully or partially qualified data set names. You can recover data sets either from FlashCopy target volumes or from dump tapes (see also "Fast replication tape support" on page 313).

Usage of fast replication recovery

When you have to recover data sets with fast replication recovery, make sure these data sets are cataloged and allocated on the same volumes on which they resided at backup time.

Due to a restriction to a maximum of 23 characters for buffer pool names in z/OS V1R8, plan the naming conventions for new copy pools carefully. For compatibility reasons, z/OS V1R8 continues to support the use of longer 30-character names for your existing copy pools. The reasons for this change are described in "RACF protection of fast replication commands" on page 318.

The following conditions apply for fast replication recovery:

- You may fully or partially qualify the data set name. At least the high-level qualifier (HLQ) must be fully or partially qualified.
- You are not allowed to use "**" as the only qualifier.
- You can use "*" in an HLQ, but not as the first qualifier.
- The catalog determines on which volumes each data set resides.
- If a volume belongs to more than one copy pool, use the following parameter:

  ```
  FROMCOPYPOOL(cpname)
  ```
- Each data set is processed independently. This means that, if one data set fails during the recovery, others may not automatically fail.
- VSAM data sets are processed at the cluster level only.

  Note: The following VSAM specific conditions apply:

  - If you specify a fully qualified VSAM component name, the fast replication recovery request will fail.
  - If you specify a partially qualified data set name, VSAM data and/or index component will be skipped.
  - If your VSAM cluster name matches a partially qualified data set name, the entire cluster will be recovered.
  - Alternate indexes (AIX®) will only be processed, if the base name of the AIX matches the data set name filtering criteria. Any AIXes that are processed, will be processed independently from their base cluster.

- The following data sets are not supported:
  - User catalogs
  - VVDS
  - VTOC index
  - GDG base

– VSAM key-range
– Migrated data sets

RACF protection of fast replication commands

In z/OS V1R8, the RACF FACILITY class resource names for DFSMShsm fast replication commands are changed. The names for discrete profiles are shortened in the third qualifier to allow you to specify the full names of copy pools, which can now be up to 23 characters long.

When you have to decide to authorize or deny the use of DFSMShsm fast replication commands, define the following new discrete profiles as shown in Table 14-1 on page 318. If you do not mask the last level qualifier (LLQ) with an asterisk (“*”), *copy_pool_name* stands for the name of a specific copy pool which you have previously defined in ISMF. See Figure 14-9 on page 314 for an example of such a definition.

Table 14-1 Changed RACF Profiles for DFSMShsm fast replication commands

Command name	**RACF FACILITY class resource name**
FRBACKUP	STGADMIN.ARC.FB.* STGADMIN.ARC.FB.copy_pool_name
FRRECOV	STGADMIN.ARC.FR.* STGADMIN.ARC.FR.copy_pool_name
FRDELETE	STGADMIN.ARC.FD.* STGADMIN.ARC.FD.copy_pool_name

Following are RACF definitions for the **FRBACKUP** command:

```
RDEFINE FACILITY STGADMIN.ARC.FB.TESTCOPYPOOL UACC(NONE)
PERMIT STGADMIN.ARC.FB.TESTCOPYPOOL CLASS(FACILITY) ID(HSM) ACCESS(READ)
SETROPTS RACLIST REFRESH
```

Before you start DFSMShsm, insure that the FACILITY class is active. DFSMShsm checks this during startup. If you have not defined the new profiles, the associated fast replication commands will fail.

In addition to the above mentioned FACILITY class resources, in z/OS V1R8 you have two more resources which are described in Table 14-2 on page 318.

Table 14-2 New RACF Profiles for DFSMShsm fast replication commands

Command	**RACF FACILITY class resource name**
Any LIST COPYPOOL and LIST DSNAME(dsname) COPYPOOL command	STGADMIN.ARC.LC.*
LIST COPYPOOL(copy_pool_name) only for the specified copy_pool_name	STGADMIN.ARC.LC.copy_pool_name

14.3 DFSMSrmm enhancements

DFSMSrmm (Removable Media Manager) provides management functions for removable media such as tape cartridges and reels as one enterprise-wide library across systems and sysplexes. DFSMSrmm also manages tape volumes and the data sets residing on them.

z/OS V1R8 DFSMSrmm has been enhanced to provide the following enhancements:

- An enterprise level interface as follows:
  - Enable resource creation and updates via the CIM interface
  - Support true E-mail addresses for the RMM NOTIFY function
- DFSMSrmm UTC implementation
- Tape data set authorization
- DFSMSrmm VRS policy management simplification
- DFSMSrmm usability items are as follows:
  - Enhancement of the `SEARCHVOLUME` command with new options
  - The `SELECT` command is supported for all search results
  - New columns for volume, data set, and VRS search result lists
  - REXX variable constraint relief
  - ISPF and ISMF data set list utility support

14.3.1 DFSMSrmm enterprise level interface

DFSMSrmm uses common information model and web based enterprise management (CIM/WBEM) to provide a remote instrumentation interface for tape management, based on open standards and with a common data model across platforms.

You can find more information on CIM in "z/OS V1R8 Common Information Model enhancements" on page 395.

DFSMSrmm can be used in a Web environment. A fully-functional Web service, as well as an xmlCIM compliant product that supports Java, such as the OpenPegasus C++ CIMOM from The Open Group, is required to use the DFSMSrmm CIM provider. The enterprise enablement feature requires z/OS V1R7. You also need WebSphere Application Server for z/OS V5.0.2 and later.

The new CIM provider allows viewing, deleting, creating, and modifying of the following DFSMSrmm resources:

- Volumes
- Data sets
- Locations
- Shelf locations

DFSMSrmm CIM classes

You can use a number of CIM classes and their usage for DFSMSrmm as shown in Table 14-3 on page 320.

Table 14-3 Supported CIM classes for DFSMSrmm in z/OS V1R8

Class name	Usage	CIM class type
IBMrmm_PhysicalVolume	Volumes that do not reside in a VTS.	Main
IBMrmm_LogicalVolume	Any volume in a VTS that is not a stacked volume.	Main
IBMrmm_Dataset	Data set name on a tape volume.	Main
IBMrmm_Owner	Owner of a tape volume. An owner can be an individual or a group defined by a RACF group name.	Main
IBMrmm_Location	Location where is tape volume is stored. This can be a system-managed tape library (automatic or manual), or a user-defined storage location.	Main
IBMrmm_ShelfLocation	A single space on a shelf for storage of removable media. DFSMSrmm defines a shelf location in the removable media library by a rack number, and a shelf location in a storage location by a bin number.	Main
IBMrmm_PhysicalLogicalVolume	Used to get logical details of a physical volume.	Association
IBMrmm_LogicalVolumeDataset	Used to get data set information from a logical volume.	Association
IBMrmm_LogicalVolumeOwner	Used to list logical volumes owned by a specified owner.	Association
IBMrmm_PhysicalVolumeCurrentLocation	Used to list physical volumes in a specified location.	Association
IBMrmm_PhysicalVolumeCurrentShelfLocation	Used to get details for a physical volume that resides in a specified shelf location.	Association
IBMrmm_DatasetOwner	Used to list all data sets owned by a specified owner.	Association
IBMrmm_SearchOperands	Used as a search operand.	Auxiliary class for search type operation
IBMrmm_DeleteOperands	Used as a search operand.	Auxiliary class for delete operation

Usage of DFSMSrmm CIM classes

You can use the web-based DFSMSrmm CIM classes for a variety of administration and information tasks. In Table 14-4 on page 321, an example of how you might use this enhancement is shown.

Table 14-4 Usage of DFSMSrmm CIM classes in z/OS V1R8

Task	Associated CIM operation

List all logical volumes, but limit it to 100 occurrences	1. Define the search limits:
	`webmcli mi http://<userid>:<password>@cimom_url :<port>/root/cimv2:IBMrmm_SearchOperands.Resource= ”IBMrmm_LogicalVolume” Operands=”Owner(*) Limit(100)”`
	2. Find volumes:
	`webmcli mi http://<userid>:<password>@cimom_url :<port>/root/cimv2:IBMrmm_LogicalVolume`

You must have the OpenPegasus CIM server up and running in order to issue web client commands against it. The variables in Table 14-4 on page 321 have the following meanings:

<userid>	Your CIM client userid.
<password>	Your CIM client password.
<cimom_url>	The IP-address of the computer, where the CIM server is running.
<port>	The port of the CIM server. This is typically port 5988.

14.3.2 DFSMSrmm UTC implementation

Today, you might have z/OS installations which span multiple time zones. When you use DFSMSrmm before z/OS V1R8, all records in the DFSMSrmm control data set (CDS) are written in local date and time. If you have to view a record from a system, that lives in another time zone, these values are not converted.

Starting with z/OS V1R8, DFSMSrmm supports multiple time zones in its CDS. In this case you have to request that the CDS has to maintain all records related to GMT. When you now look at the records in the CDS, you can see the data in your local time zone. This is independent of the location of the CDS.

UTC enablement

If you plan to enable DFSMSrmm common time support, you must first ensure that all TOD clocks (time of day) in an rmm-Plex are set to GMT. If you have the common time support enabled, DFSMSrmm will write all date and time values in common time into the CDS. It will also convert existing values. Once you have enabled this support you are no longer able to disable it.

We recommend you to set the TOD clock in CLOCKxx member of SYS1.PARMLIB to GMT.

Figure 14-12 on page 322 shows how to enable DFSMSrmm UTC using the EDGUTIL utility in a batch job.

```
//EDGUTIL  EXEC PGM=EDGUTIL,PARM='UPDATE'
//SYSPRINT DD SYSOUT=*
//MASTER   DD DSN=RMM.CONTROL.DATASET,DISP=SHR
//SYSIN    DD *
CONTROL UTC(YES)
/*
```

Figure 14-12 Enablement of DFSMSrmm UTC

Note: When you plan to use this function in an rmm-Plex with lower level systems, you must apply toleration PTFs for APARs OA13577, OA14136, and OA15873.

14.3.3 Tape data set authorization

Before z/OS V1R8, with DFSMSrmm there was no easy way to use the same RACF profile for DASD and tape data sets for full protection of the tape data sets. Starting with z/OS V1R8, DFSMSrmm introduces an option to provide tape data set authorization independent of the RACF TAPEVOL and TAPEDSN classes. This option enables you to use RACF generic DATASET profiles for both DASD and tape data sets.

If you plan to activate this support, you must activate it using new parameters in the SYS1.PARMLIB DEVSUPxx member. Table 14-6 on page 323 shows you the meaning of those parameters, you can select.

Table 14-5 Tape data set authorization using the DEVSUPxx parmlib member

Parameter	Value and meaning
TAPEAUTHDSN	Values are **YES** or **NO**. If you specify YES, you enable tape data set authorization checks in the RACF class DATASET. This is equal to DASD data sets. The data set name which you specify in your JCL to allocate it is used by the system to check your authorization to read or write that specified file. Default value is NO.
TAPEAUTHF1	Values are **YES** or **NO**. If you specify YES, you enable tape data set authorization checks in the RACF class DATASET for the first file on the same tape volume in a situation when any other file on the same tape is open. Default value is NO.
TAPEAUTHRC4	Values are **ALLOW** or **FAIL**. If you specify ALLOW, you control RACF PROTECTALL processing for tape data sets. Default value is FAIL. If FAIL is active, data sets which are not protected by a security profile cannot be accessed.
TAPEAUTHRC8	Values are **FAIL** or **WARN**. You use this new parameter to provide a managed and controlled implementation of tape data set authorization checks in the RACF class DATASET. When you specify WARN, data sets which you are normally not able to access, you can now access them by only getting a warning message issued. Default value is FAIL.

Migration considerations

In the migration phase, you might select the following options to move to your target environment:

- Select option TPRACF(CLEANUP) in the EDGRMMxx parmlib member to get help in the deletion of existing RACF TAPEVOL profiles and discrete DATASET profiles during the time when tape volumes are recycled to scratch status.

  Note: Do not use this option unless you have successfully implemented TAPEAUTHDSN parameter in the DEVSUPxx member.

- Select TAPEAUTHRC8=WARN to implement a warning level of authorization checking for tape data sets. This option gives the ability to adjust and correct generic RACF DATASET profiles to handle tape data set authorization.
- Select TAPEAUTHRC4=ALLOW to access unprotected data sets when determining which new profiles in the RACF DATASET class are required.

Note: When using IEHINITT to relabel tape volumes and it is desired to control which users can perform this task, make the RACF TAPEVOL class active. Also, define profiles in this class. It is not required to do this if labeling new tapes.

When using the TAPEAUTHDSN option or when using the RACF TAPEDSN class to protect tape data sets, define one or more generic profiles in the TAPEVOL class to cover all volumes.

It is not necessary to have the TAPEVOL class active when using the DFSMSrmm tool EDGINERS for labelling and erasing tapes.

14.3.4 DFSMSrmm VRS policy management simplification

DFSMSrmm uses a technique called vital record specifications (VRS) for its management of retention and movement of tapes in an installation. The DFSMSrmm inventory management is used to process the retention and movement policies. Before z/OS V1R8, the setup of policies required that a large number of VRS be specified because each data set name mask also contains retention and movement information.

With z/OS V1R8, DFSMSrmm uses new functions to reduce the number of VRS specifications required to manage retention and movement of volumes. The solution is to separate the data set name mask from the policy specifications.

The following new VRS functions, as shown in Table 14-6 on page 323, are implemented in this release.

Table 14-6 New DFSMSrmm VRS functions

VRS function	meaning
Data set name filter VRS with COUNT(0)	The VRS will not retain a data set.
Retention name filter VRS with COUNT(0)	The first VRS with COUNT(0) has no retention specification. The next and subsequent VRS in the chain specify the entire policy.
Use of JOBNAME(ABEND) or JOBNAME(OPEN) with a data set name mask	This option allows specification of data set masks to select special ABEND and OPEN retention. This allows multiple ABEND or OPEN VRS and uses JOBNAME to point to the appropriate one for each data set. Also, use generic jobnames in the VRS, like ABEND* or OPEN*. These options can be combined with COUNT(0).

VRS function	meaning
Maintenance of the VRS last reference date and time	DFSMSrmm helps to manage a VRS by identifying which VRS policy chains are not used. The last reference date is a value from the VRS records at the start of VRSEL processing.
Listing of unused VRS in the REPORT file and unused counts in the message file	At the end of the VRSEL processing, the unused VRS chains are listed in the REPORT file and counts of the unused VRS records are printed in the MESSAGE file. Information in the REPORT file and the last reference date in the VRS extract records can be used to manage the VRS and delete those not needed any longer.

The new functions can only be exploited when the VRSEL(NEW) operand in the EDGRMMxx parmlib member is specified.

When the new definitions are in place, the DFSMSrmm housekeeping program, EDGHSKP, will recognize the new definitions.

Note: When using VRSEL(OLD) in an installation, migrate to VRSEL(NEW) before migrating to z/OS V1R8. Otherwise, a warning is issued each time VRSEL is run. Also, EDGHSKP processing will end with a return code of 4.

A toleration APAR OA13355 is provided to prevent the continued use of the very old VRS operands STARTNUMBER and LOCATION(BOTH).

14.3.5 DFSMSrmm usability items

In z/OS V1R8, DFSMSrmm is enhanced with a number of usability items to give you more capabilities from different `SEARCH` commands and also better ISPF and ISMF data set list support. Table 14-7 on page 324 shows you these new enhancements. The options mentioned give improved and more flexible reporting opportunities.

Table 14-7 DFSMSrmm usability items

Usability item	meaning
RMM SEARCHVOLME command with option RELEASEACTION	The new operand RELEASEACTION can have the following values: ► **ALL**: List all volumes with any pending actions. ► **ERASE:** List only volumes that require erasing. ► **INIT:** List only volumes that require initialization. ► **RETURN:** List only volumes that should be returned to their owner. ► **REPLACE:** List only volumes that must be replaced by new volumes and returned to the scratch pool. ► **NOTIFY:** List only volumes for which owners must be notified. ► **SCRATCH:** List all volumes to be returned to scratch status. You can use only one or any number of the values, mentioned above.

Usability item	meaning
RMM SEARCHVOLME command with option JOBNAME	You can use the new operand JOBNAME to search for volumes which have been created by the specified jobname. You can use any characters that are allowed for jobnames and you can mask the names with % (for each one character) and * (for any string). If you do not specify JOBNAME the jobname is not used as a selection.
RMM SEARCHVOLME command with option NOJOBNAME	You can use this new operand to list those volumes that do not have any creating jobname.
RMM SEARCHVOLME command with option LOCATION(generic_location_name)	You can use a new value as an addition to the existing operand LOCATION. Use % (for each one character) and * (for any string) to mask your search for any location. This value gives you an advantage in the search for locations compared to the existing values.
SELECT command supported for all search results	SELECT is a new primary command that is available in DFSMSrmm SEARCH results lists.
New columns for volume, data set, and VRS search results lists	You can use three new columns for different search results list in DFSMSrmm: ► In the volume search result list you can find a “set retained column” which shows you the character “Y” if the volume is retained because the RETAINBY(SET) option is in use and other volumes in the set are either VRS retained or retained by volume expiration date. ► In the data set search result list, you can find a “VRS retained column” which shows you whether data sets are retained by DFSMSrmm VRS retention. In that case, the value “Y” is displayed. ► The “VRS search result list” is expanded with a new page which shows you two new columns: – **Count:** which gives you the number of days or cycles of a data set or volume to be retained. – **Store Number:** which gives you the number of days or cycles of a data set or volume to be retained in the specified location.
REXX variable constraint relief	In the past, when you used generic names or numbers as search criteria, the resulting list exceeded the storage space available to DFSMSrmm. DFSMSrmm returned as many results as possible and ended up with an error message. This enhancement enables DFSMSrmm dialog processing to create search result list for volumes and data sets with all their volumes included. You can enable it by setting your TSO profile using PROFILE VARSTORAGE(HIGH). Variables can now be stored above the 16MB line.

Usability item	**meaning**
Customization of ISPF and creation of tape data set lists	Before z/OS V1R8 DFSMSrmm, ISPF and ISMF did not support tape data set lists. Now you can configure ISPF to enable DFSMSrmm to support a limited set of the available line commands for tape data sets. The following line commands are supported: ► **I** displays a search results list showing all data sets in a multivolume set for the selected data set. ► **S** displays the individual data set details. DFSMSrmm determines the first file on the selected volume which matches the selected data set. If other data sets of the same name exist on the on the volume, the wrong details may be displayed. In that case, use the **M** line command and then the DFSMSrmm **I** line command from that list of results. ► **M** displays search results list that shows all data sets on the same volume as the selected data set. ► **D** releases the volume. If the volume is part of a multivolume set, there is an option to release all volumes in that set.

14.4 DFSMS OAM enhancements

The object access method (OAM) uses a class of data called *objects*. An object is a byte stream with a name. OAM doesn't know anything about the content, format, and structure of those byte streams. Objects can be compressed scanned images or coded data. Your applications, which access these information, are responsible for handling the data. Objects can be stored either on DASD, tape, or optical disks.

z/OS V1R8 provides OAM enhancements as follows:

- Binary large object support
- Object tape enhancements:
  - Tape recycle
  - Immediate backup copy

14.4.1 Binary large object support

Objects can have a broad range of size. Before z/OS V1R8, this meant, you must store objects larger than 32 KB in multiple rows of a DB2 table. That had major impacts of performance, because for objects, that had a size of 256 MB, you had to perform approximately 8,000 SQL insert operations into a traditional 32 KB DB2 table.

DB2 introduced large object support (LOB) and the data type binary large object (BLOB) in Version 6. Now with binary large object support in z/OS V1R8 OAM, you are able to use DB2 BLOBs for better scalability and performance. Now you can store large objects in a single LOB column within a DB2 table, where each BLOB has a size up to 256 MB.

If you want to activate BLOB support for OAM, you have to do the following steps:

1. Check, if the PTF for APAR OA12683 is applied on lower level z/OS systems in an OAMplex. If it is not applied, apply it, regardless of whether you will use the lower level in an OAMplex.

2. Run migration job CBRSMR18 from SYS1.SAMPLIB to add a new column to the DB2 object directory tables to indicate whether a DASD resident object resides in a LOB storage structure or not. Change the names in the job to reflect your installations naming conventions.
3. Run the migration job CBRILOB from SYS1.SAMPLIB to create the LOB storage structures required for OAM to exploit DB2 LOB support. This includes
   - LOB table spaces
   - Base tables
   - Base table views
   - Auxiliary tables
   - LOB indexes

   Change the names in the job to reflect your installations naming conventions.
4. Run job CBRPBIND from SYS1.SAMPLIB which has been updated to make use of the VALIDATE(RUN) option in DB2 for SMS storage groups that do not have a V_OSM_LOB_BASE_TBL view defined.
5. Modify the IEFSSNxx parmlib member as follows:

   ```
   SUBSYS SUBNAME(OAM1) INITRTN(CBRINIT)
   INITPARM('[TIME=GMT][,MSG=x][,OTIS=x][,UPD=x][,MOS=nnn][,LOB=x]')
   ```

   Shown in Table 14-8 on page 327 are the various options for LOB support when coding them in the IEFSSNxx parmlib member. LOB=x specifies whether or not OAM exploits DB2 LOB support for large objects that exceed 32 KB. This parameter is new in z/OS V1R8.

Table 14-8 LOB values in the IEFSSNxx parmlib member

LOB value	Explanation	Notice
A	Value specifies that objects for all storage groups that exceed 32KB will be stored in a OB storage structure when stored to DB2. Value indicates to OAM that the installation has created LOB storage structures and their associated V_OSM_LOB_BASE_TBL views for all object storage groups defined in the SMS ACDS	Optimal performance when you intend to store large objects (>32KB) to DB2 DB2, because OAM does not query DB2 to see if the LOB base table view exists. If the LOB base table view does not exist, the large object store fails.
P	Value indicates to OAM that the installation has created LOB storage structures and their associated V_OSM_LOB_BASE_TBL views for a partial list of object storage groups defined in the SMS ACDS	This requires OAM to query DB2 to see if the LOB base table view exists for a given object storage group for each large object stored. If the LOB base table view does exist for a given object storage group, large objects are stored in the associated LOB storage structure. If the LOB base table view does not exist, large objects are stored in the 32 KB data table.
N	Value specifies that objects that exceed 32 KB are to be stored in a 32 KB data table when stored to DB2.	Default value

Note: An IPL is not needed to activate the above named modifications.

14.4.2 Object tape enhancements

In z/OS V1R8, with OAM there are two enhancements which are related to tape handling, as follows:

- Tape recycle
- Immediate backup copy

Tape recycle

In z/OS V1R8, a new command is provided to automatically select full tape volumes based on defined specifications as recycle candidates, and to initiate a MOVEVOL with RECYCLE option for those volumes. The new OAM tape recycle command is as follows:

```
F OAM,START,RECYCLE,scope,{PV=xxx}[LIM=yy|DISPLAY]
```

You can put one of the following values as the *scope* parameter depending on the scope of the command. These values can be as follows:

name	Specifies an object or an object storage group that indicates that only tape volumes marked full belong to the specified object or object backup storage group, are considered candidates for this **RECYCLE** command.
ALLGRP	All full tape volumes that belong to all primary object storage groups defined in the ACTIVE SCDS are considered candidates for this RECYCLE command
ALLBK1	All full tape volumes that belong to all first backup storage groups defined in the ACTIVE SCDS are considered candidates for this RECYCLE command
ALLBK2	All full tape volumes that belong to all second backup storage groups defined in the ACTIVE SCDS are considered candidates for this RECYCLE command
PV=xxx	This is an optional keyword indicating the valid data threshold to be used in determining whether a volume is a candidate for RECYCLE. Full tape volumes that have a percentage of valid data less than or equal to *nnn* are candidates for RECYCLE. If PV=xxx is not specified, the percent valid to be used to determine RECYCLE candidates is derived from the PERCENTVALID default value as defined through the SETOAM command in the CBROAMxx parmlib member. Valid values for xxx are 0 to 100.
DISPLAY	This is an optional parameter that produces a list of volumes that meet criteria to be recycle candidates. This list is sorted by the percentage of valid data on each volume and is written to hardcopy system log through the CBR9875I message. This option does not initiate Recycle processing, and can be issued at anytime, whether a RECYCLE command is actively processing or not. The list of candidate volumes might be large as it shows all volumes that meet the user-specified criteria for RECYCLE. If DISPLAY is not specified then LIM=yy must be specified.
LIM=yy	If you do not specify the DISPLAY parameter, this keyword is required to indicate the maximum number of volumes to be selected for RECYCLE processing. Valid values for *yy* are 1 to 40. If LIM=*yy* is not specified, then DISPLAY must be specified.

Note: If you have systems lower than z/OS V1R8 in your environment, you must apply PTF for APAR OA12683 for coexistence reasons.

When you install the tape recycle support, you also have insert new keywords into CBROAMxx parmlib member. You can find these parameters in Table 14-9 on page 329:

Table 14-9 New keywords for CBROAMxx parmlib member

Keyword	Meaning
MAXRECYCLETASKS(*nn*)	*nn* is the maximum number of MOVELVOL tasks that can be run concurrently by the RECYCLE function. *nn* can be [0..15]. The default is 1. If 0 is specified, no RECYCLE operations can be run at the storage group or global level.
SGMAXRECYCLETASKS(*nn*)	*nn* is the maximum number of MOVEVOL tasks that can be run concurrently by the RECYCLE function for a storage group. Its value cannot exceed the value of MAXRECYCLETASKS. *nn* can be [0..15]. The default is 1. If 0 is specified, no RECYCLE operations can be run at a given storage group.
SGMAXTAPERETRIEVETASKS(*tasks*)	This parameter specifies the maximum number of tasks within the OAM address space that can concurrently read objects from tape for the storage group specified in the STORAGEGROUP parameter. *tasks* can be [1..100]. The OAM default is 1. This keyword is identical to the old MAXTAPERETRIEVETASKS keyword when specified at the storage group level. Now it is the preferred naming convention. The old keyword is kept alive for compatibility reasons.
SGMAXTAPERESTORETASKS(*tasks*)	This parameter specifies the maximum number of tape drives that can be used for writing objects to tape volumes belonging to a specific object or object backup storage group. This parameter and the SGMAXTAPERETRIEVETASKS parameter control the maximum number of tape drives that can be currently allocated to the OAM address space. *tasks* can be[1..100].
PERCENTVALID(*nnn*)	You can specify this keyword only at the global level. *nnn* is the global default percentage of valid data threshold which you use to determine wether a full tape volume is a candidate for RECYCLE processing. *nnn* can be [0..100]. The default is 0. If you use the PV= value on the RECYCLE command which overwrites the PERCENTVALID value in the SETOAM statement.

Another modify command for the OAM address space is updated in z/OS V1R8 with new parameters. In the following command, you can see the syntax to update or set the number of recycle tasks that can run concurrently:

```
F OAM,UPDATE,SETOAM,[ALL],MAXRECYC,max_recyc_tasks
```

The following command shows you the number of recycle tasks that can run concurrently for a specific storage group or for all storage groups:

```
F OAM,UPDATE,SETOAM,[ALL|group_name],SGMAXRECYC,sgmax_recyc_tasks
```

The third new action you can do with the **F OAM** command is to update or set the default percent valid threshold for RECYCLE, as follows:

```
F OAM,UPDATE,SETOAM,ALL,PERCENTV,percent_valid
```

Immediate backup copy

Starting with z/OS V1R8, OAM schedules an immediate backup copy of an object that is stored if the **backup frequency** value in the appropriate SMS management class is set to 0 and **auto backup** is specified (value Y). The second backup copy is done during the normal object storage management cycle (OSMC).The benefit of this technique is that backups are made closer to storing the objects which gives you a higher degree of availability.

Before z/OS V1R8 you were only able to make a primary copy and one or two backup copies of your objects which were typically backup up in the normal OSMC.

Note: If you have systems lower than z/OS V1R8 in your environment, you must apply PTF for APAR OA12683 for coexistence reasons.

15

Program management enhancements

In this chapter we describe the enhancements and changes to the binder and other program management utilities in z/OS V1R8.

We discuss the following:

- Extended relative-immediate instructions support
- AMBLIST enhancements
- New binder options:
  - STRIPSEC
  - STRIPCL
  - STORENX
- Modifications to the LIBRARY binder control statement and final autocall
- Diagnostics enhancements to z/OS UNIX programs calling the binder API

15.1 Extended relative-immediate instructions support

In z/OS V1R7, the binder introduced support for relative-immediate instructions. This included support for:

- All relative-immediate instructions (such as BRAS, BRASL, LARL, and so on‘) with one external symbol in their operands
- Generalized object file format (GOFF) input
- Arithmetic calculations similar to A-cons
- Cross-class references (instruction and target in the same load segment but different classes)

With z/OS V1R8, the binder extends its support for relative-immediate instructions. The extended support allows for:

- Cross-compile unit references for load modules and object modules

  To support cross-compile unit references using relative-immediate instructions, the format of the relocation dictionary (RLD) record data flag field is changed. Figure 15-1 shows the new format of the flag field.

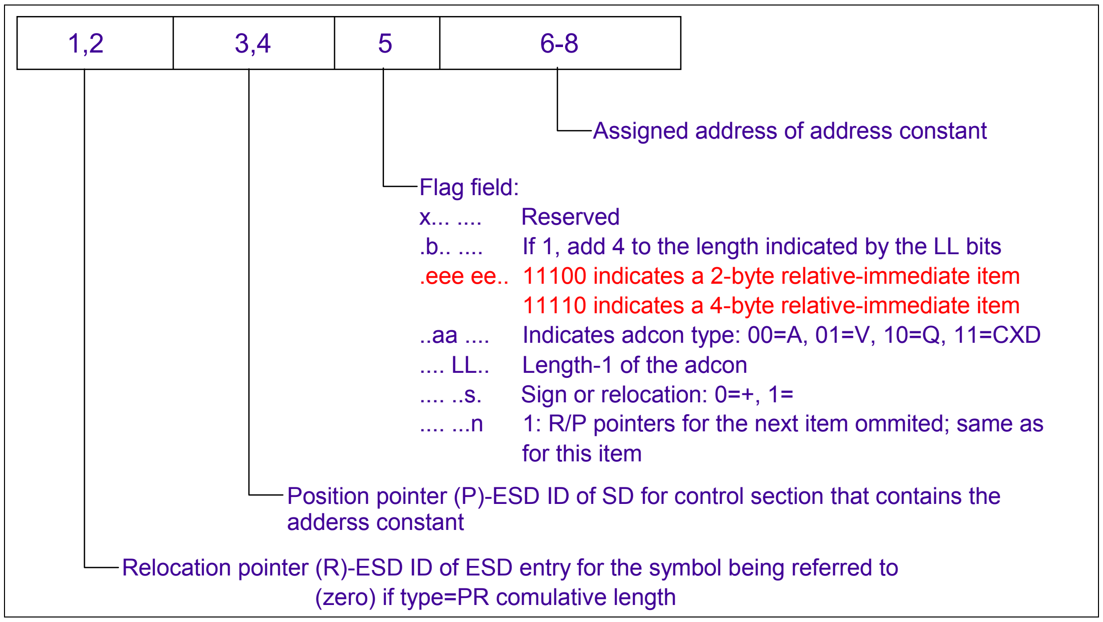

1,2
3,4
5
6-8
Assigned address of address constant
Flag field:
x... .... Reserved
.b.. .... If 1, add 4 to the length indicated by the LL bits
.eee ee.. 11100 indicates a 2-byte relative-immediate item
11110 indicates a 4-byte relative-immediate item
..aa .... Indicates adcon type: 00=A, 01=V, 10=Q, 11=CXD
.... LL.. Length-1 of the adcon
.... ..s. Sign or relocation: 0=+, 1=
....n 1: R/P pointers for the next item ommited; same as for this item
Position pointer (P)-ESD ID of SD for control section that contains the adderss constant
Relocation pointer (R)-ESD ID of ESD entry for the symbol being referred to (zero) if type=PR comulative length

Figure 15-1 Format of the data item field in the RLD

Note: The extended object module (XOBJ) format is not supported.

- Cross-segment references in program objects

  With z/OS V1R8, the loader and the NIP loader are enhanced to identify cross-segment RLDs. Relative-immediate instructions with a 2-bytes immediate field (such as BRAS) are allowed to reference symbols in the same segment. Relative-immediate instructions with a 4-bytes immediate field (such as LARL and BRASL) are allowed to reference symbols across segments, except if either segment has the RMODE64 attribute.

15.1.1 New PO format

Cross-segment support requires enhancements to the loader data in program objects to reflect the new relocation dictionary (RLD) record mappings. Therefore, the use of cross-segment references in relative-immediate instructions causes the output program object to be marked with the new program object format 5 (PO5), which is compatible with z/OS V1R8 and above.

The new PO format can be explicitly requested using the COMPAT binder option. Figure 15-2 shows the format of the new COMPAT option.

```
COMPAT={MIN | LKED | {CURRENT | CURR} | PM1 | PM2
       | {PM3 | OSV2R8 | OSV2R9 | OSV2R10 | ZOSV1R1 | ZOSV1R2}
       | {PM4 | ZOSV1R3 | ZOSV1R4} | {ZOSV1R5 | ZOSV1R6} | {ZOSV1R7}
       | {PM5 | ZOSV1R8}
```

Figure 15-2 COMPAT binder option with z/OS V1R8

Specifying `COMPAT=PM5` or `COMPAT=ZOSV1R8` marks the program object as PO5.

Note: PO5 program objects cannot be loaded, inspected, or rebound on releases prior to z/OS V1R8.

15.1.2 Compatibility considerations

When installing z/OS V1R8 from a lower level driving system, the following APARs must be installed on the driving system to allow for copying and creating directory entries of PO5 program objects:

- APAR OA13294 - Loader APAR

  Symptoms: ISPF browse fails with IEC036I 002-C8. Other programs that use IEWLXLM and IEWLCNVT may also fail.
- APAR OA13525 - DFSMS APAR for FAMS

  Symptoms: IEBCOPY will not copy PO5 members in a PDSE. Error message IGW01578E.
- APAR PK02660 - High level assembler APAR

  Symptoms: ASMA058E Invalid relative address - reloc-symbol.

  With APAR: ASMA215W relative immediate external relocation in NOGOFF object text – reloc-symbol.

15.1.3 Examples

Figure 15-3 on page 334 shows a minimal assembler program named ASMP1. The program has two CSECTs. The first CSECT is RMODE31 and the second CSECT is RMODE24. The RMODE=SPLIT binder option is used to bind this program. This causes the binder to divide the program into two separate segments. Therefore, CSECT ASMP1 must use 4 bytes of relative-immediate instructions to reference symbols in CSECT PRINTMSG. The binder detects the cross-segment relative-immediate references and automatically saves the program object in PO5 format.

```
ASMP1    CSECT
ASMP1    AMODE    31
ASMP1    RMODE    31
         YREGS    ,
         BAKR     R14,R0                     Save caller env
         LARL     R12,PRINTMSG               Establish addressability X
                                              for PRINTMSG routine
         BRASL    R9,PRINTMSG                Branch to routine
         PR       ,                          Return to caller
*
PRINTMSG CSECT
PRINTMSG AMODE    31
PRINTMSG RMODE    24
         USING    *,R12
         WTO      'PROGRAM OBJECT FORMAT 5 OS COMPAT LEVEL z/OS V1R8'
         BR       R9
*
         END      ASMP1
```

Figure 15-3 Sample program containing cross-segment relative-immediate references

15.2 AMBLIST utility enhancements

AMBLIST is a program designed to help you diagnose and repair failures in system or application programs. z/OS V1R8 introduces the following enhancements to AMBLIST.

15.2.1 AMBLIST JCL interface changes

Prior to z/OS V1R8, AMBLIST required a full path name in the DD statement specified on LISTLOAD, LISTIDR, and LISTOBJ commands and would not accept a directory path with a file name. Starting with z/OS V1R8, developers using z/OS UNIX files can now invoke the LISTLOAD, LISTIDR, and LISTOBJ commands of AMBLIST by specifying an HFS directory name on the PATH DD statement and a file name on the MEMBER statement.

You may specify more than one file name on the MEMBER statement. Long file names are supported as well.

Figure 15-4 shows an example of using the new AMBLIST JCL interface to list the contents of two object modules, dnsdomainname and ftp, located under directory /usr/lpp/tcpip/bin.

```
//PELEGAMB JOB ACCNT#,MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//AMBLIST  EXEC PGM=AMBLIST,REGION=0M
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  LISTOBJ DDN=HFS1,MEMBER=(dnsdomainname,ftp)
/*
//HFS1     DD PATH='/usr/lpp/tcpip/bin',PATHDISP=(KEEP,KEEP)
//
```

Figure 15-4 Using the new AMBLIST JCL interface

15.2.2 AMBLIST segment map table

Another enhancement to the AMBLIST utility in z/OS V1R8 is the segment map table. Using the segment map table you can see formatted information about the load segments and their contents. The segment map table is part of the output of the `OUTPUT=XREF` or `OUTPUT=MAP` parameters of the LISTLOAD control statement.

Before this enhancement, AMBLIST did not return information regarding class offsets within segments. This enhancement to the output of AMBLIST LISTLOAD makes it easier to correlate module contents as reported by AMBLIST with locations in storage.

The following example shows how to invoke AMBLIST to see the segment map table output.

```
//PELEGAMB JOB ACCNT#,MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//AMBLIST  EXEC PGM=AMBLIST,REGION=0M
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  LISTLOAD MEMBER=dnsdomainname,OUTPUT=MAP
/*
//SYSLIB   DD PATH='/usr/lpp/tcpip/bin',PATHDISP=(KEEP,KEEP)
//
```

Figure 15-5 JCL to invoke the segment map table output of AMBLIST

Figure 15-6 shows the output of the JCL in Figure 15-5.

```
                    ** SEGMENT MAP TABLE **
   CLASS       SEGMENT      OFFSET    LENGTH       ATTRIBUTES
  B_TEXT             1           0       A3A      INITIAL LOAD
  C_CODE             1         A40      5FC4      INITIAL LOAD
C_@@PPA2             1        6A08        10      INITIAL LOAD
   B_LIT             1        6A18       100      INITIAL LOAD
B_IMPEXP             1        6B18        28      INITIAL LOAD
   C_WSA             2           0        C4             DEFER
```

Figure 15-6 Segment map table output

15.3 New binder options and control statements

There are new binder options and statements introduced in z/OS V1R8.

15.3.1 STRIP options

z/OS V1R8 binder provides two new STRIP options that allow you to remove unneeded sections and classes from program objects and load modules. Using the new options can help reduce the size of program objects and load modules. The new options are invoked as `STRIPSEC=YES` and `STRIPCL=YES`.

STRIPSEC option

The STRIPSEC option allows you to remove unneeded sections from a program object or load module. To remove a section, it must be unreferenced. For a section to be considered unreferenced, it must:

- Contain no symbols that are referenced by another class or section.
- Be neither an entry point nor an alias.
- Contain no exported symbols, if DYNAM(DLL) is set.
- Not be the target of an ORDER, PAGE, INSERT, IDENTIFY, or EXPAND control statement or equivalent API function.

The default option is STRIPSEC=NO. If STRIPSEC is specified without a value, it is treated as STRIPSEC=YES.

STRIPCL option

The STRIPCL option allows you to remove unneeded classes from a program object or load module. STRIPCL=YES specifies that all classes with the REMOVEABLE class attribute are to be removed. The REMOVEABLE attribute is a new class attribute flag for GOFF files. It may be specified in GOFF files passed to the binder, and is preserved in the program object. The normal usage of the REMOVEABLE attribute is expected to be for classes composed of debug data, added by compilers such as the C/C++ compiler.

The default option is STRIPCL=NO. If STRIPCL is specified without a value, it is treated as STRIPCL=YES.

Note: Binder-owned classes (those whose names start with “B”) and classes that contain RLD entries will not be removed.

When the STRIPSEC=YES or STRIPCL=YES options are used, a report is written to the binder’s SYSPRINT DD indicating the removed classes and sections. Figure 15-7 shows an example of the new report.

```
***  R E M O V E D   C L A S S E S   A N D   S E C T I O N S  ***

CLASS NAMES (ABBREV)
C_CDA

SECTION NAMES (ABBREV)
MYSECT
PGM2

***  E N D   R E M O V E D   C L A S S E S   A N D   S E C T I O N S  ***
```

Figure 15-7 Example of the removed classes and sections report

15.3.2 Final autocall and the LIBRARY binder control statement

The AUTOCALL binder control statement indicates that the binder is to perform *incremental autocall* to resolve symbol references using only the given library as the search library. AUTOCALL asks the binder to perform an early pass on a single source to resolve symbol references, and has no effect at all over *final autocall* processing. During the final autocall phase, the binder attempts to resolve any outstanding references by searching the SYSLIB DD statement concatenation. After final AUTOCALL, if any references remain unresolved, the binder states them in its messages.

Historically, final AUTOCALL has primarily searched partitioned data sets using the SYSLIB DD statement concatenation. Since concatenation of z/OS UNIX files is not supported, incremental AUTOCALL has been used instead for z/OS UNIX sources. Most commonly, an

AUTOCALL statement will identify a z/OS UNIX archive. A z/OS UNIX archive is a z/OS UNIX file, usually with a .a file extension, which contains a number of object files along with a catalog of entry point names they can be used to resolve.

This behavior can raise difficulties when binding a complex application consisting of many program modules. Consider the following scenario, illustrated in Figure 15-8:

- A user wishes to bind an application.
- The application contains a call to module X1.
- Module X1 contains a call to module X2.
- Module X2 contains a call to module Y1.
- Module Y1 contains a call to module Y2.
- Modules X1 and Y1 are archived in Archive1.
- Modules X2 and Y2 are archived in Archive2.

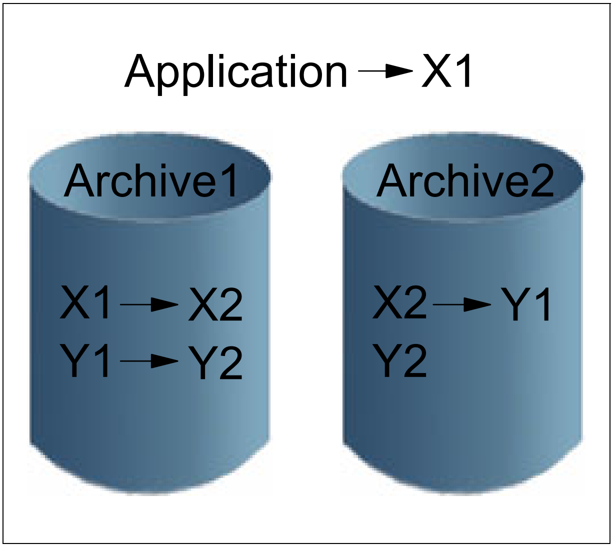


Figure 15-8 AUTOCALL scenario illustration

In order to resolve all external references in the above scenario, four AUTOCALL control statements are required:

- AUTOCALL Archive1 - Resolves X1 in Application but leaves X2 unresolved.
- AUTOCALL Archive2 - Resolves X2 in X1 but leaves Y1 unresolved.
- AUTOCALL Archive1 - Resolves Y1 in X2 but leaves Y2 unresolved.
- AUTOCALL Archive2 - Finally resolves Y2.

LIBRARY binder control statement enhancements

To help simplify the autocall process in scenarios such as the one shown above, the LIBRARY binder control statement is enhanced in z/OS V1R8. The LIBRARY option in z/OS V1R8 allows not only the specification of specific member names, but also the specification of DD names, z/OS UNIX directory path names, and archive libraries.

The new format of the LIBRARY statement is listed in Figure 15-9 on page 338.

```
LIBRARY {{ddname(membername[,...])}
        {ddname2}
        {pathname}
        {(externalreference[,...])}
        {*(externalreference[,...])}},...
```

Figure 15-9 New format of the LIBRARY binder control statement

Starting with z/OS V1R8, the LIBRARY statement also affects the final autocall phase. The behavior of LIBRARY is changed to allow dynamic extension of the SYSLIB concatenation. This allows for z/OS UNIX directories and archives to be searched during the final autocall phase. All LIBRARY statements are deferred to the final autocall phase. During final autocall, the libraries are searched in the order that the LIBRARY statements are processed, with SYSLIB last. The entire set of libraries (LIBRARY statements and SYSLIB) are searched recursively as needed.

Exclusionary LIBRARY statements can be combined, and can exclude names from either SYSLIB or new LIBRARY statements. This is useful when working with DLLs. After the final autocall phase comes the DLL resolution phase, where the binder handles EXPORT statements. It could be that certain symbols have two definitions, one that is bound statically with the application, and the other located dynamically during execution. If the intention is to resolve them dynamically, but the static version is in a concatenation that the binder might find it, then the binder must be told explicitly not to try to resolve the symbols during final autocall.

Changes to the ld utility

The ld utility combines object files and archive files into an output executable file, resolving external references. To do so, it invokes the binder under the covers, using the binder API. Before z/OS V1R8, users of the ld utility had the same problem as all users of AUTOCALL. The list of archives passed to ld with the `-l` option sometimes had to list the same archive repeatedly to get all references resolved.

In z/OS V1R8, the ld utility is changed to take advantage of the new behavior of the LIBRARY statement. Running ld with the `-l` option now results in the same behavior as described in “LIBRARY binder control statement enhancements” on page 337. The new behavior can be overridden by setting the _LD_LIBRARY environment variable to 0. One way of doing this is using the shell command **`export _LD_LIBRARY=0`**.

15.3.3 Prevent non-executable programs

Prior to z/OS V1R8, in the case where a program module does not yet exist, the binder created a new program module regardless of whether it was executable or not. With z/OS V1R8, a new binder control option, STORENX, allows you to specify the conditions under which the binder is to store non-executable program modules.

The format of the STORENX option is:

```
STORENX={YES | NOREPLACE | NEVER}
```

Figure 15-10 Format of the STORENX binder control option

STORENX=NOREPLACE This is the default value for STORENX and specifies that the binder will not replace an executable module in a program library with a non-executable version. STORENX=NOREPLACE can also be specified as STORENX=NO or NOSTORENX.

STORENX=YES	You can simply specify STORENX, which specifies that a new module replaces an existing module of the same name regardless of the executable status of either module. If the NAME statement is provided, the replace option (R) must be coded as well.
STORENX=NEVER	Specifies that the system will never save a non-executable module even when no module with the same name previously existed in the target library.

Attention: When using the `ld` shell command to invoke the binder, the default of the STORENX option is STORENX=NEVER and not STORENX=NOREPLACE.

15.4 Binder diagnostics for z/OS UNIX programs

New diagnostics enhancements have been made especially for z/OS UNIX programs calling the binder.

15.4.1 Support for file names and environment variables

Debugging z/OS UNIX programs calling the binder API is a difficult task in releases previous to z/OS V1R8. The calling program has to dynamically allocate all z/OS UNIX files needed to be passed to the binder, and then pass the binder API the DD names. Moreover, in case of problems, there is no way to change the dump data, or request a more detailed listing without changing the program source code and pass different arguments to the binder API.

In z/OS V1R8, the STARTD (start dialog) API is enhanced to provide easier control over diagnostics. The syntax of the new STARTD call is shown in Figure 15-11.

```
[symbol] IEWBIND  FUNC=STARTD
                  [,VERSION=version]
                  [,RETCODE=retcode]
                  [,RSNCODE=rsncode]
                  ,DIALOG=dialog
                  [,FILES=filelist]
                  [,EXITS=exitlist]
                  [,OPTIONS=optionlist]
                  [,PARMS=parms]
                  [,ENVARS=envars]
```

Figure 15-11 Syntax of the STARTD API call in z/OS V1R8

FILES parameter

You use the FILES parameter to specify the address of a list containing one entry for each binder file for which a DD name or file name is provided. The FILES parameter has been enhanced in z/OS V1R8 such that entries for all files on this list now accept a z/OS UNIX file name in place of a DD name. Path names must begin with a slash (/) or a period and a slash (./).

ENVARS parameter

The ENVARS parameter is a new parameter in z/OS V1R8. You use the ENVARS parameter to specify the address of a list of 31-bit pointers. Each pointer in the list contains the address of a character string that is an environment variable, in the form of `name=value`, to be passed to the binder. Each character string must be null terminated. To use the ENVARS parameter, you must also specify VERSION=6.

The form of the ENVARS parameter is compatible with the POSIX C standard `**environ` external variable (defined as `extern char **environ`). The `**environ` external variable is a pointer to an array of pointers to environment variables available to the program. The simplest way to pass the binder environment variables using the ENVARS parameter is to specify the value of the `**environ` external variable.

By using the `**environ` external variable, the programmer coding the program that calls the binder API, does not have to add special code to deal with environment variables. If the user sets any environment variables before invoking the program, they will be automatically available to the binder.

Binder environment variables

Environment variables recognized by the binder are:

- IEWBIND_DIAG - path name or DD name to be used for IEWDIAG
- IEWBIND_TRACE - path name or DD name to be used for IEWTRACE
- IEWBIND_DUMP - DD name to be used for IEWDUMP
- IEWBIND_GOFF - DD name to be used for IEWGOFF
- IEWBIND_PRINT - path name or DD name to be used for SYSPRINT
- IEWBIND_TERM - path name or DD name to be used for SYSTERM
- IEWBIND_OPTIONS - binder option string. These will be appended to options passed explicitly via the STARTD API call (and will thus take precedence).

These environment variables provide the user a simple interface to set the names of diagnosis files.

Note: The binder is not a shell utility or LE-enabled and thus cannot access the environment variables directly. They must be passed to the binder by its caller through the API. For the same reason, the environment variables are not supported for binder batch callers.

16

z/OS XL C/C++ enhancements

In this chapter, the enhancements to z/OS XL C/C++ run-time library and compiler introduced in z/OS V1R8 are described.

This chapter discusses the following run-time library enhancements:

- Multi-volume data set I/O performance
- VSAM extended addressability support
- Large format data set support
- The flockfile() family of functions
- Reinitialization of DLLs WSA
- GWP Compliance and euro locales
- The _CEE_ENVFILE_S environment variable

This chapter also describes the following compiler enhancements:

- New compiler options for optimization
- New language extensions
- HLASM invocation utility
- 64-bit IPA link phase
- Split IPA link phase listing
- Support for standard exceptions of the C++ new operator
- GONUMBER support for AMODE 64 applications
- Support for DB2 V9 coprocessor
- Integrated CICS translation

16.1 Run-time library enhancements

This section discusses enhancements and changes to the z/OS XL C/C++ run-time library in z/OS V1R8.

16.1.1 Multi-volume data set I/O performance

The `fgetpos()` and `fsetpos()` functions are used to get and set the value of the file position indicator of data sets and z/OS UNIX files. These functions also support large and multi-volume data sets. `fgetpos()` and `fsetpos()` are redesigned in z/OS V1R8 to provide better performance for multi-volume data sets.

The `ftell()`, `ftello()`, `fseek()`, and `fseeko()` functions are also used to get and change the current file position indicator. These functions are redesigned in z/OS V1R8 to provide better performance for multi-volume data sets. However, the performance enhancement is only available when using these functions to position the file position indicator of multi-volume data sets relatively backwards. The reason that `ftell()`, `ftello()`, `fseek()`, and `fseeko()` are enhanced only for relative backwards positioning, and not in all directions, is that these functions return a 32-bit value which does not allow for returning of information required for performance enhancements.

Therefore, it is recommended to use the `fgetpos()` and `fsetpos()` functions for positioning of multi-volume data sets in all directions. Using these functions generally results in better repositioning performance compared to what happens prior to z/OS V1R8 and compared to the `ftell()`, `ftello()`, `fseek()`, and `fseeko()` functions, when working with multi-volume data sets. In general, `fgetpos()` and `fsetpos()` will reduce EXCP counts and CPU time compared to what happens prior to z/OS V1R8 and compared with `ftell()`, `ftello()`, `fseek()`, and `fseeko()`.

All the changes to `fgetpos()`, `fsetpos()`, `ftell()`, `ftello()`, `fseek()`, and `fseeko()` are internal. The function formats are unchanged in z/OS V1R8.

Important: If an `fpos_t` structure, obtained from the `fgetpos()` function which was running on a release prior to z/OS V1R8 (for example, if it was saved in a file on a lower release and then read in z/OS V1R8), is then passed to the `fsetpos()` function running under z/OS V1R8, that call will not see any performance improvements.

Greater flexibility when working with multi-volume data sets

Along with the performance enhancements, z/OS V1R8 also provides greater flexibility when working with multi-volume data sets. The following tables describe the improvements in z/OS V1R8 compared with previous releases.

Table 16-1 DASD multi-volume data sets

Prior to z/OS V1R8	In z/OS V1R8
Opening multi-volume data sets is allowed for read, write or update, but not for append. Open for write is allowed only by DD name.	*Opening* multi-volume data sets by data set name or DD name is allowed for read, write, update and append modes.
Repositioning functions are available only when multi-volume data sets are opened for read.	*Repositioning* functions are available when multi-volume data sets are opened for r, r+, rb, rb+, w+, wb+, a+ and ab+ modes.
Extending multi-volume data sets opened for read update (r+, rb+) is not allowed.	*Extending* multi-volume data sets is allowed for read update mode (r+, rb+).

Table 16-2 Tape multi-volume data sets

Prior to z/OS V1R8	In z/OS V1R8
Opening multi-volume data sets is allowed for read and write (r, rb, w, wb) but not for update or append (r+, rb+, w+, wb+, a, ab, a+, ab+).	No changes.
Repositioning functions are available only when multi-volume data sets are opened for read (r, rb).	No changes.

Table 16-3 Any multi-volume data set

Prior to z/OS V1R8	In z/OS V1R8
Simultaneous readers are not supported for multi-volume data sets.	No changes.
Multivolume data sets only tolerate a single buffering mode.	No changes.

16.1.2 VSAM extended addressability support

DFSMSdfp has long supported VSAM data sets that can exceed 4 GB in size. This support is provided using the extended addressability (EA) attribute of the SMS data class ACS construct. An EA VSAM data set must be SMS-managed. The size limit for an EA VSAM data set is determined by either:

- The control interval size multiplied by 4 GB
- The volume size multiplied by 59

For example, a control interval of 4 KB yields a maximum data set size of 16 TB, while a control interval of 32 KB yields a maximum data set size of 128 TB.

With z/OS V1R8, the z/OS XL C/C++ run-time library is updated to support VSAM data sets defined with the extended addressability attribute. This allows z/OS XL C/C++ applications to read, write and position VSAM data sets beyond the 4 GB boundary.

To address positions beyond 4 GB, an 8-byte relative block address (RBA) is required. In the z/OS XL C/C++ run-time library, we call this 8-byte RBA an *XRBA*. As long as your VSAM and EA VSAM data sets do not grow larger than 4 GB, your existing applications will continue to work in the same way. When your EA VSAM data sets grow beyond 4 GB, changes to your applications are needed in order to support the XRBA.

Note: VSAM extended addressability is supported by z/OS XL C/C++ only for KSDS, ESDS, and RRDS data sets. KSDS and ESDS data sets with alternate indexes are not supported.

The rest of this section describes the enhancements and changes to the z/OS XL C/C++ run-time library introduced in support of the XRBA.

The __amrc structure

The `__amrc` is a structure defined in `stdio.h` to help you determine errors resulting from I/O operations. It is updated during system I/O and some C-specific error situations. The `__amrc` contains a 4-byte field called `__RBA`. This field is updated to contain the RBA address returned by VSAM after an ESDS or KSDS record is written out. For RRDS, the value is calculated from the record number. The `__RBA` field is usually used as input on subsequent calls to the flocate() function. As explained earlier, this field cannot be used when an EA VSAM data set grows beyond 4 GB.

In addition to the __RBA field, a new 8-byte field called __XRBA is added to the __amrc structure in support of EA VSAM data sets. During VSAM operations where the RBA is stored in the __amrc structure, both the __RBA and the __XRBA fields are updated. When the __XRBA field exceeds 4 GB-1, the __RBA will be set to -1 (meaning EOF). This is an indication that the __RBA field is not to be used anymore, since the RBA is beyond 4 GB-1. As a result, applications using the __RBA field of the __amrc structure must be changed in order to take advantage of the XRBA.

The flocate() function

The flocate() function can be used to locate a specific record within a VSAM or an EA VSAM data set given the key, RBA, or the relative record number. In z/OS V1R8, the function accepts both key lengths of 4 and 8 bytes when RBA is used for positioning. An AMODE 64 application using the __RBA field of the __amrc structure does not need to copy the value into an 8-byte container for use with flocate(). The application can now call flocate() using the __RBA field and specify the length as 4. A key length of 8 bytes is only required when working with EA VSAM data sets that are over the 4 GB boundary. When using the value 4 GB-1 (0xFFFFFFFF) as the RBA, the key length must be 8 bytes. If the key length is 4 bytes, flocate() will treat the RBA as -1 (EOF).

Applications using RBA values with flocate() must be changed in order to take advantage of the XRBA.

The fgetpos() and fsetpos() functions

The fgetpos() and fsetpos() functions are changed in z/OS V1R8 to support EA VSAM data sets and the XRBA. The changes to these functions are internal and do not require an change in the application to support EA VSAM data sets and the XRBA.

ftell(), fseek() and the non-large files version of ftello() and fseeko()

Applications running in AMODE 31 cannot use the XRBA with the ftell(), fseek(), and the non-large files version of ftello() and fseeko() functions. Both the ftell() and ftello() functions return a 4-byte value. If the current file position cannot be represented in a 4-byte value, the EOVERFLOW error description code will be set. The fseek() and fseeko() functions are unchanged.

In AMODE 64, large file behavior is automatic. Therefore, applications running in AMODE 64 have no restrictions related to support of the XRBA.

The large files version of ftello() and fseeko()

The large files version of ftello() and fseeko() is enabled when the _LRAGE_FILES feature test macro is defined and the longlong data type is available (for example, using the LANGLVL(LONGLONG) compiler option). The large files version of the ftello() function returns an 8-byte value and has been updated in z/OS V1R8 to support EA VSAM data sets and the XRBA. The large files version of the fseeko() function in z/OS V1R8 accepts an 8-byte offset as the 2nd argument, and it too is updated to support EA VSAM data sets and the XRBA.

Applications using the ftell() and fseek() function must change to use the large files version of ftello() and fseek() in order to take advantage of the XRBA.

fldata() function

The `fldata()` function is used to retrieve information about an open stream. It returns the information in a structure called `fldata_t`. In z/OS V1R8, the `fldata_t` structure is updated to support EA VSAM data sets. A new field, `__vsamEA`, is added to identify whether or not the stream refers to an EA VSAM data set.

16.1.3 Large format data set support

Starting with z/OS V1R7, DFSMSdfp provides support for large format sequential data sets, removing the size limit of 65,535 tracks per volume for the QSAM, BSAM, and EXCP access methods. To allocate a large format data set, you use the `DSNTYPE=LARGE` statement on the JCL DD statement, TSO/E ALLOCATE command, or SVC 99 (dynamic allocation).

However, z/OS V1R7 does not provide large format data sets support for high level programming languages such as C/C++. This support is provided starting with z/OS V1R8.

Support for BSAM (seek)

In order to use repositioning functions such as `ftell()`, `ftello()`, `fseek()`, `fseeko()`, `fgetpos()`, and `fsetpos()`, you must open the file without specifying the `noseek` keyword on the mode parameter of the `fopen()` or `freopen()` functions. When you omit the `noseek` keyword, z/OS XL C/C++ will open the file for processing using the BSAM access method.

In z/OS V1R8, z/OS XL C/C++ provides limited large format data sets support for BSAM. A large format data set can be opened for read (`r` or `rb` mode) only if it has no more than 65,535 tracks. A concatenated large format data set can be opened for read or read-update (`r+` or `rb+` mode) only if it has no more than 65,535 tracks. Write mode support is not available for BSAM.

Attempting to open a large format data set for read with seek, while the data set is already opened for write or append with `noseek` within the same application, is not supported and will cause the call to `fopen()` or `freopen()` to fail. The failure occurs because the writer might extend the data set beyond 65,535 tracks, but the reader is restricted to data sets no larger than 65,535 tracks.

Support for QSAM (noseek)

When you specify the `noseek` keyword on the mode parameter of the `fopen()` or the `freopen()` functions, you indicate z/OS XL C/C++ that positioning functions are not going to be used. In that case, z/OS XL C/C++ will open the file for processing using the QSAM access method, which may improve performance. z/OS XL C/C++ provides complete QSAM support for large format data sets in z/OS V1R8. This includes opening files in read and write modes with no restrictions.

But the `noseek` request is not always honored by z/OS XL C/C++. There may be cases where a `noseek` request is converted internally to a seek request. As of z/OS V1R8, it is important to understand the cases where a `noseek` request is not honored, since the complete support for large data sets it limited to `noseek` requests.

z/OS XL C/C++ will convert a `noseek` request to a `seek` request in these cases:

- The data set is opened for update mode (r+, rb+, w+, wb+, a+, ab+).
- The data set is already open for write or update mode in the same application.
- The data set is a RECFM=FBS data set opened for append mode (a, ab, a+, ab+).
- The data set is an LRECL=X data set.

- The data set is the directory of a partitioned data set (PDS or PDSE).
- The data set is a member of a PDS or PDSE where the member was not specified at allocation, but rather specified at `fopen()` or `freopen()`.

Checking whether noseek is honored

The `fldata()` function is enhanced in z/OS V1R8 to provide information about the access method used for the data set and the reason a `noseek` request was converted to a seek request. Two new fields are added to the `fldata_t` structure, returned by `fldata()`, for this purpose:

- `__access_method` - Identifies the access method used for the data set. Values include:
  - `__AM_UNSPEC` - Access method is neither BSAM nor QSAM.
  - `__AM_BSAM` - Access method is BSAM.
  - `__AM_QSAM` - Access method is QSAM.
- `__noseek_to_seek` - Identifies the reason noseek was converted to seek. Values include:
  - `__AM_BSAM_NOSWITCH` - No switch was made.
  - `__AM_BSAM_UPDATE` - The data set was opened for update.
  - `__AM_BSAM_BSAMWRITE` - The data set is already open for write or update mode in the same application.
  - `__AM_BSAM_FBS_APPEND` - The data set is a RECFM=FBS data set and opened for append.
  - `__AM_BSAM_LRECLX` - The data set is an LRECL=X data set.
  - `__AM_BSAM_PARTITIONED_DIRECTORY` - The data set is the directory for a PDS or PDSE.
  - `__AM_BSAM_PARTITIONED_INDIRECT` - The data set is a member of a PDS and the member name was not specified at allocation.

Note: `__access_method` and `__noseek_to_seek` are only valid when `__dsorgPS` or `__dsorgPO` is set.

You can also use the `__amrc` structure to check for the reason that `noseek` was converted to `seek`. A new field, `__amrc_noseek_to_seek`, is added to the `__amrc` structure for this purpose. The `__amrc_noseek_to_seek` field accepts the same values as the `__noseek_to_seek` field of the `fldata_t` structure, described above. It is updated in case of errors (for example, when seek is used, and the run-time library cannot handle the processing of a large format data set), by the z/OS XL C/C++ run-time DCB ABEND exit.

The BLOCKTOKENSIZE parameter in IGDSMSxx

The BLOCKTOKENSIZE parameter in the IGDSMSxx parmlib member of SYS1.PARMLIB allows you to specify whether applications working with large format data sets must indicate that large format data set processing is requested using the BLOCKTOKENSIZE parameter in the DCBE macro. BLOCKTOKENSIZE=REQUIRE in IGDSMSxx indicates that every open for large format data set must specify BLOCKTOKENSIZE=LARGE on the DCBE macro. BLOCKTOKENSIZE=NOREQUIRE in IGDSMSxx allows applications to access large format data sets without having to specify BLOCKTOKENSIZE=LARGE on the DCBE macro.

As of z/OS V1R8, the z/OS XL C/C++ run-time library will always specify BLOCKTOKENSIZE=LARGE on the DCBE macro when `noseek` is requested and honored. Therefore, `noseek` requests will always process a large format data set correctly, even when BLOCKTOKENSIZE=REQUIRE is specified in the IGDSMSxx.

16.1.4 The flockfile() family of functions

Starting with z/OS V1R8, z/OS XL C/C++ run-time library provides support for the `flockfile()` family of functions. These functions are defined in the Single UNIX Specification Version 3 (SUSv3). The functions provide for explicit application-level locking of `FILE*` objects. Multi-threaded applications can use these functions to define a sequence of mutually exclusive I/O statements that are executed as a unit on the same `FILE*` object. The following are thread-safe functions available with the new support:

- `void flockfile(FILE *file)`

  Acquires ownership of a FILE* object for the thread, and increments the internal lock count. If necessary, the function will wait for access to be granted.

- `int ftrylockfile(FILE *file)`

  This is the non-blocking version of flockfile(). It tries to acquire ownership of a FILE* object for the thread and increments the internal lock count if ownership is granted.

- `void funlockfile(FILE *file)`

  Decrements the internal lock count. When the count is reduced to zero, it relinquishes the ownership of a FILE* object granted to the thread.

Note: Misuse of the `flockfile()`, `ftrylockfile()`, and `funlockfile()` functions within an application can lead to deadlocks. It is your responsibility to serialize access to files properly when using these functions.

The following functions are the same as the existing `getc()`, `getchar()`, `putc()`, and `putchar()` functions, except that they are not thread-safe. However, they can be safely used in a multi-threaded program if called while the thread owns the `FILE*` object, such as after a successful `flockfile()` or `ftrylockfile()` call.

- `int getc_unlocked(FILE *stream)`

  Reads a single character from the current stream position and advances the stream position to the next character.

- `int getchar_unlocked(void)`

  This is identical to `getc_unlocked(stdin)`.

- `int putc_unlocked(int c, FILE *stream)`

  Converts `c` to an `unsigned char` and then writes `c` to the output stream at the current position.

- `int putchar_unlocked(int c)`

  This is identical to `putc_unlocked(c, stdout)`.

Example

Figure 16-1 on page 348 shows an example of an application with three threads writing to a file. They synchronize the write access to the file using the `flockfile()` and `funlockfile()` functions.

```
#define _UNIX03_SOURCE
#define _OPEN_THREADS

#include <stdio.h>
#include <pthread.h>

FILE *fp;

void *thread(char id) {
    int rc;

    /* get a lock of the file and write id */
    flockfile(fp);
    rc = putc_unlocked(id, fp);
    funlockfile(fp);

    pthread_exit(NULL);
}

void main() {

    pthread_t th1id;
    pthread_t th2id;
    pthread_t th3id;
    void      *ret;

    /* open the file */
    fp = fopen("./testlockfile.gil", "w+");

    /* create 3 threads */
    pthread_create(&th1id, NULL, thread, '1');
    pthread_create(&th2id, NULL, thread, '2');
    pthread_create(&th3id, NULL, thread, '3');

    /* wait for threads to terminate */
    pthread_join(th1id, &ret);
    pthread_join(th2id, &ret);
    pthread_join(th3id, &ret);

    fclose(fp);
}
```

Figure 16-1 Example of using the flockfile() functions

16.1.5 Reinitialization of a DLL's WSA

The writable static area (WSA) of a DLL is an area of memory that is modifiable during program execution. Typically, this area contains global variables and functions and variable descriptors for DLLs. The WSA is initialized when a DLL is loaded. In previous releases, an application had to reload the DLL in order to reinitialize its WSA.

z/OS V1R8 provides a new compiler-writer interface (CWI) service that allows the caller to reinitialize the WSA of a DLL without having to reload the DLL. The service can be used by any application that requires the reinitialization of a DLL WSA. It will provide better performance over reloading the DLL. The format of the new CWI is shown in Figure 16-2.

```
#include <edcwccwi.h>
int __static_reinit (int func_code, void *fcn);
```

Figure 16-2 Format of the __static_reinit() CWI

The *edcwccwi.h* file is the vendor interface header file. It is located in member EDCWCCWI of the SCEESAMP data set. In order to include *edcwccwi.h* in an application, the header file must be copied into a partitioned data set or an z/OS UNIX file system directory in which the C/C++ compiler will find it. *func_code* should be the value *__STATIC_REINIT_FULL* which performs a sequence of termination and initialization. *fcn* is a DLL handle pointer returned from a previous successful call to the *dllload()* or *dlopen()* functions.

An example of invoking `__static_reinit()` is shown in Figure 16-3.

```
/* Open a dynamic library and then reinitilize its WSA */

#include <edcwccwi.h>
#include <dlfcn.h>

void *handle;
int  eret;

handle = dlopen("mylib.so", RTLD_LOCAL | RTLD_LAZY);
...
eret = __static_reinit(__STATIC_REINIT_FULL, handle);
```

Figure 16-3 Invoking the __static_reinit() CWI

16.1.6 GWP compliance and euro locales

Based on annual updates for new language-territory combinations specified in the globalization white paper (GWP), and the new countries that joined the European Economic Community (EEC) in 2004, new euro locales are required.

The locales shown in Table 16-4 are added in z/OS V1R8 in support of the GWP and new EEC countries.

Table 16-4 The euro locales (EBCDIC) in support of the GWP and new ECC countries

Locale name in HFS	Country and language
Cs_CZ.IBM-1153@euro	Czech Republic, Czech
Cs_CZ.IBM-1165@euro	Czech Republic, Czech
Et_EE.IBM-1157@euro	Estonia, Estonian
Hu_HU.IBM-1153@euro	Hungary, Hungarian
Hu_HU.IBM-1165@euro	Hungary, Hungarian
Lv_LV.IBM-1156@euro	Latvia, Latvian
Lt_LT.IBM-1156@euro	Lithuania, Lithuanian

Locale name in HFS	Country and language
Pl_PL.IBM-1153@euro	Poland, Polish
Pl_PL.IBM-1165@euro	Poland, Polish
Sk_SK.IBM-1153@euro	Slovakia, Slovak
Sk_SK.IBM-1165@euro	Slovakia, Slovak
Sl_SI.IBM-1153@euro	Slovenia, Slovene
Sl_SI.IBM-1165@euro	Slovenia, Slovene

Although, when a country joins the EEC, dual currencies (the local national currency and the euro) have to be supported for a certain period of time. The POSIX model, as currently defined, does not allow the usage of dual currencies within the same locale. It only allows a single currency per locale.

Therefore, Table 16-5 shows pre-euro locales are added in z/OS V1R8 in support of the GWP and new EEC countries.

Table 16-5 Pre-euro locales (EBCDIC) in support for the GWP and new EEC countries

Locale name in HFS	Country and language
Cs_CZ.IBM-1153@preeuro	Czech Republic, Czech
Cs_CZ.IBM-1165@preeuro	Czech Republic, Czech
Et_EE.IBM-1157@preeuro	Estonia, Estonian
Hu_HU.IBM-1153@preeuro	Hungary, Hungarian
Hu_HU.IBM-1165@preeuro	Hungary, Hungarian
Lv_LV.IBM-1156@preeuro	Latvia, Latvian
Lt_LT.IBM-1156@preeuro	Lithuania, Lithuanian
Pl_PL.IBM-1153@preeuro	Poland, Polish
Pl_PL.IBM-1165@preeuro	Poland, Polish
Sk_SK.IBM-1153@preeuro	Slovakia, Slovak
Sk_SK.IBM-1165@preeuro	Slovakia, Slovak
Sl_SI.IBM-1153@preeuro	Slovenia, Slovene
Sl_SI.IBM-1165@preeuro	Slovenia, Slovene
Sv_SE.IBM-924@preeuro	Sweden, Swedish
Sv_SE.IBM-1143@preeuro	Sweden, Swedish

The z/OS XL C/C++ run-time library already contains euro locales for sweden. So only pre-euro locales are added in z/OS V1R8. The pre-euro locales are no more than a copy of the current locales. They are added so you can begin changing your applications source without having to rely on the base locale change when the euro becomes the new official currency for the countries involved. To set, change, or query a locale for a locale category or groups of categories you can use the `setlocale()` function.

z/OS V1R8 also adds support for the ASCII locales shown in Table 16-6 on page 351.

Table 16-6 ASCII locales in support for the GWP

Locale name in HFS	Country and language
bn_IN.UTF-8	India, Bengali
pa_IN.UTF-8	India, Punjabi

16.1.7 _CEE_ENVFILE_S environment variable

The `_CEE_ENVFILE` environment variable enables a list of environment variables to be set from a specified file, but it does not strip trailing white spaces from the `name=value` lines specified in the file. For this reason, it is not recommended to specify data sets with fixed-length record formats are not recommended on the `_CEE_ENVFILE` environment variable (they enable padding with blanks).

A new environment variable, `_CEE_ENVFILE_S`, is introduced in z/OS V1R8. `_CEE_ENVFILE_S` has the same behavior as `_CEE_ENVFILE`, except that it does strip trailing blanks from each `name=value` line read from the file. The behavior of `_CEE_ENVFILE` is not changed to allow existing applications to continue functioning in the same way as always.

It is possible to use the new `_CEE_ENVFILE_S` environment variable along with the existing `_CEE_ENVFILE` environment variable. In that case, environment variables specified by `_CEE_ENVFILE_S` will take precedence over those specified by `_CEE_ENVFILE`.

16.2 Compiler enhancements

This section describes enhancements to the z/OS XL C/C++ compiler. The main objective of the compiler enhancements in z/OS V1R8 is to provide better optimization and easier code portability.

16.2.1 New compiler options for optimization

Two new z/OS XL C/C++ compiler options are provided in z/OS V1R8 for code optimization. The new compiler options are `HOT` and `HGPR`.

The HOT compiler option

The `HOT` compiler option focuses on improving the code generated from z/OS XL C/C++ loop constructs. It indicates the compiler to perform high-order transformations on loops during optimization. This gives you the ability to generate more highly optimized code. In z/OS V1R8, the option will only affect the *interprocedural analysis (IPA)* compile phase processing.

IPA is a mechanism for performing optimizations across compilation unit boundaries. It also performs optimizations not otherwise available with the z/OS XL C/C++ compiler, such as inlining across compilation units, program partitioning, coalescing of global variables, code straightening, unreachable code elimination and call graph pruning of unreachable functions.

The default compiler option is `NOHOT`, which indicates no loop optimization during the IPA compile phase. In the z/OS UNIX environment, if the `xlc` utility flag option -O, -O2, or -O3 is specified, the default is `NOHOT`. If -O4 or -O5 is specified, the default is `HOT`.

The HGPR compiler option

The `HGPR` compiler option enables the compiler to exploit 64-bit general purpose registers (GPRs) for 32-bit applications running on z/Architecture hardware. `HGPR` stands for "High

GPR", which refers to the use of the additional GPR high order bits available to 64-bit instructions. Applications making use of `longlong` types should benefit from the use of native 64-bit instructions.

HGPR compiler option format

The complete format of the HGPR compiler option is shown in Figure 16-4. The default compiler option is `NOHGPR`.

```
HGPR(PRESERVE|NOPRESERVE)
```

Figure 16-4 Format of the HGPR compiler option

Use the `PRESERVE` suboption of the `HGPR` compiler option to instruct the compiler to preserve the high halves of the 64-bit GPRs that a function is using, by saving them in the prolog for the function and restoring them in the epilog. Because of performance considerations, the default suboption for `HGPR` is `NOPRESERVE`. In z/OS V1R8, the z/OS XL C/C++ compiler generates code that does not rely on the values of the high halves of the 64-bit GPRs. Therefore, the `PRESERVE` suboption is only necessary if the caller is not known to be z/OS XL C/C++ compiler generated code.

16.2.2 New language extensions

In z/OS V1R8, z/OS XL C/C++ provides new language extensions to help you port C/C++ applications to z/OS and from z/OS more easily.

The #include_next directive

Starting with z/OS V1R8, the C preprocessor supports the `#include_next` directive. `#include_next` is a C preprocessor directive in the GNU gcc extension used for specialized file inclusion. It instructs the C preprocessor to continue searching for the specified file name, and include the subsequent instance encountered after the current directory. The syntax of the `#include_next` directive is the same as the `#include` directive.

To better understand the usage of the `#include_next` directive, consider the example shown in Figure 16-5.

- The file testinc.h is located in /u/peleg/.

```
#include <stdio.h>
#include "testinc.h"

void main() {
    printloc();
}
```

Figure 16-5 /u/peleg/testinc.c

- The file testinc.h is located in /u/peleg/.

```
#include_next "testinc.h"
```

Figure 16-6 /u/peleg/testinc.h

- The file myinc.h is located in /u/peleg/.

```
void printloc() {
    printf("i am located in /u/peleg/myinc.h\n");
}
```

Figure 16-7 /u/peleg/myinc.h

- Another file named myinc.h is located in /u/peleg/include/.

```
void printloc() {
    printf("i am located in /u/peleg/include/myinc.h\n");
}
```

Figure 16-8 /u/peleg/include/myinc.h

- Compile testinc.c using this shell command:

  `cc -o testinc -I /u/peleg/include -2 testinc.c`

- The preprocessor locates testinc.h in /u/peleg/. testinc.h and contains a `#include_next` directive which directs the C preprocessor to search for myinc.h in the next include directory. Therefore, the preprocessor locates myinc.h in /u/peleg/include/. So the output of testinc is: shown in Figure 16-9.

```
i am located in /u/peleg/include/myinc.h
```

Figure 16-9 Output of testinc

Using the `#include_next` directive allows you to wrap 3rd party header files with your own implementation-specific behavior.

The #pragma extension

`The #pragma extension` is a new mechanism that allows you to restrict the use of the language features available when the `LANGLVL(EXTENDED)` compiler option is used to specific parts of your source code. As shown in Figure 16-10, you use the `#pragma extension` directive to mark the beginning of source code that will use `LANGLVL(EXTENDED)` language features, and the `#pragma extension(pop)` directive to mark its ending.

```
#pragma extension
...
/* LANGLVL(EXTENDED) code.. */
...
#pragma extension(pop)
```

Figure 16-10 Using the #pragma extension

`#pragma extension` provides a way for header file owners to use `LANGLVL(EXTENDED)` language features without requiring the users to compile their source code using `LANGLVL(EXTENDED)`. Note that this is not simply a mechanism to suppress warnings. At the point this directive is recognized, it will modify internal states of the compiler to use extended language level features, for example, recognizing keywords not acceptable in the ANSI language level.

Universal character name support for C++

In support of the C99 specification, z/OS XL C/C++ adds support of universal character names (UCN) in z/OS V1R8. The feature is enabled with the `LANGLVL(UCS)` compiler option. Using this feature allows you to specify a UCN in identifiers, character constants and string literals to designate characters that are not in the base character set.

The UCN `\Unnnnnnnn` represents the character that has an 8-digit short identifier `nnnnnnnn` (where 'n' represents a hexadecimal value). The UCN `\unnnn` represents the character that has a 4-digit short identifier `nnnn` (where `n` represents a hexadecimal value).

For example, the output of the code sample in Figure 16-11 is `New-York`.

```
printf("New\u1052York\n");
```

Figure 16-11 UCN usage in a string

The __func__ static variable support for C++

A new suboption is added to the `LANGLVL` compiler option to help debug C++ applications. The suboption is `C99__FUNC__`. When you specify the `LANGLVL(C99__FUNC__)` compiler option, the compiler automatically adds the following statement immediately after the opening brace of the each function definition, as shown in Figure 16-12.

```
static const char __func__[] = “function-name”;
```

Figure 16-12 Declaration added by LANGLVL(C99__FUNC__)

Where `function-name` is the current function's name.

The `LANGLVL(EXTENDED)` option group is changed to also include this feature.

16.2.3 High level assembler invocation utility

The **`as`** command is a z/OS UNIX shell utility that invokes the high-level assembler (HLASM) compiler. The output of the `as` utility is an object file. Using `as`, you can produce object files and use them as input for the z/OS XL C/C++ compiler. The syntax of the `as` command is shown in Figure 16-13.

```
as [--option[, option] ...] ...
   [-a[egimrsx][=file]] ...
   [--[no]gadata[=file]]
   [-moption]
   [-I PDS]
   [-o objectfile]
   [-v]
   [--[no]help]
   [--[no]verbose]
   file
```

Figure 16-13 Syntax of the as utility

To specify HLASM compiler options in the same form as the batch compiler, use the `--` parameter. You can also use the `-m` parameter to switch on or off compiler options. Specifying `-m` followed by the keyword itself, switches the option on. To switch the option off, specify the keyword prefixed with NO. Using the `-a` parameter you can turn on or off sections of the compiler listing and specify the file name to receive the output. If a file name is not specified, the output is directed to `stdout`. HLASM messages are always directed to `stderr` as well as the specified file name. The compiled file name can be a data set or a z/OS UNIX file. The output object file can also be a data set or a z/OS UNIX file.

For example, Figure 16-14 shows how to invoke the HLASM compiler for program asmp1.as, requesting the listing to include ESD and RLD information, be directed to file asmp1.list and that the object file name produced be asmp1.o.

```
as -aer=asmp1.list -o asmp1.o asmp1.as
```

Figure 16-14 Invoking the HLASM compiler from the shell

16.2.4 The 64-bit IPA link phase

During the IPA link phase, the compiler links the IPA objects that were produced by the IPA compile phase (along with non-IPA object files and load modules, if specified), does partitioning, performs optimizations, and generates the final object code. You invoke the IPA link phase by specifying the `IPA(LINK)` compiler option.

Starting with z/OS V1R8, the IPA link phase is a 64-bit process whose component name is CCNQIPA2. CCNQIPA2 allocates storage above the 2 GB bar, thus enabling the IPA link phase to handle very large applications. The actual memory used by CCNQIPA2 is very dependent on the size and complexity of the program being compiled and the level of optimization specified.

When running the z/OS XL C/C++ compiler in the z/OS UNIX environment, use the **`ulimit`** command with the `-M` parameter to set the storage limit above the bar. When running the compiler in batch or TSO/E, use the `MEMLIMIT` JCL parameter on the JOB or EXEC statement. Your system may also use the IEFUSI installation exit to limit program storage usage. If you use this exit to limit above the bar storage, you may need to update it.

16.2.5 Split IPA link phase listing

Normally, the default listing location for the z/OS XL C/C++ compiler is `stdout` or the SYSCPRT DD. You can instruct the compiler to output listing contents into a file by using the `LIST` or `INLRPT` (for the inlined functions report) options. These compiler options can be useful when the source file is large and there is a large amount of details in the listing. The IPA link phase processes the whole application, instead of just one source file. This sometimes causes the listing file to become too large, which makes it difficult to process with editing or search utilities.

The new `SPLITLIST` compiler option enables the z/OS XL C/C++ compiler to write the IPA Link phase listing to multiple PDS members, PDSE members, or a z/OS UNIX directory. The `NOSPLITLIST` compiler option disables the splitting action. The option is used only in the IPA link phase, and the location of the files must be specified by the `LIST` or `INLRPT` option. If the `LIST` or `INLRPT` option is not used to specify a location, the default is `stdout` or the SYSCPRT DD. Since `stdout` and SYSCPRT cannot be split, the compiler issues an error message if the `SPLITLIST` option is specified without the `LIST` or `INLRPT` options.

The z/OS XL C/C++ compiler will create several files in the PDS, PDSE or z/OS UNIX directory specified with the `SPLITLIST` option. Each member or file will contain a different section of the IPA link phase listing. Table 16-7 shows which of the created files contains which section of the IPA link phase listing.

Table 16-7 IPA link phase section names

File name created with SPLITLIST	IPA link phase listing section
part0	Partition 0 listing
part1	Partition 1 listing
part2	Partition 2 listing
objmap	Object file map
srcmap	Source file map
inlrpt	Inline report
options	IPA link options
cuopts	Compiler options map
globsym	Global symbols map
messages	Messages and summary

The file names are limited to 8 characters and are the same when the output is directed to a PDS, PDSE, or z/OS UNIX directory. In the example shown in Figure 16-15 we run the compiler with the SPLITLIST option, directing the output to the ./listing directory. All the files described in Table 16-7 are created under the ./listing directory.

```
c89 -Wl,I,'SPLITLIST,LIST(./listing)' main.o -o a.out
```

Figure 16-15 Example of running the z/OS XL C/C++ compiler with the SPLITLIST option

16.2.6 Support for standard exceptions of the C++ new operator

`NEWEXCP` is a new suboption of the `LANGLVL` compiler option introduced in z/OS V1R8. The purpose of the `LANGLVL(NEWEXCP)` compiler option is to control whether or not the C++ `new` operator will throw an exception when memory allocations fail. The default compiler option is `NONEWEXCP`, which means an exception is not thrown. When `LANGLVL(NEWEXCP)` is specified, the standard exception `std::bad_alloc` is thrown when the requested memory allocation fails.

This option applies only to the `throw` versions of the new operator. It does not apply to the `nothrow` versions of the `new` operator. Furthermore, `NEWEXCP` does not apply to the following:

- Class-specific `new` operators
- User-defined `new` operators
- `New` operators with placement arguments

Thus, only the following new operators are governed by this option:

- `void *operator new(size_t) throw(std::bad_alloc);`
- `void *operator new[](size_t) throw(std::bad_alloc);`

The same functionality as the `LANGLVL(NEWEXCP)` compiler option is provided with a pragma directive. This pragma takes precedence over the compiler option. It can be specified only once and before any statements in the source file. The syntax of the pragma directive is shown in Figure 16-16 on page 357.

```
#pragma operator_new(throwsexception)
```

Figure 16-16 Pragma directive to control new operator exceptions

16.2.7 GONUMBER support for AMODE 64 applications

When you specify the `LP64` compiler option, the compiler creates AMODE 64 code, utilizing the z/Architecture 64-bit instructions. `LP64` instructs the compiler to use 64-bits for the `long` data type and for pointers (as opposed to the ILP32 compiler option which instructs the compiler to use 32-bits for `int`, `long` and pointers). With `LP64`, the `int` data type remains 32-bits.

Prior to z/OS V1R8, the compiler used the DWARF Version 3 debugging format for AMODE 64 applications, since the existing format of the line numbers table was inadequate. With z/OS V1R8, specifying `GONUMBER` with the `LP64` option is supported. The `GONUMBER` compiler option generates a line numbers table that corresponds to the input source file for the object module. The inclusion of the line numbers table allows Language Environment to provide you with more details in the error trace back information when an exception occurs. For example, the source line number where exception occurs.

To support this new feature, Language Environment 64-bit CEEDUMP processing is enhanced. A new statement column is added to the traceback table in CEEDUMP to reflect the compile unit statement number. Additionally, a new traceback section named ‘Fully Qualified Names’ is added. This new section provides, when available, the fully qualified data set name or path name of the source code and load module name.

Figure 16-17 shows an example of the new Fully Qualified Names section.

```
Fully Qualified Names
DSA       Entry      Program Unit                                                Load Module
00000006  foobar3    AQMVSOE:/u/alfcar/tools/h3.c                                ./a.out
00000007  foobar2    AQMVSOE:/u/alfcar/tools/h3.c                                ./a.out
00000008  foobar     PLPSC://'ALFCAR.C(H2)'                                      ./a.out
00000009  main       AQMVSOE:/u/alfcar/tools/h1.c                                ./a.out
```

Figure 16-17 Example of the Fully Qualified Names section in CEEDUMP

16.2.8 Support for the DB2 V9 coprocessor

The `SQL` option enables the z/OS XL C/C++ compiler to process embedded SQL statements. As of z/OS V1R5, the compiler supports the interaction with the DB2 V7 and V8 SQL statement coprocessor. As of z/OS V1R8, the compiler also supports the DB2 V9 SQL statement coprocessor.

16.2.9 Integrated CICS translation

A new z/OS XL C/C++ compiler option is now available to allow for easier integration of C/C++ applications with CICS statements. The new `CICS` compiler option enables you to imbed CICS statements in your C/C++ source and pass them directly to the z/OS XL compiler without the need for an explicit preprocessing step. In addition, the integrated CICS translation support allows for comments and macros within imbedded CICS statements, which was not supported by the CICS translator. Integrated CICS translation is supported for use with CICS Transaction Server for z/OS 3.1 and above.

The `CICS` compiler option also allows passing suboptions to the integrated CICS translator in the form `CICS(suboption,...)`. The specified suboptions are passed directly to the integrated CICS translator and have no effect on the z/OS XL C/C++ compiler. This allows for easier migration to new CICS Transaction Server releases while maintaining compatibility with different z/OS XL C/C++ compiler versions. As an alternative, you can use the new `#pragma XOPTS` directive to pass suboptions to the integrated CICS translator.

Using the integrated CICS translator can make it easier to maintain C/C++ applications, especially in the z/OS UNIX environment as it saves you running the CICS translator step.

17

Language Environment enhancements

This chapter contains a description of the new features in Language Environment in z/OS V1R8.

The following topics are discussed:

- Dynamic allocation of SYSMDUMP
- Support for non-XPLink to XPLink function pointers
- Tracing of XPLink/non-XPLink transitions in applications
- New callable services

17.1 Dynamic allocation of SYSMDUMP

Not all jobs run with a SYSABEND, a SYSUDUMP, or a SYSMDUMP DD statement. If these jobs ABEND, most of the diagnosis information is not saved, and cannot be used to diagnose the problem causing the ABEND. In some cases, a job only rarely ABENDs, which makes it more difficult to obtain meaningful information to help diagnose the problem.

Starting with z/OS V1R8, Language Environment dump processing is enhanced to detect that a dump data set was not specified using SYSABEND, SYSUDUMP, or SYSMDUMP and that one needs to be allocated dynamically. This will allow you to receive a dump containing diagnosis information that would otherwise be lost. The dump produced dynamically by Language Environment run-time is an unformatted dump, the same as the SYSMDUMP dump. Once the dump is produced, you can use IPCS to format it.

17.1.1 DYNDUMP run-time options

The support for dynamic allocation of dump data sets is provided using the new Language Environment run-time `DYNDUMP` option. The syntax of the `DYNDUMP` run-time options is shown in Figure 17-1.

```
DYNDUMP(hlq,NODYNAMIC|DYNAMIC|FORCE|BOTH,TDUMP|NOTDUMP)
```

Figure 17-1 Syntax of the DYNDUMP run-time options

The first parameter of the `DYNDUMP` run-time options specifies a high level qualifier (HLQ) for the dump data set name that will be dynamically allocated by Language Environment. The HLQ is limited to 26 characters, including the dot separators, and must be a valid z/OS data set name. z/OS UNIX file names are not supported. Language Environment will concatenate a time stamp consisting of the julian date and time the dump was taken. If you specify less than 26 characters on the HLQ and there is enough room left in the data set name, Language Environment will also concatenate the job name (as taken from the value of the &JOBNAME system symbol) or the process id.

Multilevel security considerations

If your system is using multilevel security, the SECLABEL will always be used as the second qualifier. In a system with multilevel security, if you specify an HLQ containing multiple qualifiers, only the first qualifier will be used, followed by the SECLABEL. To summarize, the format of the data set name is:

- When the application is not exec()ed and is not multilevel security:

  `hlq.Djjj.Thhmmsst.jobname`

- When the application is exec()ed and not multilevel security:

  `hlq.Djjj.Thhmmsst.Ppppppppp`

- When the application is multilevel security and not exec()ed:

  `hlq.MLS-level.Djjj.Thhmmsst.jobname`

- When the application is both multilevel security and exec()ed:

  `hlq.MLS-level.Djjj.Thhmmsst.Ppppppppp`

17.1.2 Specifying the DYNDUMP run-time options

The syntax of the `DYNDUMP` run-time options is shown in the following sections.

HLQ option

You can specify two different keywords for the HLQ parameter, as follows:

***USERID**	In that case Language Environment is to use the user id associated with the job step task as the HLQ for the dump data set. If HLQ is not specified (left blank), `*USERID` is used.
***TSOPREFIX**	In that case Language Environment is to use the TSO/E prefix. The prefix is only valid in a TSO/E environment. If the job is running in an environment where the TSO/E prefix is not available, the user id will be used.

Second option for ABEND actions

The second parameter of the `DYNDUMP` run-time options specifies the action that Language Environment is to take in case of U4039 ABENDs:

NODYNAMIC	Instructs Language Environment not to create a dump in case of a U4039 ABEND.
DYNAMIC	Instructs Language Environment to dynamically allocate a dump data set dynamically according to the HLQ parameter and write the dump to it, but only if SYSABEND, SYSUDUMP, and SYSMDUMP were not specified.
FORCE	Instructs Language Environment to allocate a dump data set and write the dump to it even if SYSABEND, SYSUDUMP, or SYSMDUMP were specified.
BOTH	The same as `FORCE`, but in addition to the IPCS readable dump, Language Environment will also write a transaction dump.

Third option for other ABENDs

The last parameter of the `DYNDUMP` run-time option indicates Language Environment what action to take in case of any other U40xx ABENDs. There are only two options for this parameter:

TDUMP	Instructs Language Environment to dynamically allocate a dump data set according to the HLQ parameter and write a dump to it.
NOTDUMP	Instructs Language Environment to dynamically allocate a dump data set. In that case, diagnosis information will not be saved.

17.1.3 Dynamic allocation processing

When Language Environment starts dynamic allocation processing for an ABEND, the following message is issued:

```
CEE3798I Attempting to take a dump for ABEND Uxxxx to data set: data set
```

After a dump is successfully allocated and written by Language Environment, the following message is issued:

```
CEE3797 Language Environment has dynamically created a dump.
```

In case of an error while allocating the dump data set, Language Environment issues the following message:

```
CEE3796I An attempt to dynamically take a dump was not successful. The error
return code was returncode and the reason code was reasoncode
```

These messages are issued using a WTO and not normal Language Environment message processing, so that they appear on the console and job log. Other messages might be issued by RACF or the IEATDUMP service in case of error.

17.2 Support for non-XPLink to XPLink function pointers

Prior to z/OS V1R8, function pointers created on a non-XPLink program could not be used on an XPLink program, unless the function pointer was passed explicitly as a function argument. Passing the function pointer as a member of a structure, for example, was not supported and would cause an operation exception when used to call a function. Writing programs that do support mixing of non-XPLink and XPLink function pointers across DLLs was still possible prior to z/OS V1R8, but it required additional application code in the XPLink application, such as using the `__bldfxfd()` CWI function and identifying each function pointer with a `__callback` type declaration, which had performance implications.

z/OS V1R8 removes this restriction and allows a function pointer to be passed from a non-XPLink program to an XPLink program. With the new support, an XPLink program is able to correctly use a non-XPLink function pointer in all situations; for example, in the following scenarios which were previously unsupported:

- As a global variable
- Part of a structure that is passed as a parameter
- Part of a union that is passed as a parameter
- Passing it as an explicit return value

Function pointer limitations

However, the following limitations still exist:

- DLLs must be built using the program management binder.
- The non-XPLink source module must be compiled using the `DLL` and `GOFF` z/OS XL C/C++ compiler options and without the `XPLINK` or `LP64` compiler options.
- Non-XPLink assembler DLLs are not supported.

This support does not require any changes to the application code. It only requires the non-XPLink programs to be recompiled with the `DLL` and `GOFF` compiler options. But since compatibility is only supported across DLL boundaries, the non-XPLink compiled program and the XPLink compiled programs must be in separate DLLs.

New support example

Consider the following example, which shows a usage scenario for the new support:

- fp.c is a non-XPLink program. It calls the bar() function defined in an XPLink DLL and passes it a pointer to the non-XPLink function foo().

```
#include <stdio.h>

struct NonXPLINKfunc {
    float  parm;
    int  (* func_ptr) (float);  /* function pointer */
};

int foo(float a) { return (int) a*a; }

/* function exported from the XPLink DLL */
extern int bar(struct NonXPLINKfunc);

int main() {

    /* func_ptr points to a Non-XPLink function */
    struct NonXPLINKfunc s = { 2.0, &foo };

    /* calling the XPLink DLL function */
    printf("bar=%d\n",  bar(s));
}
```

Figure 17-2 Source code for fp.c

- dllfp.c is an XPLink DLL. It uses the function pointer created by fp.c to call the foo() function. This was not supported in previous releases.

```
struct NonXPLINKfunc {
    float parm;
    int   (* func_ptr) (float); /* the function pointer */
};

int bar(struct NonXPLINKfunc p) {

    /* calling the callback via the pointer */
    return p.func_ptr(p.parm);
}

#pragma export(bar)
```

Figure 17-3 Source code for dllfp.c

- To compile and link dllfp.c as an XPLink DLL, and compile fp.c with the DLL and GOFF compiler options, see Figure 17-4, which shows a program that is to run in an XPLink environment.

```
PELEG @ SC75:/u/peleg>cc -o dllfp -Wc,XPLINK -Wl,XPLINK,DLL dllfp.c
PELEG @ SC75:/u/peleg>cc -o fp -Wc,DLL,GOFF fp.c dllfp.x
PELEG @ SC75:/u/peleg>export _CEE_RUNOPTS='XPLINK(ON)'
PELEG @ SC75:/u/peleg>./fp
bar=4
PELEG @ SC75:/u/peleg>
```

Figure 17-4 Running fp

17.3 Tracing of XPLink/non-XPLink transitions

Using XPLink can greatly increase the performance of C/C++ applications. However, transitions between XPLink and non-XPLink code can reduce the application's effectiveness. It is often difficult or impossible to determine where in the application these transitions are occurring. To help you trace these transitions, Language Environment in z/OS V1R8 provides a new trace level for the `TRACE` run-time option.

The `TRACE` run-time option controls run-time library tracing activity, the size of the in-storage trace table, the type of trace events to record, and it determines whether a dump containing, at a minimum, the trace table should be unconditionally taken when the application terminates. The format of the `TRACE` run-time option is shown in Figure 17-5.

```
TRACE(ON|OFF,table_size,DUMP|NODUMP,LE=0|1|2|3|8|20)
```

Figure 17-5 Format of the TRACE run-time option

New trace records

The new trace level is `LE=20`, which provides tracing of transitions between XPLink code to non-XPLink code and between non-XPLink code to XPLink code in an application. Tracing is available only for 31-bit applications. When specifying `LE=20`, two new trace records might be generated:

- Trace record 00000007 - XPLink code calls non-XPLink code. Entry format:

  ```
  ModuleNameOfCallingFunction:NameOfCallingXplinkFunction
  -->ModuleNameOfCalledFunction:NameOfCalledNonXplinkFunction
  ```

- Trace record 00000008 - non-XPLink code calls XPLink code. Entry format:

  ```
  ModuleNameOfCallingFunction:NameOfCallingNonXplinkFunction
  -->ModuleNameOfCalledFunction:NameOfCalledXplinkFunction
  ```

For both entry types, the module name is 16 bytes long and the function name is 32 bytes long. If the module name is longer than 16 bytes or the function name is longer than 32 bytes, an extra trace entry is taken. For module names longer than 32 bytes or function names longer than 64, truncation occurs and only the first 32 or 64 bytes appear in the trace table entry. In some case, such as when a DLL is freed, a module name might not be located. The trace entry displays the module name as UNKNOWN, in such a case.

LE trace output

Figure 17-6 on page 365 shows the output of a Language Environment trace containing the new trace entries 00000007 and 00000008. For example, when looking at the first entry in the trace table, we see a transition from non-XPLink code to XPLink code, indicated by entry type 00000008. The trace entry shows function EDCZHINV, the Language Environment XPLink main() invocation routine in module CELHV003, calls the user XPLink function main() in module a859c01x.

```
Displacement                        Trace Entry in Hexadecimal                                  Trace Entry in EBCDIC
------------ ------------------------------------------------------------------------   --------------------------------
  +000000    Time 23.04.44.393727    Date 2005.04.29    Thread ID... 2050D5F000000000
  +000010    Member ID.... 03    Flags..... 000000    Entry Type..... 00000008
  +000018    C3C5D3C8 E5F0F0F3 40404040 40404040  7AC5C4C3 E9C8C9D5 E5404040 40404040  |CELHV003        :EDCZHINV        |
  +000038    40404040 40404040 40404040 40404040  4060606E 81F8F5F9 83F0F1A7 40404040  |                 -->a859c01x     |
  +000058    40404040 7A948189 95404040 40404040  40404040 40404040 40404040 40404040  |    :main                        |
  +000078    40404040 404040F1                                                         |       1                         |

  +000080    Time 23.04.44.399327    Date 2005.04.29    Thread ID... 2050D5F000000000
  +000090    Member ID.... 03    Flags..... 000000    Entry Type..... 00000007
  +000098    C3C5C5D7 D3D7D2C1 40404040 40404040  7AC3C5C5 D7C8E3D3 C3404040 40404040  |CEEPLPKA        :CEEPHTLC        |
  +0000B8    40404040 40404040 40404040 40404040  4060606E C3C5C5D7 D3D7D2C1 40404040  |                 -->CEEPLPKA     |
  +0000D8    40404040 7AC3C5C5 D7E3D3D6 D9404040  40404040 40404040 40404040 40404040  |    :CEEPTLOR                    |
  +0000F8    40404040 404040F1                                                         |       1                         |

  +000100    Time 23.04.44.425623    Date 2005.04.29    Thread ID... 2050D5F000000000
  +000110    Member ID.... 03    Flags..... 000000    Entry Type..... 00000007
  +000118    81F8F5F9 83F0F1A7 40404040 40404040  7A948189 95404040 40404040 40404040  |a859c01x        :main            |
  +000138    40404040 40404040 40404040 40404040  4060606E 81F8F5F9 83F0F240 40404040  |                 -->a859c02      |
  +000158    40404040 7A86A495 83F16D84 F1404040  40404040 40404040 40404040 40404040  |    :func1_d1                    |
  +000178    40404040 404040F1                                                         |       1                         |

  +000180    Time 23.04.44.427092    Date 2005.04.29    Thread ID... 2050D5F000000000
  +000190    Member ID.... 03    Flags..... 000000    Entry Type..... 00000008
  +000198    81F8F5F9 83F0F240 40404040 40404040  7A86A495 83F16D84 F1404040 40404040  |a859c02         :func1_d1        |
  +0001B8    40404040 40404040 40404040 40404040  4060606E C3C5D3C8 E5F0F0F3 40404040  |                 -->CELHV003     |
  +0001D8    40404040 7A979989 95A38640 40404040  40404040 40404040 40404040 40404040  |    :printf                      |
  +0001F8    40404040 404040F1
```

Figure 17-6 Example of LE=20 trace entries

17.4 New Language Environment callable services

z/OS V1R8 provides new callable services for Language Environment applications. These callable services can be invoked from applications generated with the following IBM compiler products:

- IBM z/OS XL C/C++
- C/C++ for MVS/ESA
- IBM SAA® AD/Cycle® C/370™
- Enterprise COBOL for z/OS
- Enterprise PL/I for z/OS
- IBM COBOL for OS/390 & VM
- IBM COBOL for MVS & VM
- IBM PL/I for MVS & VM

You can also invoke the Language Environment callable services from assembler programs that use the CEEENTRY and associated macros.

Restriction: Language Environment callable services are not supported for AMODE 64 applications.

In the rest of this section we describe the new Language Environment callable services introduced in z/OS V1R8. For more information about Language Environment callable services, see *z/OS Language Environment Programming Reference*, SA22-7562.

CEE3AB2 callable service

This new callable service is an enhancement of the CEE3ABD callable service. CEE3AB2 provides you with a means to terminate an enclave, supplying your own defined ABEND code and reason code. It also allows you to request the type of cleanup processing and the type of dump suitable for your needs. CEE3AB2 does not return control to the application. The format of CEE3AB2 is shown in Figure 17-7.

```
CEE3AB2(abendcode,reasoncode,cleanup)
```

Figure 17-7 Format of the CEE3AB2 callable service

CEE3AB2 callable service parameters

The syntax of the CEE3AB2 callable service is as follows:

abendcode `abendcode` is a full word integer, no greater than 4095, specifying the ABEND code that the application is to terminate with. Under CICS, this integer ABEND code is converted to EBCDIC.

reasoncode `reasoncode` is a full word integer, specifying the reason code that the application is to terminate with. The default reason code is 0.

cleanup `cleanup` is a full word integer between 0 and 5, indicating the type of cleanup processing for the application. The meaning of the 0-5 values is:

0	Issue the ABEND without cleanup.
1	Issue the ABEND with normal enclave termination processing.
2	Issue the ABEND with enclave termination processing honoring the TERMTHDACT run-time option for taking a system dump of the user address space but suppressing the CEEDUMP.
3	Issue the ABEND with enclave termination processing suppressing both the CEEDUMP and system dump if requested by the TERMTHDACT run-time option.
4	Issue the ABEND with enclave termination processing honoring the TERMTHDACT run-time option for taking a CEEDUMP but suppressing the system dump.
5	Issue the ABEND with enclave termination processing forcing a system dump of the user address space and suppressing the CEEDUMP.

If an illegal value is specified on the `cleanup` parameter, the ABEND is issued without cleanup.

CEE3PR2 callable service

CEE3PR2 is an extension of the existing CEE3PRM callable service. It allows you to query the parameter string specified to the program at invocation. CEE3PR2 returns the length of the parameters strings in an output parameter, unlike CEE3PRM which is limited to a fixed 80 characters length parameters string.

The format of the CEE3PR2 callable service is shown in Figure 17-8.

```
CEE3PR2(stringlength,parmstring,feedback)
```

Figure 17-8 Format of the CEE3PR2 callable service

CEE3PR2 parameters

Following is the format of the CEE3PR2 parameters:

stringlength As input, `stringlength` is a full word integer specifying the size of the buffer pointed by the `parmstring` parameter. When `stringlength` is 0 or less, it indicates CEE3PR2 to return the whole parameters string. As output, the `stringlength` parameter holds the actual size of the whole parameters string.

parmstring `parmstring` is the address of a buffer in which CEE3PR2 returns the parameters string specified to the program at invocation.

feedback `feedback` is an integer feedback code returned by CEE3PR2. Feedback code CEE000 means the service was successful. Feedback code CEE3K3 means the parameters string returned by CEE3PR2 was truncated because of insufficient buffer size.

CEE3INF callable service

CEE3INF is a new callable service that returns information about the environment the application is running in. The environmental information include the system and subsystems under which the application is running, the Language Environment environment, initialized languages and Language Environment version. For example, you can use the CEE3INF callable service to determine at run-time whether your application is running under CICS or in a batch environment.

CEE3INF callable service parameters

The format of the CEE3INF callable service is shown in Figure 17-9.

```
CEE3INF(subsys,envinfo,memberid,gpid,feedback)
```

Figure 17-9 Format of the CEE3INF callable service

subsys `subsys` is the address of a full word returned by CEE3INF containing information about the system and the subsystem the application is running under. The format of the returned full word is:

0	Currently executing in the CICS environment
1	Currently executing in a CICS_PIPI environment
2-3	Reserved for other specific CICS environments
4	Currently executing in a TSO environment
5	Currently executing in a Batch environment
6	Currently executing in a z/OS UNIX environment
7-28	Reserved for future use
29	Currently executing on z/VSE™
30	Currently executing on z/OS
31	Currently executing on z/OS.e

envinfo `envinfo` is the address of a full word returned by CEE3INF containing information representing the available Language Environment environments for the application. The format of the returned full word is:

0	Currently executing in a PIPI environment
1	Currently executing in a PIPI-Main environment
2	Currently executing in a PIPI-Sub environment
3	Currently executing in a PIPI-Subdp environment
4	Currently executing in a PICI environment
5	Currently executing in a nested enclave
6	LRR is active in the current enclave
7	Run-time reuse is active in the current environment
8	XPLINK(ON) is in effect in the current enclave
9	POSIX(ON) RTO in effect in the current enclave
10	At least one pthread has been created in this enclave
11	Currently executing on the IPT
12	Multithreaded fork is in effect in the current enclave
13-14	AMODE, B'00' means AMODE 24 and B'10' means AMODE 31
15-31	Reserved for future use

memberid `memberid` is the address of a full word returned by CEE3INF containing information about the member languages that are initialized in the enclave. The format of the returned full word is:

0-2	Reserved
3	C/C++
4	Reserved
5	COBOL
6	Reserved
7	Fortran
8-9	Reserved
10	PL/I
11	Enterprise PL/I
12-23	Reserved

gpid `gpid` is the address of a full word returned by CEE3INF which represents the version of Language Environment that created the enclave. The full word is in the format PPVVRRMM, where PP is the product number, VV is the version number, RR is the release number and MM is modification level.

feedback `feedback` is an integer feedback code returned by CEE3INF.

CEEENV callable service

The CEEENV callable service provides a language neural mechanism for manipulating z/OS UNIX environment variables. Using CEEENV you can do the following:

- Obtain the value for an existing environment variable.
- Create a new environment variable with a value.

- Clear all environment variables.
- Delete an existing environment variable.
- Overwrite the value for an existing environment variable.

CEEENV callable service parameters

The format of the CEEENV callable service is shown in Figure 17-10.

```
CEEENV(functioncode,namelength,name,valuelength,value,feedback)
```

Figure 17-10 Format of the CEEENV callable service

functioncode `functioncode` is a full-word integer describing the desired function:

1 Searches the environment table for environment variable specified by name and if found returns a pointer to the value.

2 Adds an environment variable to the environment table. Does not overwrite an existing environment variable.

3 Clears all environment variables from the environment table.

4 Deletes an environment variable from the environment table.

5 Overwrites an existing environment variable in the environment table and adds the environment variable if it does not exist.

namelength `namelength` is a full-word integer containing the length of the `name` variable. `name` is the address of a buffer holding the name of the environment variable you wish to perform one the CEEENV functions on. The `name` value is ignored if `functioncode` is 3.

Note: Applications should not define environment variables that begin with the characters “_BPXK_”, “_EDC_” or ”_CEE_”. This might cause conflicts with variable names reserved for z/OS, that begin with those characters.

valuelength `valuelength` is a full-word integer containing the length of the `value` variable. value is the address of a buffer holding the value for the environment variable to be set or modified. value is ignored if functioncode is 1,3 or 4.

feedback `feedback` is an integer feedback code returned from CEEENV. CEE000 means the service completed successfully, CEE51O means there was not enough memory available, CEE51P means that a bad input character detected for name or value, CEE51Q means that a bad address was detected for the ENVAR anchor or environment variable array, CEE51R means a parameter to the environment variable processing routine contained a value that was not valid and CEE51S means the specified environment variable name already exists.

CEE3ABD callable service

CEE3ABD is an existing Language Environment variable that is enhanced in z/OS V1R8. In previous releases, CEE3ABD was incapable of controlling the type of dumps that are generated during enclave termination processing. In z/OS V1R8, new cleanup values are added to support different types of enclave termination.

CEE3ABD callable service parameters

The format of the CEE3ABD callable service is shown in Figure 17-11.

```
CEE3ABD(abendcode,cleanup)
```

Figure 17-11 Format of the CEE3ABD callable service

abendcode `abendcode` is a full-word integer containing the desired ABEND code for the application. The value must not be greater than 4095.

cleanup `cleanup` is a full-word integer between 0-5 indicating the type of cleanup processing required for the ABEND. The acceptable values are:

0 Issue the ABEND without clean-up.

1 Issue the ABEND with normal enclave termination processing.

2 Issue the ABEND with enclave termination processing honoring the TERMTHDACT run-time option for taking a system dump of the user address space but suppressing the CEEDUMP.

3 Issue the ABEND with enclave termination processing suppressing both the CEEDUMP and system dump if requested by the TERMTHDACT run-time option.

4 Issue the ABEND with enclave termination processing honoring the TERMTHDACT run-time option for taking a CEEDUMP but suppressing the system dump.

5 Issue the ABEND with enclave termination processing forcing a system dump of the user address space and suppressing the CEEDUMP.

Notice that CEE3ABD does not provide a way to specify a reason code with your ABEND. If you need to specify a reason code with your ABEND, use the CEE3AB2 callable service. When the TRAP(OFF) run-time option is specified, CEE3ABD always behaves as if cleanup code 0 was specified. CEE3ABD does not return control to the calling application.

18

GRS enhancements

This chapter describes the enhancements to global resource scheduling (GRS) in z/OS V1R8, as follows:

- Setting address space ENQ limits
- Performance of resource contention
- SETGRS operator command enhancements
- GRS storage constraint
- New GRS health checks

18.1 Global resource serialization (GRS)

In a star complex, global resource serialization uses an XES lock structure to serialize requests for global resources. All systems in a star complex must be members of the same sysplex and be connected to a coupling facility containing the global resource serialization lock structure, ISGLOCK, to manage contention for global resources. For practical purposes, where global resource serialization star complex is concerned, the terms sysplex and complex are synonymous. No channel-to-channel (CTC) connection of systems, other than those managed by XCF, are supported by global resource serialization in a star complex.

ISGLOCK lock structure

When a system in a star complex issues an ENQ, DEQ, ISGENQ, or RESERVE request for a global resource, global resource serialization converts the request to a lock request against the ISGLOCK lock structure. Global resource serialization uses the ISGLOCK lock structure to coordinate the requests to ensure proper resource serialization across all systems in the complex. The status of each request is returned to the system that originated the request. Based on the results of these lock requests, global resource serialization will respond to the requester with the outcome of the serialization request.

Global ENG/DEQ processing overview

In a star complex, requests for ownership of global resources will be handled through ISGLOCK, the lock structure, on a coupling facility that is fully connected with all the systems in the sysplex. Global resource serialization uses the ISGLOCK lock structure to reflect a composite system level view of the interest in every global resource, for which there is at least one requester. In general, global resource serialization alters this composite view each time you change the set of requesters for the resource.

Each time an ENQ request is received, global resource serialization processing analyzes the state of the resource request queue for the resource. If the new request alters the composite state of the queue, an IXLLOCK macro request is made to reflect the changed state of the resource for the requesting system. If the resource is immediately available, the requester is then granted ownership of the resource.

If the resource is not immediately available, global resource serialization will maintain the request in the waiting state. When the appropriate DEQ request is received, global resource serialization will either resume or post the requester (depending on the ENQ options).

18.1.1 Setting address space ENQ limits

GRS enforces a limit on the maximum number of concurrent ENQ, ISGENQ, RESERVE, GQSCAN, and ISGQUERY requests issued by an address space. The general purpose of enforcing a limit is to prevent a run away ENQ address space from exhausting GRS private storage. In z/OS releases earlier than V1R8, the maximum number of concurrent requests is:

- 4,096 for unauthorized requests.
- 250,000 for authorized requests.

These are system-wide maximums. They apply to all address spaces in the system. Note that ENQ RET=CHNG, ENQ RET=TEST, and their equivalent ISGENQ requests do not increase the number of concurrent requests. However, GQSCAN and ISGQUERY REQINFO=QSCAN requests, where the filled answer area results in a request token for continuation, do increase the number concurrent requests.

Current application ENQ problems

As installations grow and applications get more complex, the number of concurrent ENQ requests may rise to exceed the maximum values. An unauthorized workload such as CICS may already exceed the current maximum values in your environment. Future authorized workloads such as DB2 V8 may pose a similar problem. Some installations are already zapping the maximum values to allow for a greater number of concurrent ENQs than the system-wide defaults.

New limits for ENQs

GRS is enhanced in z/OS V1R8 to allow greater flexibility and control over the system-wide maximums. In z/OS V1R8, the default values are:

- 16,384 for unauthorized requests
- 250,000 for authorized requests

z/OS V1R8 allows you to set the system-wide maximum values. Furthermore, it is now possible for an authorized caller to set its own address space-specific limits.

Current installation modifications

In order to provide support for setting the maximums, some non-programming interfaces in z/OS V1R8 are changed. If you are already zapping the maximums at your installation (a non-programming interface), the changes in z/OS V1R8 may cause some incompatibilities. Prior to z/OS V1R8, it was necessary to zap the GVTCREQ and GVTCREQA fields in the GVT to change the maximums. When you upgrade to z/OS V1R8, there are new mechanisms for setting the maximums.

Note: The GVTCREQ and GVTCREQA fields are no longer zappable in z/OS V1R8.

To check whether you are currently zapping the maximums in the GVT at your installation, follow this procedure (remember this is only true for pre-z/OS V1R8 releases):

- From ISPF, you can use ISRDDN, as follows:

  ISRDDN provides a BROWSE command for browsing storage and loaded modules. We use the BROWSE command to check the values in the GVT fields.

- From the ISRDDN command line, enter this command:

  ```
  BROWSE 0.+10?+1B0?
  ```

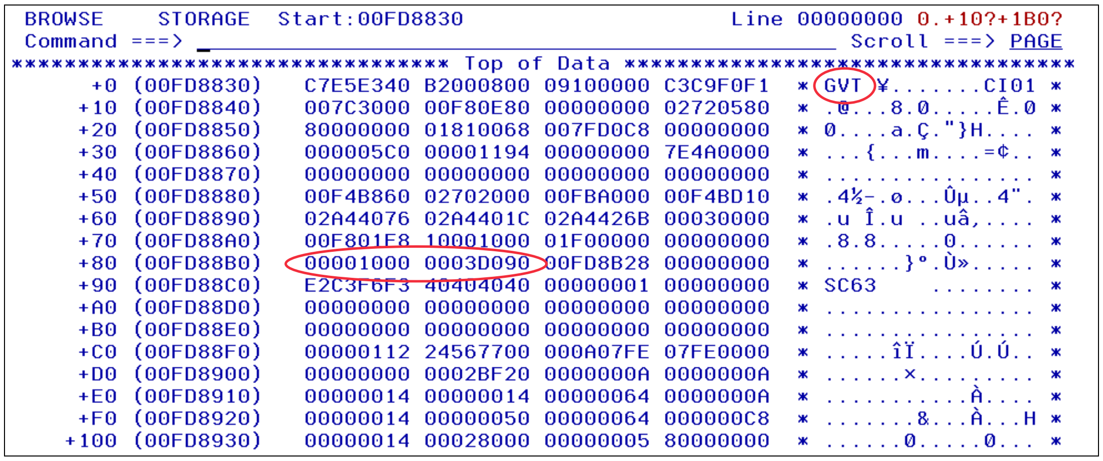
  This command uses the pointer at offset X'1B0' of the CVT to get to the GVT. The pointer to the CVT is located at offset X'10' in virtual storage. After issuing the command, a browse screen is displayed.

- To verify you are really looking at the GVT, check for the "GVT" eye catcher at offset X'0', as shown in Figure 18-1 on page 374.
- Check the GVTCREQ field value at offset X'80' and the GVTCREQA field value at offset 84.

  If you see the default values in the GVTCREQ and GVTCREQA fields, then you are not zapping the GVT in your installation. When using the default values, the GVTCREQ field holds a value of 4,096 (X'1000') and the GVTCREQA field holds a value of 250,000 (X'3D090'), as displayed in Figure 18-1 on page 374.

- Check the GVTCREQ field value at offset X'80' and the GVTCREQA field value at offset 84.

  If you see the default values in the GVTCREQ and GVTCREQA fields, then you are not zapping the GVT in your installation. When using the default values, the GVTCREQ field holds a value of 4,096 (X’1000’) and the GVTCREQA field holds a value of 250,000 (X’3D090’), as displayed in Figure 18-1.



```
 BROWSE     STORAGE   Start:00FD8830                   Line 00000000 0.+10?+1B0?
 Command ===> _                                                Scroll ===> PAGE
******************************** Top of Data **********************************
      +0 (00FD8830)    C7E5E340 B2000800 09100000 C3C9F0F1  * GVT ¥.......CI01 *
     +10 (00FD8840)    007C3000 00F80E80 00000000 02720580  * .@...8.0.....Ê.0 *
     +20 (00FD8850)    80000000 01810068 007FD0C8 00000000  * 0....a.Ç."}H.... *
     +30 (00FD8860)    000005C0 00001194 00000000 7E4A0000  * ...{...m....=¢.. *
     +40 (00FD8870)    00000000 00000000 00000000 00000000  * ................ *
     +50 (00FD8880)    00F4B860 02702000 00FBA000 00F4BD10  * .4½-.ø...Ûµ..4". *
     +60 (00FD8890)    02A44076 02A4401C 02A4426B 00030000  * .u Î.u ..uâ,.... *
     +70 (00FD88A0)    00F801F8 10001000 01F00000 00000000  * .8.8.....0...... *
     +80 (00FD88B0)    00001000 0003D090 00FD8B28 00000000  * ......}°.Ù»..... *
     +90 (00FD88C0)    E2C3F6F3 40404040 00000001 00000000  * SC63    ........ *
     +A0 (00FD88D0)    00000000 00000000 00000000 00000000  * ................ *
     +B0 (00FD88E0)    00000000 00000000 00000000 00000000  * ................ *
     +C0 (00FD88F0)    00000112 24567700 000A07FE 07FE0000  * .....îÏ....Ú.Ú.. *
     +D0 (00FD8900)    00000000 0002BF20 0000000A 0000000A  * ......×......... *
     +E0 (00FD8910)    00000014 00000014 00000064 0000000A  * ...........À.... *
     +F0 (00FD8920)    00000014 00000050 00000064 000000C8  * .......&...À...H *
    +100 (00FD8930)    00000014 00028000 00000005 80000000  * ......0.....0... *
```

Figure 18-1 Verifying GVT fields using the ISRDDN BROWSE command

ISGADMIN service

The ISGADMIN service allows you to programmatically change the maximum number of concurrent ENQ, ISGENQ, RESERVE, GQSCAN, and ISGQUERY requests in an address space. This is useful for subsystems such as CICS and DB2, which have large numbers of concurrently outstanding ENQs, query requests, or both. Using ISGADMIN you can set the maximum limits of unauthorized and authorized concurrent requests. It is impossible to set the maximums lower than the system-wide default values.

New keywords in the GRSCNFxx parmlib member

To allow you to control the system-wide maximums, two new keywords are added to the GRSCNFxx parmlib member:

ENQMAXU(value)	Identifies the system-wide maximum of concurrent ENQ requests for unauthorized requesters. The ENQMAXU range is 16,384 to 99,999,999. The default is 16,384.
ENQMAXA(value)	Identifies the system-wide maximum of concurrent ENQ requests for authorized requesters. The ENQMAXA range is 250,000 to 99,999,999. The default is 250,000.

SETGRS ENQMAXU and SETGRS ENQMAXA operator commands

z/OS V1R8 also provides operator commands to allow you to dynamically set the system-wide maximums. Use the SETGRS ENQMAXU command to set the system-wide maximum number of concurrent unauthorized requests, and the SETGRS ENQMAXA command to set the system-wide maximum number of concurrent authorized requests.

The format of the new operator commands is shown in Figure 18-2.

```
SETGRS ENQMAXU=nnnnnnnn[,NOPROMPT|NP]
SETGRS ENQMAXA=nnnnnnnn[,NOPROMPT|NP]
```

Figure 18-2 Format of the SETGRS ENQMAXU and ENQMAXA operator commands

GRS messages for concurrent requests

When an address space exceeds 80% of the maximum concurrent requests, GRS issues the ISG368E message. When the number of concurrent requests in the address space drops below 75% of the maximum, the ISG369I message is issued and the ISG368E message is DOMed. You can use the NOPROMPT parameter to automate the response to message ISG368E.

In general, the SETGRS ENQMAXU and SETGRS ENQMAXA commands are intended for emergency situations and not for normal operations. Keep in mind that the system-wide maximums are intended to protect GRS from run away ENQ address spaces. Increasing the system-wide maximums makes GRS more vulnerable to run away ENQ address spaces. If you have an address space that requires a higher maximum than the system-wide maximums, the recommended action is to change the application to use the ISGADMIN service to request a higher address space specific maximum.

18.1.2 Contention notification system movement

Resource contention can result in poor system performance. When resource contention lasts over a long period of time, it can result in program starvation or deadlock conditions.

When running in ring mode, each system in the GRS complex is aware of the complex wide ENQs, which allows each system to issue the appropriate ENF 51 contention notification event. However, when running in star mode, each system only knows the ENQs that are issued by itself. To ensure that the ENF 51 contention notification event is issued on all systems in the GRS complex in a proper sequential order, one system in the complex is appointed as the sysplex wide contention notifying system (CNS). All ENQ contention events are sent to the CNS, which then issues a sysplex wide ENF 51.

GRS provides two APIs to query for contention information:

ISGECA	Obtains waiter and blocker information for GRS managed resources.
ISGQUERY	Obtains the status of resources and requester of those resources.

GRS issues an event notification facility (ENF) signal 51 to notify monitoring programs to track resource contention. This is useful in determining contention bottlenecks, preventing bottlenecks, and potentially automating correction of these conditions.

During an IPL, or if the CNS can no longer perform its duties, any system in the complex can act as the CNS. You can determine which system is the current CNS with the D GRS command. In the example shown in Figure 18-3, you can see that the CNS is SC74.

```
D GRS
ISG343I 13.44.48 GRS STATUS 571
SYSTEM    STATE              SYSTEM    STATE
SC75      CONNECTED          SC74      CONNECTED
GRS STAR MODE INFORMATION
  LOCK STRUCTURE (ISGLOCK) CONTAINS 1048576 LOCKS.
  THE CONTENTION NOTIFYING SYSTEM IS SC74
  SYNCHRES:      YES
  ENQMAXU:     16384
  ENQMAXA:    250000
  GRSQ:   CONTENTION
```

Figure 18-3 Output of the DISPLAY GRS command

The SETGRS CNS operator command

The CNS can take up a considerable amount of system resources. You may prefer that it does not reside on your main production system, or you may prefer that it does not reside on a test system with insufficient capacity to handle the function. Starting with z/OS V1R8, you can choose the system to act as the CNS with the SETGRS CNS command. The format of this command is shown in Figure 18-4.

```
SETGRS CNS=system-name[,NOPROMPT|NP]
```

Figure 18-4 Format of the SETGRS CNS command

Figure 18-5 shows an example of changing the CNS from system SC74 to system SC75.

```
 SETGRS CNS=SC75
*020 ISG366D CONFIRM REQUEST TO MIGRATE THE CNS TO SC75. REPLY CNS=SC75 TO
   CONFIRM OR C TO CANCEL.
 R 20,CNS=SC75
 IEE600I REPLY TO 020 IS;CNS=SC75
 ISG364I CONTENTION NOTIFYING SYSTEM MOVED FROM SYSTEM SC74 TO SYSTEM SC75.
   OPERATOR COMMAND INITIATED.
 IEE712I SETGRS  PROCESSING COMPLETE
 D GRS
 ISG343I 15.26.46 GRS STATUS 638
 SYSTEM   STATE          SYSTEM   STATE
 SC75    CONNECTED       SC74    CONNECTED
 GRS STAR MODE INFORMATION
   LOCK STRUCTURE (ISGLOCK) CONTAINS 1048576 LOCKS.
   THE CONTENTION NOTIFYING SYSTEM IS SC75
  SYNCHRES:    YES
  ENQMAXU:    16384
  ENQMAXA:   250000
   GRSQ:   CONTENTION
```

Figure 18-5 Setting the CNS using SETGRS command

Setting CNS during the IPL

It is currently not possible to set the CNS from the GRSCNFxx parmlib member, because GRSCNFxx parsing is done too early in system initialization for setting the CNS across the sysplex. If you require that the CNS reside on a specific system, it is recommended that you use the SETGRS CNS operator command with the NOPROMPT parameter. You can add the command to the COMMNDxx parmlib member to have it issued automatically at IPL. An example of the SETGRS CNS command with NOPROMPT is shown in Figure 18-6.

```
SETGRS CNS=SC74,NP
IEE712I SETGRS  PROCESSING COMPLETE
ISG364I CONTENTION NOTIFYING SYSTEM MOVED FROM SYSTEM SC75 TO SYSTEM SC74.
  OPERATOR COMMAND INITIATED.
```

Figure 18-6 Setting the CNS using the SETGRS command with NOPROMPT

Coexistence with down-level systems

The support for CNS movement across systems in a GRS star complex requires that all systems in the complex run z/OS V1R8. Systems running z/OS V1R7 with APAR OA11382 are also supported. If any system in the complex is an earlier release than z/OS V1R7 with the required APAR, the SETGRS CNS command cannot be issued by any member of the complex.

If the CNS system fails, one of the remaining systems in the complex automatically becomes the new CNS. You can use automation for the rare case that the CNS moves from one system to another. Depending on the trigger (operator command or system failure), the system issues one of the following messages in case the CNS moves from one system to another:

```
ISG364I CONTENTION NOTIFYING SYSTEM MOVED FROM SYSTEM  xxxx TO SYSTEM yyyy.
OPERATOR COMMAND INITIATED.

ISG364I CONTENTION NOTIFYING SYSTEM MOVED FROM SYSTEM  xxxx TO SYSTEM yyyy.
SYSTEM INITIATED.
```

18.1.3 SETGRS GRSQ operator command

When SVCDUMP is executing in an address space to dump local storage, tasks may be left non-dispatchable for a long time while GRS is gathering GRS information. When SDATA=GRSQ is used to collect GRS serialization information during SYSMDUMP and SVCDUMP processing, all ENQ resource information from all images in the GRS complex is gathered and placed into the dump. This can be time-consuming in GRS star mode because GRS has to collect and merge information from all systems in the sysplex, which can cause excessive times and sometimes result in transaction timeouts.

The GRSQ keyword in the GRSCNFxx parmlib member allows you to limit the data collected by GRS at dump processing and therefore lessen the time required to complete the dump. You can specify three suboptions on the GRSQ keyword in the GRSCNFxx:

CONTENTION	Causes GRSQ processing to collect all ENQ resources on the local system, and only collect information about global resources in contention in the rest of the GRS complex.
LOCAL	causes GRSQ processing to collect all ENQ resources on the local system only. This is *not* a recommended value and should only be used as a circumvention. The GQSCAN that is issued does not collect data from the other systems.
ALL	Causes GRSQ processing to collect all ENQ resources from all systems in the entire GRS complex.

The default option is CONTENTION.

Note: The GRSQ keyword was introduced in z/OS V1R7.

SDATA=GRSQ option

During SVC dump and SYSMDUMP processing, if you specify the SDATA=GRSQ option, it indicates to the system to also collect GRS information. As a result, ENQ resource information from all systems in the GRS complex is gathered and placed into the dump. This can be time-consuming when running in star mode because GRS needs to collect and merge information from all systems in the sysplex. The time it takes for the process to complete increases as more systems are added to the sysplex or the number of concurrent ENQs increases.

New command to set GRSQ

z/OS V1R8 allows you to dynamically change the setting of the GRSQ keyword using the SETGRS GRSQ command. The format of this operator command is shown in Figure 18-7.

```
SETGRS GRSQ=ALL|CONTENTION|LOCAL
```

Figure 18-7 Format of the SETGRS GRSQ operator command

The SETGRS GRSQ command is only available when running in star mode. The effect of this command is system-specific. The specified option is only taken on the system where the command is issued.

Figure 18-8 shows an example of the SETGRS GRSQ command:

```
SETGRS GRSQ=ALL
ISG370I GRSQ ON SYSTEM SC75 HAS BEEN SET TO ALL.
IEE712I SETGRS   PROCESSING COMPLETE
```

Figure 18-8 Output of the SETGRS GRSQ operator command

Note: To allow GRSQ to be set via are hoping

the SETGRS command, this is available on z/OS V1R7 via APAR OA11382.

APAR OA06591 is part of the z/OS V1R7 base and rolled back to z/OS V1R2 changes processing to not mark tasks non-dispatchable while the global resource information is being collected.

APAR OA07975 is part of the z/OS V1R7 base and rolled back to z/OS V1R4 allows the GRSQ keyword on GRSCNFxx parmlib member.

18.1.4 GRS storage constraint

GRS has its own internal storage management system to make the best overall use of its private storage and maintain performance for its various services. In systems with many data sets left open for long periods of time, it is possible to exhaust storage. If left unchanged, this problem would only worsen as system capacity increases. This limit provides the most constraint relief for GRS STAR mode systems.

GRS storage management utilizes a fixed-size block of private above-the-line storage. An ABEND09A, Rsn 00008108x and waitstate occurs when that storage runs out.

z/OS V1R8 enhancement

Some serialization products reference and sometimes alter GRS blocks. With z/OS V1R8, GRS provides APIs as required so they no longer need to directly reference GRS blocks. Now, ISGECA, ENF 51, or ISGQUERY may be used as an alternative to running GRS queues.

18.2 Latch deadlocks

In current releases, there is an improper use of the GRS latch services that can cause deadlocks. Previously, no checks were made when a request was made for a latch that the requester already owns.

ENQ services check to see whether a requester already owns the resource. GRS Latch services had not been making this check, such that deadlocks can result. This new checking is optionally specified at the time of the obtain. It is optional in case current exploiters intentionally utilize the old behavior.

There are two levels, ISGLCRT_DEADLOCKDET1 and ISGLCRT_DEADLOCKDET2. The first level involves the first two checks when exclusive ownership is requested. The second level additionally involves the check when shared ownership is requested, which is more robust but can impede latch contention processing performance.

Requesting shared when the requester already owns the latch shared does not entail a deadlock scenario, so is not checked.

19

z/OS XML (eXtensible Markup Language)

This chapter describes the changes made in z/OS V1R8 XML and discusses the following:

- Introduction to XML
- z/OS XML
- Rules to create an XML document
- Standards used by XML
- Strategic importance of XML
- z/OS XML system services functions
- The query function
- The PARSE function
- z/OS XML exits

19.1 Introduction to XML

The introduction of XML covers the following areas:

- What is XML
- Rules to create an XML document
- Standards used by XML
- Importance of XML
- Strategic importance of XML

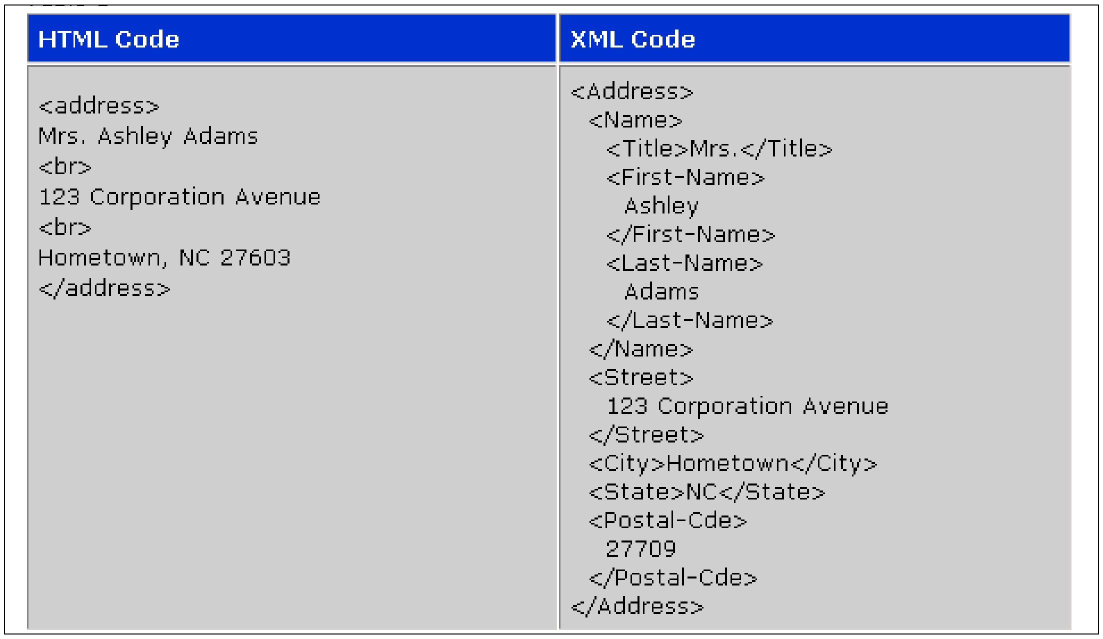
19.1.1 What is XML

The World Wide Web Consortium (W3C) developed XML to define markup languages. It derived from and is a subset of Standard Generalized Markup Language (SGML). Originally XML was designed to meet the challenges of large-scale electronic publishing. XML plays a role in the exchange of a wide variety of data on the Web and elsewhere.

XML allows the definition, transmission, validation, and interpretation of data between applications. It is a meta-language: a language for defining other markup languages, interchange formats, and message sets.

XML provides a way to mark up the content of a document (the traditional idea, as in SGML) or data. The data could be, for example, a purchase order or similar business document. With XML, the tags used for the markup describe what something is. This is in contrast to HTML, for example, where the tags describe how something should appear in a browser.

Figure 19-1 on page 382 shows a comparison between HTML and XML code for the address tag.



HTML Code

```
<address>
Mrs. Ashley Adams
<br>
123 Corporation Avenue
<br>
Hometown, NC 27603
</address>
```

XML Code

```
<Address>
  <Name>
    <Title>Mrs.</Title>
    <First-Name>
      Ashley
    </First-Name>
    <Last-Name>
      Adams
    </Last-Name>
  </Name>
  <Street>
    123 Corporation Avenue
  </Street>
  <City>Hometown</City>
  <State>NC</State>
  <Postal-Cde>
    27709
  </Postal-Cde>
</Address>
```

Figure 19-1 The difference between the HTML and the XML codes

Managing XML data

As an example, an algorithm to find the city in HTML code would be very difficult. With XML, the task would be very simple. XML allows you to assign specific meaning to the tags, such as <City>Hometown</City>. An algorithm could easily be written to locate the content surrounded by the <City> and </City> tags, technically known as the <City> element.

XML fills the need for a universal language that is also platform independent and enables common methods for the exchange of data. You can create applications that can easily extract the data they need from XML documents and any kind of data can be described and communicated through XML.

XML is quickly becoming the data representation language of choice for managing data interchange between applications and business platforms.

Using XML effectively requires the management of XML definitions (metadata) for data interfaces (for example, communication between devices) and positioning of data elements with respect to business data standards. Effective use of XML requires a defined XML management strategy with associated standards, processes and tools. Conforming to these standards enables the sharing of structured data which is the key to technology to enable e-business.

Therefore, IBM needs to manage XML data and its relationship to business data standards. Developers need to deploy consistent XML interfaces and data to render efficient application-to-application (A2A) and business-to-business (B2B) communications. It will also affect the quality of data that will potentially be used for effective business reporting and management. In addition to using XML, Software Group needs to deliver products that handle XML so that our clients can also manage XML data. Today, the Web Services Interface is very important and Simple Object Access Protocol (SOAP) messages expressed in XML are used within the Web Services Interfaces.

19.1.2 Rules to create an XML document

Following are the rules to create an XML document:

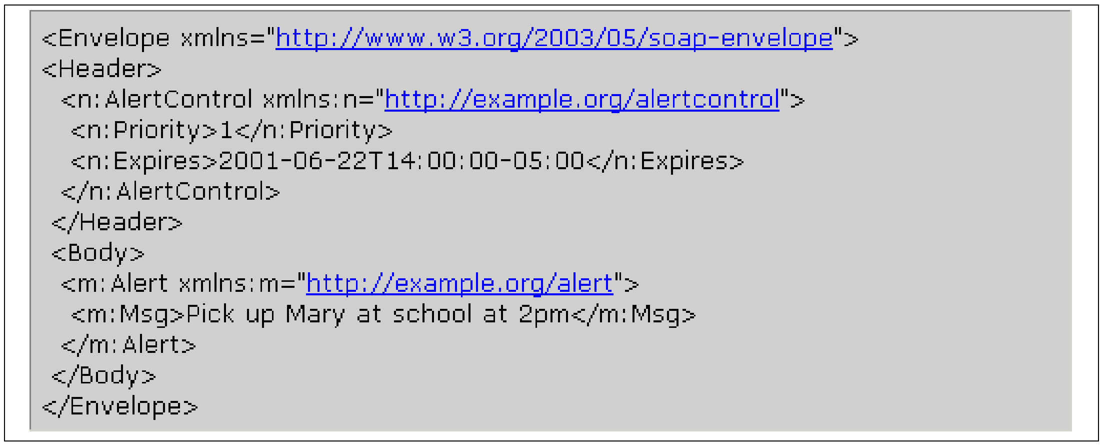
Tag	A tag is the text between the left angle bracket (<) and the right angle bracket (>). There are starting tags (such as <Name>) and ending tags (such as </Name>).
Element	An element is the starting tag, the ending tag, and everything in between. The following describes what an element is <Department>Industrial Design</Department>.
Attribute	An attribute is a name-value pair inside the starting tag of an element. In the following <City state="NC">Hometown</City>), state is an attribute of the <City> element. The “NC” is the value of the attribute.
XML declaration	`<?xml version=”1.1”?>`) is the XML declaration that provides basic information about the document to the parser.
Comment tag	A comment begins with <!-- and ends with -->. A comment cannot contain a double hyphen (--) except at the end; with that exception, a comment can contain anything.
Empty element	An empty element can contain no content. It has two patterns: A short pattern like <Middle-Name/>, and a long pattern like <Middle-Name></Middle-Name>. The shorter version still has an ending tag of “/>”. An XML parser treats them in the same way. If an XML document references an XML schema and the XML schema checks for

that element, make sure that element is in your XML document. But leave it empty if you don't have data.

Nesting tags Nesting tags enables the XML users to describe hierarchical structures as well as sequences. Nesting requirements mean that a well-formed XML document can be treated as a tree structure of elements. Many XML specs will casually refer to the term XML tree when referring to the structure of elements. The following is a nested tag:

```
<Name>
<Title>Mrs.</Title>
<First-Name>Ashley</First-Name>
<Last-Name>Adams</Last-Name>
</Name>
```

XML Namespace Namespace is a method of qualifying the element and attribute names used in XML documents by associating them with a Universal Resource Identifier (URI). A URI is a string of characters that identifies an Internet Resource (IR). The most common URI is the Uniform Resource Locator (URL), which identifies an Internet domain address along with other system identifiers. Another, not so common, type of URI is the Universal Resource Name (URN). An XML namespace is a collection of element names, of attribute names identified by a URI reference, which are used in XML documents and define the scope of the element and attribute names. Element and attribute names defined in the same namespace must be unique.An XML document can have a default namespace (using 'xmlns=') and any element can belong to the default, or another specified namespace. The collection of defined elements and attributes within the same namespace are said to be in the same "XML vocabulary." Figure 19-2 shows some examples of the use of namespace

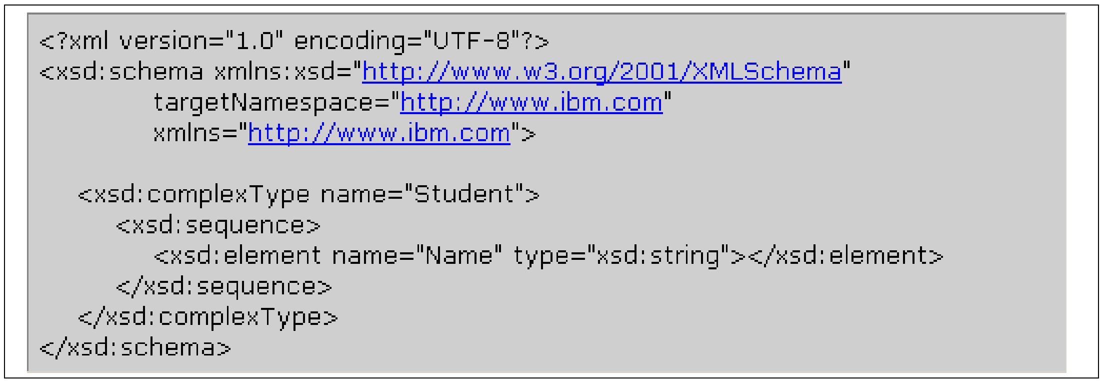


```
<Envelope xmlns="http://www.w3.org/2003/05/soap-envelope">
<Header>
  <n:AlertControl xmlns:n="http://example.org/alertcontrol">
   <n:Priority>1</n:Priority>
   <n:Expires>2001-06-22T14:00:00-05:00</n:Expires>
  </n:AlertControl>
 </Header>
 <Body>
  <m:Alert xmlns:m="http://example.org/alert">
   <m:Msg>Pick up Mary at school at 2pm</m:Msg>
  </m:Alert>
 </Body>
</Envelope>
```

Figure 19-2 The use of a namespace

Definition of XML documents

XML schemas express shared vocabularies and allow computers to carry out rules defined by people. These rules are expressed by the definitional statements within the XML schema (XSD) or DTD. They provide a means for defining the structure, content and semantics of XML documents. Figure 19-3 on page 385 shows what an XML schema is.



```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://www.ibm.com"
        xmlns="http://www.ibm.com">

   <xsd:complexType name="Student">
      <xsd:sequence>
         <xsd:element name="Name" type="xsd:string"></xsd:element>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```

Figure 19-3 XML schema example

Validating an XML document

To avoid errors, XML documents must be structured to follow the rules specified in their XML specifications (for example, XSD or DTD). The XML specification requires a parser to reject any XML document that does not follow the basic rules. A parser consists of code that validates an XML document by trying to read the document and interpret its contents. If an element has a start tag and no ending tag, you would get an error from the parser. If you have an XML document with an associated DTD or XML schema, the XML document will be validated against them to determine if it is a well-formed and valid document.

XML parsers

The two basic types of parsers are Document Object Mode (DOM) and Simple API for XML (SAX). Both DOM and SAX give you access to the information stored in XML documents. However, both of them take very different approaches to giving you access to your information. SAX enables you to parse a portion of a file or process a file from start to end, extracting information from specific nodes. A DOM parser parses an entire XML file before you can do anything with the content. When working with large files, this takes more time and resources. Web browsers, such as Internet Explorer 4.0 and later, Mozilla, and Opera also contain an XML parser. This parser reads the XML code, processes and validates the data. From this point, the data can be used by other applications or objects for further processing. XML with the correct syntax is well-formed XML. XML validated against a DTD or XML schema is valid XML.

19.1.3 Standards used by XML

To process the XML documents, the following XML standards are listed:

- XML Namespace - XML Namespace 1.0 is now a W3C Recommendation. XML namespaces provide a simple method for qualifying element and attribute names used in eXtensible Markup Language documents by associating them with namespaces identified by URI references.
- eXtensible Markup Language (XML) - As of November 2003, XML 1.1 is now the proposed recommendation by the W3C approval process. This means that the standard is stable and can be fully embraced by Web and tools developers. The following statement <?xmlversion=”1.1”?> indicates the version level.
- eXtensible Stylesheet Language Family (XSLT/XSL/XSLFO)
  - eXtensible Stylesheet Language Transformations (XSLT) is a markup language used to express how a processor creates a transformed result from an instance of XML

information. XSLT is designed for the transformation required for using XSL and is also used for arbitrary transformation requirements.

  - eXtensible Stylesheet Language (XSL), known today as eXtensible Stylesheet Language Formatting Objects (XSLFO), defines a set of elements (called formatting objects) that represents how data is formatted. XSL FO is used to transform XML-based data into HTML or other presentation formats. XSL provides a superset of the Cascading Style Sheets (CSS) language functionality and allows developers to build a presentation structure that is different from the data structure. For instance, XSL can be used to transform an XML-based purchase order number into a bulleted list in one HTML view and into a footnote in a second HTML view. CSS may still be used for simply structured XML data but cannot present information in an order different from how it was received. XSL will also be able to generate CSS along with HTML.

- XML Document Object Model (XML DOM)

  The XML DOM tracks the W3C standard Document Object Model (DOM) to allow programmatic access to structured data through scripting, so developers can interact with and compute on XML-based data consistently. XML DOM is essentially an API that defines a standard way in which developers can interact with the nodes of an XML document.

- Simple Object Access Protocol (SOAP)

  SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined datatypes, and a convention for representing remote procedure calls and responses. SOAP can potentially be used in combination with a variety of other protocols; however, the only bindings defined in this document describe how to use SOAP in combination with HTTP and HTTP Extension Framework.

- XML Query (XQuery

  XQuery is designed to be a language in which queries are concise and easily understood, but also flexible enough to query a broad spectrum of XML information sources, including both databases and documents.

- XHTML

  eXtensible Hypertext Markup Language (XHTML) is intended to replace HTML. The current version XHTML 1.0, W3C's first recommendation of XHTML, reformulates HTML as an XML application. XML applications are easier to process and maintain and are designed to work in conjunction with XML-based user agents.

- XML Path Language (XPath)

  XPath is a syntax language that describes locations in XML documents. It is used in XSLT style sheets to indicate the parts of an XML document that you want to transform.

- XML Linking Language (XLink) and XML Pointer Language (XPointer)

  Both XLink and XPointer are designed to create links between files.

  - XLink enables you to link documents to be expressed in XML. Elements within an XML document can be remapped using architectural forms, and attributes can be remapped using XLink.
  - XPointer is used to recognize specific elements within an XML document. This would be used to create links to particular elements when used with XLink. XPointer searches the “id” attributes within the XML document for a match. When a match is found on the “id” attribute, the XPointer identifies the element so it can become the target of an individual link.

- XML Forms (XForms)

  XForms is the name given by W3C for a specification of Web forms that can be used with a wide variety of platforms including desktop computers, hand-held devices, information appliances, and even paper.

XML is evolving

XML has evolved considerably over the past few years. In the beginning it was a way to mark up data and used DTDs to validate the data. Soon to follow, the W3C added new technologies—such as XML Schema, eXtensible Stylesheet Language Transformations (XSLT), and XPath—to support and enhance its usefulness. Today, XML can generate PDFs, word-processing documents, and even graphics.

XML now supports typed APIs that simplify the conversion of XML schema types to Common Language Runtime (CLR) types, while preventing coding errors in the process. The APIs allow you to copy XML data to CLR objects alter them and then copy it back to the XML data source. XML schema enables you to move relational data to a CLR object and back. Also, APIs can check XQuery statements at compile time, since it is a typed language.

19.1.4 Importance of XML

IBM is focusing on XML for the following reasons:

- XML is a key technology that will help our clients and business partners realize the promise of e-business. XML enables business-to-business communication by using data standards. By following the document rules defined in the DTD or XML schema, we can easily validate data that is exchanged using its “open” standardized definition.
- XML makes it easy to share structured data across the Web so that the data is well understood.
- XML enables smart agents. When XML-structured data is sent to an agent, it's much easier for the agent to understand exactly what the data means and how it relates to other pieces of data that it may already know.
- XML enables smart searches. XML-structured data enables much more accurate and useful search results.

Why XML is gaining importance

Because XML is a data representation standard, accessing, integrating and replicating data across the enterprise is now an easier task for most organizations. In many cases, productivity gains in the enterprise require an emphasis on data. Executive and management dashboards, synchronizing operations and ensuring consistency of information are becoming top-of-mind issues for both business and IT executives. Businesses are becoming more interdependent with their supply chains and require the effective sharing of information at ever increasing levels.

Business initiatives are propelling a renewed focus on data. The on demand requirements for reconfigurable processes and plug-in software packages require standardized interfaces that can be provided by XML data standards. XML supports well defined standard definitions of data and is self-describing, readable by both humans and computers, and can easily be translated to other XML “vocabularies” in the domain of worldwide enterprises.

Finally, XML adoption is being driven by some of the same characteristics that prompted acceptance of other ubiquitous technologies such as Ethernet and Transmission Control Protocol/Internet Protocol. XML is simple, approachable and not loaded down with every possible combination of services and capabilities.

IBM support of XML

IBM actively participates in, and leads, many industry projects and consortium working groups designed to advance XML as a standard used to define and share data over the Internet.

IBM co-chairs the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) effort to develop the Electronic Business XML (ebXML) Framework for business-to-business interchange.

IBM has created an XML Standards Repository where IBM internal developers can create, maintain and share standardized XML artifacts. These artifacts are created using approved namespaces and become part of the vocabulary of standardized XML, when approved by other peer developers.

IBM has created an XML Namespace Registry, where developers can register their specific namespaces. As namespace is the first step in defining a managed XML vocabulary, this allows them to use a pre-allocated section of the IBM XML Standards Repository to contain the XML artifacts associated with their vocabulary.

19.1.5 Strategic importance of XML

XML will become the primary data representation for the exchange of data, including databases, application servers, browsers, message brokers, and middleware. XML can be used to represent both structured and semi-structured data. XML is eXtensible, platform-independent, and supports internationalization by being fully Unicode compliant. XML is a text-based format. It can be read and edited using standard text editing tools.

Adoption of XML is increasing in many different product lines. Diverse software categories such as application servers, relational databases, message brokers and content management systems all have increased support for XML. The U.S. federal government's Chief Information Officers Council makes XML one of the cornerstone technologies that can improve the productivity and efficiency of federal agencies. The U.S. Navy, a leader in the application of technology, made XML the standard for data formatting and metadata definition. The emerging Web services standards use XML as one of the core elements.

Note: The following formula shows how XML is part of the e-business:

Networking (TCP/IP) + Program loading (Web model) + Programming (Java components) + Data (XML) = e-business.

19.2 z/OS XML system services functions

These functions are used through the z/OS XML system services API. The z/OS XML system services contains two primary functions:

- Querying XML documents

  XML documents contain characteristics that can affect their ability to be parsed. One characteristic is the encoding scheme of the document. The z/OS XML parser must know it before parsing. The query service allows the caller to acquire this information, and it can then it can pass it to the z/OS XML parser. The z/OS XML parser can use the correct encoding scheme to parse the document.

  The following callable services are for querying an XML document:

  ```
  GXL1QXD is used in 31-bit mode.
  ```

```
GXL4QXD is used in 64-bit mode.
```

- Parsing XML documents

  The parsing process consists of three fundamental steps: initiate the parse process, parse the document, and terminate the parse process. The parse process can be repeated for multiple documents.

  The following callable services are used for parsing:

  ```
  GXL1CTL – perform a parser control operation in 31-bit mode.
  GXL1INI – initialize a parse instance in 31-bit mode.
  GXL1PRS – parse an input stream in 31-bit mode.
  GXL1QXD – query an XML document in 31-bit mode.
  GXL1TRM – terminate a parse instance in 31-bit mode.
  GXL4CTL – perform a parser control operation in 64-bit mode.
  GXL4INI – initialize a parse instance in 64-bit mode.
  GXL4PRS – parse an input stream in 64-bit mode.
  GXL41QXD – query an XML document in 64-bit mode.
  GXL4TRM – terminate a parse instance in 64-bit mode.
  ```

19.2.1 The query function

An XML document contains declarations that may need special handling during a parse. For instance, if the encoding of the document to parse is unknown, the query service provided by z/OS XML parser can be used to help determine the encoding in order to provide the correct Coded Character Set IDentifier (CCSID) to the parser, when the actual parse is performed. In order for the caller to query an XML document, all the caller needs to do is use service GXL1QXD (31-bit mode) or GXL4QXD (64-it mode). This service allows the caller to obtain all the XML characteristics of the document. These characteristics can be either the default values or those contained in an XML declaration.

Once these characteristics are obtained, the caller can then determine the encoding scheme needed to parse the document, along with any additional steps that may be needed. For example, if the document in question uses an encoding scheme of UTF-16, it will require that the z/OS XML parser also uses the UTF-16 encoding scheme when parsing this document. The caller would use the GXL1QXD (31-bit mode) or GXL4QXD (64-bit mode) service to ascertain the encoding type of the document being parsed. Once this information is acquired, z/OS XML parser can be initialized using GXL1INI (31-bit mode) or GXL4INI (64-bit mode)

Note: GXL1QXD (31-bit mode) or GXL4QXD (64-bit mode) are the only services that provide support for both UTF-16 (little endian) and UTF-16 (big endian), whereas the other services only support UTF-16 (big endian).

19.2.2 The parse function

The parsing process consists of three steps: initiating the parse process, parsing the document, and terminating the parse process. The parse process can be repeated for multiple documents. To parse a document you need an input buffer, the z/OS XML parser, and an output buffer. These three components and their interrelationships make up the processing model.

There may be more than one input and output buffer, depending on the size of the document being parsed. If the document is sufficiently large, the caller may find it necessary to provide it to the parser in several pieces, using buffer spanning to maintain the document structure as it is being parsed. Similarly, the caller may need to provide multiple buffers to contain the data stream that the z/OS XML parser generates. Document processing is the creation of the output buffers from the parsed input data.

Parsing an XML document

Before the z/OS XML parser can parse an XML document, it must establish a context in which it can operate. The initialization routine, GXL1INI (31-bit mode) or GXL4INI (64-bit mode), creates this context and passes a piece of memory where the z/OS XML parser establishes a parse instance memory area (PIMA). This is the area where the z/OS XML parser creates a base for the internal data structures it uses to complete the parse process.

Note: A particular PIMA must only be used during the parse of a single XML document at a time. Only after the parse is complete can a PIMA be reused for the parse of another document. The minimum size for the PIMA is 128k.

Parse instance memory area (PIMA)

In addition to control information, the PIMA is used as a memory area to store temporary data required during the parse. When the z/OS XML parser needs more storage than was provided in the PIMA, additional storage is allocated. Everything that the z/OS XML parser needs to complete the parse of a document is kept in the PIMA, and any associated memory extensions that parser may allocate during the parse process.

The caller must also provide input and output buffers on each call to GXL1PRS (31-bit mode) or GXL4PRS (64-bit mode). In the event that either the text in the input buffer is consumed, or the parsed data stream fills the output buffer, the z/OS XML parser will return XRC_WARNING, along with a reason code indicating which buffer (possibly both) needs the caller's attention. It also indicates the current location, and the number of bytes remaining in each buffer, by updating the buffer_addr and buffer_bytes_left parameters passed on the parse request.

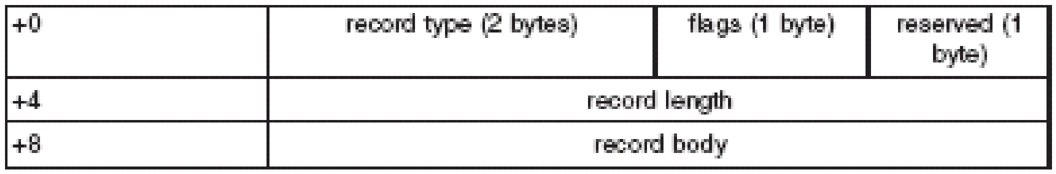
This process is referred to as *buffer spanning*. If the entire document has been processed when the z/OS XML parser returns to the caller, the parse is complete and the caller proceeds accordingly. If the caller requires another document to be parsed, it has the option of terminating the current parse instance by calling GXL1TRM (31-bit mode) or GXL4TRM (64-bit mode). This will free up any resources that the z/OS XML parser may have acquired and resets the data structures in the PIMA so that it can be used for the next parse. If the caller needs to parse another document, it will have to call GXL1INI (31-bit mode) or GXL4INI (64-bit mode) again to either completely re-initialize the existing PIMA area or initialize a new PIMA area from scratch. Another option is to use the finish/reset function of the GXL1CTL (31-bit mode) or GXL4CTL (64-bit mode) z/OS XML parser service to reset the PIMA so that it can be reused.

This is a lighter-weight operation that preserves certain information that can be reused across parsing operations for multiple documents. This improves the performance for subsequent parses, since this information can be reused instead of being rebuilt from scratch. Reusing the PIMA in this way is particularly beneficial to callers that need to handle multiple documents that use the same symbols (for example, namespaces and local names for elements and attributes). The PIMA can only be reused in this way when the XML documents are in the same encoding and the same z/OS XML parser options are used.

Parsed data model

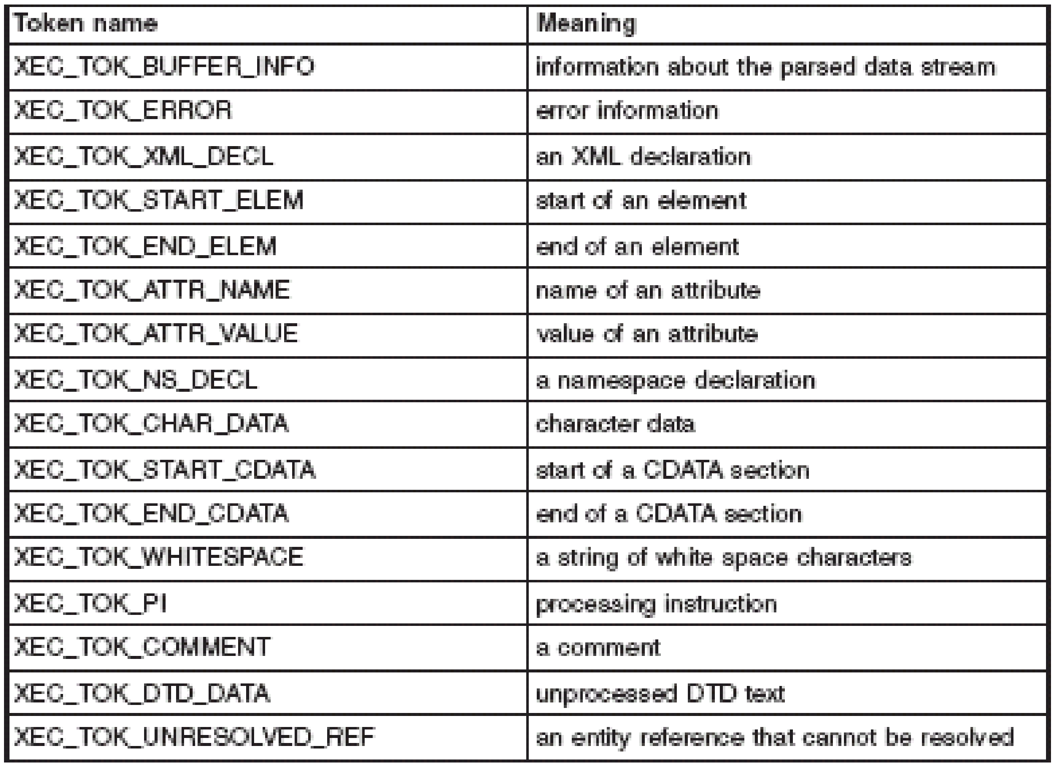
Here we provide information on the data model used to represent the contents of the output buffer. The caller needs to understand this data model so that it can effectively process the parsed data stream that has been created in the output buffer. The z/OS XML parser produces a structured data stream resulting from the parse process. It is a feature that distinguishes the z/OS XML parser from most other XML parsers.

The parsed data stream consists of a set of self-describing records representing the output of the parser. These records provide a structure to the data stream that allows a consumer to navigate the data stream as needed. Some of the records represent the actual semantic content of the parsed document, while others provide metadata about the parse itself. Each record in the parsed data stream consists of a common header, followed by information that is specific to a given record type. Figure 19-4 shows a description of the common header record and Figure 19-5 shows the record types.



+0	record type (2 bytes)	flags (1 byte)	reserved (1 byte)
+4	record length		
+8	record body		

Figure 19-4 A common record header



Token name	Meaning
XEC_TOK_BUFFER_INFO	information about the parsed data stream
XEC_TOK_ERROR	error information
XEC_TOK_XML_DECL	an XML declaration
XEC_TOK_START_ELEM	start of an element
XEC_TOK_END_ELEM	end of an element
XEC_TOK_ATTR_NAME	name of an attribute
XEC_TOK_ATTR_VALUE	value of an attribute
XEC_TOK_NS_DECL	a namespace declaration
XEC_TOK_CHAR_DATA	character data
XEC_TOK_START_CDATA	start of a CDATA section
XEC_TOK_END_CDATA	end of a CDATA section
XEC_TOK_WHITESPACE	a string of white space characters
XEC_TOK_PI	processing instruction
XEC_TOK_COMMENT	a comment
XEC_TOK_DTD_DATA	unprocessed DTD text
XEC_TOK_UNRESOLVED_REF	an entity reference that cannot be resolved

Figure 19-5 This picture shows the different record types

19.2.3 z/OS XML exits

The system services exit interface is made of exits running in either 31-bit or 64-bit mode. They are GXLFSTxx (free memory), GXLGSTxx (get memory) and GXLSYMxx (StringID service). The xx suffix is either 31 or 64. It means that the exit runs either in 31-bit or in 64-bit mode.

A string identifier service creates a unique 4-byte numerical value (StringID) that corresponds to a string parsed from the document. This exit allows the caller to control the individual

StringID values that the z/OS XML parser uses and serves as an efficient mechanism to communicate these values between caller and parser. If no StringID service is specified, StringIDs are not exploited by the z/OS XML parser and the parsed data stream will contain only length/value pairs for all parsed strings.These exits are all passed the address of a system service work area. This work area is storage obtained by the caller and used to store any information to make communication between the caller and the exits easier.

Tip: These exit routines must not call any of the services provided in the z/OS XML parser API, either directly or indirectly.

Figure 19-6 on page 392 explains how the parsing works.

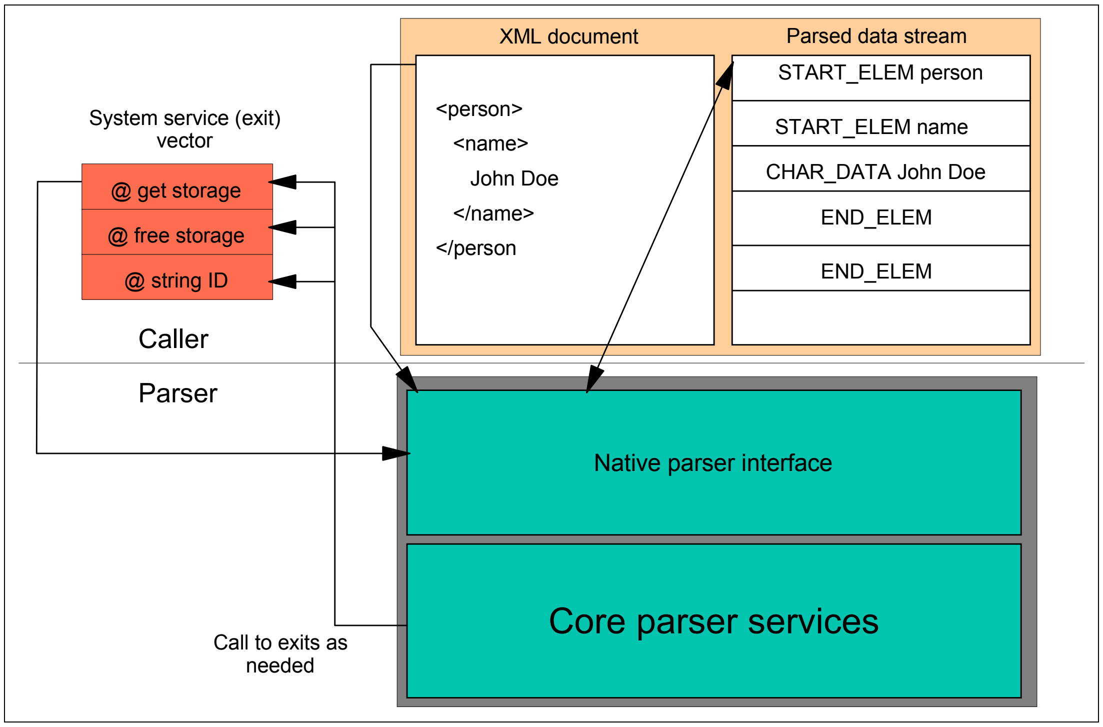


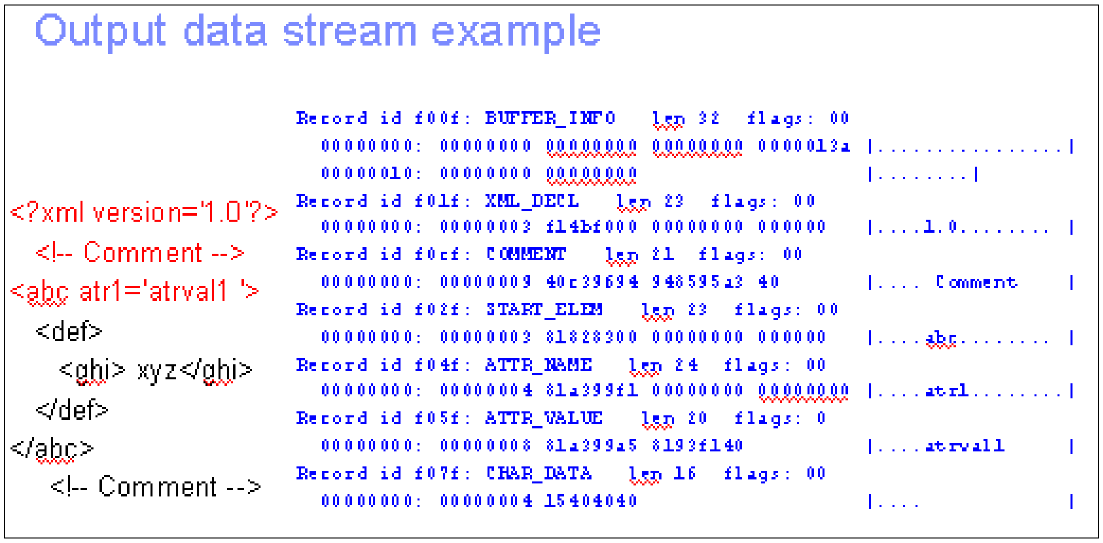
Figure 19-6 The parsing process

Contents of the output buffer

Figure 19-7 on page 393 shows the content of the output buffer. The output buffer contains the parsed data stream that results from the parse process. This data stream will contain the parsed XML document contents, along with header and any error information that was produced during the parse.

Each record in the output consists of:

- Record ID (example: F00F is a buffer information record)
- Length of the whole record, including the header
- A flag field. Currently the only use of the flag field is to indicate when a record spans into another output buffer.
- Zero or more values


Output data stream example

```
<?xml version='1.0'?>
  <!-- Comment -->
<abc atr1='atrval1 '>
  <def>
    <ghi> xyz</ghi>
  </def>
</abc>
   <!-- Comment -->
```

```
Record id f00f: BUFFER_INFO   len 32  flags: 00
  00000000: 00000000 00000000 00000000 0000013a |................|
  00000010: 00000000 00000000                   |........|
Record id f01f: XML_DECL   len 23  flags: 00
  00000000: 00000003 f14bf000 00000000 000000   |....1.0........ |
Record id f0cf: COMMENT   len 21  flags: 00
  00000000: 00000009 40c39694 948595a3 40       |.... Comment    |
Record id f02f: START_ELEM   len 23  flags: 00
  00000000: 00000003 81828300 00000000 000000   |....abc........ |
Record id f04f: ATTR_NAME   len 24  flags: 00
  00000000: 00000004 81a399f1 00000000 00000000 |....atr1........|
Record id f05f: ATTR_VALUE   len 20  flags: 0
  00000000: 00000008 81a399a5 8193f140          |....atrval1     |
Record id f07f: CHAR_DATA   len 16  flags: 00
  00000000: 00000004 15404040                   |....            |
```

Figure 19-7   Example of an output data stream

For additional information about z/OS XML, see *z/OS XML System Services User's Guide and Reference*, SA23-1350.

20

z/OS V1R8 Common Information Model enhancements

This chapter contains the following:

- Introduction to CIM
- Tools used by CIM
- CIM object manager
- Updates to z/OS CIM
- Security with CIM

Important: To make it simple, the purpose of implementing CIM on the z/OS platform is to enable you to manage the operating system without knowing its infrastructure. What makes CIM so difficult to figure out is its complexity, which comes from the use of object programming (the concept of class, of methods), the use of XML (the XML language, the concept of schema, the concept of namespaces), the use of HTTP, the use of providers, the use of MOF, the use of CIMOM, the use of the repository, the use of the CIM/XML protocol, and its connection with WBEM. The goal of this chapter is to give a general idea of what CIM is, what it is made of, how it works, and to present the updates.

20.1 Introduction to CIM

The Common Information Model (CIM) is a standard data model developed by a consortium of major hardware and software vendors (including IBM) called the Distributed Management Task Force (DMTF). CIM is part of the Web Base Enterprise Management (WBEM) standards defined by DMTF. CIM works in conjunction with WBEM. WBEM and CIM use the client/server architecture. WBEM includes a set of technologies that enables the interoperable management of an enterprise network. The WBEM core set of standards includes a data model, the CIM standard; an encoding specification, CIM-XML encoding specification; and a transport mechanism, CIM operations over HTTP. The open-source implementation of the DMTF, of the CIM and the WBEM standards represents Pegasus. On z/OS, CIM version 2.5.1 is implemented.

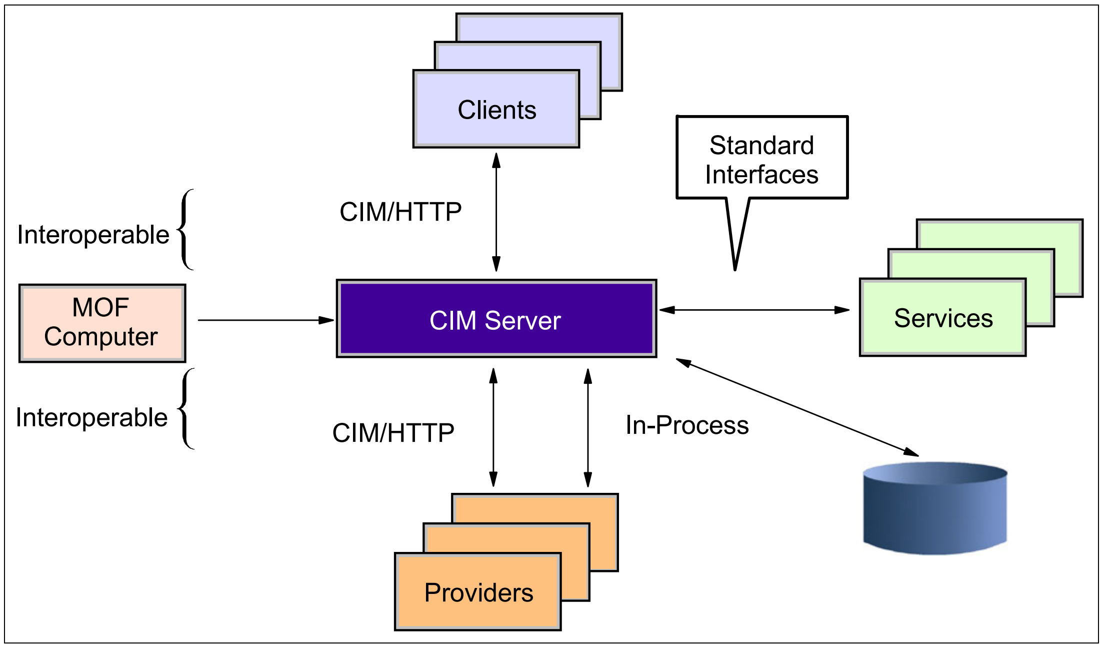


Figure 20-1 The Pegasus architecture

20.1.1 CIM server on z/OS systems

CIM provides a model for describing and accessing data across an enterprise. CIM consists of both a specification and a schema. The specification defines the details for integration with other management models, while the schema provides the actual model descriptions.

With support for the CIM server on systems running z/OS, users can access z/OS resources through an extendible industry standard model. CIM for z/OS includes:

- Instrumentation for server resources on the system are called providers. The providers, which are based on a subset of the standardized CIM classes, gather data on a system. CIM clients can work with this data by accessing the providers through the CIM server.
- An open source implementation of the CIM server that manages communication between clients and providers. The CIM server also provides several management functions, including security, and a set of commands that provide configuration and management functions to administrators. The CIM server implementation on z/OS is based on the OpenPegasus CIM server 2.4.1 from The Open Group.
- A CIM schema that defines an information model for representing systems management functions. For z/OS 1.8, CIM Schema version 2.9 is supported by the CIM server.

- An implementation of the standardized formats for communication between clients and the CIM server, called CIM in XML, and called CIM Operations over HTTP.
- The CIM server for z/OS supports most of the CIM operations defined in the *CIM Operations over HTTP* specification by the DMTF. Figure 20-2 illustrates how the CIM server works in the z/OS environment: A CIM client application requests the CIM server to return information about z/OS resources, in this case about basic OS data as well as RMF metrics. The CIM server invokes the corresponding z/OS OS management and RMF monitoring providers, which retrieve the requested data associated to z/OS system resources. The z/OS RMF monitoring provider invokes the RMF Distributed Data Server (DDS), which in turn collects RMF Monitor III performance data. The CIM server consolidates the data from the providers and returns it back to the calling client through the CIM/XML protocol.

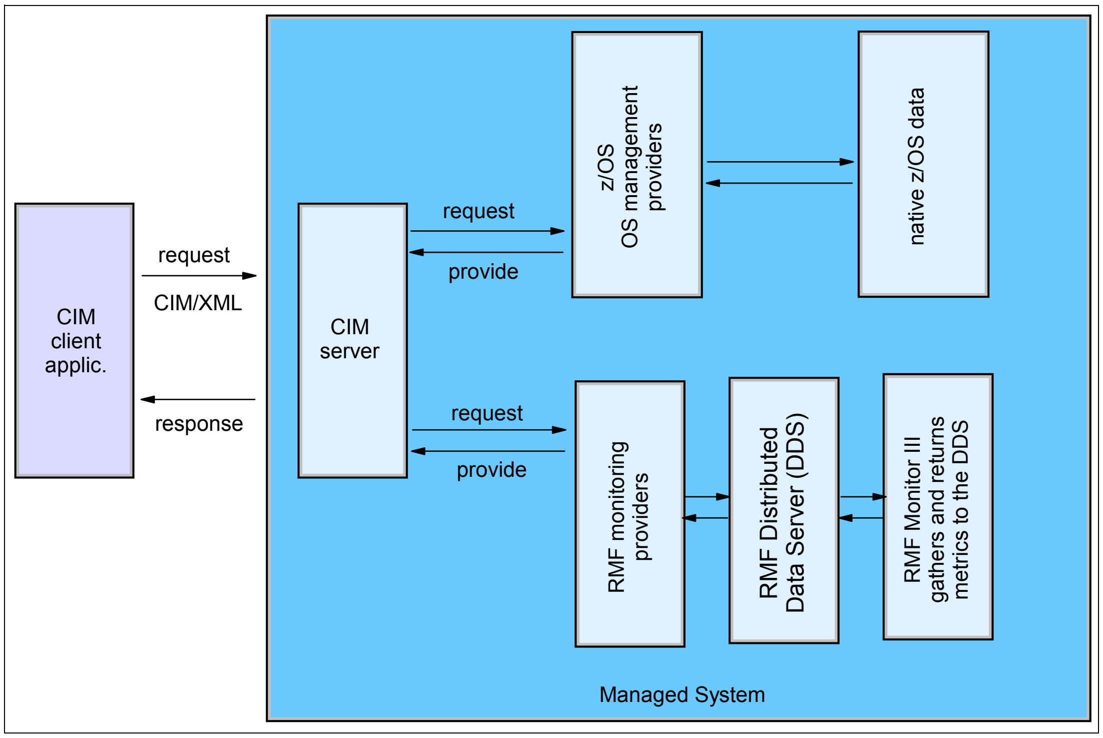


Figure 20-2 Tasks of the CIM server in a z/OS environment

20.2 Tools used by CIM

The following tools are used by CIM:

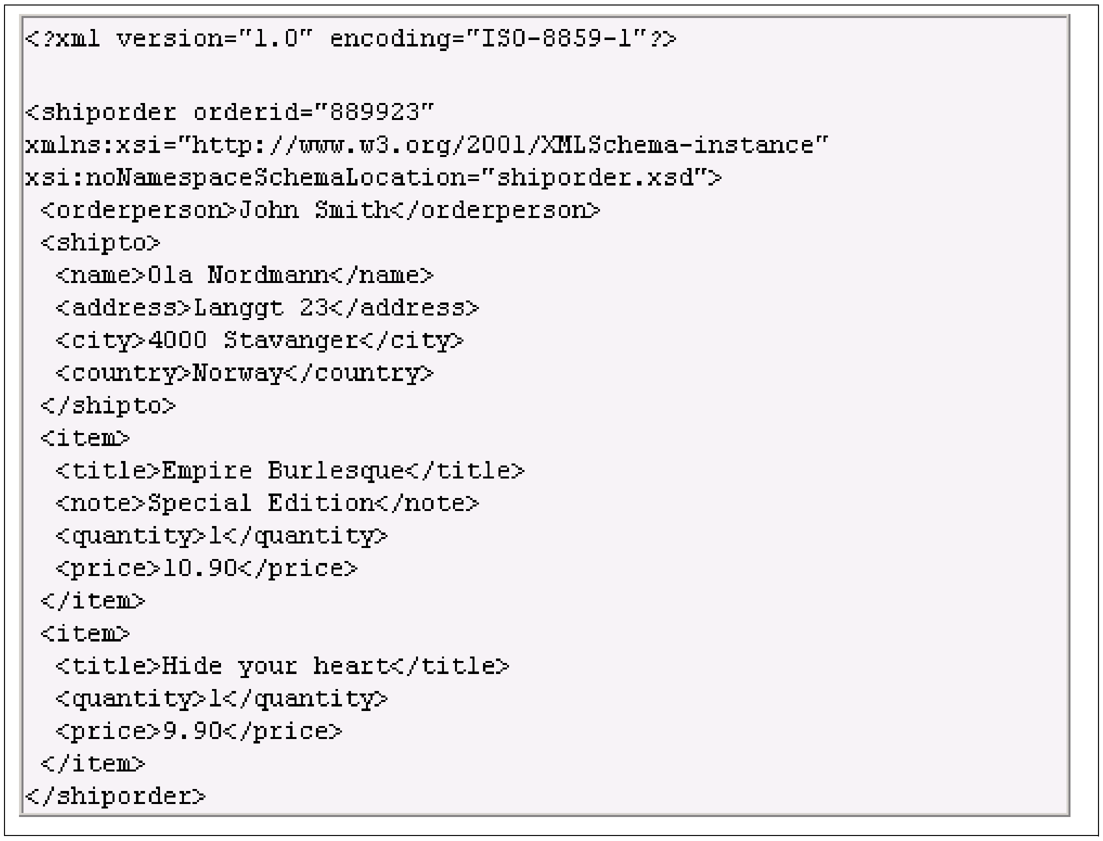
- CIM/XML protocol
- Class
- Repository
- Managed Object Format
- Provider
- CIM Object Manager (CIMOM)
- CIM operations over HTTP

20.2.1 CIM/XML protocol

The CIM/XML protocol is used to send requests to the CIM server and to receive responses from the CIM server. The CIM/XML deals with the XML schemas. A schema describes the structure of an XML document. A schema defines the following:

- Elements that can appear in a document
- Attributes that can appear in a document
- Which elements are child elements
- The order of child elements
- The number of child elements
- Whether an element is empty or can include text
- Data types for elements and attributes
- Default and fixed values for elements and attributes

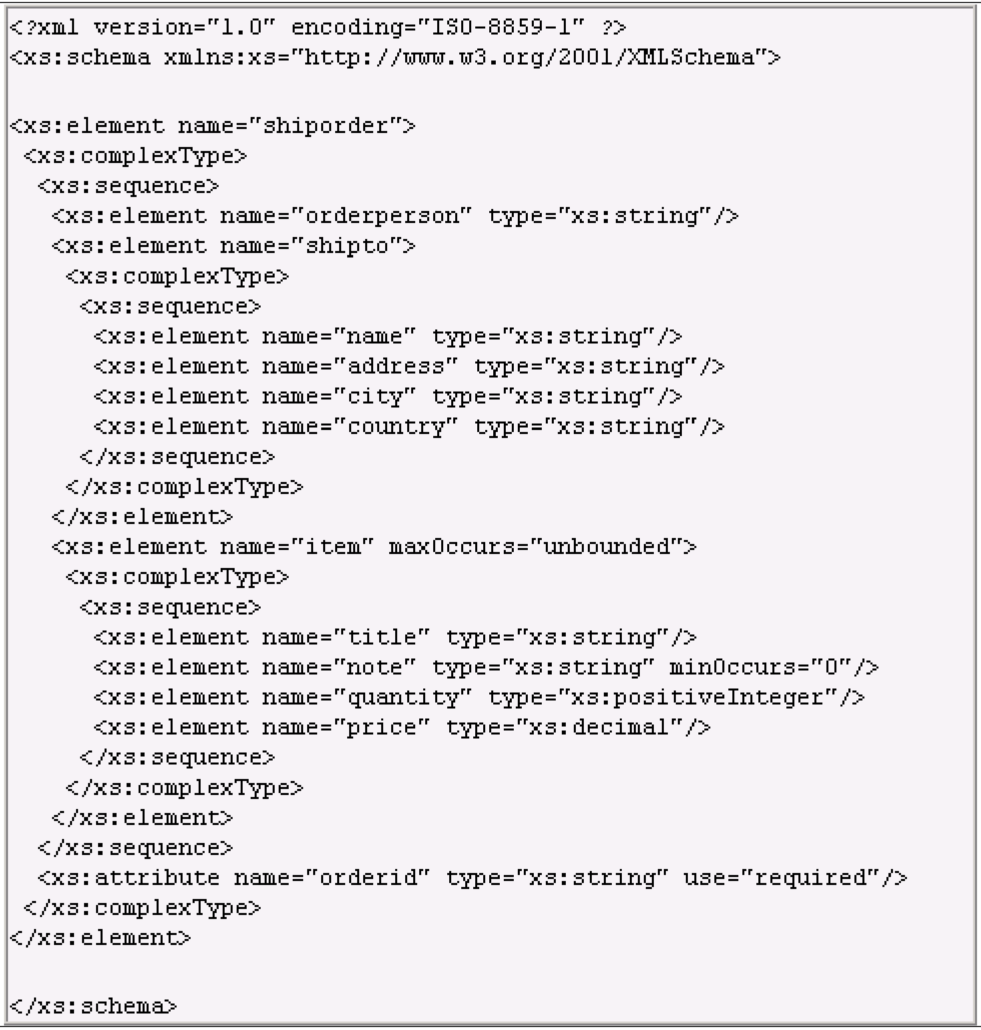
Figure 20-3 is an example of an XML document and its schema.



```
<?xml version="1.0" encoding="ISO-8859-1"?>

<shiporder orderid="889923"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="shiporder.xsd">
 <orderperson>John Smith</orderperson>
 <shipto>
  <name>Ola Nordmann</name>
  <address>Langgt 23</address>
  <city>4000 Stavanger</city>
  <country>Norway</country>
 </shipto>
 <item>
  <title>Empire Burlesque</title>
  <note>Special Edition</note>
  <quantity>1</quantity>
  <price>10.90</price>
 </item>
 <item>
  <title>Hide your heart</title>
  <quantity>1</quantity>
  <price>9.90</price>
 </item>
</shiporder>
```

Figure 20-3 The shiporder XML document

Figure 20-4 on page 399 shows a schema of the shiporder.



```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="shiporder">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="orderperson" type="xs:string"/>
   <xs:element name="shipto">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="address" type="xs:string"/>
      <xs:element name="city" type="xs:string"/>
      <xs:element name="country" type="xs:string"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="item" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="note" type="xs:string" minOccurs="0"/>
      <xs:element name="quantity" type="xs:positiveInteger"/>
      <xs:element name="price" type="xs:decimal"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="orderid" type="xs:string" use="required"/>
 </xs:complexType>
</xs:element>

</xs:schema>
```

Figure 20-4 The schema of the shiporder

20.2.2 Class

A class is related to the object programming. It is made up of attributes and methods. As an example, you can create a class called book. Then you can define methods to run against this class. One of them could be to lend the book, another to sell the book, and so on. There are two types of classes in the CIM server:

- OS management classes
- OS monitoring classes

The OS management classes are implemented by the providers of z/OS CIM. The OS monitoring classes are implemented by the z/OS RMF monitoring providers. There are network classes that are implemented by the providers of z/OS Communication Server.

20.2.3 Repository

The CIM Server repository contains the definitions of classes and instances that describe managed objects and the relationships among them. The repository is not available for use by clients or providers for static or persistent data storage.

20.2.4 Managed Object Format (MOF)

The Managed Object Format (MOF) is a formal description of the classes and associations that originate in CIM schemas. MOF is typically its own grammar, but can be translated to XML using a document type definition available from the DMTF. A MOF file can be compiled using cimmof or cimmof1 and then stored in the repository.

20.2.5 CIM provider

Upon a CIM client request, a provider is started to satisfy the request. Following is a flow of a CIM request:

- A client issues a request for data or an operation; the request is encoded in xmlCIM by the Client API (if used) and sent to the CIM server.
- The CIM server receives the xmlCIM request, decodes it, and authenticates it by determining whether the username and password are valid (authentication is automatic for local requests (see the connectLocal() client function).
- The CIM server determines whether the user is authorized to perform the requested operation in the specified namespace.
- The CIM server searches the provider registration information to determine which provider can service the request.
- The CIM server loads the appropriate shared library (if it’s not already loaded), and calls the appropriate provider.
- The provider has the responsibility to determine whether the client is authorized to issue the request if necessary (in addition to the simple namespace authorization already performed by the CIM server).
- The provider performs whatever operations are required to satisfy the request, and delivers information requested, as appropriate, to the CIM server.
- The CIM server encodes the information received from the provider into xmlCIM and returns it in its reply to the client.

20.2.6 CIM object manager

This is the common conceptual framework for data management that receives, validates, and authenticates the CIM requests from the client application. It then directs the requests to the appropriate component or device provider.

20.2.7 CIM operations over HTTP

CIM operations over HTTP allow implementations of CIM to operate in an open, standard manner. It describes how CIM operations are encoded in the HTTP payload using XML and defines the syntax and semantics of the requests and their corresponding responses.

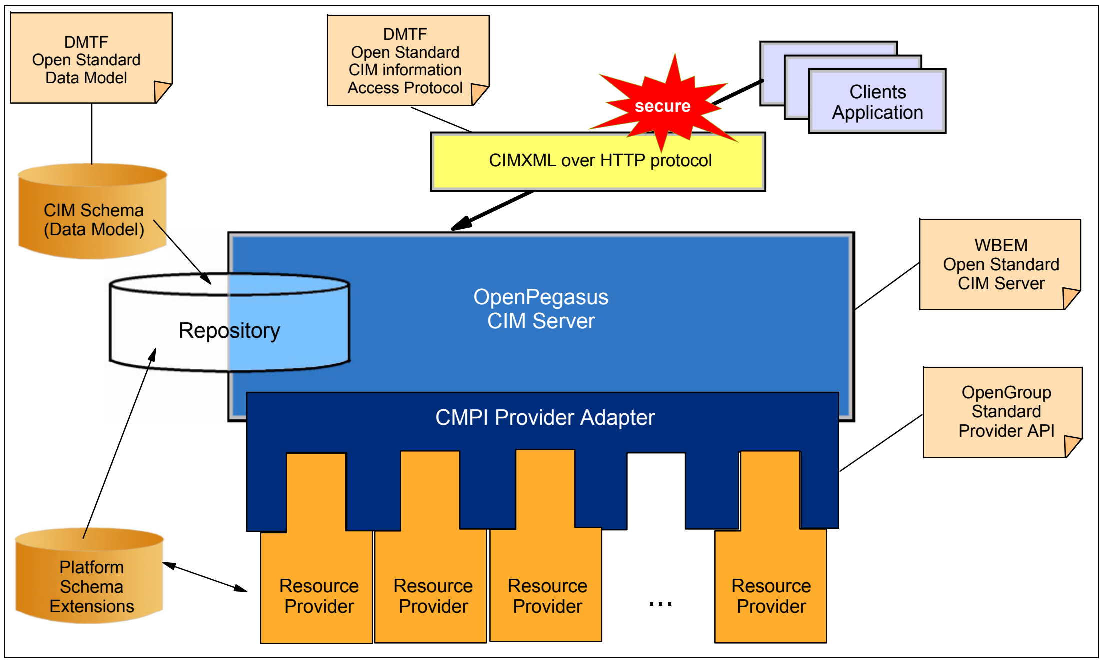


Figure 20-5 The CIM server architecture

20.3 Updates to z/OS CIM

z/OS V1R8 includes a new version of the Common Information Model (CIM). The upgrade covers the CIM Schema to 2.9, the HTTP chunking, and the capability to run CIM providers in a separate address space. In addition, a command line interface is planned to be provided to execute CIM Client requests against the CIM server. Improvements are planned to be implemented in the areas of security, reliability, and scalability of the CIM server.

20.3.1 CIM Schema 2.9

With CIM 2.9, DMTF implemented improvements to its technical release processes. DMTF releases incremental CIM feature sets as each set is finalized (rather than a single annual revision of the schema). CIM 2.9.0 introduces new diagnostic components needed for the DMTF's common diagnostic model (CDM), version 2. With the inclusion of CDM 2.0's enhanced multi-vendor diagnostic capabilities, CIM 2.9.0 plays a key role in improving system reliability, availability, and serviceability for end users. CDM's standards-based diagnostics improve system test capabilities. It makes possible the interoperability and the ability to reuse diagnostic tests across platforms. In addition, CDM creates diagnostic instrumentation at the driver level and generates fault events that can be utilized by platform management applications. In version 2.0, CDM delivers new standardized logging mechanisms and tighter synergy with the other models in CIM. The files for CIM 2.9.0 can be downloaded from the DMTF Web site at:

```
http://www.dmtf.org/standards/cim
```

20.3.2 Chunked transfer encoding

The hypertext transfer protocol originally allowed only one request per TCP connection. However, establishing a TCP connection is fairly expensive timewise, so that some implementors of HTTP/1.0 added so-called *Keep-Alives* to keep a connection open after a request was completed and to allow further requests to be made over that connection. Unfortunately, this was not well defined and is broken in the face of proxies. HTTP/1.1 defines persistent connections correctly and even makes them the default.

In the Pegasus enhancement proposal #140, a specification requires that transfer encoding with value="chunked" be supported by any client claiming to be HTTP 1.1 compliant. The description of this specification is in Section 3.6.1 at:

```
http://www.ietf.org/rfc/rfc2616.txt
```

The Pegasus server is currently compliant because HTTP does not require servers to send chunked even if the client requests it to do so.

This feature allows the message to be sent over the network in "chunks" as they are sent down by the server application. Each chunk is a complete CIM object. Each time an XML encoded object is sent to the client, the memory is released. This realizes a big memory savings since both XML encoded objects and application objects are released after they are sent to the client as they become available.

Persistent connections

The HTTP persistent connections have the following advantages:

- By opening and closing fewer TCP connections, CPU time is saved in routers and hosts (clients, servers, proxies, gateways, tunnels, or caches), and memory used for TCP protocol control blocks can be saved in hosts.
- HTTP requests and responses can be pipelined on a connection. Pipelining allows a client to make multiple requests without waiting for each response, allowing a single TCP connection to be used much more efficiently, with much lower elapsed time.
- Network congestion is reduced by reducing the number of packets caused by TCP opens, and by allowing TCP sufficient time to determine the congestion state of the network.
- Latency on subsequent requests is reduced since there is no time spent in TCP's connection opening handshake.
- HTTP can evolve more gracefully, since errors can be reported without the penalty of closing the TCP connection. Clients using future versions of HTTP might optimistically try a new feature, but if communicating with an older server, retry with old semantics after an error is reported.

20.3.3 The Out-of-Process provider support

For z/OSV1R7, a provider always runs in the same address space as the z/OS CIM server, which has the following undesirable side effects:

- Providers are able to crash the CIM server, intentionally or accidentally. The CIM server has no defense against a provider simply calling exit(). The risk of provider stability issues increases with support for indication providers. Just as the CIM server is vulnerable to provider programming errors, providers themselves are vulnerable to faults by other providers.
- Providers can access internal CIM server data and functions, potentially altering the state of the CIM server. The provider lifecycle is controlled by the CIM server rather than the provider owner. For example, a provider may wish to gather data independent of client requests. In the z/OS V1R7 arrangement, the provider does not get loaded or initialized until it is needed to fulfill a client request. If a provider requires to run APF authorized this requires the whole CIM server to run authorized, since a single address space can only run one authorization model.

In the Out-of-Process provider scenario the CIM server address space relies on trusting the Provider Agent address spaces. This trust can be established by using the must-stay-clean flag support in the UNIX System Services. The CIM server address space enables the flag and from that point in time cannot load any non-program controlled library. When a new address space is created by the CIM server address space, it automatically inherits the must-stay-clean flag and thus the Provider Agent address space can check whether the CIM server address space is clean.

The Out-of-Process feature introduced with z/OS V1R8 solves these problems. Providers can be managed in separate address spaces rather than loading and calling provider libraries directly within the CIM server process. This converts the CIM server process into a daemon process that starts off several server processes (Provider Agent process). Providers are then run in threads by the Provider Agent. The Provider Agent communicates with the CIM server process in memory via Pipes. The trust base between both address spaces can be secured with the new z/OS feature.

The code path length to process a request increases when the Out-of-Process feature is turned on. Most of this additional path length occurs because of the serialization/deserialization of request and response messages flowing between the CIM server and Provider Agent address space.

20.3.4 The CIMCLI interface

OpenPegasus provides a command-line interface called CLI through which you can perform CIM client requests/operations. On z/OS, this interface is called *cimcli*. It implements all of the DMTF CIM operations except for the modify and create class/instance operations. Each execution of cimcli invokes a CIM operation with the corresponding parameters equivalent to the CIM operations defined in the CIM Operations over HTTP specification. In addition to the basic CIM operations defined in this specification, this command-line interface implements a number of other specific operations that support testing and querying CIM servers, including operations to query for namespaces and to get all instances in a namespace. The command line client is invoked from the USS shell and has the following syntax:

```
cimcli [operation] [CIMObject] [options] *[extra parameters]
```

20.3.5 Security in CIM

Security of the CIM server was improved and is more fine-granular configurable than for z/OS CIM server 1.7. The base structure is kept for an easier understanding of the way the different security functions play together. For RACF users, the WBEM class is created using the new RACF feature for dynamic creation of classes.

Network security is insured the same way as for CIM server 1.7 via the Communications Server feature TTLS. That way all communication between a CIM client and the CIM server is encrypted and tap-proof. The main target for the security model was to make sure that no user could gain access that wasn't granted before. For this the user is first authenticated via userid and password. In the next step a user is authorized against the CIMSERV profile in the SAF WBEM class, depending on the kind of request. The CIM server differentiates between requests only receiving system information, changing properties of the managed system, and operative changes (controlling the CIM server and Schema data).

In z/OS V1R8, the ability to secure single providers is introduced. The profile to be used for this additional check has to be specified in the provider registration mof. The CIM server checks user access against the SAF security profile. In provider registration (provider-based authorization) the profile the CIM server shall use for this check. A CIM operations driven against a provider who defined a SAF profile now will only be executed if the request clients user ID does have access to that profile being defined forthe SAF WBEM class. The last step taken to secure processing is to run providers processing a request in a separate thread running with the security settings of the requesting client userid (thread-level security).

The CIM server runtime environment security is split into authentication and authorization. Authentication is enabled for the CIM server. The CIM server supports authorization via the RACF WBEM class, in which the single profile CIMSERV currently restricts access to the CIM server. For authorization purposes to specific z/OS system resources, the CIM server executes requests under the user ID from which the request was generated. To do this, the CIM server uses thread-level security, which is provided by the UNIX System Services. Additionally, the CIM server is enabled for the enhanced security model. Under the enhanced security model, the CIM server does not load any dynamic link library that is not program controlled, in particular it does not load any such provider DLL. Setting up security for the CIM server includes the following steps:

1. Define a RACF class and profile for the CIM server.
2. Define a user ID for the CIM server and grant it access to the CIM server's RACF profile.
3. Configure the CIM server's authorization model.
4. Grant client users and administrators access to the CIM server.
5. Optionally configure HTTPs for the CIM server.

Authorization is based on z/OS resource CIM authorization (RACF WBEM class).

21

Enterprise Workload Manager

This chapter discusses the following:

- Enterprise Workload Manager (EWLM) performance enhancement for DB2 Distributed Data Facility (DDF)
- EWLM support for WLM execution delay monitoring services
- Assign Service/Report class based on EWLM classification
- EWLM zAAP support

21.1 WLM performance enhancement for DB2 DDF

Enterprise Workload Manager (EWLM) is a product of the IBM Virtualization Engine™ solution that dynamically monitors and manages distributed heterogeneous workloads to achieve user-defined business goals. It helps your IT infrastructure "think" like you do, like a business. Monitoring of business transactions now includes resources spanning multiple hardware platforms, operating systems, networks, and applications. Starting from service-level objectives established for your business applications, it learns the relationships between the servers, the application middleware instances, and your business processes.

Performance tuning and workload management is happening vertically while most business transactions in a three-tier architecture (Web, application, data) encompass a horizontal work flow. Thus, a number of problems frequently occur that are difficult and time-consuming to resolve. For example, simple performance problems at one tier can materialize as major resource contention issues on some other tiers. Sorting out cause from effect can take hours, days, or sometimes weeks.

Reporting information from a business perspective lets you rapidly understand when goals are not being achieved, where the bottlenecks are, and what the business impact is. Detailed internal performance statistics are maintained by EWLM, explaining where time was spent and why, so you can quickly zoom into the area where investigation is required. It will also help you avoid investigating non-problem situations.

The statistics maintained by EWLM form the basis for the introduction of goal-oriented autonomic management techniques, like those from the z/OS platform. So, while you are looking at a potential problem, the environment is adapting itself to mitigate, or even eradicate, the problem. As is the case with the z/OS Workload Manager, the EWLM product provides management capabilities to monitor workloads and dynamically reallocate available resources to meet business objectives.

Important: EWLM has chosen to use the Application Response Measurement (ARM) 4.0 standard as the suggested way for middleware applications to instrument their software for EWLM. But in z/OS, existing middleware such as DB2, WAS, CICS, and IMS have used z/WLM proprietary instrumentation interfaces for years. The idea is to map this existing instrumentation to Application Response Measurement (ARM) calls as much as possible, so that EWLM can simply pick up that data.

21.1.1 What is DDF

OS/390's Distributed Data Facility (DDF) is a built-in component of DB2 for z/OS. It provides the connectivity to and from other databases (or servers like DB2 Connect™) over the network. DB2 for z/OS and OS/390 DDF support two network protocols, SNA and TCP/IP, as well as two database communication protocols called DB2 Private Protocol and Distributed Relational Database Architecture (DRDA).

DDF implements a full DRDA Application Server and Application Requester. However, the DB2 Private Protocol functionality has not been enhanced for a long time and, although still supported, we do not encourage people to use it, especially when you are just starting with DDF. DDF is DB2's transaction manager for distributed database connections. Unlike CICS or IMS, connections via DDF are not localized to a few address spaces within z/OS. With DRDA, connections can come from literally anywhere within the bounds of the SNA or TCP/IP network that DB2 is operating in. For this reason DDF has developed very mature thread management strategies to be able to handle thousands of connections from anywhere. DDF runs as an additional address space in the DB2 subsystem. The address space name is xxxxDIST, where xxxx is the DB2 subsystem name. DDF uses SRBs instead of TCBs, which reduces CPU time. MVS enclaves are used in exchanging data across address spaces. This enables proper management by Workload Manager (WLM) of the work coming into DB2 through DDF.

21.1.2 DB2 DDF instrumentation

Instrumentation provides the data to create the topology of a transaction. It describes what server instances or application instances a transaction flows through. In a typical Web transaction (for example, buy books) the transaction starts at a Web server (edge server), flows to an application server, then to a database server. Without instrumentation, the transaction would look like three separate processes with three separate response times rather than one transaction. This is key to EWLM, that is, providing end-to-end topology views and statistics for business transactions. To report on these transactions and potentially manage the EWLM management domain based on these statistics, Application Response Measurement (ARM) 4.0 has been implemented for application instrumentation.

Currently there are three types of applications instrumented with ARM 4.0:

- WebSphere Application Server
- DB2 UDB
- Web servers

To monitor work requests in a multi-tiered heterogeneous environment, you must identify and track the performance of those requests across server boundaries. It is possible to collect this data using versions of middleware that have been instrumented with the ARMV4.0 standard.

To collect this data, the application that EWLM supports goes through the following ARM function calls:

- arm _ register_ application
- arm _ start_ application
- arm _ register_ transaction
- arm _ start_ transaction
- arm _ block_ transaction
- arm _ unblock_ transaction
- arm _ stop _ transaction
- arm _ stop _ application
- arm _ destroy_ application

Note: A transaction in an EWLM environment is encapsulated between arm_start_transaction and arm_stop_transaction function calls issued by an application or middleware.

21.1.3 Update to the DB2 DDF instrumentation

This update is due to a performance problem. Using EWLM generated extra calls to WLM. These additional calls generated CPU overhead. To fix this, the EWLM enclaves services are lightweighted.

Note: To use the update you have to install EWLM V2.1 and apply APAR PK11801.

21.1.4 Migration consideration

In order to use this feature in pre-z/OS V1R8 releases, APAR OA12005 must be applied.

21.2 WLM execution delay monitoring services

Using the execution delay monitoring services, workload management knows how well work is executing, and locates the delays. The execution delay monitoring services are for complex work manager configurations that process across systems in a sysplex, but do not allow MVS to individually manage resource consumption of the transactions. The services allow MVS to recognize additional address spaces that are processing transactions. When the execution delay monitoring services are used, MVS can allocate resources for address spaces based on the behavior of the transactions being serviced by them. The services also provide execution delay information, so that your customers can determine where work is being delayed. They can then adjust the work manager configuration to consistently meet the goals. Only response time goals can be used with execution delay services.

Execution delay monitoring is mutually exclusive with enclaves in the same address space, so you must choose which function best suits your needs. Enclave services provide more granular resource control and reporting than execution delay monitoring services, but do not provide the capability for the work manager to report its own view of transaction states. The subsystem work manager uses the execution delay monitoring services to tell workload management about their view of the current state of a work request, such as ready state, idle state, or waiting state. The actual state may be different. For example, a work request may be active from the subsystem’s view, but might be delayed by a page fault, or for CPU access. The information is kept in performance blocks, also called monitoring environments. The monitoring environments represent work wherever it executes: across multiple dispatchable units, address spaces, and systems.

21.2.1 Changes to WLM to participate in EWLM

Previously, WLM execution delays monitoring services could not participate in EWLM. Thus, customers could not see what was going on on the z/OS platform via EWLM. To fix this, the interfaces of the execution delay monitoring services were created. The following services were impacted:

- IWM4MCRE
- IWM4MINI
- IWMRPT
- IWMMNTFY
- IWMMEXTR
- IWMMXFER
- IWMCLSFY

To exploit this new feature, CICS and IMS have to enhance their instrumentation.

In order to use this feature in pre-z/OS V1R8 releases, APAR OA12935 must be applied.

Tip: The MVS operator command D WLM,AM,A,LL and the IPCS command WLMDATA ARMAGENT {DETAIL/SUMMARY} are added.

21.3 Assign service/report class on EWLM classification

With EWLM, there are now two workload policies: EWLM and WLM. The EWLM policy manages the workload on different platforms whereas the WLM policy manages the workload on the z/OS platform only. Thus, the scope of the EWLM policy is broader than the scope of the WLM policy. As an example, when running a transaction across different platforms EWLM is able to view how the transaction is working on every platform whereas WLM can only see what is going on the z/OS platform. To associate a work request to a service class, a work manager passes some attribute values that EWLM matches against filters and z/OS WLM matches against subsystem work qualifiers. The risk is that the attribute values for work requests received by the entry application of an EWLM Management Domain might be different from those passed to z/OS WLM, especially if the classification on the distributed platform happened in an application environment different from the entry point on z/OS. For this reason it may not always be possible to have a one-to-one mapping for EWLM service classes and z/OS WLM service classes. To fix this, a few changes to WLM were made, which we discuss now.

21.3.1 Updates to WLM

The following changes were implemented:

- Subsystem EWLM in the WLM policy
- New WLM work qualifiers
- IWMCLSFY service

Subsystem EWLM in the WLM policy

The subsystem type EWLM comprises a special function of WLM: it assigns EWLM transaction classes and service classes to WLM service classes. Thus, WLM supports the following:

- Assigning EWLM service classes to WLM service or report classes
- Assigning EWLM transaction classes to WLM service or report classes

Assigning several EWLM service or transaction classes to a WLM service class means: End-to-end work requests that are already classified in one of those EWLM service or transaction classes are managed on z/OS according to the goals of the assigned WLM service class. If an EWLM service or transaction class is correlated with a WLM service class, WLM assigns the specified service class right away instead of going through the z/OS classification process. Thus, end-to-end work processed by a z/OS subsystem can be managed towards different goals than local z/OS work and it can also be reported in different report classes than local z/OS work. This function has the following advantages:

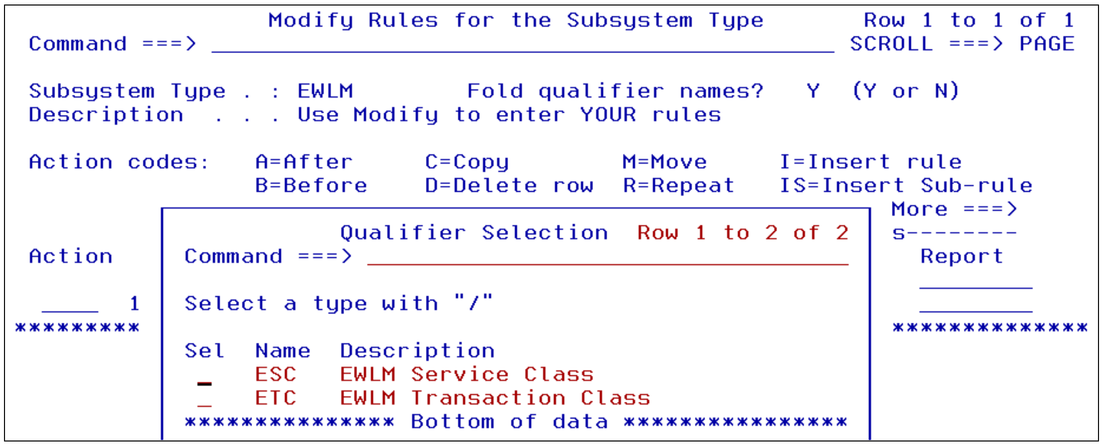
- Management is still controlled by WLM and is done according to the goal of the service class assigned to that work on z/OS.
- On the EWLM control center, the performance administrator has access to a report that integrates the end-to-end work within the same transaction class for reporting and management.

New work qualifiers

Classification rules are the rules you define to categorize work into service classes, and optionally report classes, based on work qualifiers. A work qualifier is what identifies a work request to the system. The first qualifier is the subsystem type that receives the work request.

There is one set of classification rules in the service definition for a sysplex. The rules are the same regardless of what service policy is in effect; a policy cannot override classification rules. You should define classification rules after you have defined service classes, and ensure that every service class has a corresponding rule.

In order to assign a WLM service class and a WLM report class according the EWLM classification, the work qualifiers EWLM service class (ESC) and the EWLM transaction class (ETC) were designed, as shown in Figure 21-1.



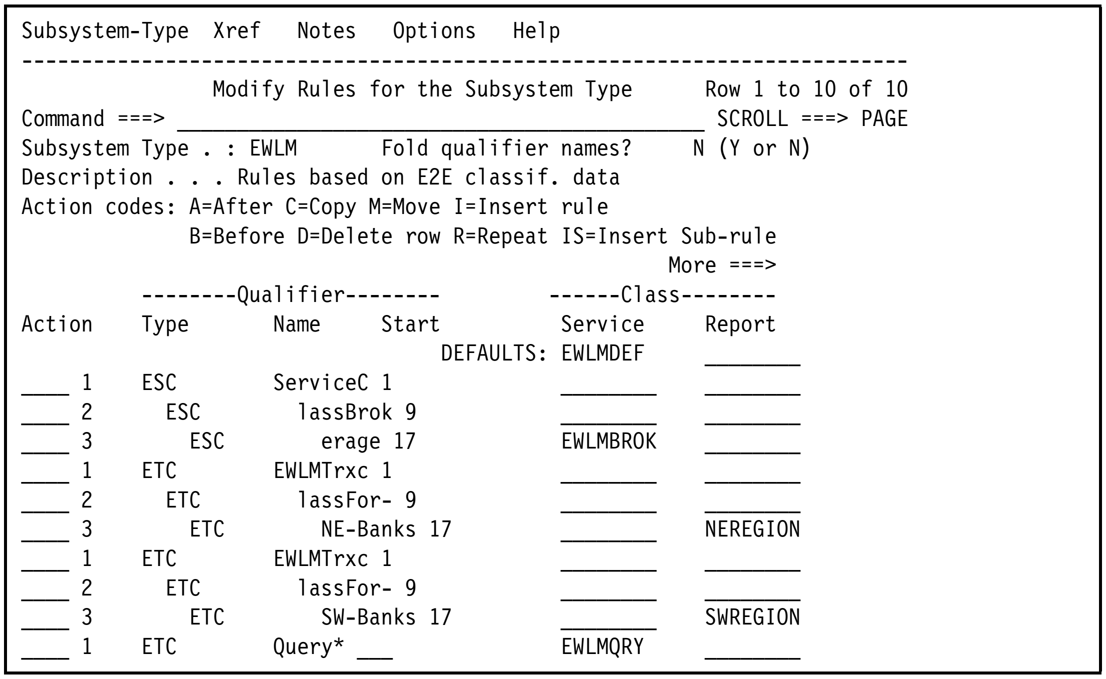
```
                        Modify Rules for the Subsystem Type       Row 1 to 1 of 1
 Command ===> ____________________________________________ SCROLL ===> PAGE

 Subsystem Type . : EWLM        Fold qualifier names?   Y  (Y or N)
 Description  . . . Use Modify to enter YOUR rules

 Action codes:   A=After     C=Copy        M=Move     I=Insert rule
                 B=Before    D=Delete row  R=Repeat   IS=Insert Sub-rule
                                                              More ===>
                           Qualifier Selection  Row 1 to 2 of 2   s--------
 Action       Command ===> __________________________________      Report
                                                                   ________
  ____   1    Select a type with "/"                               ________
*********                                                         **************
              Sel  Name  Description
               _   ESC   EWLM Service Class
               _   ETC   EWLM Transaction Class
              *************** Bottom of data ****************
```

Figure 21-1   EWLM work qualifiers

To exploit the special function of subsystem EWLM, you can define classification rules that allow to assign EWLM service classes or EWLM transaction classes to WLM service or report classes. The following classification rules (or work qualifiers) are valid:

ESC	Used for EWLM service classes
ETC	Used for EWLM transaction classes

Figure 21-2 on page 411 shows what the definitions could look like.



```
Subsystem-Type  Xref   Notes   Options   Help
--------------------------------------------------------------------------
                Modify Rules for the Subsystem Type      Row 1 to 10 of 10
Command ===> ____________________________________________ SCROLL ===> PAGE
Subsystem Type . : EWLM       Fold qualifier names?     N (Y or N)
Description . . . Rules based on E2E classif. data
Action codes: A=After C=Copy M=Move I=Insert rule
              B=Before D=Delete row R=Repeat IS=Insert Sub-rule
                                                      More ===>
          --------Qualifier--------         ------Class--------
Action    Type       Name     Start          Service     Report
                                   DEFAULTS: EWLMDEF     ________
____ 1    ESC        ServiceC 1              ________    ________
____ 2      ESC        lassBrok 9            ________    ________
____ 3        ESC        erage 17            EWLMBROK    ________
____ 1    ETC        EWLMTrxc 1              ________    ________
____ 2      ETC        lassFor- 9            ________    ________
____ 3        ETC        NE-Banks 17         ________    NEREGION
____ 1    ETC        EWLMTrxc 1              ________    ________
____ 2      ETC        lassFor- 9            ________    ________
____ 3        ETC        SW-Banks 17         ________    SWREGION
____ 1    ETC        Query* ___              EWLMQRY     ________
```

Figure 21-2 Sample classification rules panel

Important:

- ESC and ETC are only supported for subsystem EWLM.
- The maximum length of an EWLM ESC or ETC name is 64 characters. To specify the entire name, eight rules would be required. Thus, the maximum length for ESC and ETC is 32 characters.
- There is no service class required for the classification rule and there is no default service class required for subsystem EWLM. If the processing of the classification rules does not result in an assignment of a valid WLM service class, the usual subsystem type-specific WLM classification rules are applied to assign a WLM service class.
- EWLM service or transaction class names can only be specified in EBCDIC characters. Therefore, to allow for defining a correlation, the transaction class names in the EWLM domain policy should only contain characters that have an EBCDIC representation.
- EWLM service or transaction class names are specified in mixed case (fold qualifier name = N) and the comparison is also done case-sensitive.
- Subrules must use the same qualifier as their parents, that is, either ESC or ETC rules.
- The specified correlation rules are processed top down. This means that the first correlation found for an EWLM service or transaction class will be applied.

WLM IWMCLSFY service

The purpose of this service is to factor in available information about an arriving work request in order to associate a service class and possibly a report class with it. To assign a work request to a WLM service class and a report class based on the EWLM service class or on the EWLM transaction class, the following changes were made to IWMCLSFY:

- EWLM_CHCORR: An optional output parameter, which contains the cross-platform Enterprise Workload Management (EWLM) child correlator associated with the instantiated sub work request.

- EWLM_CHCTKN: An optional output parameter, which contains the cross-platform Enterprise Workload Management (EWLM) child correlator token associated with the instantiated sub work request.
- PLISTVER=7: An optional input parameter that specifies the version of the macro. The range version of the macro is now from 0 to 7. PLISTVER determines which parameter list the system generates. PLISTVER=7 supports EWLM_CHCORR, EWLM_CHCTKN and EWLM_OUTCORR in addition to the parameters supported by version 0 to 7.

21.3.2 Migration considerations

In order to use this feature in pre-z/OS V1R8 releases, APAR OA12784 should be applied. This APAR increases the functionality level to 17 and the WLM level of the policy to "LEVEL017".

Note:

- The functionality level is increased (17) if the policy contains the subsystem EWLM with specified default service/report class and/or ESC/ETC rules.
- The WLM level in the policy header is set to LEVEL017, if the policy is activated from a system with OA12784 applied.
- Lower level systems in the same sysplex (without OA12784 applied) will ignore the ESC/ETC rules. Classification will only use the subsystem-specific classification rules (as before).
- Lower level systems in the same sysplex (without OA12784 applied) will not be able to activate/extract/modify a new policy (with the new constructs <=> functionality level >= 17) that is installed by a system that has OA12784 applied.
- Lower level systems in the same sysplex (without OA12784 applied) will be able to activate/extract/modify a new policy (without the new constructs <=> functionality level <17) that is installed by a system that has OA12784 applied.
- A policy ISPF data set with the new constructs (functionality level 17) cannot be modified with a WLM Administrative Application that does not have OA12784 applied.

Until you have installed OA12784 on all the systems in the sysplex, avoid modifying, installing, or activating a service definition or policy from a system that has OA12784 installed.

21.4 EWLM zAAP support

In z/OS V1R8, WLM includes zAAP data in the CPU using and delay samples as well as in the CPU service times reported to EWLM for processes using zAAPs. But EWLM cannot differentiate CP, zAAP, or zIIP processor activities.

21.4.1 Migration considerations

To use this feature on pre-z/OS V1R8 releases, APAR 0A12786 needs to be applied.

22

IBM Health Checker for z/OS in z/OS V1R8

The objective of IBM Health Checker for z/OS is to identify potential problems before they impact system availability or, in the worst case, cause outages. It checks the current active z/OS and sysplex settings and definitions for a system and compares the values to those suggested by IBM or defined by you. It is not meant to be a diagnostic or monitoring tool, but rather a continuously running preventative that finds deviations from best practices.

IBM Health Checker for z/OS produces output in the form of detailed messages to inform you of both potential problems and suggested actions to take. Note that these messages do not mean that IBM Health Checker for z/OS has found problems that you need to report to IBM. IBM Health Checker for z/OS output messages simply inform you of potential problems so that you can take action.

In this chapter, we discuss the following topics:

- The Health Checker enhancements
- The new checks

22.1 The IBM Health Checker for z/OS

Health Checker for z/OS is a tool developed to address component configuration and setup errors commonly made by installations. The goal of this tool is to avoid outages by identifying potential problems before they impact the availability of your installation.

This tool became a new component incorporated in z/OS V1R7, but it is also available as a download from the Web for use in older z/OS versions.

Health Checker for z/OS checks the current active z/OS sysplex settings and definitions for a system and compares the values to those suggested by IBM or defined by you. As mentioned, it is not meant to be a diagnostic or monitoring tool, but rather a continuously running preventative that finds deviations from best practices.

22.1.1 Health checks

A *check* is actually a program or routine that identifies potential problems before they impact availability or, in some worst cases, cause outages. A check is owned, delivered, and supported by the component, element, or product that writes it, and processing is as follows:

- You can update or override some check values using either SDSF or statements in the HZSPRMxx parmlib member or the MODIFY command. You might want to apply these installation updates or overrides if some check values are not suitable for your environment or configuration.

  As shown in Figure 22-1 on page 415, you can override some check values in the following ways:

  – Statements in the HZSPRMxx parmlib member.

    Make permanent overrides by creating policies in the HZSPRMxx SYS1.PARMLIB member.

  – Using the MODIFY command.

  – The SDSF interactive command panel identifies when SDSF can be used.

- Check output is created when a check issues its output as WTOs and other messages, which you can view using:

  – SDSF

  – The HZSPRINT utility

  – A log stream that collects a history of check output

  If a check finds a deviation from best practices or a potential problem, it issues a WTO message. We call these WTO messages *exceptions*. Check exception messages include not only a description of the potential problem found, including the severity, but also information about what to do to fix the potential problem.

Save the check results using System Logger

IBM Health Checker for z/OS retains only the check results from the last iteration of a check in the message buffer. If you want to retain a historical record of check results (which is useful), you must define and connect to a log stream. When you have a log stream connected, the system writes check results to the log stream every time a check completes.

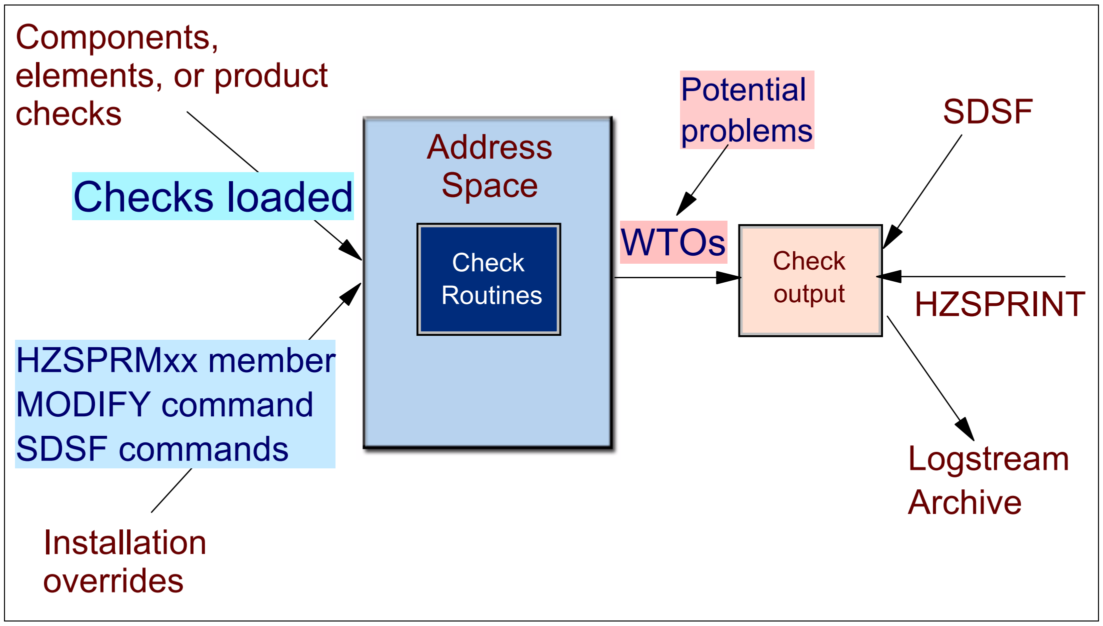


Figure 22-1 Health Checker for z/OS runs in its own address space

22.1.2 How Health Checker for z/OS can identify problems

IBM Health Checker for z/OS output messages simply inform you of potential problems so that you can take action on your installation. The goal is to know *where* you are running a risk. Then, you can be aware and take preventive actions.

The following situations are examples of how IBM Health Checker for z/OS can help by identifying potential problems:

- Changes in defaults or configuration values that occur dynamically over the life of an IPL.

  Checks that look for changes in these values should run periodically to keep the installation aware of changes.
- Threshold levels approaching the upper limits, especially those that might occur gradually.
- Single points of failure in a configuration.
- Unhealthy combinations of configurations or values that an installation might not check.
- Configuration abnormalities in what was believed to be a stable system.
- Unexpected values on a system. Investigation reveals changes had been correctly made to that system, but not replicated on other systems.
- Default configurations that were never optimized for performance.
- Outdated settings that do not support all current applications.
- Mismatched naming conventions that can lead to an outage.

Tip: Combine Health Checker for z/OS messages with an automation product, like System Automation. This is the one way to have your installation running as defined by your installation, and to have actions proceed on without human intervention. Using this method, only a part of the messages will need your action, and you can focus on the high severity exceptions.

22.1.3 Health Checker for z/OS processing

As illustrated in Figure 22-2 on page 416, Health Checker for z/OS functions in the following way:

1. Check values provided by components.

   Each check includes a set of pre-defined values, such as:

   – Interval, or how often the check will run

   – Severity of the check, which influences how check output is issued

   – Routing and descriptor codes for the check

   You can update or override some check values using either SDSF or statements in the HZSPRMxx parmlib member, or by using the MODIFY command.

2. Check output.

   A check issues its output as WTOs and other messages, which you can view using SDSF, the HZSPRINT utility, or a log stream that collects a history of check output. If a check finds a deviation from best practices or a potential problem, it issues a WTO message known as an exception, as previously mentioned. Check exception messages include not only a description of the potential problem found, including the severity, but also information about what to do to fix the potential problem.

3. Resolve check exceptions.

   To get the best results from IBM Health Checker for z/OS, let it run continuously on your system so that you will know when your system has changed dynamically from best practice values. When you get an exception, resolve it using the information in the check exception message or by overriding check values, so that you do not receive the same exceptions over and over. You can use either SDSF or the HZSPRMxx parmlib member, or the IBM Health Checker for z/OS MODIFY (`F hzsproc`) command to manage checks.

4. If you solve an exception by changing a product setting or system control, it is a good policy to rerun the checks related to this action, to guarantee that the problem identified was fixed.

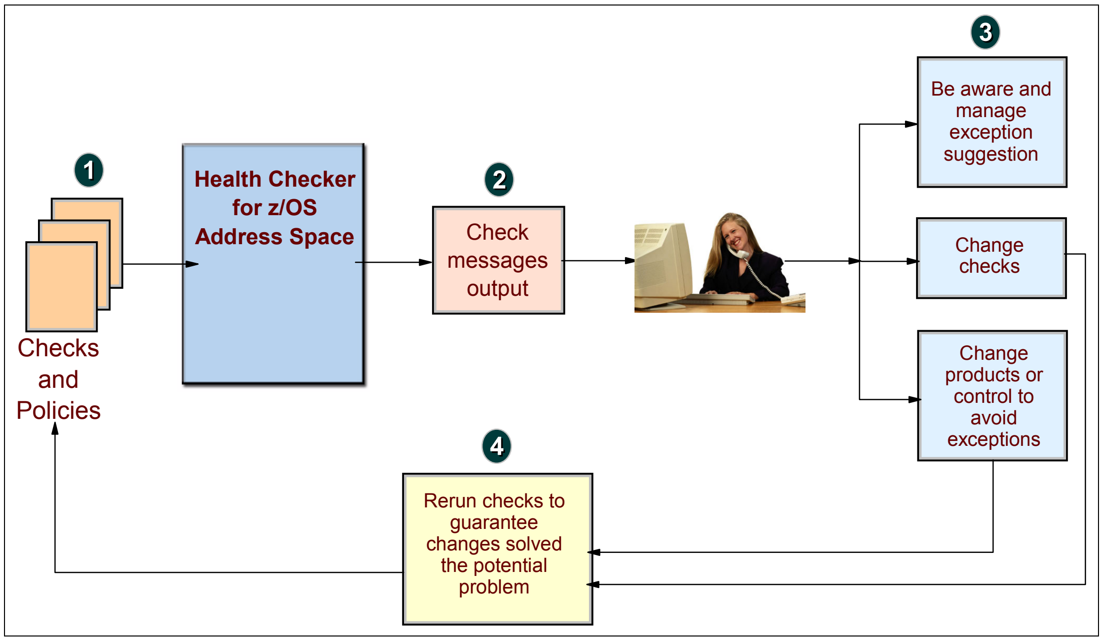

1
Checks and Policies
Health Checker for z/OS Address Space
2
Check messages output
3
Be aware and manage exception suggestion
Change checks
Change products or control to avoid exceptions
4
Rerun checks to guarantee changes solved the potential problem

Figure 22-2 Flow of the Health Checker for z/OS

IBM Health Checker for z/OS procedure

The IBM Health Checker for z/OS procedure must contain the following:

```
//HZSPROC JOB JESLOG=SUPPRESS
//HZSPROC PROC HZSPRM='00'
//HZSSTEP EXEC PGM=HZSINIT,REGION=0K,TIME=NOLIMIT,
// PARM='SET PARMLIB=&HZSPRM'
//HZSPDATA DD DSN=SYS1.&SYSNAME..HZSPDATA,DISP=OLD
// PEND
// EXEC HZSPROC
```

The value for PARM= must resolve to SET PARMLIB=(x1,...,xn). You can also use an ADD or REPLACE command in place of SET, because the command is issued during the initialization phase.

User interfaces to manage checks

You can manage checks by making dynamic or temporary changes to current checks. You make these decisions by deactivating, adding, or temporarily updating check values, using the following ways for administrators to interact with Health Checker for z/OS:

- SDSF panels or (E)JES panels
  - CK command
- MODIFY command
  - Temporary check changes
- HZSPRMxx parmlib member
  - Permanent check changes

22.2 z/OS V1R8 enhancements

The enhanced Health Checker framework addressed the following problems:

- Adding checks using the exit HZSADDCHECK
- Authorized code needed to run the checks
- No National Language Support (NLS) translation
- The Health Checker policy
- No parsing

In the following sections, we explain each item in more detail.

22.2.1 Adding checks using the HZSADDCHECK exit

Adding a check is complex and requires the following steps:

- Write a check routine that gathers information, compares current values with suggested settings or looks for configuration problems, and issues messages with the results of the check.
- Create a message table for the check output. The message table defines the check output messages issued by the check routine.
- Create a HZSADDCHECK exit routine. The HZSADDCHECK exit routine adds one or more checks, and, for each, provides the default values. This authorized routine runs in

the IBM Health Checker for z/OS address space, called by the IBM Health Checker for z/OS dynamic exit, HZSADDCHECK.

- Add the check to IBM Health Checker for z/OS by adding the HZSADDCHECK exit routine to the HZSADDCHECK exit, and then having the system run the exit routines to add checks to the IBM Health Checker for z/OS.

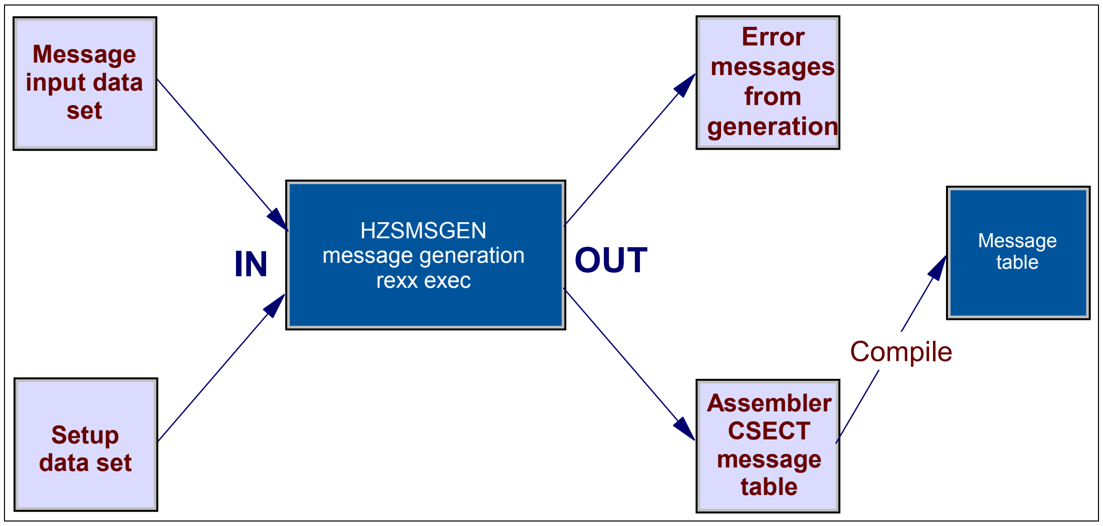

Message input data set
Setup data set
IN
HZSMSGEN message generation rexx exec
OUT
Error messages from generation
Assembler CSECT message table
Compile
Message table

Figure 22-3 The shaded boxes are the parts that the check developer must provide

22.2.2 z/OS V1R8 enhancement to add a check

To alleviate the problems of adding a check, a check can now be added using the statement ADDCHECK in the HZSPRMxx parmlib member. However, IBM checks are still added via the HZSADDCHECK dynamic exit.

Authorized code needed to run the checks

To check for potential problems, a check routine can be used. A *check routine* is a program that gathers installation information and looks for problems, and then issues the check results in messages. IBM Health Checker for z/OS writes the check messages as WTOs or to the message buffer.

The check routine runs in the IBM Health Checker for z/OS address space, which has superuser authority (UID(0)). Such a check routine is called a *local check routine*. The problem is that some checks might affect the performance of the Health Checker address space due to I/O-intensive operations, serialization or waits.

Remote check enhancement

The enhancement is to create remote checks. A *remote check* runs as tasks in the address space of the caller. For example, a remote check might run in a server address space so that it obtains the necessary data about the server space and also reads data from data sets.

Local and remote check routines share a basic structure, but there are differences between them. A remote check requires synchronization and communication between the remote check routine and IBM Health Checker for z/OS. This is important because if your check hangs in the HZSPROC address space, it can affect the performance of IBM Health Checker for z/OS and all the other checks.

IBM Health Checker for z/OS tracks remote checks for you. If the caller's address space where the remote check is running fails, IBM Health Checker for z/OS treats the check as if it had been deleted. If the IBM Health Checker for z/OS address space terminates, then upon restart it restarts any remote checks that were defined to the system when the address space terminated, unless they have been explicitly deleted.

Note: You should write a remote check in the following cases:

1. You cannot access the data you need for your check from the IBM Health Checker for z/OS address space.
2. Your check requires potentially disruptive actions, such as I/O-intensive operations, serialization, or waits.

Figure 22-4 on page 419 shows the parts of a remote check, and whether they run within the IBM Health Checker for z/OS address space or not.

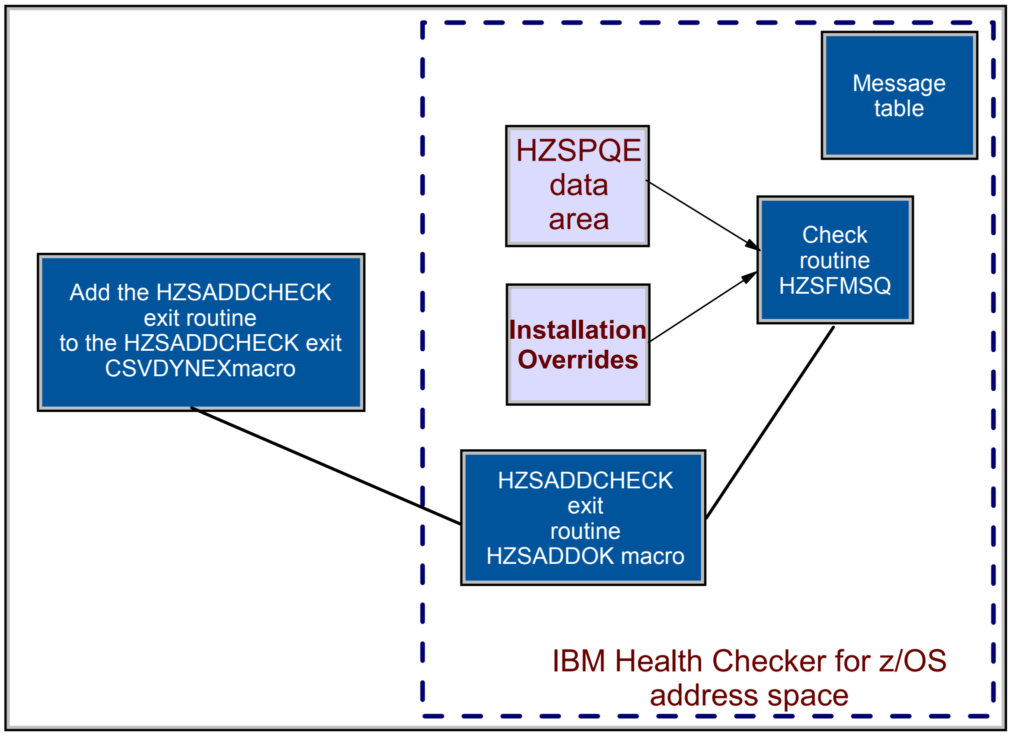

Message table
HZSPQE data area
Check routine HZSFMSQ
Add the HZSADDCHECK exit routine to the HZSADDCHECK exit CSVDYNEXmacro
Installation Overrides
HZSADDCHECK exit routine HZSADDOK macro
IBM Health Checker for z/OS address space

Figure 22-4 Shaded boxes indicate which parts the check developer must provide

22.2.3 No National Language Support (NLS) translation

The implementation of the National Language Support (NLS) translation is due to the use of the check exception WTO messages. If you want to generate skeletons for message translations for your check exception WTO messages, it will impact the way you code the message text for your messages in the message input data set.

For example, if you want to generate NLS skeletons for your messages, you must break up message text in the message input data set into lines of 71 characters or less. The line length is calculated based the total length of the message text, and the maximum length that each insert is defined.

When you use HZSMSGEN, you can specify NLSCHECK(Y) to specify that the system enforce the NLS length guideline. Then at check runtime, when an exception message is issued with the HZSFMSG service, the system also compares the length of each variable in the NLS skeletons to the defined length of the variable in the message input data set. If the skeleton and the WTO differ, then the translation fails and the system abends the check. See the “Creating the message input for your check” section in *IBM Health Checker for z/OS User’s Guide*, SA22-7994, for more information.

Note: To ensure that you can generate your exception messages successfully, and that messages will translate successfully at runtime, use the following guidelines:

1. On the first line of the message text, remember that the message identifier, or number, can require up to 11 characters.
2. You must use “<lines></lines>” tags to define a new line before you reach the WTO limit of 71 characters.
3. Specify <mv class="variable_class" xreftext="maxlen(nnn)>" for all the variables in your exception messages to define the maximum length possible for each variable. This will make it much easier for you to calculate where you need to insert a <lines></lines> tag to break up a message text to avoid exceeding the 71 character limit.

   If you do not specify maxlen, then you must allow for the maximum space allowed for the type of variable when calculating where you want to break your line with <lines></lines>.
4. For a predefined system symbol, which is resolved at check run time, you must allow for the maximum space allowed for the element when calculating the number of characters it will take up.
5. System symbols and variables can be longer than 71 characters themselves, as long as they follow a new line indicator (<lines></lines> tags together) or are the very last items in the message text.

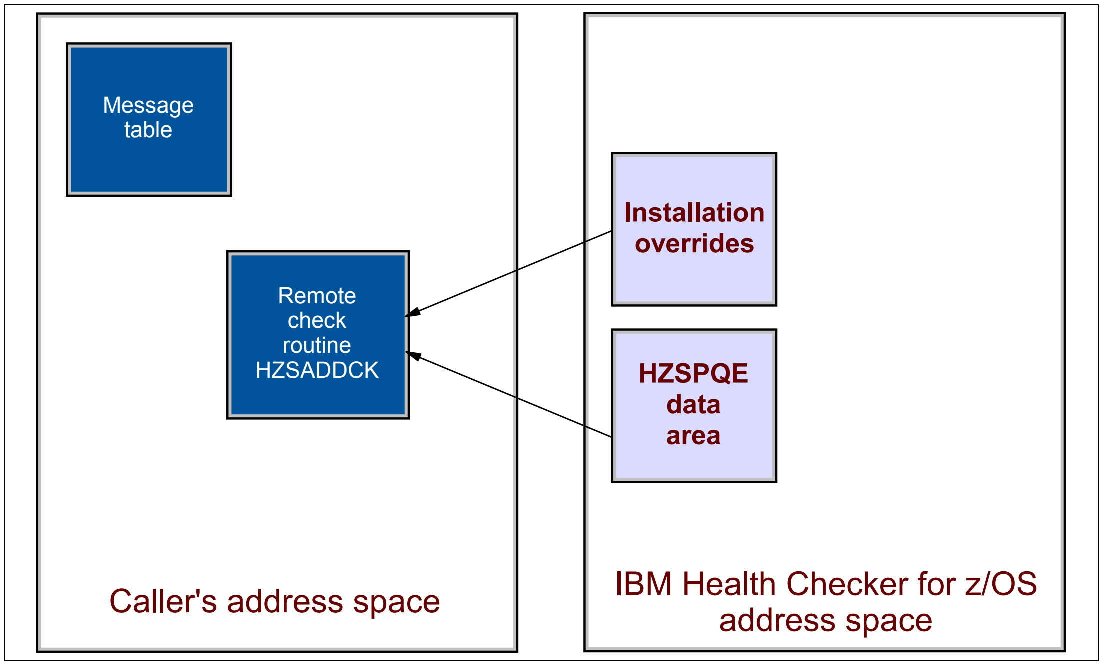


Figure 22-5 The parts involved in creating a message table

22.2.4 The Health Checker policy

Prior to z/0S V1R8, customers could not create multiple Health Checker policies. The default Health Checker policy name was DEFAULT. An IBM Health Checker for z/OS policy simply consists of a set of policy statements in an HZSPRMxx member (or members) currently in use for a system. The system applies the information in your active IBM Health Checker for z/OS policy to all existing checks and to any new checks you add. IBM Health Checker for z/OS processes information from the active policy every time checks are added or refreshed, every time you activate a new policy, and whenever you restart IBM Health Checker for z/OS.

Solution

To help customers who had different policy names, IBM added the POLICY(policyname) parameter to some policy statements in the HZSPRMxx parmlib member. The following policy statements in HZSPRMxx parmlib member were updated:

- DISPLAY
- ADD
- ADDREPLACE
- REMOVE

For more information about this topic, refer to the section “Define multiple policies in one HZSPRMxx parmlib member” in *IBM Health Checker for z/OS User’s Guide*, SA22-7994.

Tip: The parameter STMT=my_stmt_name used with the policy statements ADD POLICY and ADDREPLACE POLICY is now optional. The system calculates the value of my_stmt_name. The numeric value is incremented automatically so that there is no risk to have a duplicate statement name value within a policy.

22.2.5 No parsing

The implementation of the HZSCPARS service was due to the use of parameters that users specify using the policy statements UPDATE or POLICY UPDATE in HZSPRMxx or the MODIFY command F hzsproc,UPDATE or F hzsproc,POLICY UPDATE command, and the risk of errors when specifying them.

Solution

To prevent the risk of errors, IBM implemented the HZSCPARS service. When HZSCPARS finds a parameter error, it issues appropriate error messages using the REASON=PARSxxxx reason values on the HZSFMSG macro. This means that your check routine does not have to issue error messages for parameter errors. Your check routine can also use REASON=PARSxxxx on the HZSFMSG REQUEST=HZSMSG to issue parsing error messages in the course of doing its own parameter parsing.

Use the HZSCPARS REQUEST=PARSE in your check routine to allocate a parameter area, mapped by mapping macro HZSZCPAR, that describes the parsed parameters for the check. You can free this parameter area using the HZSCPARS REQUEST=FREE request. For a local check, if you do not free the parameter area, then the system will delete the parameter area upon return from the check routine.

Note: Your check routine must still issue the HZSFMSG REQUEST=STOP request when HZSCPARS finds a parameter error.

23

Storage clear for PL/I applications

This chapter describes initial storage clear for PL/I applications and discusses the following topics:

- The purpose of the new CLEAR sub-option value for the STORAGE run-time option support
- How the new CLEAR sub-option value for the STORAGE run-time option support fits into the existing Language Environment run-time option support
- How to use the new CLEAR sub-option value for the STORAGE run-time support

23.1 Purpose of the new CLEAR sub-option value

To reduce migration and enhance the PL/I performance issues for PL/I users, the new third sub-option value, CLEAR, was added to the STORAGE run-time option. This value will clear the unused portion of the initial stack storage prior to invoking the main procedure.

This new function will help you to do the following:

- Clear the unused portion of the Language Environment initial stack segment before the main routine gets control
- Remove the migration barrier between the pre-Language Environment PL/I run-time library and the Language Environment run-time library

23.2 How the new CLEAR sub-option value fits into LE

In previous versions of the Language Environment (LE), the run-time library behaves differently than the pre-Language Environment PL/I run-time library; that is, the pre-Language Environment PL/I run-time library clears the stack storage before giving control back to the main procedure. That limitation causes the following problems:

- The Language Environment has the STORAGE run-time option. However, its stack initialization sub-option only clears the storage for each stack frame.
- Migration and performance issues for PL/I users.

z/OS V1R8 provides initial storage clear for PL/I applications to help ease application migration by enhancing the STORAGE run-time option.

This new sub-option is supported under the following products and components:

- CICS
- CICS OTE
- DB2
- IMS
- C++
- LRR
- ILC
- Language Environment pre-initialization (PIPI)
- Preinit compatibility (PICI)
- C-MTF
- Batch
- TSO/E

This new sub-option is *not* supported under the following functions:

- XPLINK, 64-bit, authorized LE
- Downward growing stack is not affected by this change
- System programmer C (SPC)

23.3 How to use the new CLEAR sub-option value

The new CLEAR sub-option is invoked as follows:

- By specifying CLEAR as the third sub-option of STORAGE run-time option
  - STORAGE(,,CLEAR,)

The third sub-option may now be specified in three ways, as follows:

- A one-byte hex value
  - STORAGE(,,30,)
- NONE - no initialization
  - STORAGE(,,NONE,)
- CLEAR - clear unused portion of initial up-stack
  - STORAGE(,,CLEAR,)

Note the sample shown in Figure 23-1.

```
Options Report for Enclave main 06/27/05 3:45:54 PM
Language Environment V01 R08.00

LAST WHERE SET          OPTION
-------------------------------------------------------------------
Installation default    ABPERC(NONE)
Installation default    ABTERMENC(ABEND)
…
Invocation command      STORAGE(NONE,NONE,CLEAR,0)
```

Figure 23-1 Options Report and the Invocation command STORAGE

23.3.1 Restrictions and additional information

In a 64-bit environment, CLEAR specified as the third sub-option will cause a failure in TSO, batch, CELQDOPT, and so on.

For the environment variable _CEE_RUNOPTS, CLEAR must be allowed to be specified even for AMODE64 applications for a spawning process (that is, 64-bit spawns a 31-bit process and vice versa).

If the CLEAR value is specified, then CLEAR will be treated as if NONE was specified by AMODE64 applications and as if CLEAR was specified by AMODE31 applications.

CLEAR can be specified in mixed case.

CLEAR *cannot* be specified as the first, second, or fourth sub-options of the STORAGE run-time option.

23.3.2 Messages processed when the CLEAR value is used

The following error messages are returned for specifying CLEAR as the first or second sub-options of STORAGE run-time option:

- `CEE3608I The following messages pertain to the invocation command run-time options`
- `CEE3616I The string 'CLEAR' was not a valid or supported sub option of the run-time option STORAGE in this release`

The following error messages are returned for specifying CLEAR as the fourth sub-option of the STORAGE run-time option:

- `CEE3608I The following messages pertain to the invocation command run-time options`
- `CEE3614I An invalid character occurred in the numeric string 'clear' of the run-time option STORAGE`

23.3.3 Interactions and dependencies

There are no hardware or software dependencies. A coexistence APAR PK02614 is available for the following releases with PTFs:

- UK08989 for z/OS V1R4
- UK08990 for z/OS V1R5
- UK08991 for z/OS V1R6
- UK08992 for z/OS V1R7

23.3.4 Prerequisites for installation

With Language Environment, existing users who do not wish to take advantage of the new CLEAR value do not need to make any changes to their run-time options.

24

HCM/HCD enhancements

Here we describe the ease-of-use enhancements to HCD/HCM, as follows:

- Hide/show connection
- Include PCHID in cable label
- Default switch port name
- Performance improvement for defining spanned CHPIDs
- Automatic activity logging
- HCD report enhancements
- Export/Import IODF from HCM
- Locate by user field
- Multiple delete
- Extract controller physical description file
- Filtering using wildcard
- Enhancements with importing/exporting data
- Server Time Protocol (STP) link support
- New utilities to support complex configuration definitions
- Enhanced OS device group change
- Performance data integration
- Compare reports for IODFs
- Compare reports for HCM configuration files
- Restrictions

24.1 The hide/show connection

In the previous versions of z/OS, you could not “fine-tune” HCM documentation (for example, to take diagram screenshots of selected objects and connections).

z/OS V1R8 provides an HCM function to show and hide connections between objects in the HCM configuration diagram, and now you have more means to fine-tune a diagram view or a screenshot of a selected part of an I/O configuration.

For the following interface objects in a configuration diagram, you can hide or show their connections to other objects:

- CHPIDs
- Switch ports
- Controller channel interfaces

The context menus for these objects contain new choices: Show Connection or Hide Connection. You can also show or hide multiple connections in one step by selecting the appropriate objects and then selecting the new choices from the View menu bar.

Using these functions, you can do the following:

- Hide a connection between two configuration objects in the (cropped) HCM configuration diagram
- Re-show a hidden connection
- Modify cropped HCM diagrams to exclude non-interesting connections for the current view before taking a printout
- Communicate with different user groups

24.1.1 Implementation

The Hide Connection function is invoked by certain objects’ context menus. An HCM diagram shows objects and connections between objects. Currently there are certain ways to hide and show objects, as follows:

- Object-type base (via **View** → **Filter**)
- Object base (via an object’s context menu, which is available for processors, partitions, and controllers)
- Enhanced cropping (which shows selected objects and all connected objects)

To hide and show objects, use the Change View Attributes dialog, which is called by choosing the Filter diagram function from the View menu, as shown in Figure 24-1 on page 429.

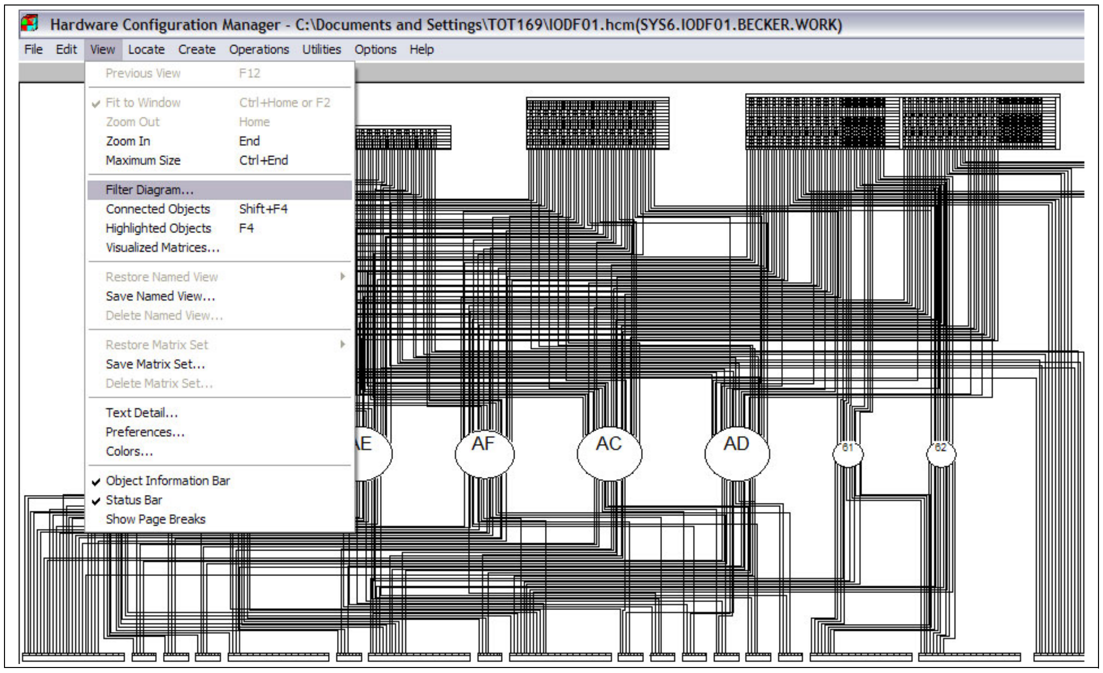


Figure 24-1 View window before selecting the filter diagram function

The filter diagram allows you to specify which processors, partitions, and controllers are displayed to reduce the number of objects in the current view.

The default setting for the Include only List is to show all DASD controllers; you can change the default and then click **OK**.

You can use the Save Named View option from the View menu to save the current view attributes to a named view. If you save the named view, it will become the most recently used named view.

Use the dialog Change View Attributes to include and exclude the types of objects you want to see, as shown in Figure 24-2 on page 430.

Note that you may set exceptions to the selected controller types in the Change View Attributes dialog with the Exceptions dialog.

Exceptions allow you to include or exclude individual controllers in the class from the view. After having selected the object types, you can now select one or more objects and press F4 (Highlighted Objects) to further reduce the diagram to display just those objects.

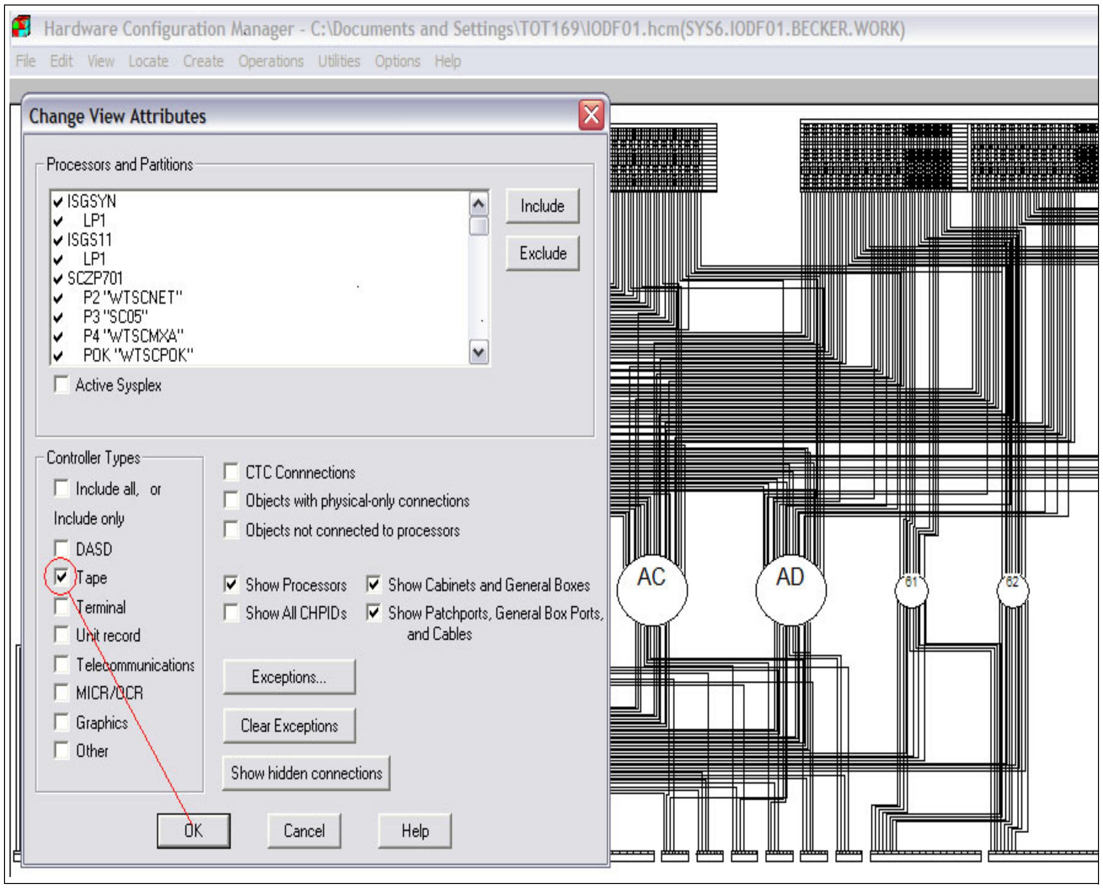
Pressing F4 considers the settings made in the Change View Attributes dialog, while pressing Shift+F4 crops the diagram without considering these settings.

To return to the original diagram, select **Clear Exceptions** in the Change View Attributes dialog. Alternatively, you can press the F12 key to restore the view that you had before pressing F4 or Shift+F4.

As shown in Figure 24-2 on page 430, the following controller types can be selected to tailor your view:

- ▶ DASD, the default
- ▶ Tape
- ▶ Terminal

- Unit record
- Telecommunications
- MICR/OCR
- Graphics
- Other


Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKER.WORK)

Change View Attributes

Processors and Partitions

- ✓ ISGSYN
- ✓ LP1
- ✓ ISGS11
- ✓ LP1
- ✓ SCZP701
- ✓ P2 "WTSCNET"
- ✓ P3 "SC05"
- ✓ P4 "WTSCMXA"
- ✓ POK "WTSCPOK"

☐ Active Sysplex

Controller Types

☐ Include all, or

Include only

- ☐ DASD
- ☑ Tape
- ☐ Terminal
- ☐ Unit record
- ☐ Telecommunications
- ☐ MICR/OCR
- ☐ Graphics
- ☐ Other

- ☐ CTC Connnections
- ☐ Objects with physical-only connections
- ☐ Objects not connected to processors
- ☑ Show Processors
- ☑ Show Cabinets and General Boxes
- ☐ Show All CHPIDs
- ☑ Show Patchports, General Box Ports, and Cables

Figure 24-2 View window after selecting the filter diagram function and choosing Tape

“Hide connection” enables you to explicitly hide individual connections between objects as shown in Figure 24-3 on page 431. Supported interface objects are as follows:

- CHPIDs (also known as Channel Interfaces)
- Switch Ports (also known as Director Ports)
- Controller CU Interfaces (also known as Channel Interfaces)

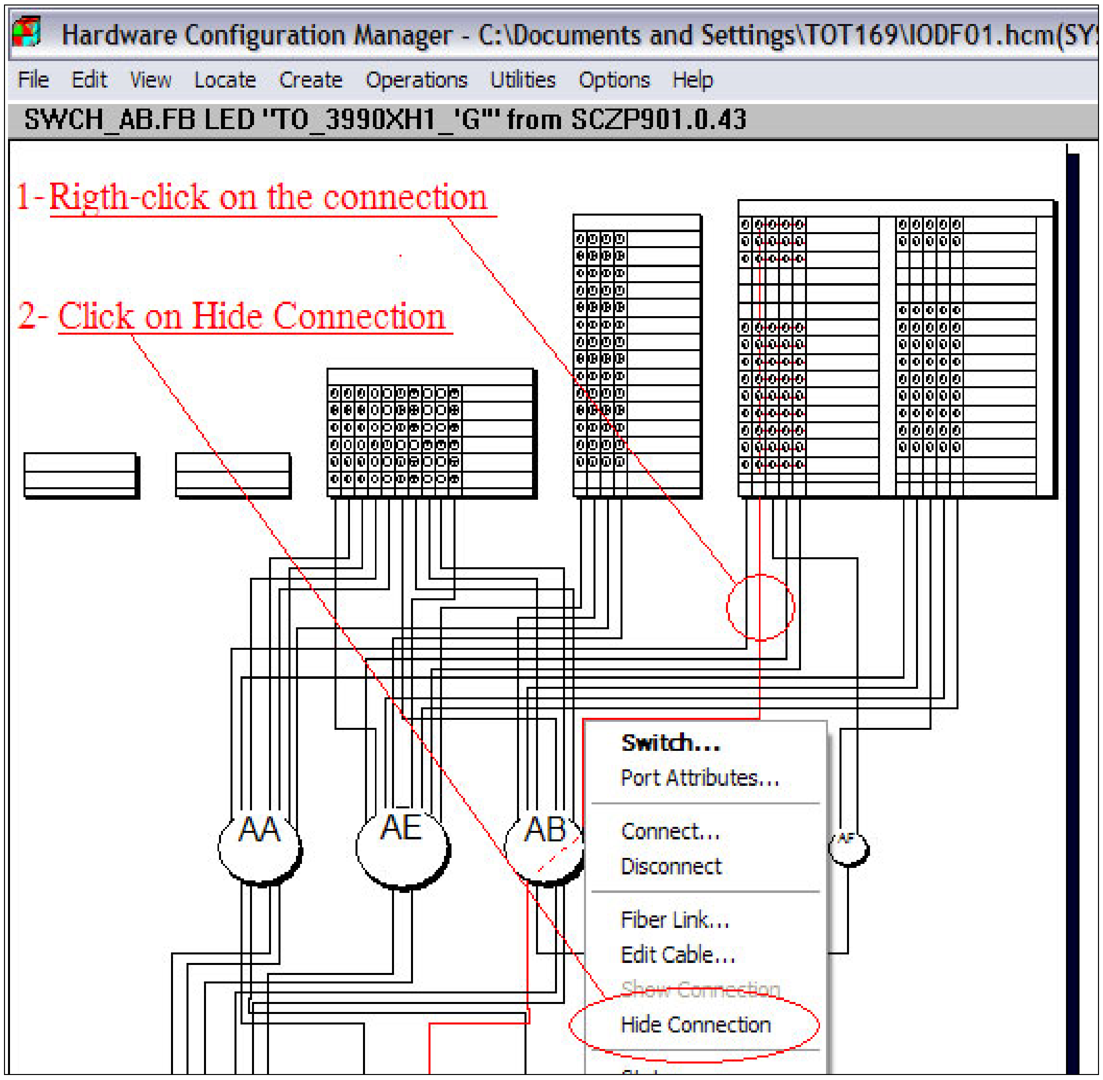

Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SY
File Edit View Locate Create Operations Utilities Options Help
SWCH_AB.FB LED "TO_3990XH1_'G'" from SCZP901.0.43
1-Rigth-click on the connection
2- Click on Hide Connection
AA
AE
AB
AF
Switch...
Port Attributes...
Connect...
Disconnect
Fiber Link...
Edit Cable...
Show Connection
Hide Connection

Figure 24-3 Steps to hide a connection with the Hide Connection function

The Show Connection function is available on individual object interfaces (whose connection is hidden), as shown in Figure 24-4 on page 432.

And globally for all currently hidden interfaces (in Filter Diagram) Show Connection enables you to re-show hidden connections.

This menu entry is available on both ends of a hidden connection.

Hidden connections are usually indicated by a “stub”.

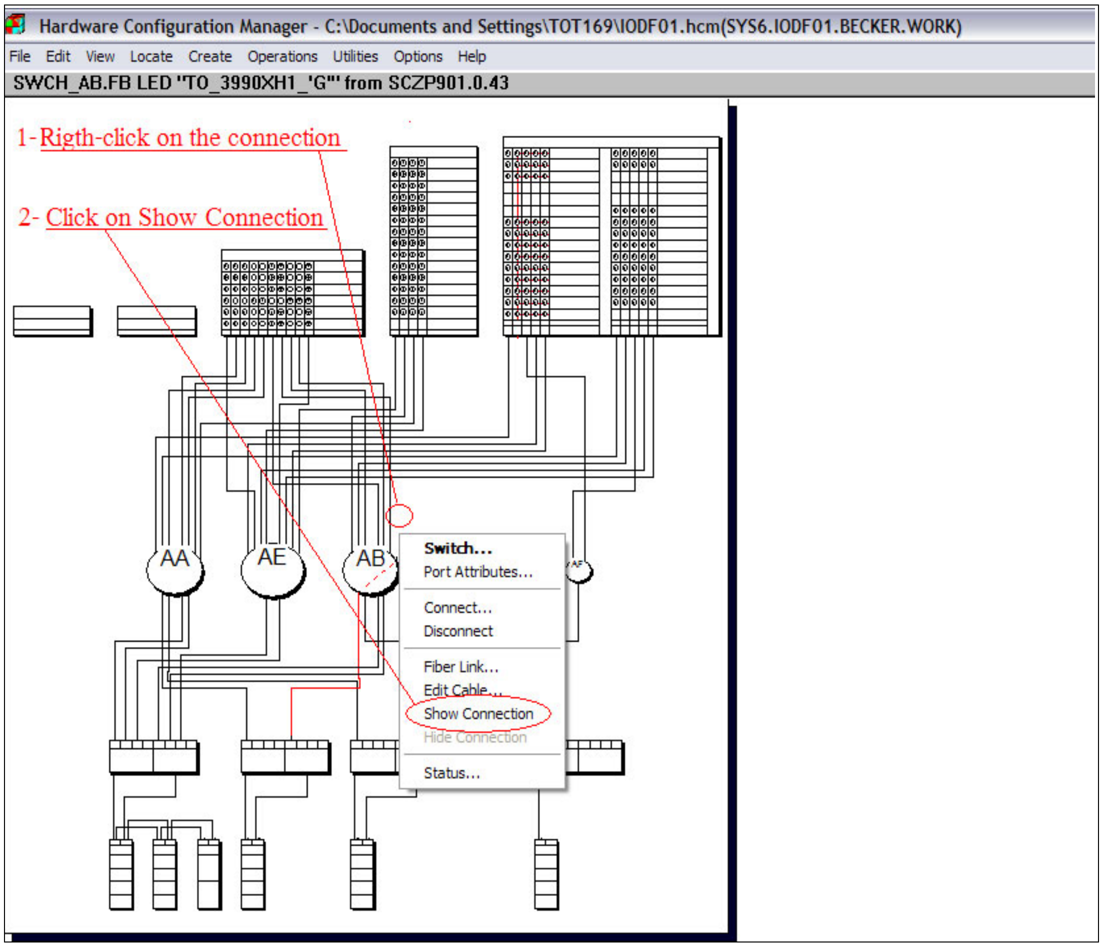
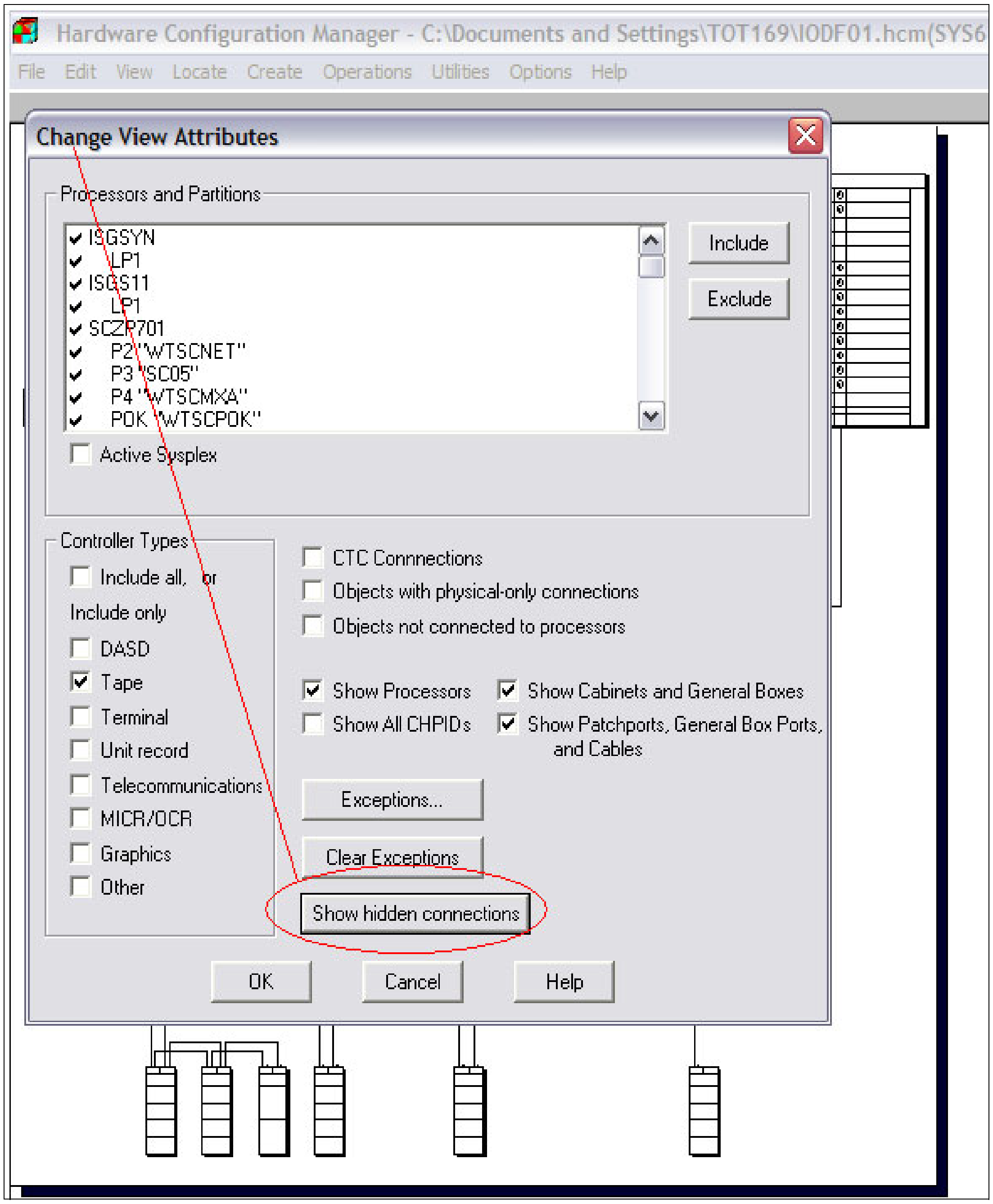

Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKER.WORK)
File Edit View Locate Create Operations Utilities Options Help
SWCH_AB.FB LED "TO_3990XH1_'G'" from SCZP901.0.43
1-Rigth-click on the connection
2- Click on Show Connection
AA
AE
AB
Switch...
Port Attributes...
Connect...
Disconnect
Fiber Link...
Edit Cable...
Show Connection
Hide Connection
Status...

Figure 24-4 Steps to show a hidden connection using the Show Connection function

Additionally there is a function to re-show all currently hidden connections in one step, as follows:

- Invocation via "Show hidden connections" in Change View Attributes dialog (via **View** → **Filter Diagram**), as shown in Figure 24-5 on page 433.


Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6

File Edit View Locate Create Operations Utilities Options Help

Change View Attributes

Processors and Partitions

- ✓ ISGSYN
- ✓ LP1
- ✓ ISGS11
- ✓ LP1
- ✓ SCZP701
- ✓ P2 "WTSCNET"
- ✓ P3 "SC05"
- ✓ P4 "WTSCMXA"
- ✓ POK "WTSCPOK"

Include | Exclude

- [ ] Active Sysplex

Controller Types

- [ ] Include all, or

Include only

- [ ] DASD
- [✓] Tape
- [ ] Terminal
- [ ] Unit record
- [ ] Telecommunications
- [ ] MICR/OCR
- [ ] Graphics
- [ ] Other

- [ ] CTC Connections
- [ ] Objects with physical-only connections
- [ ] Objects not connected to processors
- [✓] Show Processors
- [✓] Show Cabinets and General Boxes
- [ ] Show All CHPIDs
- [✓] Show Patchports, General Box Ports, and Cables

Exceptions... | Clear Exceptions | Show hidden connections

OK | Cancel | Help

Figure 24-5 Steps to show a hidden connection using the Show hidden connections function

24.2 Include PCHID in the cable label

One of HCM's purposes is to document an I/O configuration. You must be able to fine-tune HCM document output (for example, cable documents) to include physical information on cable labels.

To accomplish this, an option was introduced to select the cable label format.

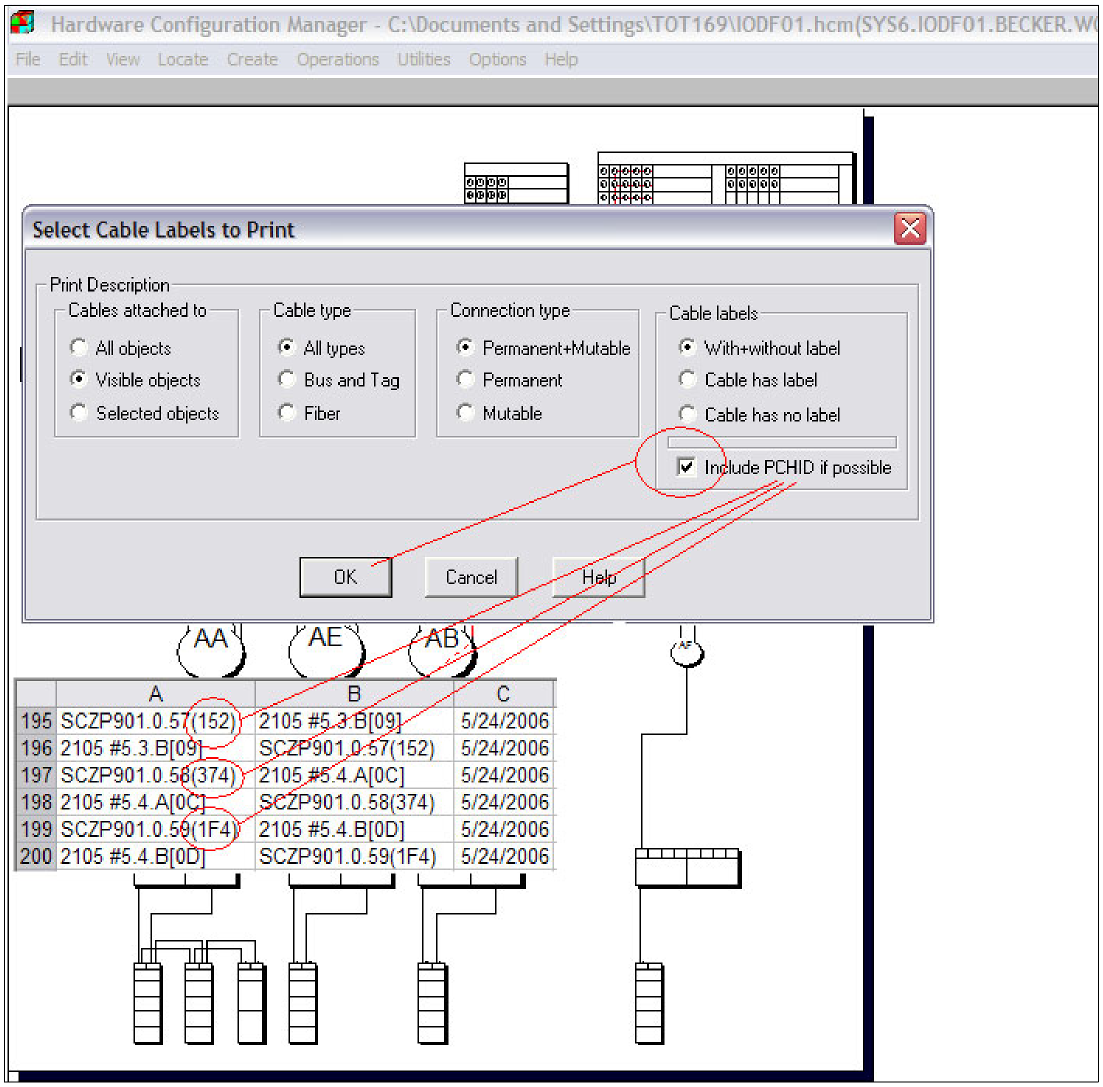
In documents, cable labels can optionally include PCHIDs, in addition to CHPIDs.

Using this option, you can:

- Select whether the label of a channel cable includes PCHID information
- Better identify the cable label on the channel
- Improve communication between different user groups

This new function is invoked via **File** → **Print Cable labels**.

The resulting output now shows an additional PCHID number (in parentheses), as shown in Figure 24-6.


Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKER.W
File Edit View Locate Create Operations Utilities Options Help
OK
Cancel
Help
AA
AE
AB

Select Cable Labels to Print

Print Description

Cables attached to	Cable type	Connection type	Cable labels
All objects	[✓ All types]	[✓ Permanent+Mutable]	[✓ With+without label]
[✓ Visible objects]	Bus and Tag	Permanent	Cable has label
Selected objects	Fiber	Mutable	Cable has no label

[✓ Include PCHID if possible]

	A	B	C
195	SCZP901.0.57(152)	2105 #5.3.B[09]	5/24/2006
196	2105 #5.3.B[09]	SCZP901.0.57(152)	5/24/2006
197	SCZP901.0.58(374)	2105 #5.4.A[0C]	5/24/2006
198	2105 #5.4.A[0C]	SCZP901.0.58(374)	5/24/2006
199	SCZP901.0.59(1F4)	2105 #5.4.B[0D]	5/24/2006
200	2105 #5.4.B[0D]	SCZP901.0.59(1F4)	5/24/2006

Figure 24-6 Additional PCHID number (in parentheses)

24.3 Default switch port names

In the previous versions of HCM, modifying each port name for ports of a FICON or ESCON director was time-consuming because each port had to be modified manually. Assigning default names to all switch ports was not possible.

z/OS V1R8 provides a means to assign the “Connected to” information that HCM provides on the “Edit Port Matrix ([IIS]) <matrix name> for Switch <switch ID>” dialog as default for switch port names.

As a result, assigning port names for connected switch ports can now be done in a time-saving manner, and you can:

- Generate switch port names based on connectivity
- Keep or overwrite existing port names
- Enter switch port names productively
- Define switch port names consistently

24.3.1 Implementation of default switch port names

You can use default switch port names as follows:

- Assign them to each switch port on the “Edit Port Matrix ([IIS]) <matrix name> for Switch <switch ID>” dialog
- Invoke the function by pressing the new Default Port Names button

In case a user-defined switch port name is defined, and the defined switch port name and the default switch port name differ, a message is issued, as shown in Figure 24-7 on page 436.

Pressing **Yes** or **No**, only the switch port is shown in the message.

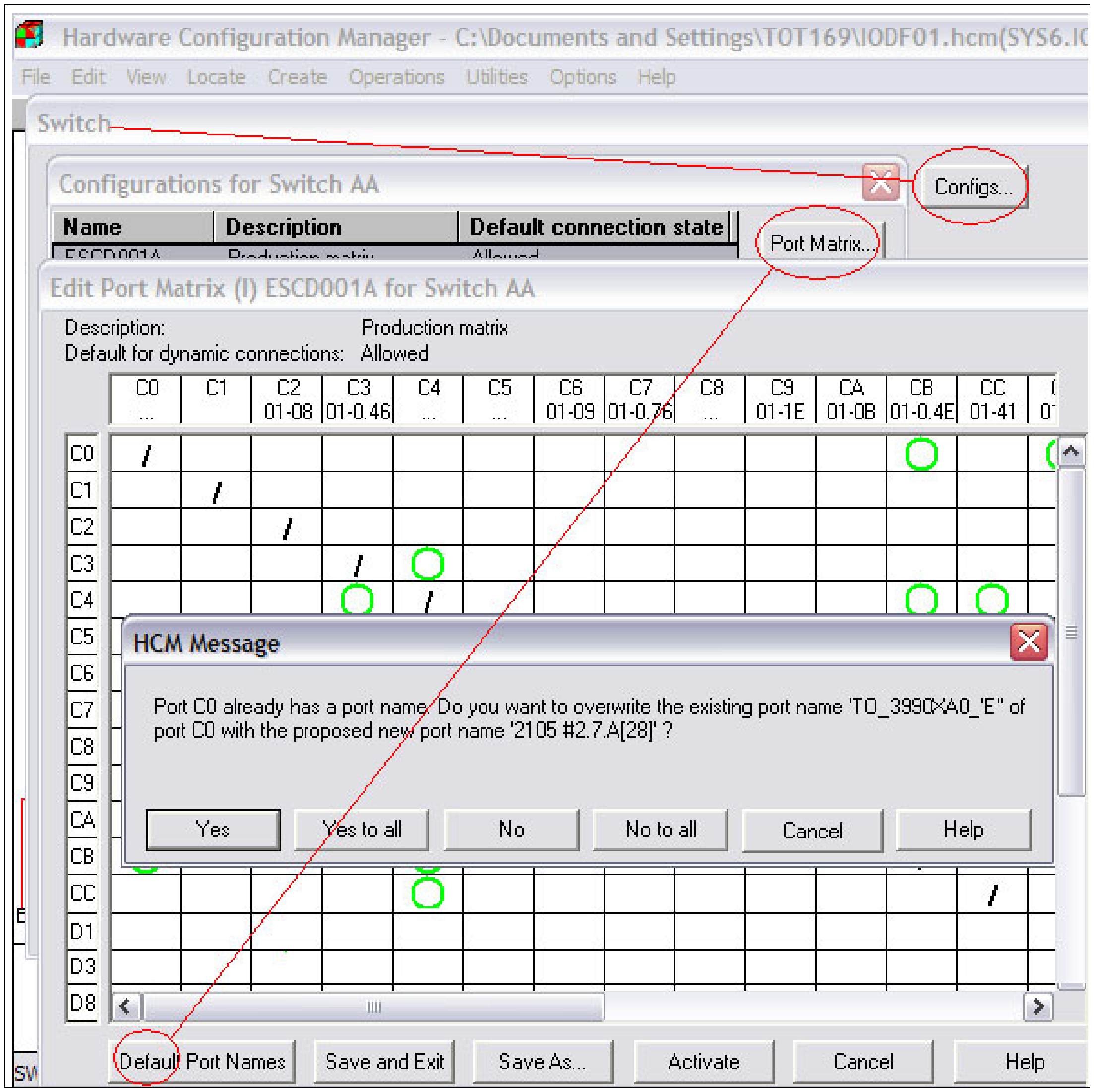
Pressing **Yes to all** or **No to all**, the switch port is shown in the message plus all following switch ports for which a conflict might be reported.

Pressing **Cancel** aborts the whole action and no switch port name gets modified, regardless of possible previous message replies.

As default, the “Connect to” information shown on the “Edit Port Matrix ([IIS]) <matrix name> for Switch <switch ID>” dialog can be assigned to the port names. This is done by pressing **Default Port names**, which can be found on the bottom left part of the dialog, as shown in Figure 24-7 on page 436.

- When you press this new button, HCM assigns the “Connect to” information to the port names. In case a switch port does not yet contain a name (indicated by a blank value), the Connect to” information is copied without further notification. In case a port name is already defined, and in case this port name differs from the “Connect to” information, HCM issues a message that a conflict exists for a given switch port. You can now select how to continue, as follows:
  - Pressing **Yes** or **No** answers the message for just this switch port. Depending on the answer, the switch port name is either overwritten with the “Connected to” information (when **Yes** was pressed), or it remains what it is (when **No** was pressed).
  - If **Yes to all** is selected, then the switch port name mentioned in the message, as well as all following switch port names for which a conflict exists, is overwritten without further notification.
  - If **No to all** is pressed, then the switch port name mentioned in the message, as well as all following switch port names for which a conflict exists, are not overwritten, without further notification.
  - If **Cancel** is pressed, the new function is aborted and no switch port name gets modified, even if that message appeared before if you selected **Yes** for another switch port name.

Figure 24-7 shows the “Edit Port Matrix ([I|S]) <matrix name> for Switch <switch ID>” dialog with the new Default Port Names button on the bottom left side of the dialog.


Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IC
File Edit View Locate Create Operations Utilities Options Help
Switch
Configurations for Switch AA
Configs...
Port Matrix...
Edit Port Matrix (I) ESCD001A for Switch AA
HCM Message
Yes
Yes to all
No
No to all
Cancel
Help
Default Port Names
Save and Exit
Save As...
Activate
Cancel
Help

Name	Description	Default connection state
ESCD001A	Production matrix	Allowed

Description: Production matrix
Default for dynamic connections: Allowed

Port C0 already has a port name. Do you want to overwrite the existing port name 'TO_3990XA0_'E'' of port C0 with the proposed new port name '2105 #2.7.A[28]' ?

Figure 24-7 Edit port matrix with an HCM message

24.4 Performance improvement for defining spanned CHPIDs

In the previous versions of HCM, defining new CHPIDs, modifying a CHPID’s access or candidate list, and adding a new partition in the access or candidate list of existing CHPIDs can be very time consuming.

z/OS V1R8 provides a different design, resulting in better performance for HCM:

- Defining a large number of (spanned) CHPIDs
- Modifying a CHPID’s access and or candidate list
- Adding a new partition in the access and or candidate list of existing CHPIDs

Now you can:

- More efficiently define new or update existing spanned channel paths in HCM

- Increase productivity due to improved response time when defining or updating channel paths

24.4.1 Migration considerations

No dialog has been modified, nor has a new function been added or an old function been removed. The only important information here is that performance is dramatically enhanced.

The performance improvement for defining spanned CHPIDs for HCM requires HCD SPE APAR OA14334. Also required is HCD APAR OA15071.

24.5 Automatic activity logging

z/OS V1R8 provides automatic activity logging to facilitate documentation about changes made in the IODF. Now, entries that describe the changes in the IODF are automatically generated in the already existing activity log file.

You can benefit from this function by:

- Using automatic activity logging
- Maintaining a change log of IODF updates
- Having activity log entries complete and consistent
- Receiving an additional diagnostic capability for service purposes with the change log

24.5.1 Implementation of automatic activity logging

HCD is now able to generate activity log entries automatically. For this purpose, use the HCD profile option CHANGE_LOG = YES and enable activity logging for the IODF used. HCD then creates activity log entries for actions such as add, change, delete, connect, or disconnect on HCD objects. These entries are proposals and presented in the activity log panel when ending an HCD session or changing the currently accessed IODF. They can be modified before leaving the activity log panel, and are stored in the activity log file.

In the following situations automatic activity logging is invoked or called:

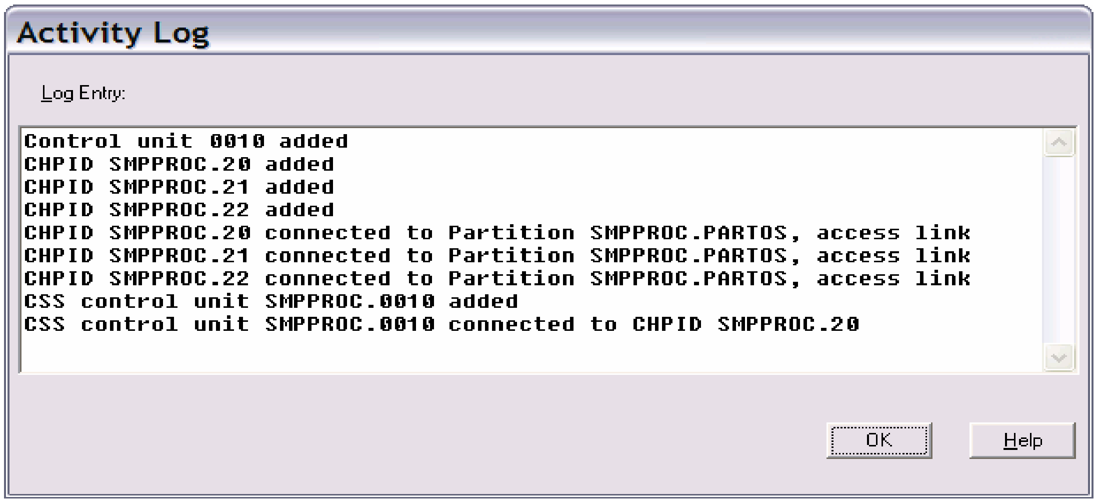
- ACTLOG = YES when defining an IODF
- New HCD profile option CHLOG = YES
- New HCD profile option CHLOG_VOL = xxx
- Exiting HCD or HCM
- Changing the IODF

New external output, such as:

- Change log file with data set name <<IODFname.CHLOG>>.
- Contents of the change log file are written to the HCD trace data set with the TRACE command and ID = CLOG and LEVEL = 8.

Contents of the change log file can be traced in a readable format with the command **`trace on,id=clog,level=8`**.

When leaving an HCM session or changing the IODF, the activity log window is displayed, as shown in Figure 24-8, which contains the automatically generated entries. These entries document the changes made to objects defined in the IODF.

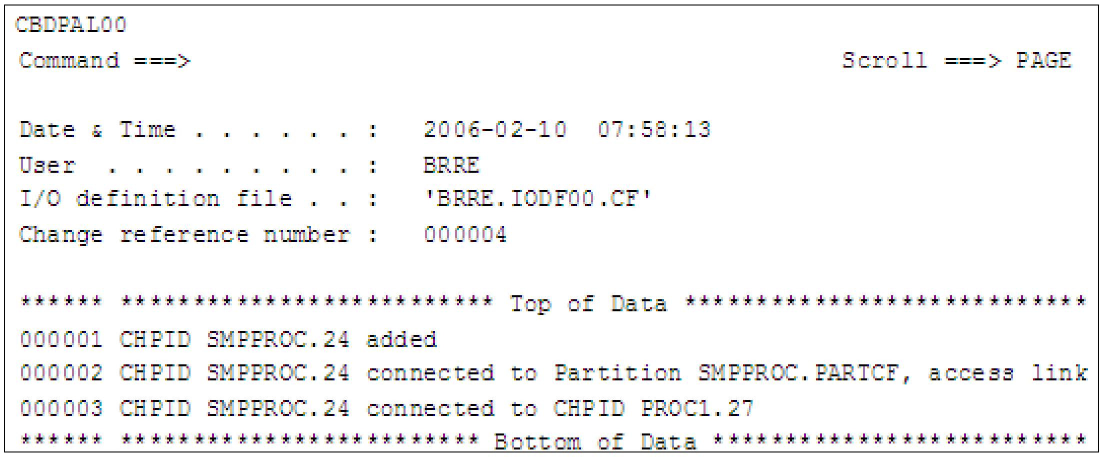


```
Control unit 0010 added
CHPID SMPPROC.20 added
CHPID SMPPROC.21 added
CHPID SMPPROC.22 added
CHPID SMPPROC.20 connected to Partition SMPPROC.PARTOS, access link
CHPID SMPPROC.21 connected to Partition SMPPROC.PARTOS, access link
CHPID SMPPROC.22 connected to Partition SMPPROC.PARTOS, access link
CSS control unit SMPPROC.0010 added
CSS control unit SMPPROC.0010 connected to CHPID SMPPROC.20
```

Figure 24-8 Activity Log panel from HCM

When leaving an HCD session or changing the IODF in access, the activity log panel is displayed, as shown in Figure 24-9, which contains the automatically generated entries in the editable part of the activity log panel. These entries document the changes made to objects defined in the IODF, such as:

- The header section remains unchanged. It consists of date and time, the ID of the user who modified the IODF, the name of the IODF, and a change reference number.
- You can accept the generated entries unchanged, modify them, and/or add additional information.



```
CBDPAL00
Command ===>                                               Scroll ===> PAGE

Date & Time . . . . . . . :   2006-02-10  07:58:13
User  . . . . . . . . . . :   BRRE
I/O definition file . . . :   'BRRE.IODF00.CF'
Change reference number   :   000004

****** ************************* Top of Data ****************************
000001 CHPID SMPPROC.24 added
000002 CHPID SMPPROC.24 connected to Partition SMPPROC.PARTCF, access link
000003 CHPID SMPPROC.24 connected to CHPID PROC1.27
****** ************************ Bottom of Data **************************
```

Figure 24-9 Activity Log panel from HCD

24.6 HCD report enhancements

In the previous versions of HCD, the Processor Summary Report should provide information about the support level of a processor (MR0525051128), and the IOCDS Report should provide POR information (MR0920053553).

z/OS V1R8 introduces a new table in the Processor Summary Report to enhance the IOCDS report with POR information.

Now HCD reports show more information, as follows:

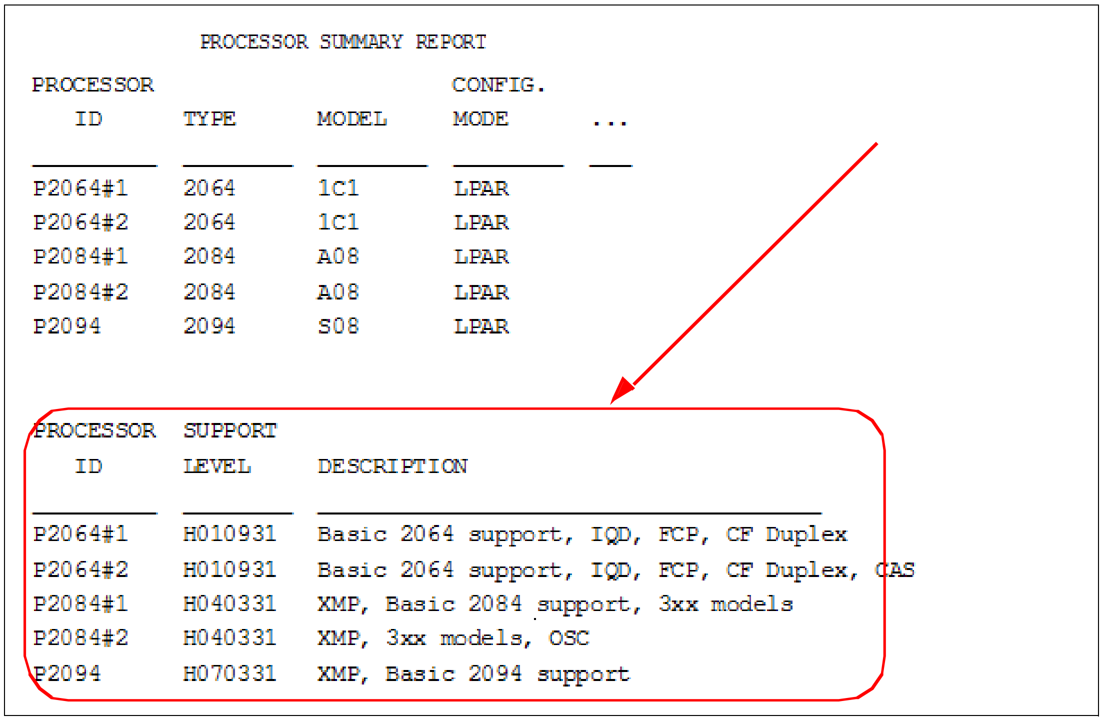
- Processor Summary Report

  Depending on the processor type/model, there may be more than one support level for a processor type. Since the support level of a processor, defined in the IODF, provides important information about H/W features, this information is now part of the Processor Summary Report.

- IOCDS Report

  The IOCDS Report shows all IOCDSs configured in the S/390 microprocessor cluster and defined in the currently accessed IODF. The report is now extended to show the same information as the HCD Panel (option 2.11, action code "work with IOCDS").

The Processor Summary Report shown in Figure 24-10 has a marked area showing the added processor support-level information for each defined processor.



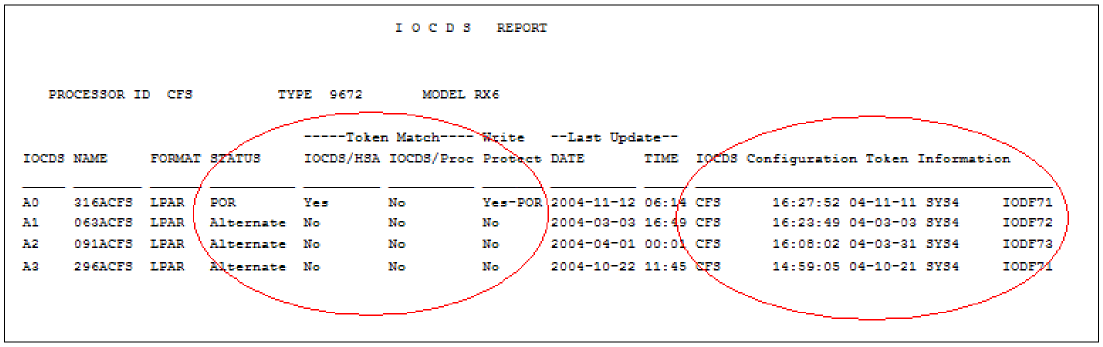
```
              PROCESSOR SUMMARY REPORT

PROCESSOR                          CONFIG.
   ID       TYPE      MODEL       MODE       ...

_________  ________  ________  ________  ___
P2064#1    2064      1C1       LPAR
P2064#2    2064      1C1       LPAR
P2084#1    2084      A08       LPAR
P2084#2    2084      A08       LPAR
P2094      2094      S08       LPAR


PROCESSOR  SUPPORT
   ID      LEVEL     DESCRIPTION

_________  ________  _____________________________________
P2064#1    H010931   Basic 2064 support, IQD, FCP, CF Duplex
P2064#2    H010931   Basic 2064 support, IQD, FCP, CF Duplex, CAS
P2084#1    H040331   XMP, Basic 2084 support, 3xx models
P2084#2    H040331   XMP, 3xx models, OSC
P2094      H070331   XMP, Basic 2094 support
```

Figure 24-10 Processor Summary Report

Figure 24-11 on page 440 shows two marked areas that show the added POR information in the IOCDS Report, such as:

- IOCDS - system-defined identifier of the IOCDS.
- NAME - user-defined name of the IOCDS.
- FORMAT - IOCDS format (BASIC or LPAR).
- STATUS - indicates the status of the IOCDS: Alternate, POR, Invalid.
- Token Match-IOCDS/HSA - indicates whether the IOCDS token matches the current HSA token.
- Token Match-IOCDS/Proc - indicates whether the IOCDS token matches the current processor token in the IODF.

- Write Protect - indicates whether the IOCDS is write-protected or not.
- Last Update DATE/TIME - date and time of the last update.
- IOCDS Configuration Token Information - configuration token information stored in the support element.



```
                                          I O C D S   REPORT


   PROCESSOR ID  CFS          TYPE  9672       MODEL RX6

                                   -----Token Match---- Write   --Last Update--
 IOCDS NAME      FORMAT STATUS      IOCDS/HSA IOCDS/Proc Protect DATE       TIME  IOCDS Configuration Token Information
 _____ ________  ______ _________   _________ __________ _______ __________ _____ _______________________________________
 A0    316ACFS   LPAR   POR         Yes       No         Yes-POR 2004-11-12 06:14 CFS      16:27:52 04-11-11 SYS4      IODF71
 A1    063ACFS   LPAR   Alternate   No        No         No      2004-03-03 16:49 CFS      16:23:49 04-03-03 SYS4      IODF72
 A2    091ACFS   LPAR   Alternate   No        No         No      2004-04-01 00:01 CFS      16:08:02 04-03-31 SYS4      IODF73
 A3    296ACFS   LPAR   Alternate   No        No         No      2004-10-22 11:45 CFS      14:59:05 04-10-21 SYS4      IODF71
```

Figure 24-11   IOCDS Report

24.7 Export/import IODF from HCM

z/OS V1R8 provides a dialog to export and import an IODF to and from the workstation, to allow you to transfer IODFs to their workstations so you can easily send them to IBM service for problem diagnosis or deployment and archival purposes.

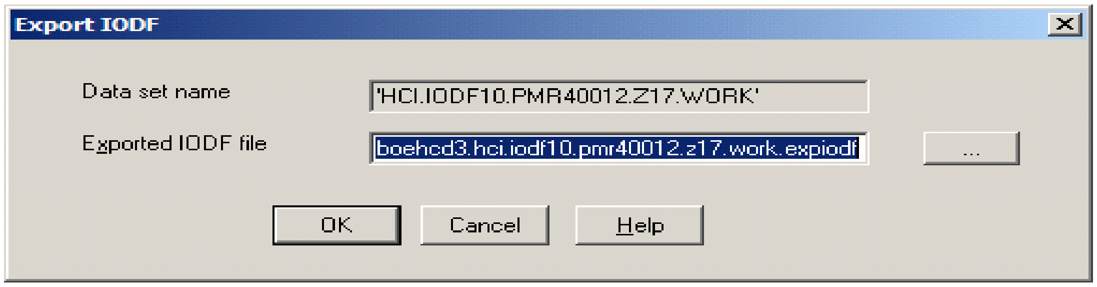
When you open problem records (PMRs) against HCD/HCM, Level-3 support often asks your IODF to diagnose the problem. This process is now simplified: HCM can export the IODF directly to the workstation, preserving the correct data set properties of the IODF. The exported IODF can be compressed using standard workstation tools such as ZIP. The resulting file can be sent to Level-3 support as an e-mail attachment.

Depending on the network infrastructure of your data center, the new export/import function may also be used to archive IODFs using workstation-based tools and possibly to distribute an IODF to several hosts.

The function Export IODF is invoked from the HCM IODF list with the Export button. The IODF data set is preselected from the IODF list, and a corresponding workstation file name is suggested (but can be changed).

The function Import IODF is invoked from the HCM IODF list with the Import button. The exported file can be imported into a new IODF or an existing IODF. If the IODF already exists, the Replace option must be checked.

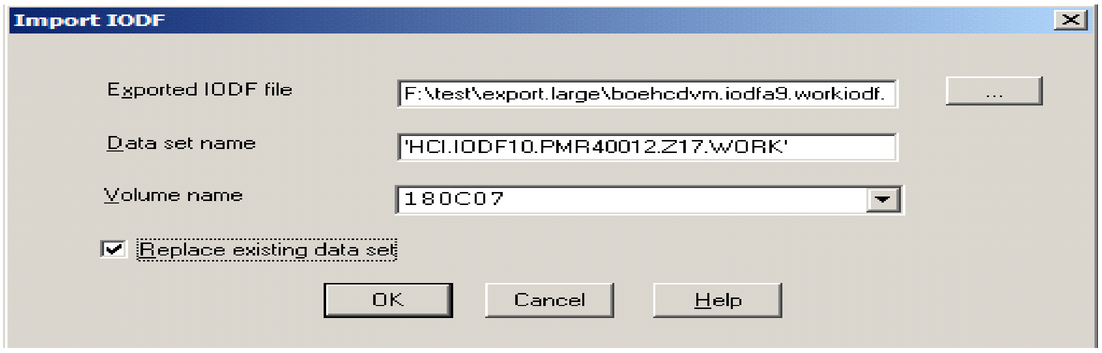
IODFs can be exported and saved to a file. Saved exported IODF files can be imported again, as shown in Figure 24-12 on page 441 and Figure 24-13 on page 441.



Data set name: 'HCI.IODF10.PMR40012.Z17.WORK'

Exported IODF file: boehcd3.hci.iodf10.pmr40012.z17.work.expiodf

Figure 24-12 Exporting an IODF



Exported IODF file: F:\test\export.large\boehcdvm.iodfa9.workiodf.

Data set name: 'HCI.IODF10.PMR40012.Z17.WORK'

Volume name: 180C07

[✓] Replace existing data set

Figure 24-13 Importing an IODF

24.8 Locate by user field

z/OS V1R8 HCM provides a new Locate One or More User Fields dialog to locate all physical objects that have the same (filtered) user field entries.

Objects of different object type do not need to be located separately one after another, but can be located within one invocation of a Locate dialog.

Customers widely exploit HCM's User Fields in which they can specify arbitrary, user-defined data. Typical information stored in the user fields is, for instance:

- Location information
- Object coordinates within a data center
- Asset numbers

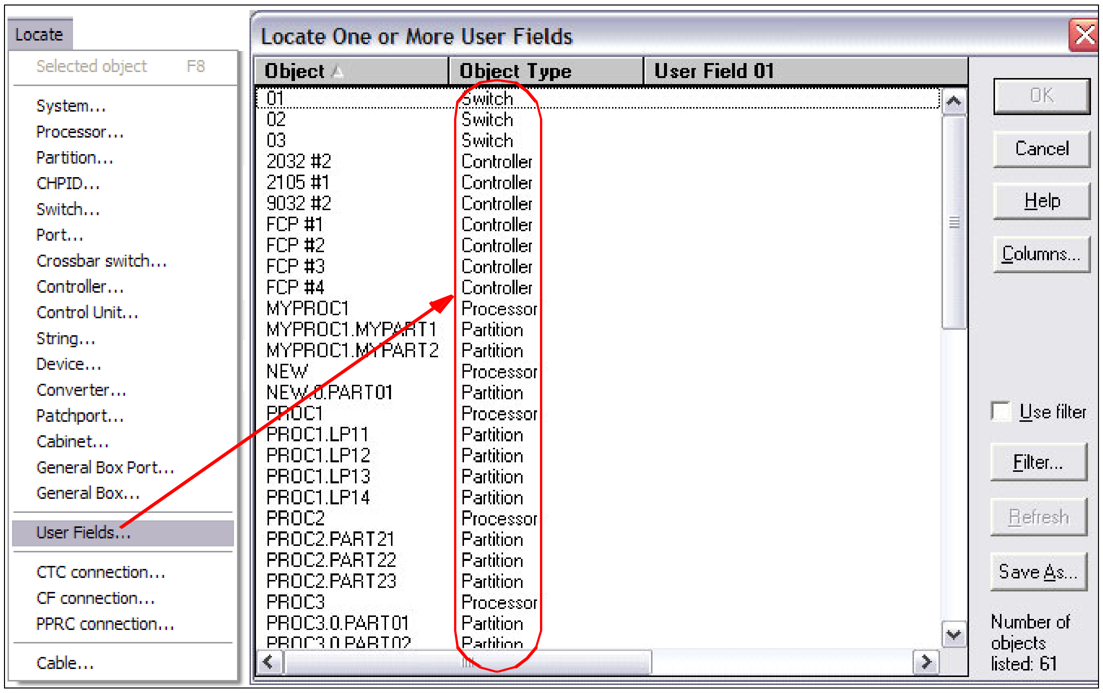
Using another HCM function named Enhanced Cropping, the diagram can be cropped to only contain the previously selected objects. This view can then be used for creating textual or graphical reports.

24.8.1 Implementation

Locate User Field is invoked via the User Fields submenu of the Locate main menu. This function is used as any other, already existing Locate function.

Objects that provide user fields are listed in the Locate One or More User Fields dialog (that is, objects shown are all processors, partitions, directors, controllers, strings, units, cabinets, general boxes, converters, and crossbar switches).

The new Locate dialog pretty much works the same as all other already existing Locate dialogs, so there is nothing new from this point of view. However, the object types that provide user fields are processors, partitions, directors, controllers, strings, units, cabinets, general boxes, converters, and crossbar switches, as shown in Figure 24-14.



Object	Object Type	User Field 01
01	Switch	
02	Switch	
03	Switch	
2032 #2	Controller	
2105 #1	Controller	
9032 #2	Controller	
FCP #1	Controller	
FCP #2	Controller	
FCP #3	Controller	
FCP #4	Controller	
MYPROC1	Processor	
MYPROC1.MYPART1	Partition	
MYPROC1.MYPART2	Partition	
NEW	Processor	
NEW.0.PART01	Partition	
PROC1	Processor	
PROC1.LP11	Partition	
PROC1.LP12	Partition	
PROC1.LP13	Partition	
PROC1.LP14	Partition	
PROC2	Processor	
PROC2.PART21	Partition	
PROC2.PART22	Partition	
PROC2.PART23	Partition	
PROC3	Processor	
PROC3.0.PART01	Partition	
PROC3.0.PART02	Partition	

Number of objects listed: 61

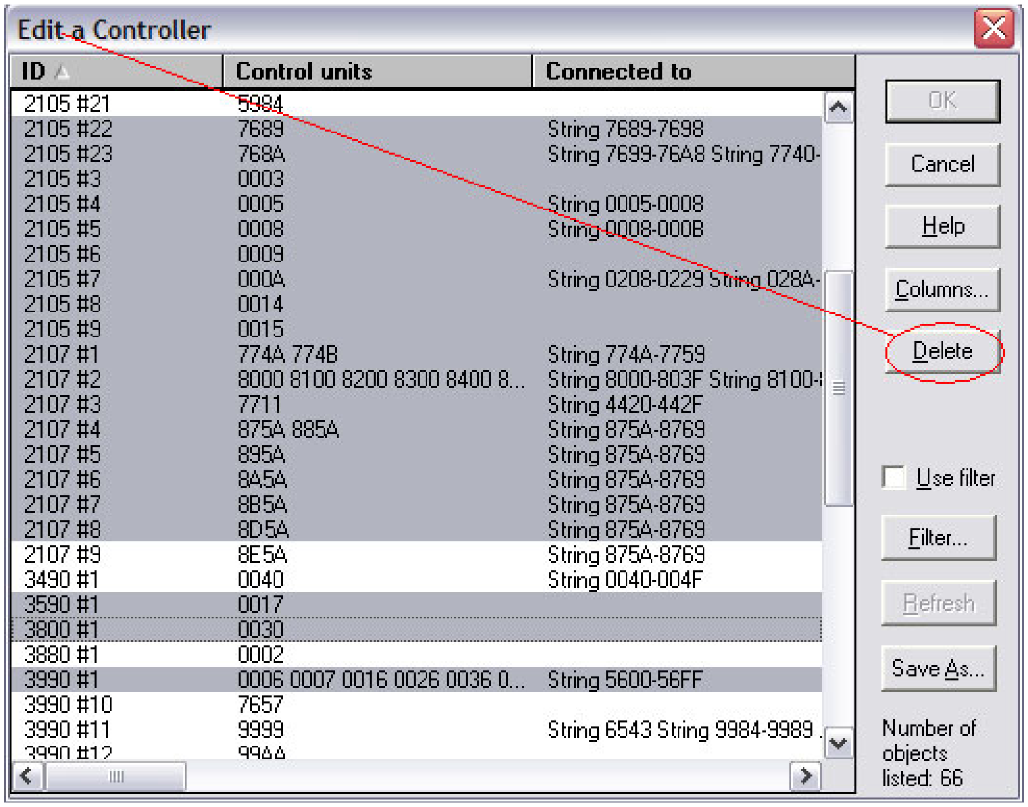
Figure 24-14 'The Locate One or More User Fields dialog

24.9 Multiple deletes

z/OS V1R8 HCM provides a Delete button on the Edit a <object type> dialogs where it is necessary, so that you can easily delete multiple objects of the same type (available for object types processor, partition, CHPID, switch, crossbar switch, controller, control unit, string, device, cabinet, and general box).

In case of data center or DASD consolidation, several “old” objects may no longer be needed and can be deleted. Also, after an HCM resynchronization step where control units and devices have been deleted via HCD before, the physical containers of the control units and devices, that is, the controllers and strings, remain.

The Edit a <object type> dialogs are multi-select enabled. Hence, several objects of the same type, for instance several controllers as shown on this snapshot, can be selected at once. Using the new Delete button, all selected objects (here: all selected controllers) get deleted, as shown in Figure 24-15.



ID	Control units	Connected to
2105 #21	5984	
2105 #22	7689	String 7689-7698
2105 #23	768A	String 7699-76A8 String 7740-
2105 #3	0003	
2105 #4	0005	String 0005-0008
2105 #5	0008	String 0008-000B
2105 #6	0009	
2105 #7	000A	String 0208-0229 String 028A-
2105 #8	0014	
2105 #9	0015	
2107 #1	774A 774B	String 774A-7759
2107 #2	8000 8100 8200 8300 8400 8...	String 8000-803F String 8100-
2107 #3	7711	String 4420-442F
2107 #4	875A 885A	String 875A-8769
2107 #5	895A	String 875A-8769
2107 #6	8A5A	String 875A-8769
2107 #7	8B5A	String 875A-8769
2107 #8	8D5A	String 875A-8769
2107 #9	8E5A	String 875A-8769
3490 #1	0040	String 0040-004F
3590 #1	0017	
3800 #1	0030	
3880 #1	0002	
3990 #1	0006 0007 0016 0026 0036 0...	String 5600-56FF
3990 #10	7657	
3990 #11	9999	String 6543 String 9984-9989 .
3990 #12	99AA	

Figure 24-15 Shows edit a controller dialog

24.10 Extract Controller PDF

z/OS V1R8 provides a means to save the physical description of a given controller into a new physical description file, and to modify the labels of a controller adapter, a controller adapter interface, a controller device adapter, and a controller device adapter interface.

z/OS V1R8 also provides a dialog for each above-mentioned labels where the labels can be modified. With this you no longer need to manually copy and change physical description files.

Saving a new physical description file from an existing controller enables you to adjust a given controller according to your needs, and to save this physical description as a reference for other controllers on which the new physical description can be applied.

With the ability to change the labels of a controller adapter, a controller adapter interface, a controller device adapter, and a controller device adapter interface, you can adjust your physical descriptions even more, according to your needs.

Now a controller PDF can be generated from a defined controller and does not need to be manually written.

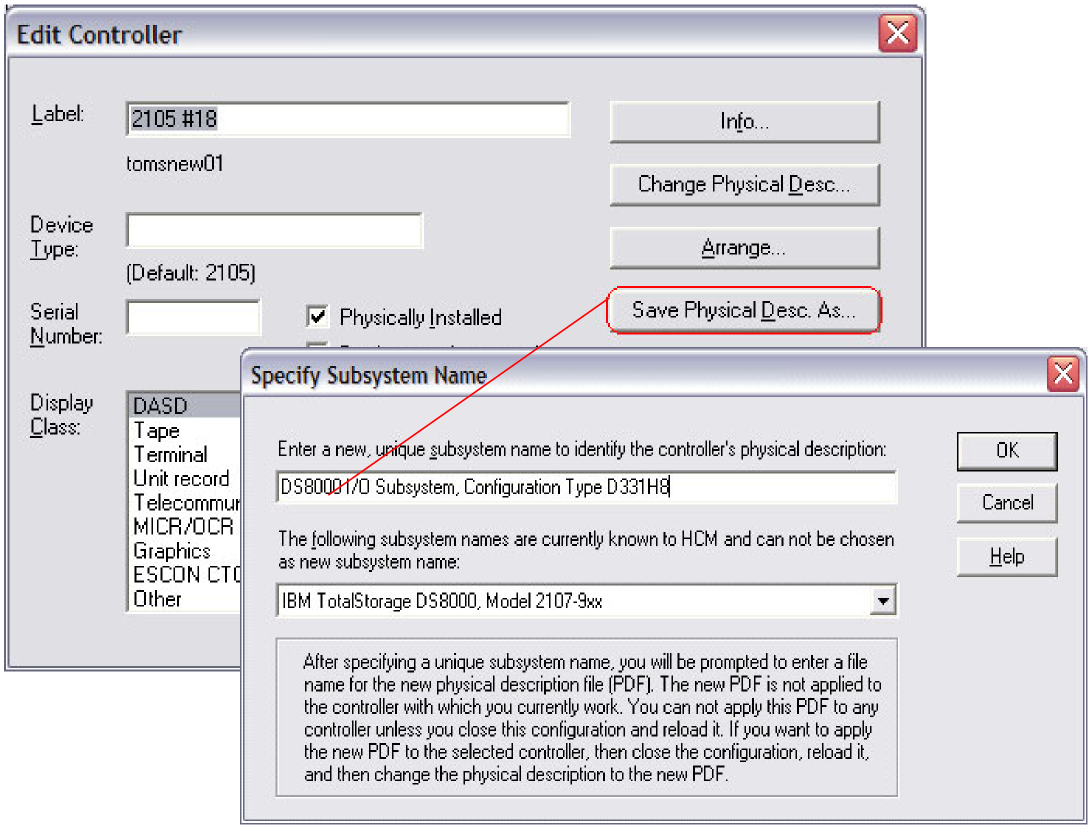
Modification of the labels of controller adapters and interfaces is made easier.

24.10.1 Implementation of the Extract Controller PDF function

This function is invoked via the Save Physical Desc. As button on the Edit Controller dialog.

A new physical description file can be created from an adjusted controller definition via this new button. Pressing it brings up the new Specify Subsystem Name dialog, as shown in Figure 24-16. Here, you must enter a new unique subsystem name which has not been used for any other physical description file that is known to HCM. The combo box contains all subsystem names that are currently known to HCM. You can verify which names have already been used in order to make sure that the new name is indeed unique and not yet used.

Physical description files for controllers which are known to HCM are stored in the CPDFA, CPDFB, and CPDFC directories of the HCM directory into which HCM was installed.



Enter a new, unique subsystem name to identify the controller's physical description:

DS8000 I/O Subsystem, Configuration Type D331H8

The following subsystem names are currently known to HCM and can not be chosen as new subsystem name:

IBM TotalStorage DS8000, Model 2107-9xx

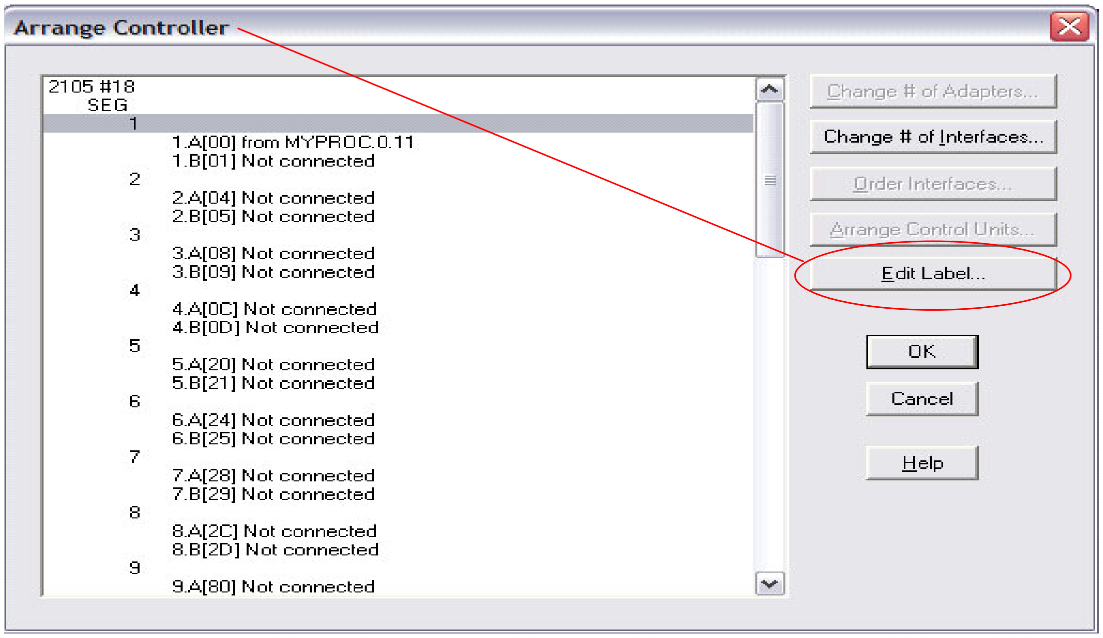
After specifying a unique subsystem name, you will be prompted to enter a file name for the new physical description file (PDF). The new PDF is not applied to the controller with which you currently work. You can not apply this PDF to any controller unless you close this configuration and reload it. If you want to apply the new PDF to the selected controller, then close the configuration, reload it, and then change the physical description to the new PDF.

Figure 24-16 Specify Subsystem Name dialog

In order to modify the label of a controller adapter, a controller adapter interface, and controller device adapter, or a controller device adapter interface, a new Edit Label dialog is provided on the Arrange Controller dialog, which itself is available via the Arrange button on the Edit Controller dialog.

Depending on whether a controller adapter, a controller adapter interface, and controller device adapter, or a controller device adapter interface is selected on the Arrange Controller dialog, a subsequent dialog is invoked where you can modify the selected label. The next two pages show the details on the four new dialogs to modify the corresponding labels.

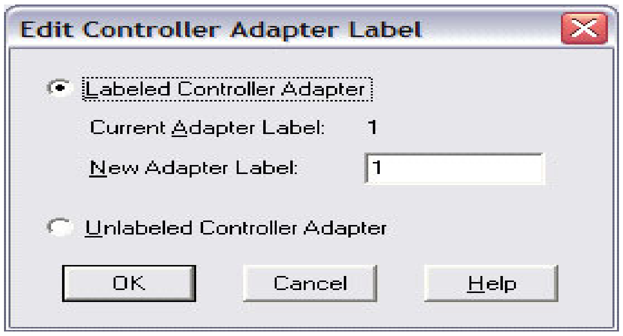
With the Extract Controller PDF function, editing the labels of a controller adapter, a controller adapter interface, a controller device adapter, and a controller device adapter interface is available via the new Edit Label button, as shown in Figure 24-17.



```
2105 #18
     SEG
          1
             1.A[00] from MYPROC.0.11
             1.B[01] Not connected
          2
             2.A[04] Not connected
             2.B[05] Not connected
          3
             3.A[08] Not connected
             3.B[09] Not connected
          4
             4.A[0C] Not connected
             4.B[0D] Not connected
          5
             5.A[20] Not connected
             5.B[21] Not connected
          6
             6.A[24] Not connected
             6.B[25] Not connected
          7
             7.A[28] Not connected
             7.B[29] Not connected
          8
             8.A[2C] Not connected
             8.B[2D] Not connected
          9
             9.A[80] Not connected
```

Figure 24-17 The Edit Label button

When editing a controller device adapter label, a new dialog is available, as shown in Figure 24-18.



[✓ Labeled Controller Adapter]

Current Adapter Label: 1

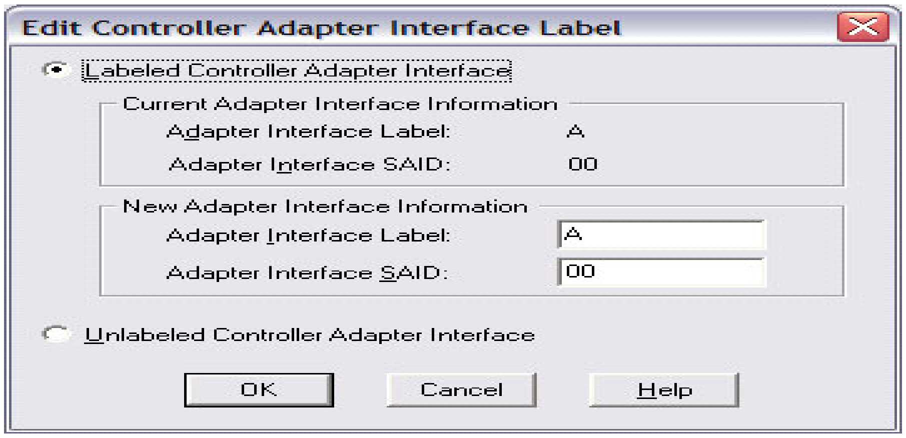
New Adapter Label: 1

Unlabeled Controller Adapter

Figure 24-18 Edit Controller Adapter Label dialog

You can either specify a label for the controller adapter, or you can choose to mark this controller adapter as unlabeled.

When editing a controller device adapter interface, a new dialog is available, shown in Figure 24-19 on page 446.


Edit Controller Adapter Interface Label

(●) Labeled Controller Adapter Interface

Current Adapter Interface Information
Adapter Interface Label: A
Adapter Interface SAID: 00

New Adapter Interface Information
Adapter Interface Label: A
Adapter Interface SAID: 00

( ) Unlabeled Controller Adapter Interface

Figure 24-19 Edit Controller Adapter Interface Label dialog

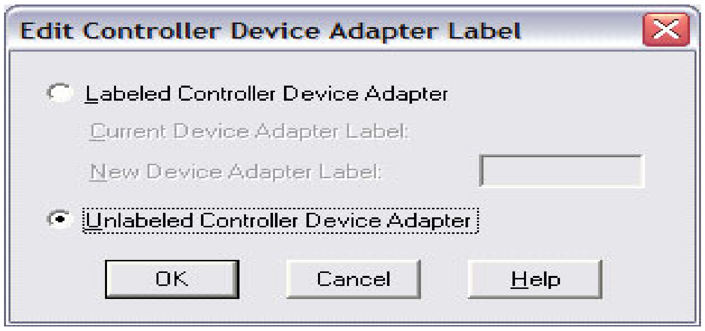
You can either specify a label and SAID for the controller adapter interface, or you can choose to mark this controller adapter interface as unlabeled. Unlabeled controller device adapter interfaces also have no SAID defined.

In case a controller adapter has been selected on the Arrange Controller dialog, the Edit Controller Adapter Label dialog is shown. You can select to either use this adapter as labeled adapter or as unlabeled adapter. In the first case, the currently assigned adapter label is shown, and a new adapter label then needs to be entered.

In case a controller adapter interface has been selected on the Arrange Controller dialog, the Edit Controller Adapter Interface Label dialog is shown. You can select to either use this adapter interface as labeled adapter interface or as unlabeled adapter interface. In the first case, the currently assigned adapter interface label as well as the adapter interface SAID is shown, and a new adapter interface label and adapter interface SAID then need to be entered.

When editing a controller device adapter label, a new dialog is available, shown in Figure 24-20.

You can either specify a label for the controller device adapter, or you can choose to mark this controller device adapter as unlabeled.


Edit Controller Device Adapter Label

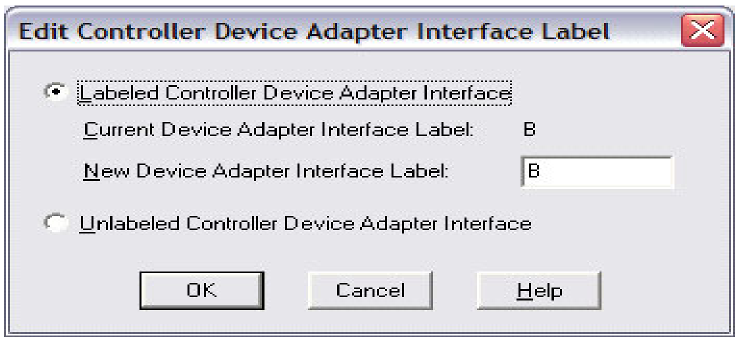
( ) Labeled Controller Device Adapter
Current Device Adapter Label:
New Device Adapter Label:

(●) Unlabeled Controller Device Adapter

Figure 24-20 Edit Controller Device Adapter Label dialog

When editing a controller device adapter interface, as shown in Figure 24-21 on page 447, a new dialog is available.

You can either specify a label for the controller device adapter interface, or you can choose to mark this controller device adapter interface as unlabeled.



(● Labeled Controller Device Adapter Interface)
Current Device Adapter Interface Label: B
New Device Adapter Interface Label: B
(○ Unlabeled Controller Device Adapter Interface)

Figure 24-21 Edit Controller Device Adapter Interface Label dialog

24.11 Filtering using wildcards

z/OS V1R8 allows you to use the wildcard character "*" (asterisk) at any place in a filter criteria where "*" stands for an arbitrary number of characters.

Filtering for data gets easier with wildcards, for instance for locating objects that match a particular entry in a user field.

This enhancement is useful for the existing Edit a <object type> and Locate One or More <object type> dialogs, but it is also especially of interest for the new Locate User Fields function of HCM.

The current rules of the wildcard character allow its usage only at the very beginning or the very end of the filter criteria string. Now, the wildcard character can be used anywhere in the filter criteria where "*" stands for an arbitrary number of characters, which also includes 0 characters. Note that "*" could also be a valid character in a user-defined data field.

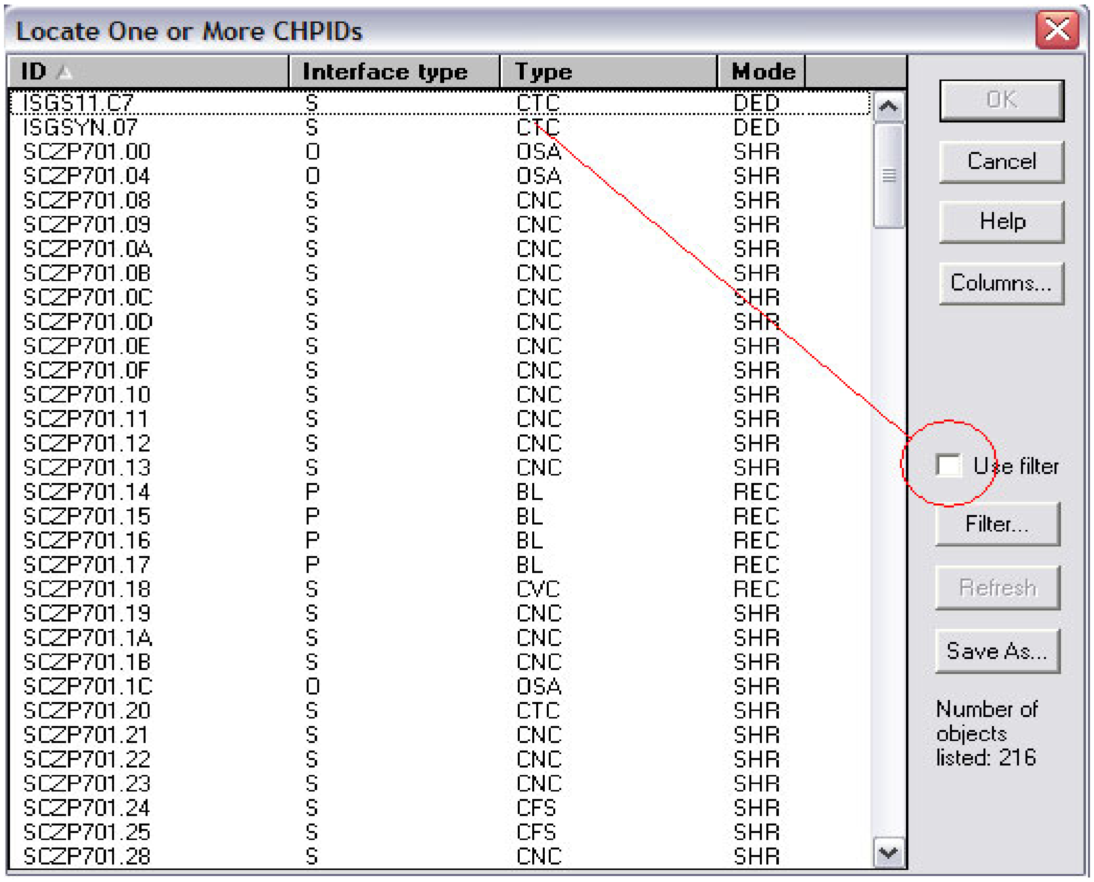
24.11.1 Implementation of the wildcard

The enhanced filtering function now supports a more appropriate usage of the wildcard character.

To set the filter, press the Filter button, which invokes a dialog where you can set up to 8 filter criteria.

Filtering has been available in HCM for a long time, and the way of switching filtering on and off has not changed. You just select the checkbox Use Filter in the Edit a <object type> and Locate One or More <object type> dialogs, and the selected filter is then activated. In order to set or to modify the current filter criteria, you need to press the Filter button, which invokes the <object type> Filter dialog.

To activate filtering, select the "Use filter" checkbox, as shown in Figure 24-22 on page 448.



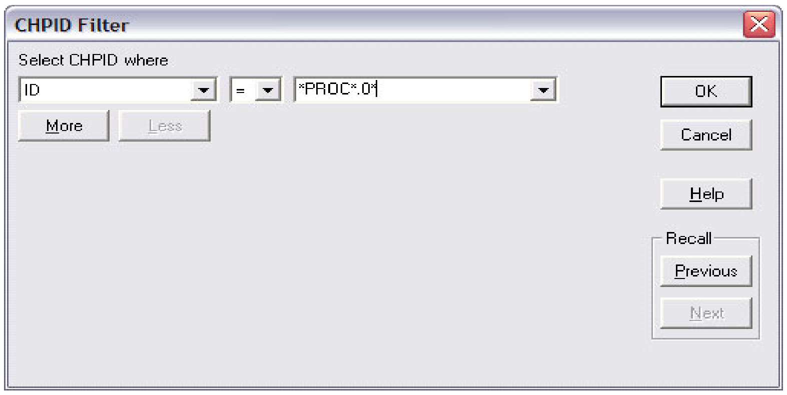
ID	Interface type	Type	Mode
ISGS11.C7	S	CTC	DED
ISGSYN.07	S	CTC	DED
SCZP701.00	O	OSA	SHR
SCZP701.04	O	OSA	SHR
SCZP701.08	S	CNC	SHR
SCZP701.09	S	CNC	SHR
SCZP701.0A	S	CNC	SHR
SCZP701.0B	S	CNC	SHR
SCZP701.0C	S	CNC	SHR
SCZP701.0D	S	CNC	SHR
SCZP701.0E	S	CNC	SHR
SCZP701.0F	S	CNC	SHR
SCZP701.10	S	CNC	SHR
SCZP701.11	S	CNC	SHR
SCZP701.12	S	CNC	SHR
SCZP701.13	S	CNC	SHR
SCZP701.14	P	BL	REC
SCZP701.15	P	BL	REC
SCZP701.16	P	BL	REC
SCZP701.17	P	BL	REC
SCZP701.18	S	CVC	REC
SCZP701.19	S	CNC	SHR
SCZP701.1A	S	CNC	SHR
SCZP701.1B	S	CNC	SHR
SCZP701.1C	O	OSA	SHR
SCZP701.20	S	CTC	SHR
SCZP701.21	S	CNC	SHR
SCZP701.22	S	CNC	SHR
SCZP701.23	S	CNC	SHR
SCZP701.24	S	CFS	SHR
SCZP701.25	S	CFS	SHR
SCZP701.28	S	CNC	SHR

Number of objects listed: 216

Figure 24-22 Locate One or More CHPIDs dialog

Figure 24-23 shows filter criteria that selects all CHPIDs where PROC is part of the processor name, and where the channel path ID starts with 0, as follows:

- PROC2064.0D
- PROC2094.1.04
- MYPROCA.03



Select CHPID where

ID = *PROC*.0*

Figure 24-23 The CHPID filter dialog

24.11.2 Interactions and dependencies

The items Automatic Activity Logging and Export/Import IODF' require HCD SPE APAR OA14334.

24.11.3 Installation

With the z/OS V1R8 HCM, install HCD SPE APAR OA14334.

24.12 Enhancements with importing/exporting data

24.12.1 New column SSID in the CU table

The Export Data function now exports the subsystem ID of a control unit (needed to define PPRC links) in the new optional column SSID of table CU.txt. This column can also be imported.

24.12.2 Import Data tolerates unknown columns

The Import Data function was modified to enhance backward compatibility. When finding an unknown column header in an exported table, earlier versions of this function issued an error message and could not perform the data import.

The enhanced function now issues an informational message stating that the unknown column will be ignored, and the data import continues. This allows older versions of HCM to import tables that were exported by newer versions of HCM (which may write additional data columns not known to the older version).

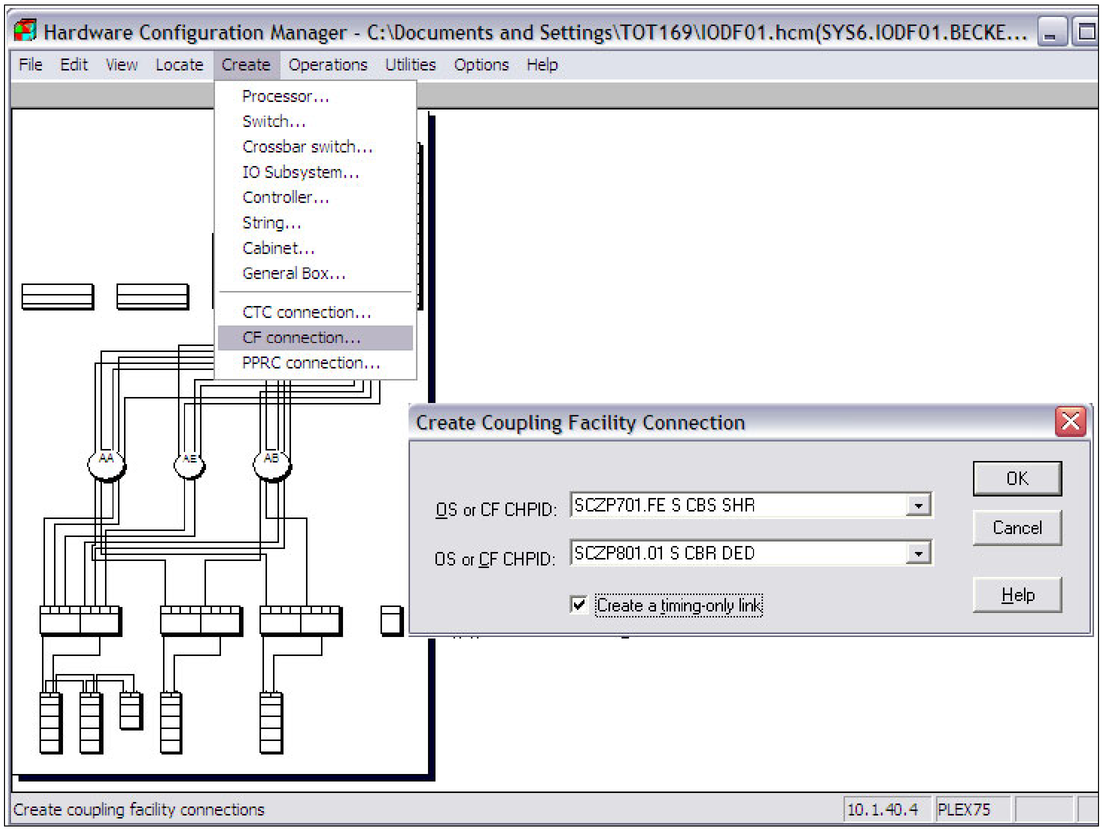
24.12.3 Exporting customized user field names

If you specified customized user field names for an object, for example, Location or Power consumption for a processor, and if you selected to have a header line in the exported files, then these customized names are used in this header line instead of their generic names (User field 1... User field 20), and are also used in the HCM Compare Reports.

24.13 Server Time Protocol (STP) link support

HCM supports Server Time Protocol (STP) links between two zSeries (z890, z990, or higher) processors.

In the Create Coupling Facility dialog, you can select two CHPIDs capable for Coupling Facilities, and then click an option to create an STP link (timing-only link), as shown in Figure 24-24 on page 450.


Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKE...
File Edit View Locate Create Operations Utilities Options Help
Processor...
Switch...
Crossbar switch...
IO Subsystem...
Controller...
String...
Cabinet...
General Box...
CTC connection...
CF connection...
PPRC connection...
Create Coupling Facility Connection
OK
Cancel
Help
Create coupling facility connections
10.1.40.4
PLEX75

OS or CF CHPID: SCZP701.FE S CBS SHR

OS or CF CHPID: SCZP801.01 S CBR DED

☑ Create a timing-only link

Figure 24-24 Creating a timing-only link function

24.14 New utilities to support complex configurations

z/OS V1R8 HCM provides new utilities (wizards) for various complex, cumbersome and therefore error-prone configuration definition tasks to let you perform these tasks quickly and efficiently.

You no longer need to perform the copy tasks by using HCD. A resynchronization step is no longer required, and physical updates are done at the same time that logical definitions are established, which saves further overall time for finishing the copy task.

24.14.1 Copy processor

HCM provides copy and repeat functions that HCD provides as well. Objects that can be copied are processors, channel subsystems, partitions, operating systems, eligible device tables (EDTs), and esoteric device groups.

Using this function, you can have the following copy scenarios:

Source	Target
SMP processor	SMP processor
XMP processor	XMP processor
SMP processor	Channel Subsystem
Channel Subsystem	SMP Processor
Channel Subsystem	Channel Subsystem
Partition	Partition

Note that the corresponding target object must not yet exist in the configuration. That means that, for instance, the target SMP processor, the target XMP processor, or the target partition must not yet exist in the IODF.

If you copy a CSS or an SMP processor into a CSS, then this CSS must not yet exist, but the XMP processor that should hold the target CSS *must* exist. A similar statement is true for partitions: the target partition must not yet exist, but the processor (and the channel subsystem in case of an XMP processor) must already exist.

Implementation

The Wizard Page Flow Overview shown in Figure 24-25 on page 452 shows which wizard page is invoked for which wizard and for which scenario.

This is an overview of which wizard page is used for which Copy wizard and in which scenario. Note that some wizard pages are used for several wizards, and this shows the wizard flow for all wizards in all possible scenarios.

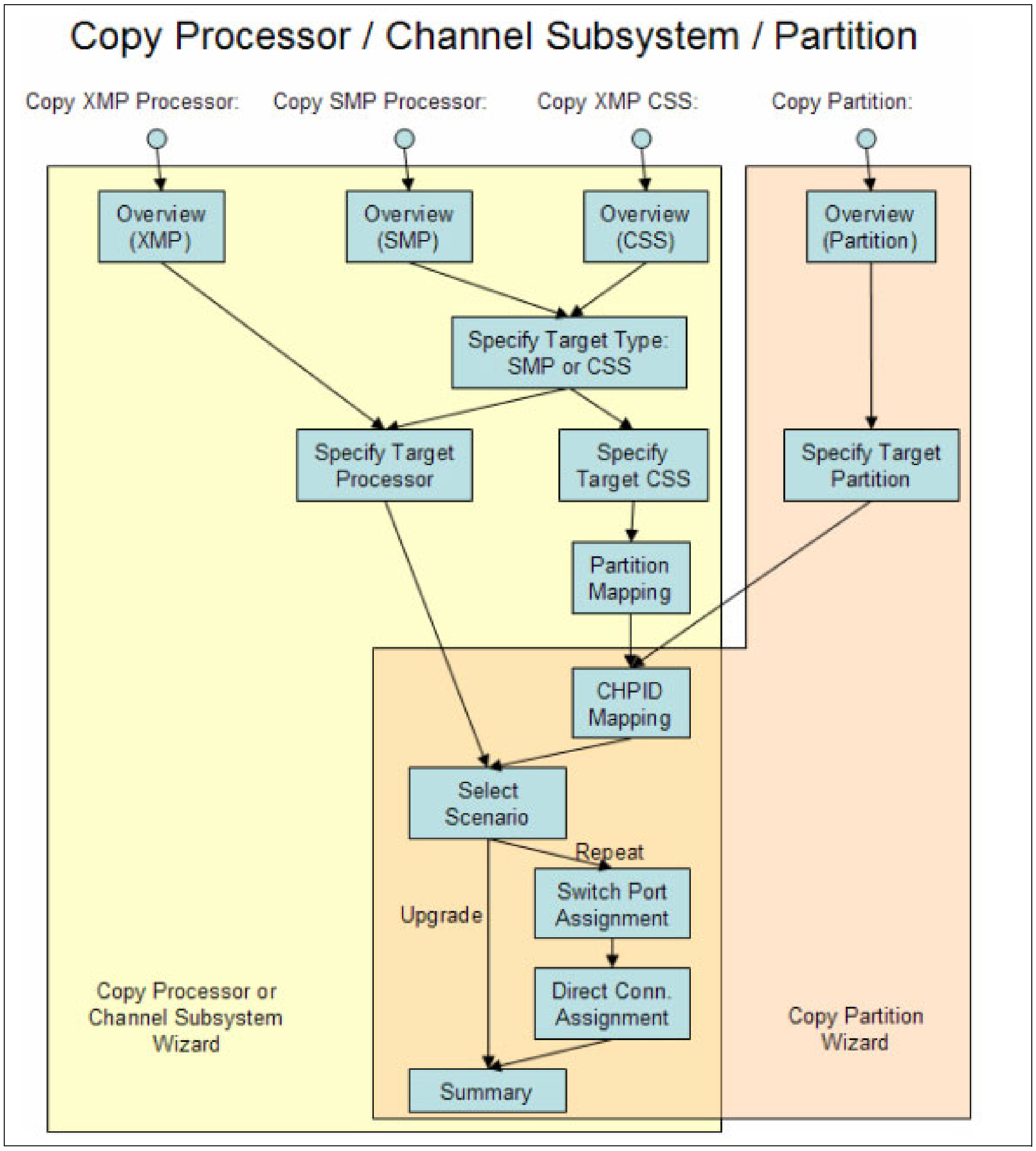
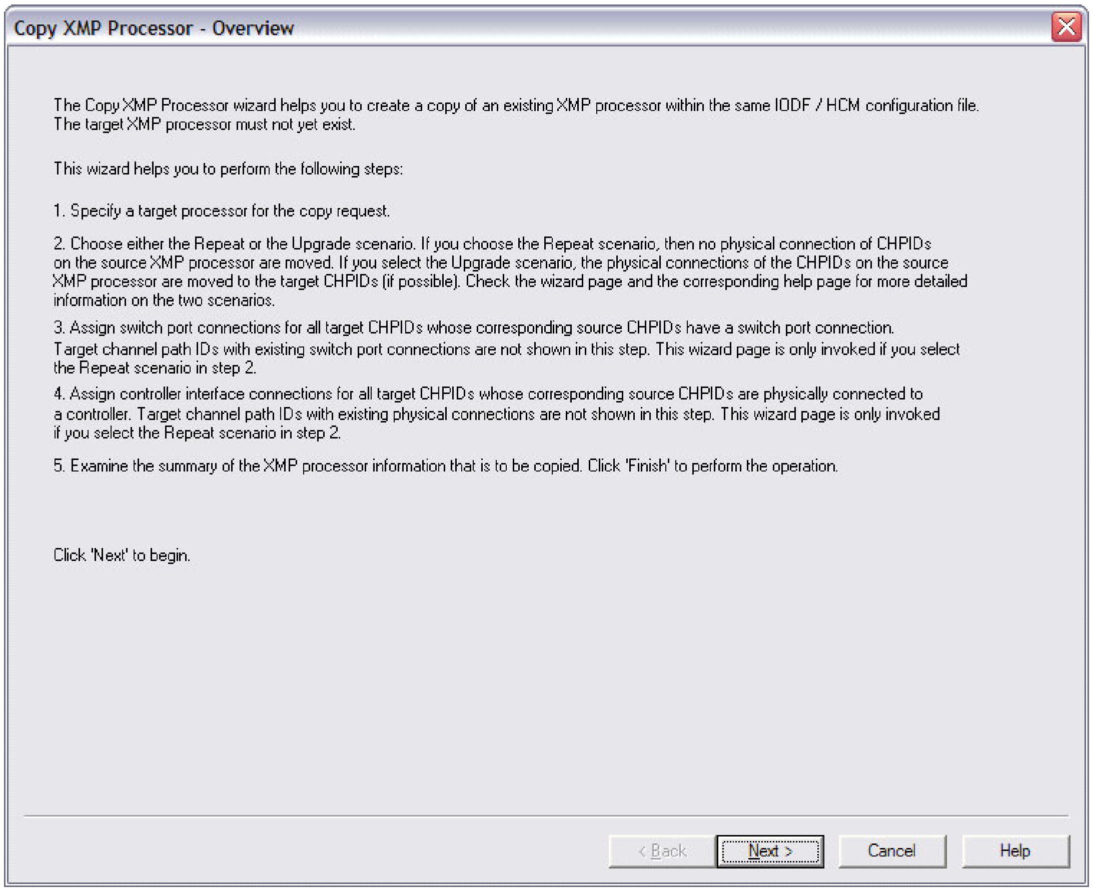


Figure 24-25 Wizard page flow overview

Copy XMP processor

The overview page describes all steps you need to complete in order to copy an XMP processor, as shown in Figure 24-26 on page 453.


Copy XMP Processor - Overview

The Copy XMP Processor wizard helps you to create a copy of an existing XMP processor within the same IODF / HCM configuration file. The target XMP processor must not yet exist.

This wizard helps you to perform the following steps:

1. Specify a target processor for the copy request.

2. Choose either the Repeat or the Upgrade scenario. If you choose the Repeat scenario, then no physical connection of CHPIDs on the source XMP processor are moved. If you select the Upgrade scenario, the physical connections of the CHPIDs on the source XMP processor are moved to the target CHPIDs (if possible). Check the wizard page and the corresponding help page for more detailed information on the two scenarios.

3. Assign switch port connections for all target CHPIDs whose corresponding source CHPIDs have a switch port connection. Target channel path IDs with existing switch port connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 2.

4. Assign controller interface connections for all target CHPIDs whose corresponding source CHPIDs are physically connected to a controller. Target channel path IDs with existing physical connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 2.

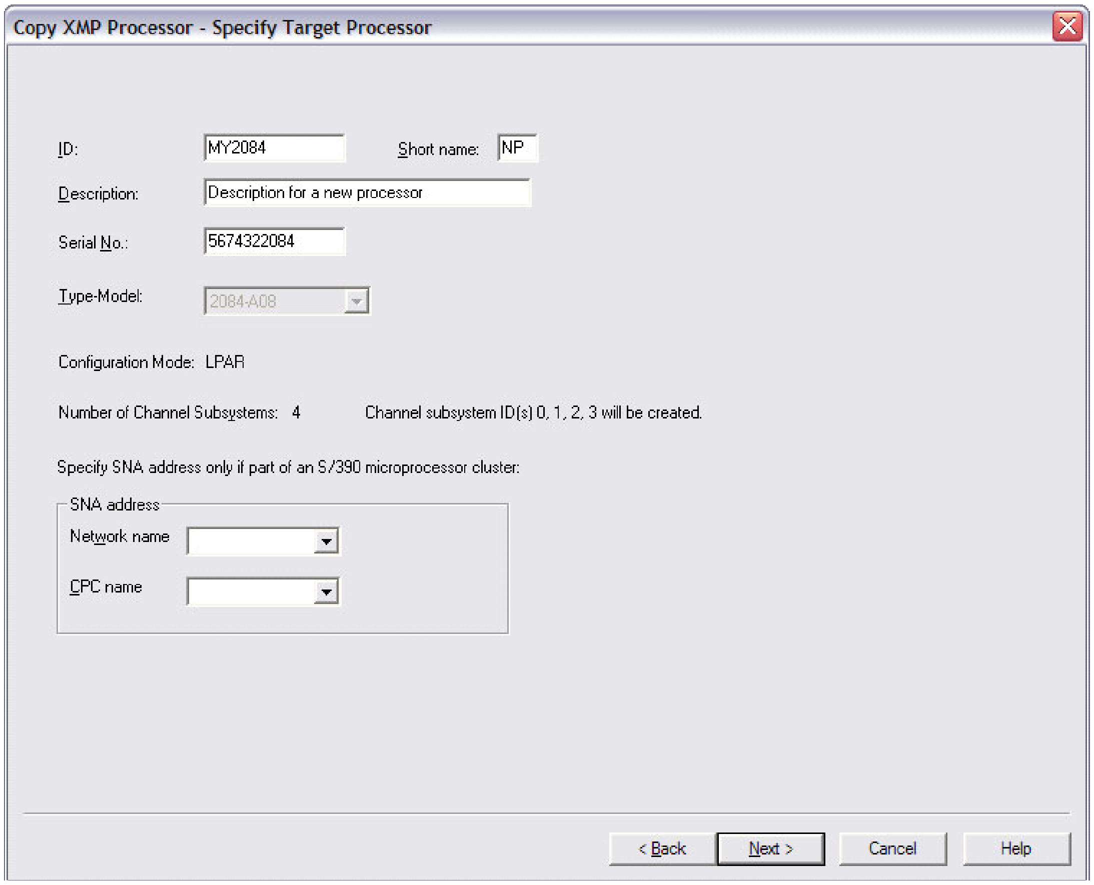
5. Examine the summary of the XMP processor information that is to be copied. Click 'Finish' to perform the operation.

Click 'Next' to begin.

Figure 24-26 Copying an XMP processor overview

Enter the data for the new XMP processor in the Specify Target Processor page, shown in Figure 24-27 on page 454. All channel subsystems of the source XMP processor will be copied. The type and model of the new processor equals that of the source processor.

Note: You cannot modify the number of channel subsystems because all channel subsystems which are defined to the source processor will be copied to the target processor. The IDs of the channel subsystems that are defined to the source XMP processor are mentioned in the center of the wizard page.


Copy XMP Processor - Specify Target Processor

ID: MY2084 Short name: NP

Description: Description for a new processor

Serial No.: 5674322084

Type-Model: 2084-A08

Configuration Mode: LPAR

Number of Channel Subsystems: 4 Channel subsystem ID(s) 0, 1, 2, 3 will be created.

Specify SNA address only if part of an S/390 microprocessor cluster:

SNA address

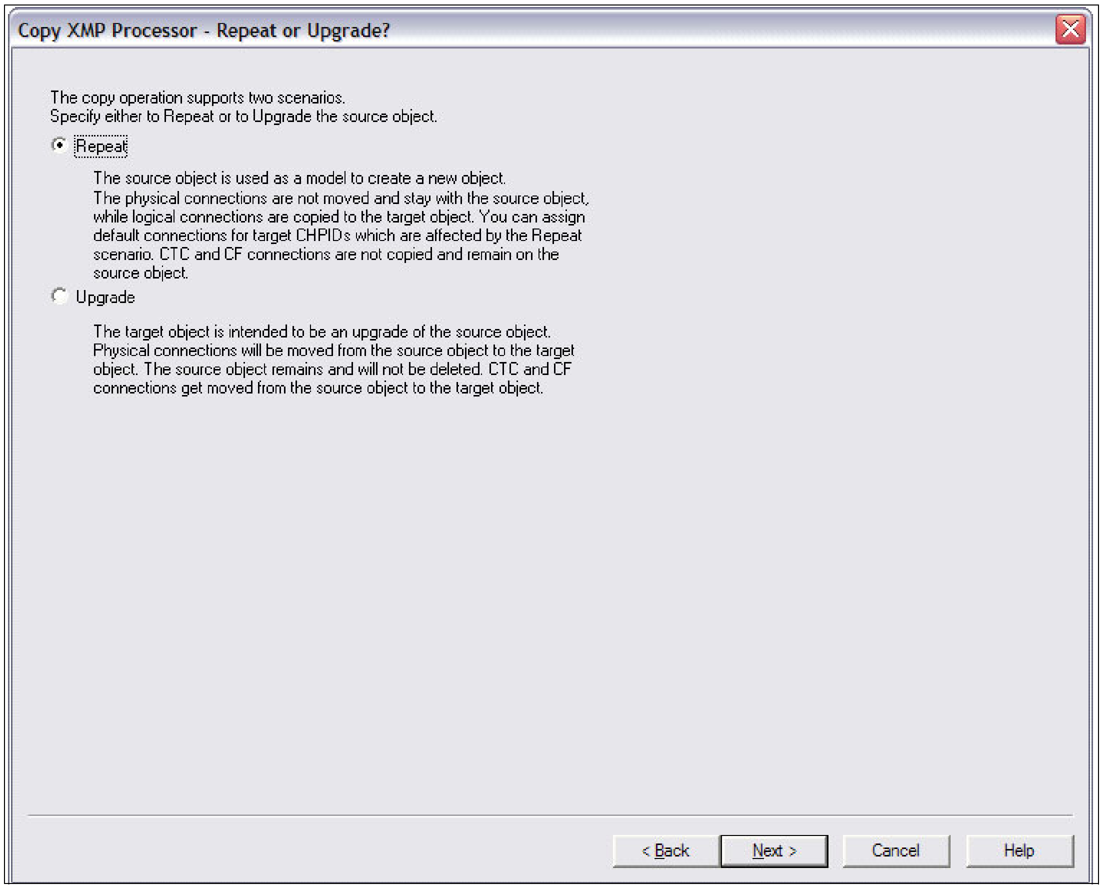
Network name

CPC name

Figure 24-27 Copying an XMP processor page

As shown in Figure 24-28 on page 455, you need to decide for either the Repeat or the Upgrade scenario, as follows:

- Upgrade - Move physical objects of CHPID connections to the target CHPIDs. CTC and CF connections are copied.
- Repeat - Physical objects of CHPID connections remain at the source processor. CTC and CF connections are not copied with the exception of internal CF connections (ICP CHPID connections).



The copy operation supports two scenarios.
Specify either to Repeat or to Upgrade the source object.

[✓ Repeat]

The source object is used as a model to create a new object. The physical connections are not moved and stay with the source object, while logical connections are copied to the target object. You can assign default connections for target CHPIDs which are affected by the Repeat scenario. CTC and CF connections are not copied and remain on the source object.

Upgrade

The target object is intended to be an upgrade of the source object. Physical connections will be moved from the source object to the target object. The source object remains and will not be deleted. CTC and CF connections get moved from the source object to the target object.

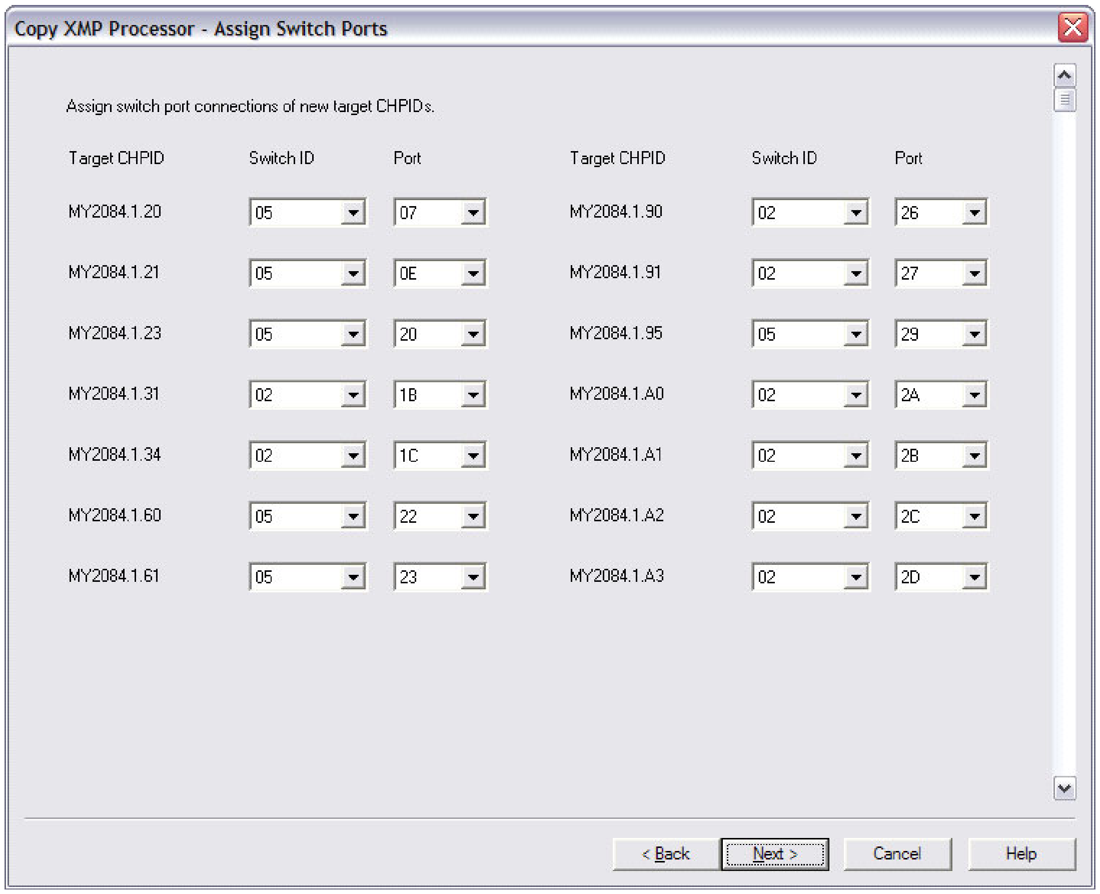
Figure 24-28 Copying an XMP processor - Repeat or Upgrade page

Figure 24-29 on page 456, shows the Assign Switch Ports page that applies only for the Repeat scenario, and assign switch ports for target CHPIDs, which are copies of source CHPIDs, which themselves are connected to switch ports.

Select a switch and a port of this switch to connect it to the already determined target CHPIDs.

This makes it possible to keep a physical link between the CHPID and connected control units and devices in case the connection runs over a switch.

Note that you cannot connect the target CHPID of the copy operation to the same switch port as the source CHPID. You can, however, connect to a different switch port of the same switch (if available) in order to keep the same logical connections between the CUs and devices to the CHPID or processor.



Copy XMP Processor - Assign Switch Ports

Assign switch port connections of new target CHPIDs.

Target CHPID	Switch ID	Port	Target CHPID	Switch ID	Port
MY2084.1.20	05	07	MY2084.1.90	02	26
MY2084.1.21	05	0E	MY2084.1.91	02	27
MY2084.1.23	05	20	MY2084.1.95	05	29
MY2084.1.31	02	1B	MY2084.1.A0	02	2A
MY2084.1.34	02	1C	MY2084.1.A1	02	2B
MY2084.1.60	05	22	MY2084.1.A2	02	2C
MY2084.1.61	05	23	MY2084.1.A3	02	2D

Figure 24-29 Assigning a switch ports page

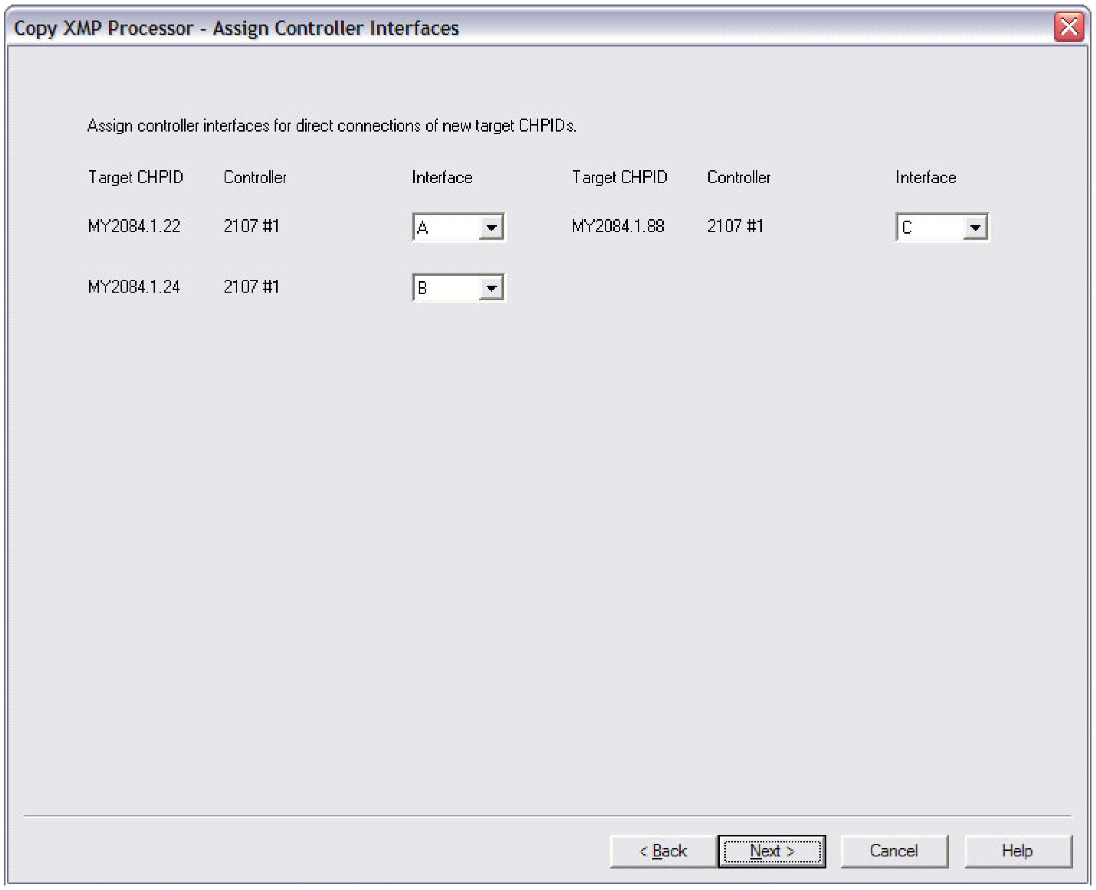
The Assign Controller Interfaces page, as shown in Figure 24-30 on page 457, assigns controller interfaces for target CHPIDs, which are copies of source CHPIDs, which themselves are connected to controller interfaces.

The copied target CHPID must be connected to the same controller in order to keep the logical and physical connections in sync.

You do not want to have an invalid configuration without a logical connection, so you have to establish the required physical connection between the target CHPID of the copy operation and the controller to which the source CHPID is connected.

Note that HCM will connect to unlabeled interfaces if no labeled controller interface is left. HCM allows to manually change the controller device connections and also the order of the devices.

Check the HCM user‘s guide for more details about the existing functionality for moving controller interfaces within the controller.



Assign controller interfaces for direct connections of new target CHPIDs.

Target CHPID	Controller	Interface	Target CHPID	Controller	Interface
MY2084.1.22	2107 #1	A	MY2084.1.88	2107 #1	C
MY2084.1.24	2107 #1	B			

Figure 24-30 Assigning a controller interfaces page

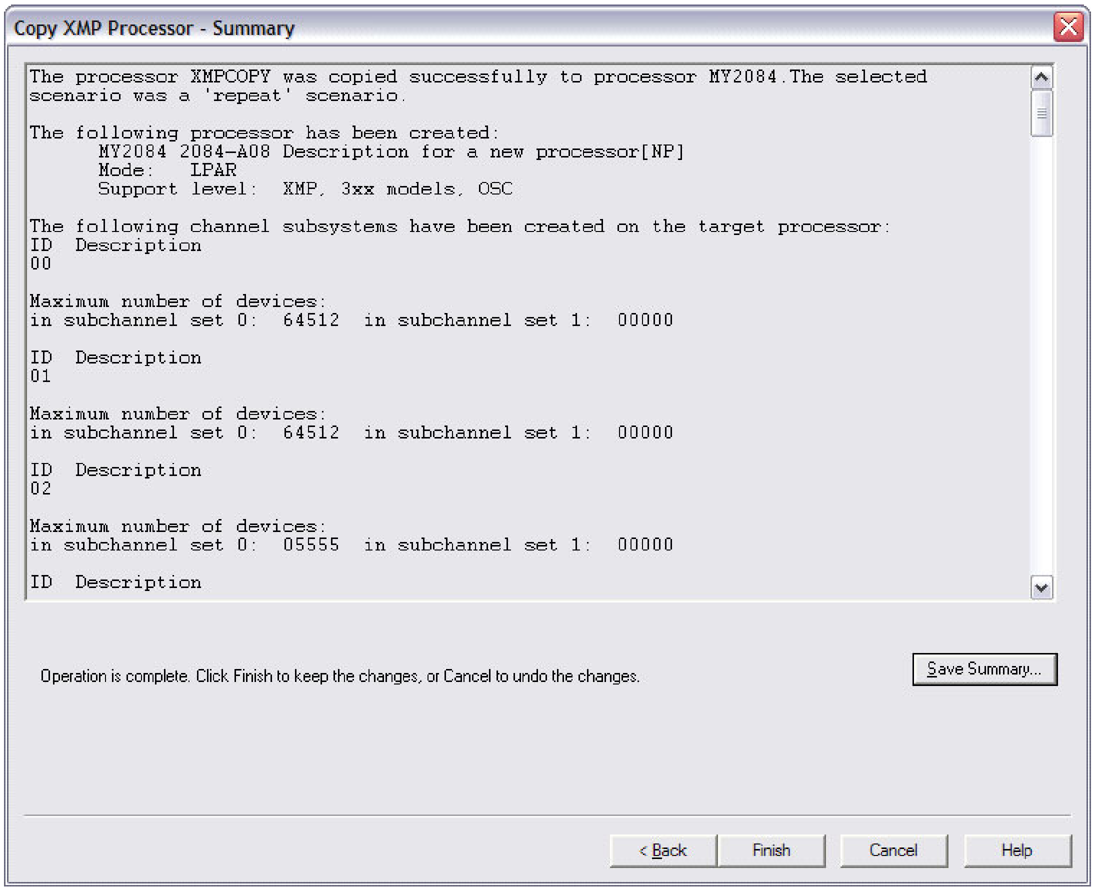
The Summary page describes all actions that have been performed for the source and target processor.

Note that modifications might apply for the source processor in the upgrade scenario.

All objects have been copied, which means:

- The XMP processor object itself
- All channel subsystems
- All partitions
- All CHPIDs
- All CHPID access and candidate lists for all CHPIDs
- All CHPID<->CU connections
- All processor<->CU related connections
- All device<->processor and CSS connections

Using the Save Summary button, you can create a detailed report of all changes that have been applied to the configuration, as shown in Figure 24-31 on page 458.



```
The processor XMPCOPY was copied successfully to processor MY2084.The selected
scenario was a 'repeat' scenario.

The following processor has been created:
      MY2084 2084-A08 Description for a new processor[NP]
      Mode:   LPAR
      Support level:  XMP, 3xx models, OSC

The following channel subsystems have been created on the target processor:
ID  Description
00

Maximum number of devices:
in subchannel set 0:  64512  in subchannel set 1:  00000

ID  Description
01

Maximum number of devices:
in subchannel set 0:  64512  in subchannel set 1:  00000

ID  Description
02

Maximum number of devices:
in subchannel set 0:  05555  in subchannel set 1:  00000

ID  Description
```

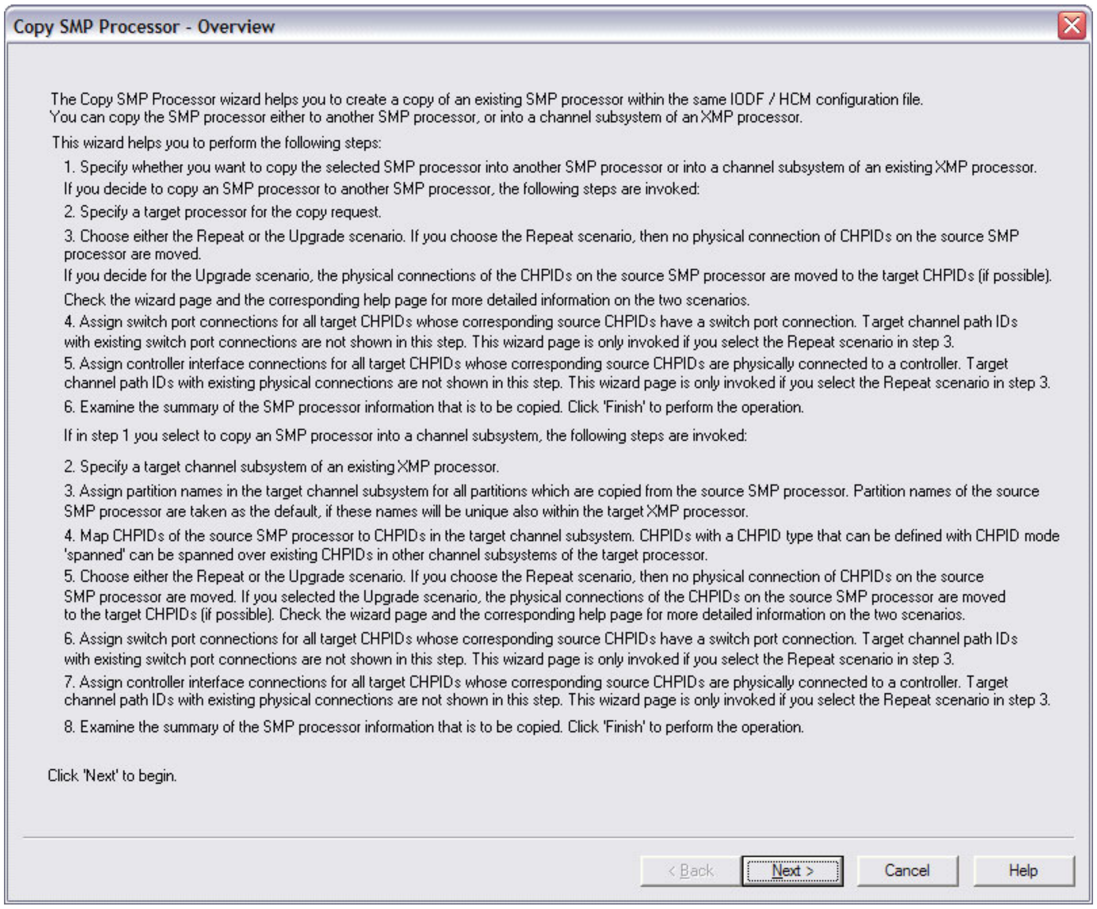
Operation is complete. Click Finish to keep the changes, or Cancel to undo the changes.

Figure 24-31 The summary page

Copy the SMP processor

The overview page, as shown in Figure 24-32 on page 459, describes all steps that you need to perform in order to copy an SMP processor.

The target can either be a new SMP processor or a channel subsystem of an existing XMP processor.


Copy SMP Processor - Overview

The Copy SMP Processor wizard helps you to create a copy of an existing SMP processor within the same IODF / HCM configuration file.
You can copy the SMP processor either to another SMP processor, or into a channel subsystem of an XMP processor.

This wizard helps you to perform the following steps:

1. Specify whether you want to copy the selected SMP processor into another SMP processor or into a channel subsystem of an existing XMP processor.
If you decide to copy an SMP processor to another SMP processor, the following steps are invoked:
2. Specify a target processor for the copy request.
3. Choose either the Repeat or the Upgrade scenario. If you choose the Repeat scenario, then no physical connection of CHPIDs on the source SMP processor are moved.
If you decide for the Upgrade scenario, the physical connections of the CHPIDs on the source SMP processor are moved to the target CHPIDs (if possible).
Check the wizard page and the corresponding help page for more detailed information on the two scenarios.
4. Assign switch port connections for all target CHPIDs whose corresponding source CHPIDs have a switch port connection. Target channel path IDs with existing switch port connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.
5. Assign controller interface connections for all target CHPIDs whose corresponding source CHPIDs are physically connected to a controller. Target channel path IDs with existing physical connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.
6. Examine the summary of the SMP processor information that is to be copied. Click 'Finish' to perform the operation.

If in step 1 you select to copy an SMP processor into a channel subsystem, the following steps are invoked:

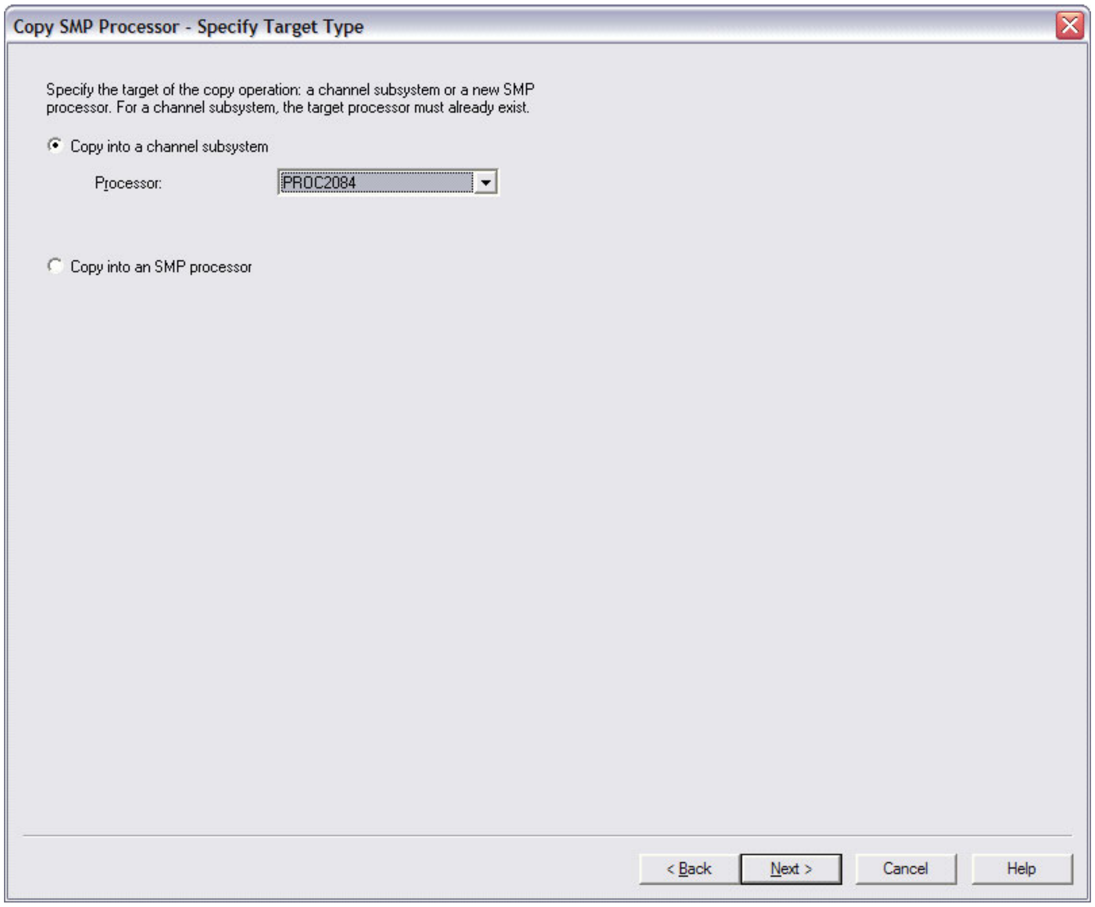
2. Specify a target channel subsystem of an existing XMP processor.
3. Assign partition names in the target channel subsystem for all partitions which are copied from the source SMP processor. Partition names of the source SMP processor are taken as the default, if these names will be unique also within the target XMP processor.
4. Map CHPIDs of the source SMP processor to CHPIDs in the target channel subsystem. CHPIDs with a CHPID type that can be defined with CHPID mode 'spanned' can be spanned over existing CHPIDs in other channel subsystems of the target processor.
5. Choose either the Repeat or the Upgrade scenario. If you choose the Repeat scenario, then no physical connection of CHPIDs on the source SMP processor are moved. If you selected the Upgrade scenario, the physical connections of the CHPIDs on the source SMP processor are moved to the target CHPIDs (if possible). Check the wizard page and the corresponding help page for more detailed information on the two scenarios.
6. Assign switch port connections for all target CHPIDs whose corresponding source CHPIDs have a switch port connection. Target channel path IDs with existing switch port connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.
7. Assign controller interface connections for all target CHPIDs whose corresponding source CHPIDs are physically connected to a controller. Target channel path IDs with existing physical connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.
8. Examine the summary of the SMP processor information that is to be copied. Click 'Finish' to perform the operation.

Click 'Next' to begin.

Figure 24-32 Copying an SMP processor overview page

In the Specify Target Type page, shown in Figure 24-33 on page 460, you must decide whether to copy:

- Into a channel subsystem of an existing XMP processor
- Into a new SMP processor



Specify the target of the copy operation: a channel subsystem or a new SMP processor. For a channel subsystem, the target processor must already exist.

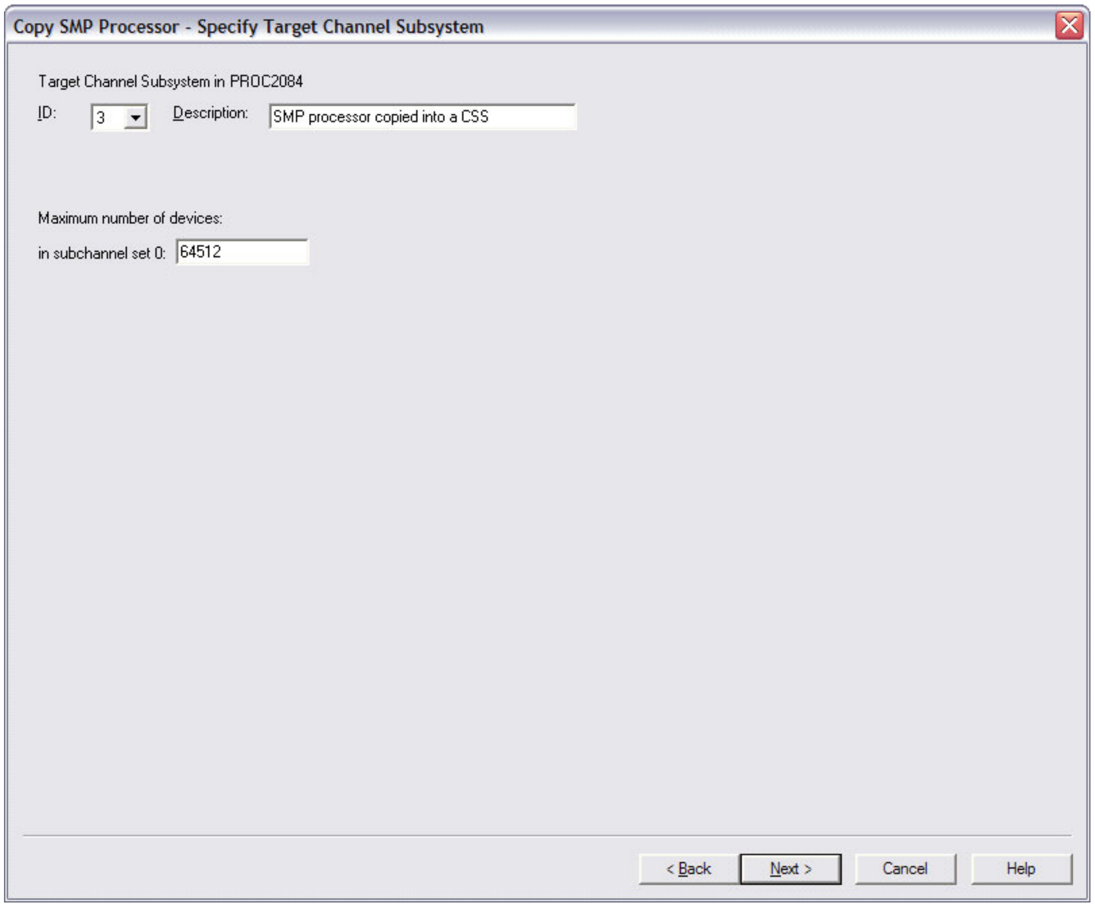
(●) Copy into a channel subsystem

Processor: PROC2084

( ) Copy into an SMP processor

Figure 24-33 Specifying a target type page

In the Specify Target Channel Subsystem page, shown in Figure 24-34 on page 461, you enter all required information to create a channel subsystem.



Copy SMP Processor - Specify Target Channel Subsystem

Target Channel Subsystem in PROC2084

ID: 3 Description: SMP processor copied into a CSS

Maximum number of devices:

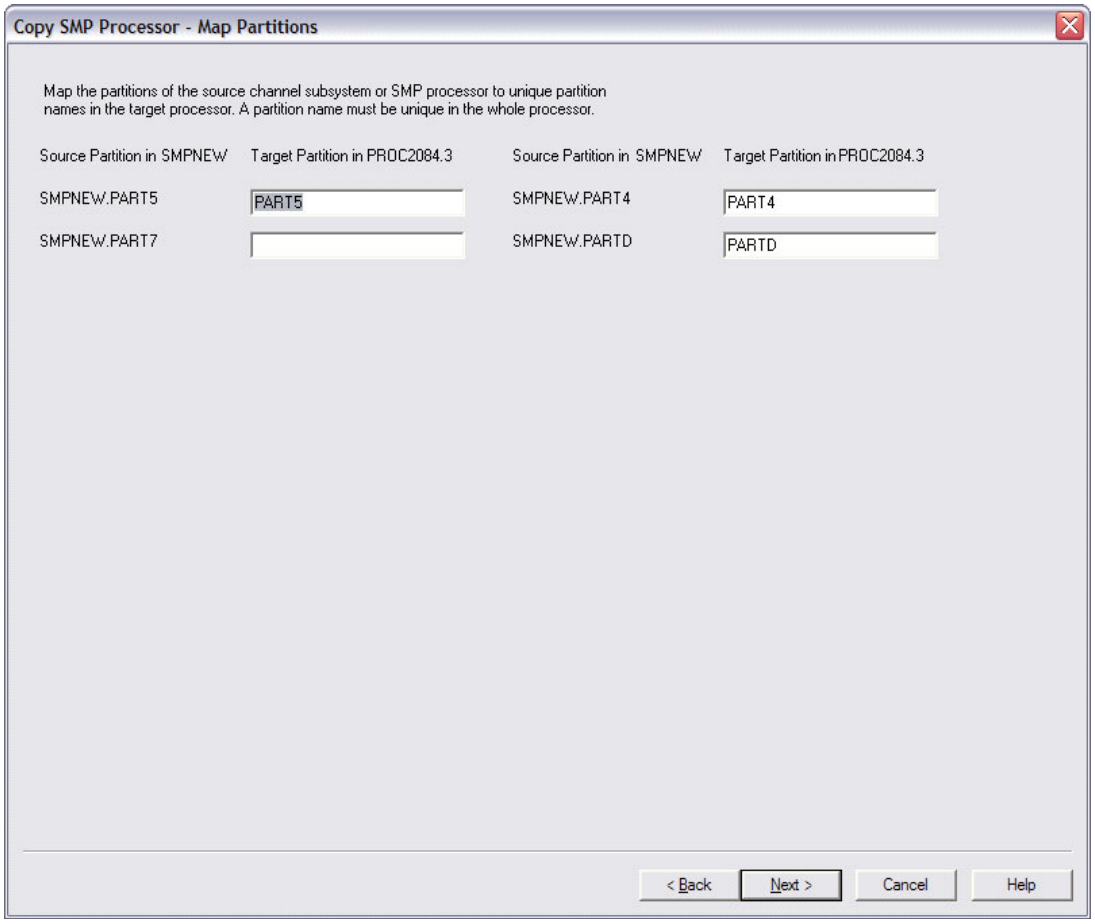
in subchannel set 0: 64512

Figure 24-34 Specifying a target channel subsystem page

In the Map Partitions page, shown in Figure 24-35 on page 462, partitions in the source SMP processor are mapped to new partitions in the target. Be aware that:

- Partition names must be unique within the XMP processor. If a source partition name is also unique in the target processor, then this name is proposed.
- You must specify a target partition name for each of the source partitions that get copied.
- You must not leave a target partition name BLANK, or HCM will issue an error message.
- In case you enter a duplicate name, HCM will also reject this name.

Note that the partition IDs also get copied, and that you cannot change the partition usage in the copy process. This ensures, for instance, that CHPID access and candidate lists can be set up correctly.


Copy SMP Processor - Map Partitions

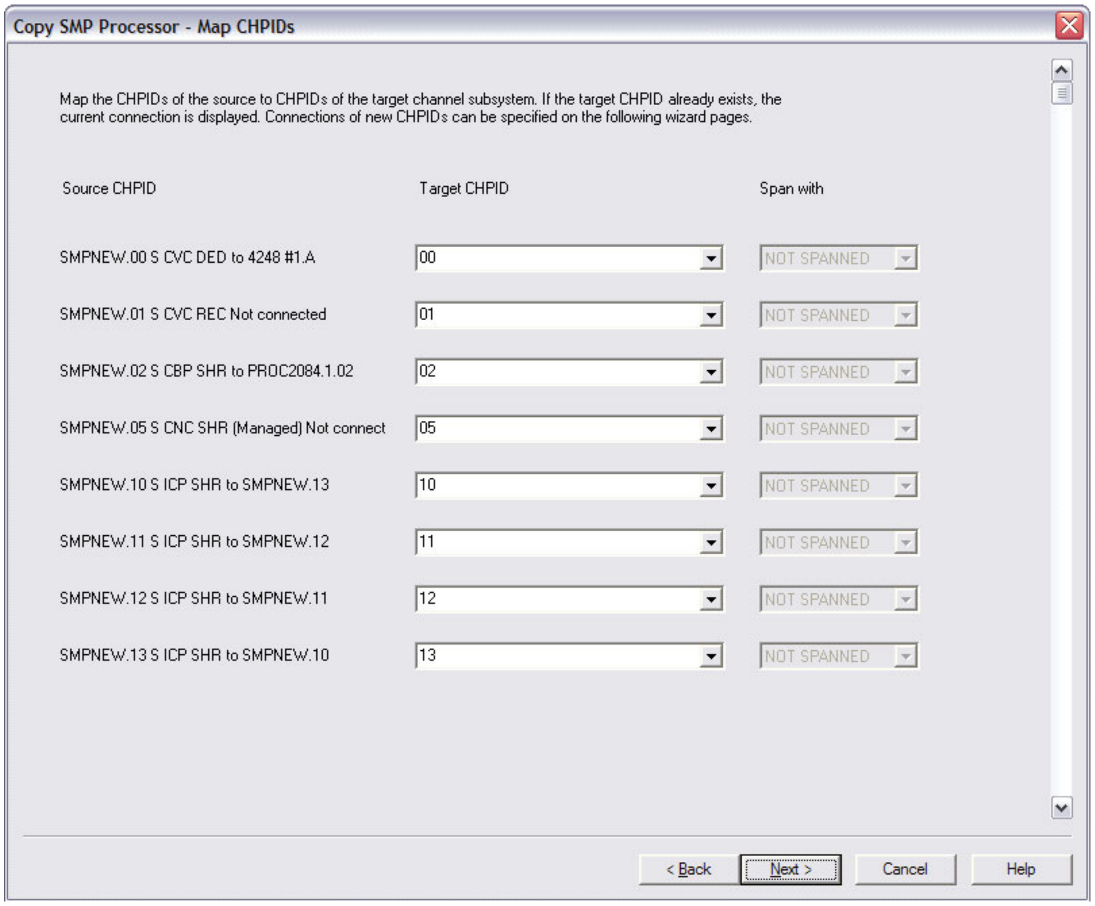
Map the partitions of the source channel subsystem or SMP processor to unique partition names in the target processor. A partition name must be unique in the whole processor.

Source Partition in SMPNEW	Target Partition in PROC2084.3	Source Partition in SMPNEW	Target Partition in PROC2084.3
SMPNEW.PART5	PART5	SMPNEW.PART4	PART4
SMPNEW.PART7		SMPNEW.PARTD	PARTD

Figure 24-35 Mapping a partitions page

The Map CHPIDs page, shown in Figure 24-36 on page 463, maps source CHPIDs to target CHPIDs, possibly by spanning target CHPIDs with instances of other CSSs.

In other words, you can span the new target CHPID instances with CHPIDs in other instances provided that the channel path IDs coincide.


Copy SMP Processor - Map CHPIDs

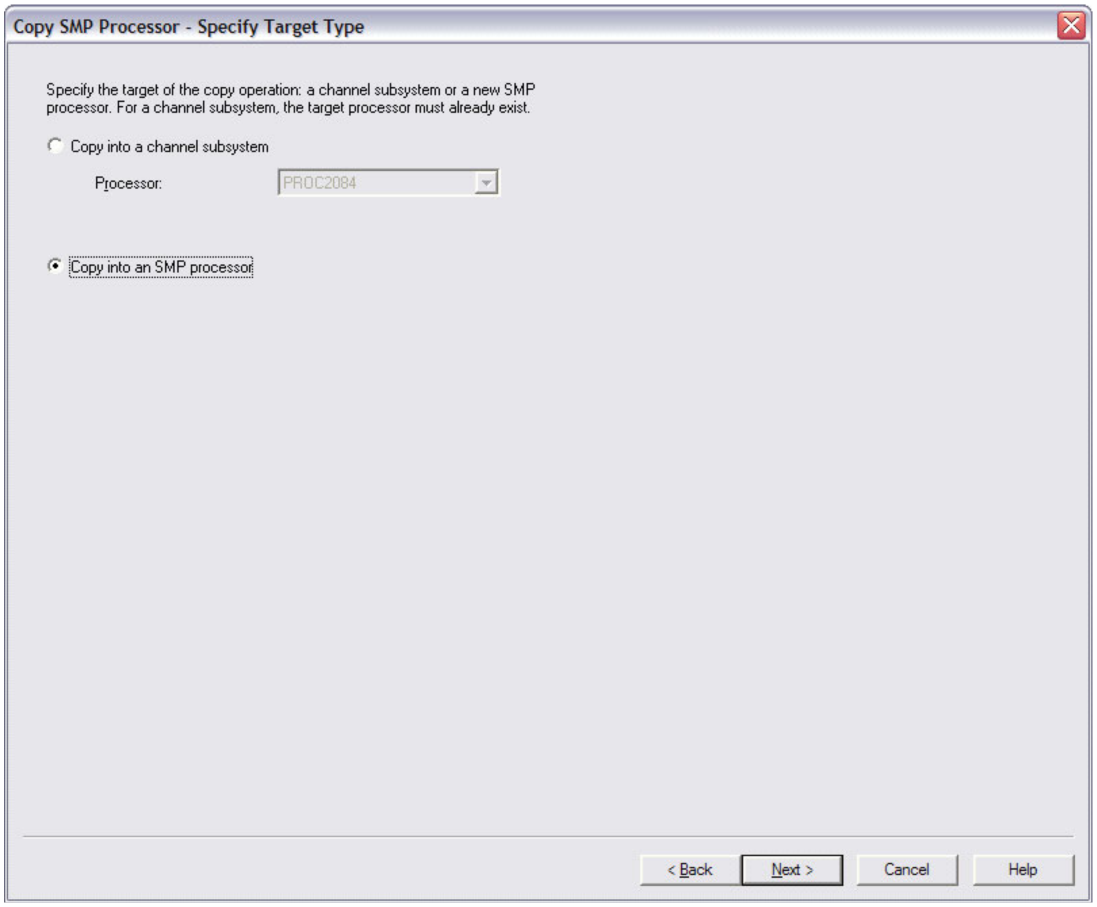
Map the CHPIDs of the source to CHPIDs of the target channel subsystem. If the target CHPID already exists, the current connection is displayed. Connections of new CHPIDs can be specified on the following wizard pages.

Source CHPID	Target CHPID	Span with
SMPNEW.00 S CVC DED to 4248 #1.A	00	NOT SPANNED
SMPNEW.01 S CVC REC Not connected	01	NOT SPANNED
SMPNEW.02 S CBP SHR to PROC2084.1.02	02	NOT SPANNED
SMPNEW.05 S CNC SHR (Managed) Not connect	05	NOT SPANNED
SMPNEW.10 S ICP SHR to SMPNEW.13	10	NOT SPANNED
SMPNEW.11 S ICP SHR to SMPNEW.12	11	NOT SPANNED
SMPNEW.12 S ICP SHR to SMPNEW.11	12	NOT SPANNED
SMPNEW.13 S ICP SHR to SMPNEW.10	13	NOT SPANNED

Figure 24-36 The Map CHPIDs page

The Specify Target Type page, shown in Figure 24-37 on page 464, shows that the decision option can also be to copy an existing SMP processor into a different, new SMP processor.

We are still in the Copy SMP Processor, but we now decide to copy the source SMP Processor into a new SMP processor, and not into a channel subsystem.



Specify the target of the copy operation: a channel subsystem or a new SMP processor. For a channel subsystem, the target processor must already exist.

[ ] Copy into a channel subsystem

Processor: PROC2084

[✓ Copy into an SMP processor]

Figure 24-37 Specifying the target type page

The Specify Target Processor wizard page is the next page, which follows the previously shown Copy SMP Processor – Specify Target Type page.

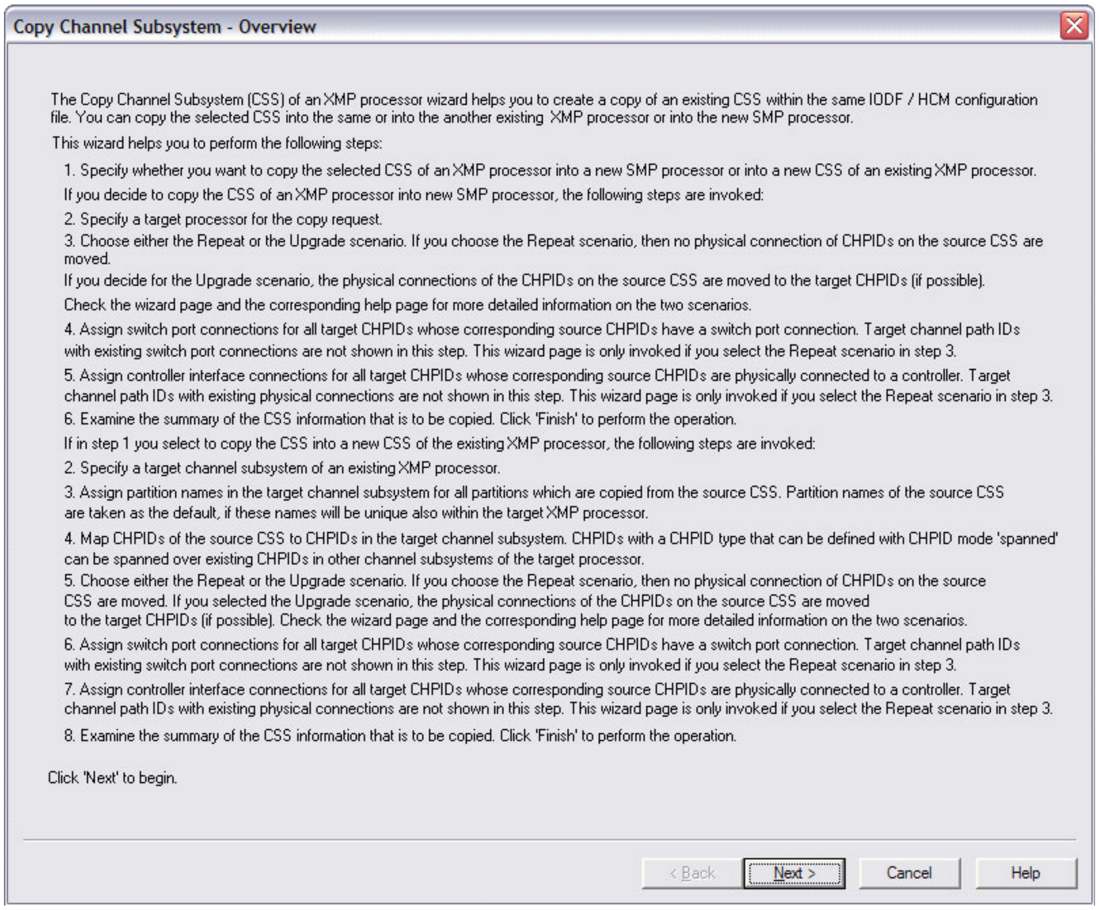
The Specify Target Processor wizard page was already shown for the Copy XMP Processor case.

All other pages for the Copy SMP Processor wizard have also been shown before during our discussion of the Copy XMP Processor wizard.

24.14.2 Copy channel subsystem

The Copy Channel Subsystem overview page describes the steps you must perform to copy a channel subsystem. The target can either be a new SMP processor or a channel subsystem of an existing XMP processor.

The Overview page shows a description of steps you need to perform when working with the wizard, shown in Figure 24-38 on page 465.


Copy Channel Subsystem - Overview

The Copy Channel Subsystem (CSS) of an XMP processor wizard helps you to create a copy of an existing CSS within the same IODF / HCM configuration file. You can copy the selected CSS into the same or into the another existing XMP processor or into the new SMP processor.

This wizard helps you to perform the following steps:

1. Specify whether you want to copy the selected CSS of an XMP processor into a new SMP processor or into a new CSS of an existing XMP processor.

If you decide to copy the CSS of an XMP processor into new SMP processor, the following steps are invoked:

2. Specify a target processor for the copy request.

3. Choose either the Repeat or the Upgrade scenario. If you choose the Repeat scenario, then no physical connection of CHPIDs on the source CSS are moved.

If you decide for the Upgrade scenario, the physical connections of the CHPIDs on the source CSS are moved to the target CHPIDs (if possible).

Check the wizard page and the corresponding help page for more detailed information on the two scenarios.

4. Assign switch port connections for all target CHPIDs whose corresponding source CHPIDs have a switch port connection. Target channel path IDs with existing switch port connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.

5. Assign controller interface connections for all target CHPIDs whose corresponding source CHPIDs are physically connected to a controller. Target channel path IDs with existing physical connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.

6. Examine the summary of the CSS information that is to be copied. Click 'Finish' to perform the operation.

If in step 1 you select to copy the CSS into a new CSS of the existing XMP processor, the following steps are invoked:

2. Specify a target channel subsystem of an existing XMP processor.

3. Assign partition names in the target channel subsystem for all partitions which are copied from the source CSS. Partition names of the source CSS are taken as the default, if these names will be unique also within the target XMP processor.

4. Map CHPIDs of the source CSS to CHPIDs in the target channel subsystem. CHPIDs with a CHPID type that can be defined with CHPID mode 'spanned' can be spanned over existing CHPIDs in other channel subsystems of the target processor.

5. Choose either the Repeat or the Upgrade scenario. If you choose the Repeat scenario, then no physical connection of CHPIDs on the source CSS are moved. If you selected the Upgrade scenario, the physical connections of the CHPIDs on the source CSS are moved to the target CHPIDs (if possible). Check the wizard page and the corresponding help page for more detailed information on the two scenarios.

6. Assign switch port connections for all target CHPIDs whose corresponding source CHPIDs have a switch port connection. Target channel path IDs with existing switch port connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.

7. Assign controller interface connections for all target CHPIDs whose corresponding source CHPIDs are physically connected to a controller. Target channel path IDs with existing physical connections are not shown in this step. This wizard page is only invoked if you select the Repeat scenario in step 3.

8. Examine the summary of the CSS information that is to be copied. Click 'Finish' to perform the operation.

Click 'Next' to begin.

Figure 24-38 Copying the channel subsystem overview page

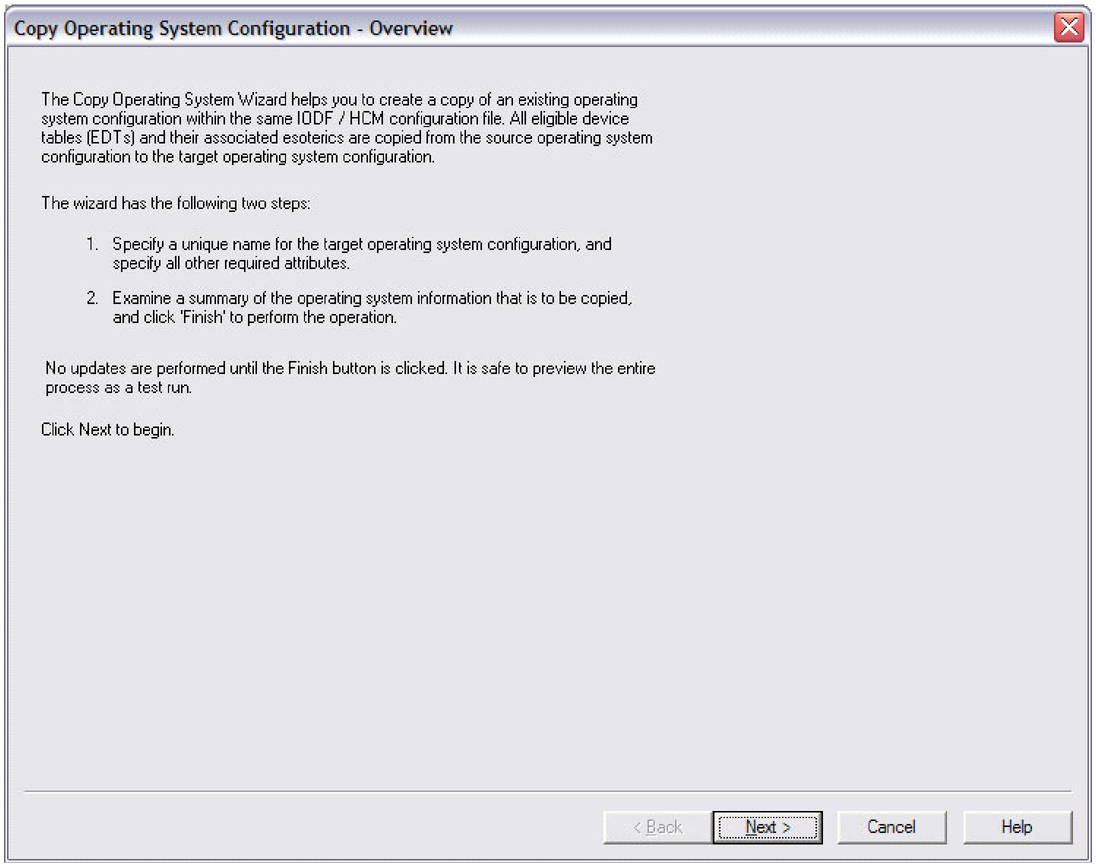
All wizard pages that might be shown when a channel subsystem is copied have already been shown in the previous pages.

As shown on the Wizard Page Flow Overview page in Figure 24-25 on page 452, the pages that might be invoked are the same as for the Copy SMP Processor wizard.

24.14.3 Copy operating system configuration

The Copy Operating System Configuration - Overview page shown in Figure 24-39 on page 466 describes the steps you need to perform in order to copy an operating system.

The Overview page shows a description of steps you need to perform when working with the wizard.



Copy Operating System Configuration - Overview

The Copy Operating System Wizard helps you to create a copy of an existing operating system configuration within the same IODF / HCM configuration file. All eligible device tables (EDTs) and their associated esoterics are copied from the source operating system configuration to the target operating system configuration.

The wizard has the following two steps:

1. Specify a unique name for the target operating system configuration, and specify all other required attributes.
2. Examine a summary of the operating system information that is to be copied, and click 'Finish' to perform the operation.

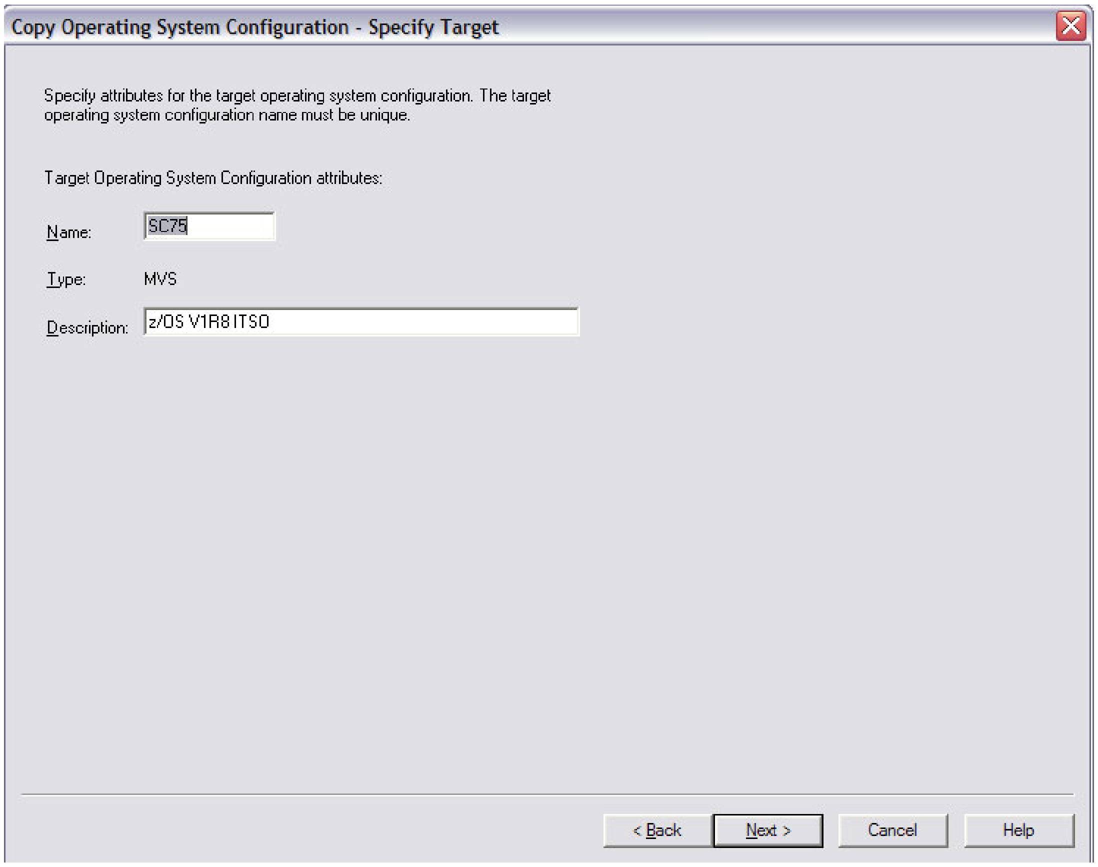
No updates are performed until the Finish button is clicked. It is safe to preview the entire process as a test run.

Click Next to begin.

Figure 24-39 Copying the operating system configuration overview page

In the Specify Target page shown in Figure 24-40 on page 467 you specify the operating system-related data needed to create the target operating system.

You need to specify the new name of the new operating system, and you can not change the type of the operating system. The type of the new operating system equals the type of the source operating system of the copy request.


Copy Operating System Configuration - Specify Target

Specify attributes for the target operating system configuration. The target operating system configuration name must be unique.

Target Operating System Configuration attributes:

Name: SC75

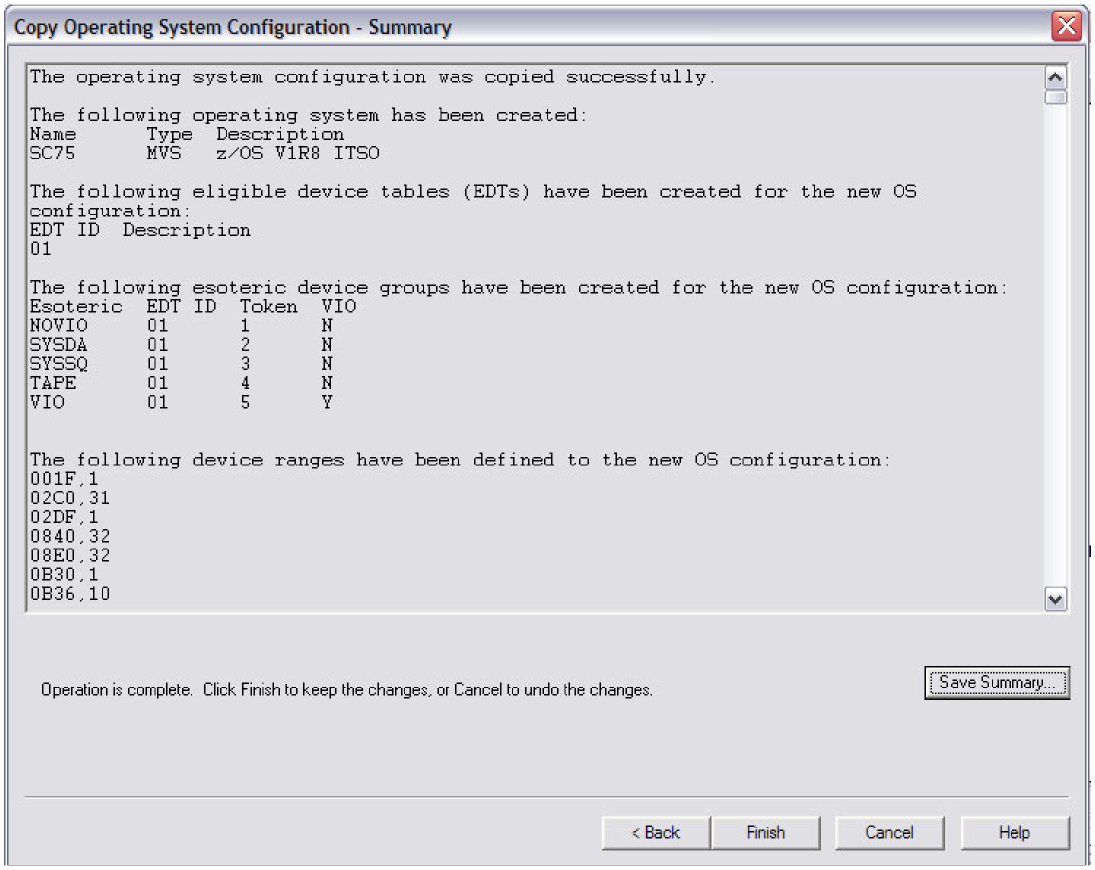
Type: MVS

Description: z/OS V1R8 ITSO

Figure 24-40 Copying the operating system configuration specify target page

The Summary page shown in Figure 24-41 on page 468 describes the actions that were taken during the Copy Operating System wizard, as follows:

- A new operating system was created.
- If the source operating system had EDTs defined, they were also copied.
- If the source operating system had esoterics defined, they were also copied.
- If the source operating system had devices defined to it, the device<->operating system definitions were also copied.



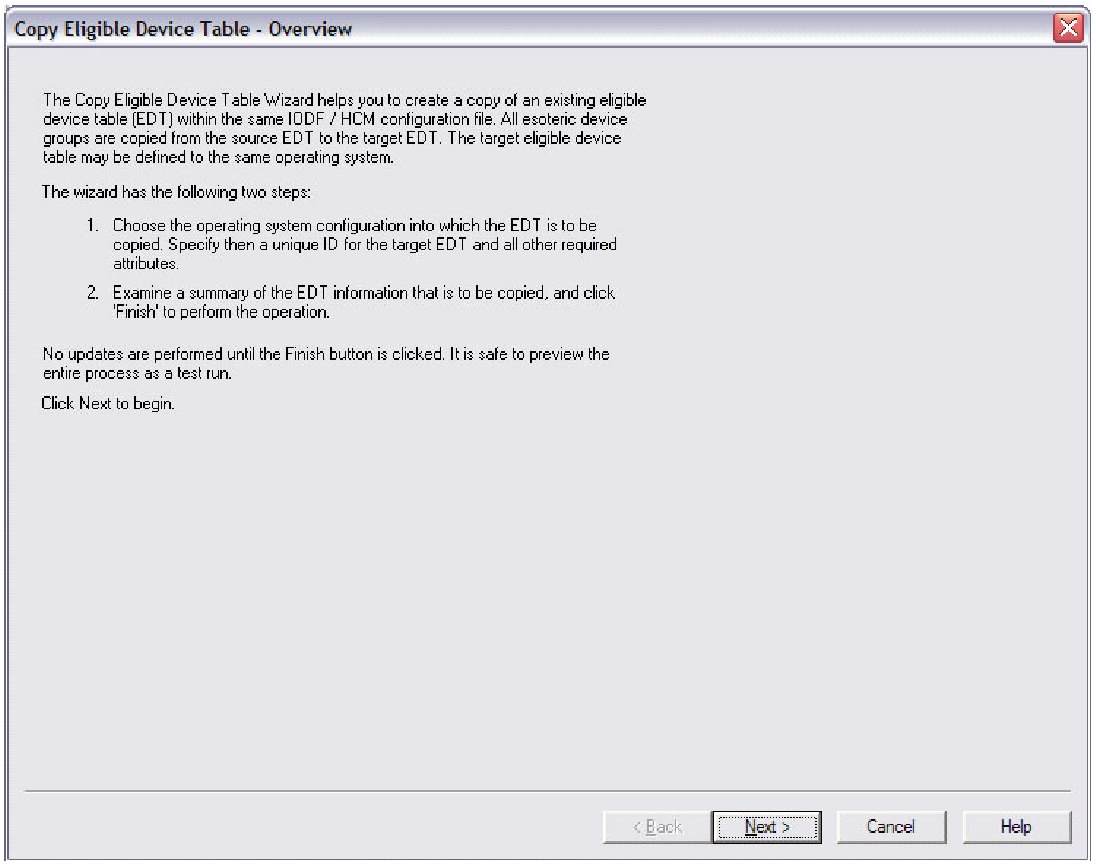
```
The operating system configuration was copied successfully.

The following operating system has been created:
Name      Type  Description
SC75      MVS   z/OS V1R8 ITSO

The following eligible device tables (EDTs) have been created for the new OS
configuration:
EDT ID  Description
01

The following esoteric device groups have been created for the new OS configuration:
Esoteric  EDT ID  Token  VIO
NOVIO     01      1      N
SYSDA     01      2      N
SYSSQ     01      3      N
TAPE      01      4      N
VIO       01      5      Y


The following device ranges have been defined to the new OS configuration:
001F,1
02C0,31
02DF,1
0840,32
08E0,32
0B30,1
0B36,10
```

Operation is complete. Click Finish to keep the changes, or Cancel to undo the changes.

Figure 24-41 Copying the operating system configuration summary page

24.14.4 Copying an eligible device table

The Copy Eligible Device Table Overview page shown in Figure 24-42 on page 469 describes the steps you need to perform in order to copy an Eligible Device Table.


Copy Eligible Device Table - Overview

The Copy Eligible Device Table Wizard helps you to create a copy of an existing eligible device table (EDT) within the same IODF / HCM configuration file. All esoteric device groups are copied from the source EDT to the target EDT. The target eligible device table may be defined to the same operating system.

The wizard has the following two steps:

1. Choose the operating system configuration into which the EDT is to be copied. Specify then a unique ID for the target EDT and all other required attributes.
2. Examine a summary of the EDT information that is to be copied, and click 'Finish' to perform the operation.

No updates are performed until the Finish button is clicked. It is safe to preview the entire process as a test run.

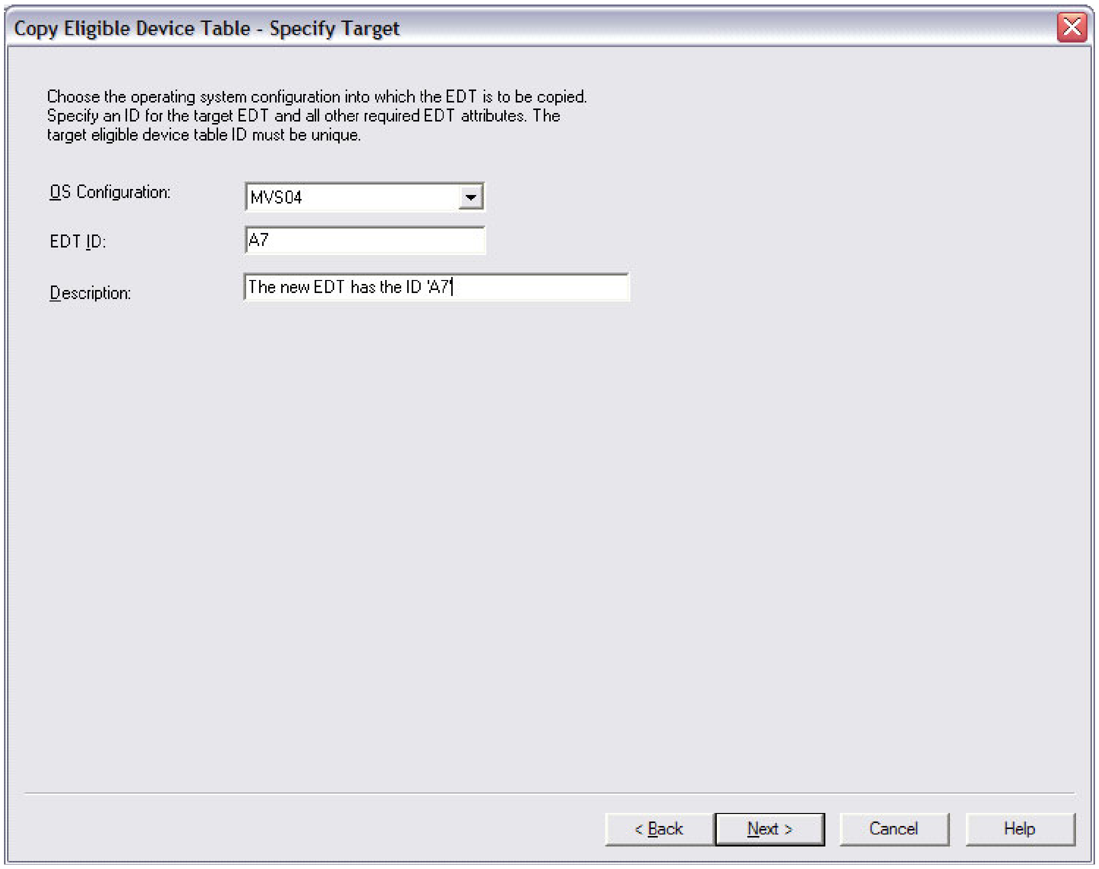
Click Next to begin.

Figure 24-42 Copying the eligible device table overview page

On the Specify Target page shown in Figure 24-43 on page 470 you specify the target operating system as well as the target EDT information.

The default is to copy the EDT within the same operating system.

The operating system must already exist in the configuration, while the EDT that gets specified must not yet exist.



Choose the operating system configuration into which the EDT is to be copied. Specify an ID for the target EDT and all other required EDT attributes. The target eligible device table ID must be unique.

OS Configuration: MVS04

EDT ID: A7

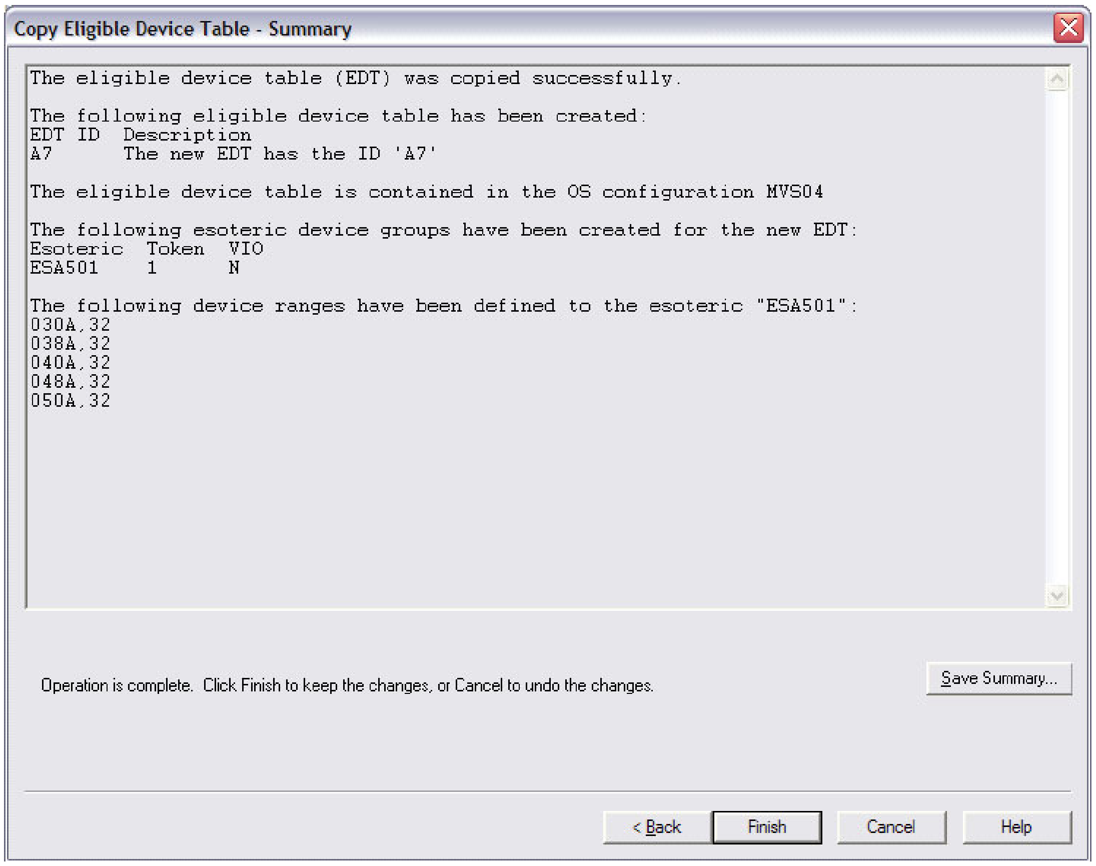
Description: The new EDT has the ID 'A7'

Figure 24-43 Copying the eligible device table specify target

The Summary page as shown in Figure 24-44 on page 471 describes the actions that were performed for the target Eligible Device Table, as follows:

- The source EDT was copied to the target EDT, and the target EDT was newly created.
- If the source operating system had Esoteric Device Tables defined, they were also copied.
- If the source operating system had devices defined to it, the device<->operating system definitions were also copied.

Using the Save Summary button, you can create a detailed report of all changes that were been applied to the configuration.



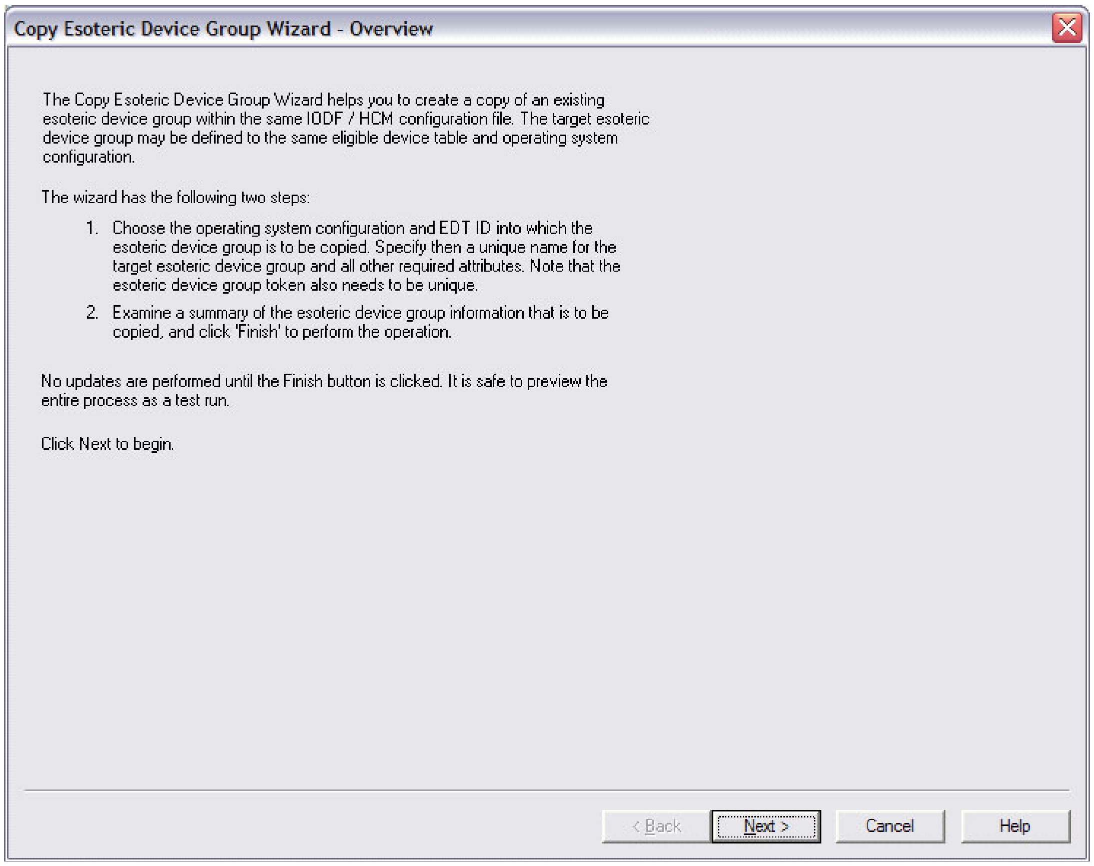
```
The eligible device table (EDT) was copied successfully.

The following eligible device table has been created:
EDT ID  Description
A7      The new EDT has the ID 'A7'

The eligible device table is contained in the OS configuration MVS04

The following esoteric device groups have been created for the new EDT:
Esoteric  Token  VIO
ESA501    1      N

The following device ranges have been defined to the esoteric "ESA501":
030A,32
038A,32
040A,32
048A,32
050A,32
```

Operation is complete. Click Finish to keep the changes, or Cancel to undo the changes.

Figure 24-44 Copying the eligible device table summary page

24.14.5 Copying an Esoteric Device Group

The Overview page shown in Figure 24-45 on page 472 describes the steps you have to perform in order to copy an Esoteric Device Group.



The Copy Esoteric Device Group Wizard helps you to create a copy of an existing esoteric device group within the same IODF / HCM configuration file. The target esoteric device group may be defined to the same eligible device table and operating system configuration.

The wizard has the following two steps:

1. Choose the operating system configuration and EDT ID into which the esoteric device group is to be copied. Specify then a unique name for the target esoteric device group and all other required attributes. Note that the esoteric device group token also needs to be unique.
2. Examine a summary of the esoteric device group information that is to be copied, and click 'Finish' to perform the operation.

No updates are performed until the Finish button is clicked. It is safe to preview the entire process as a test run.

Click Next to begin.

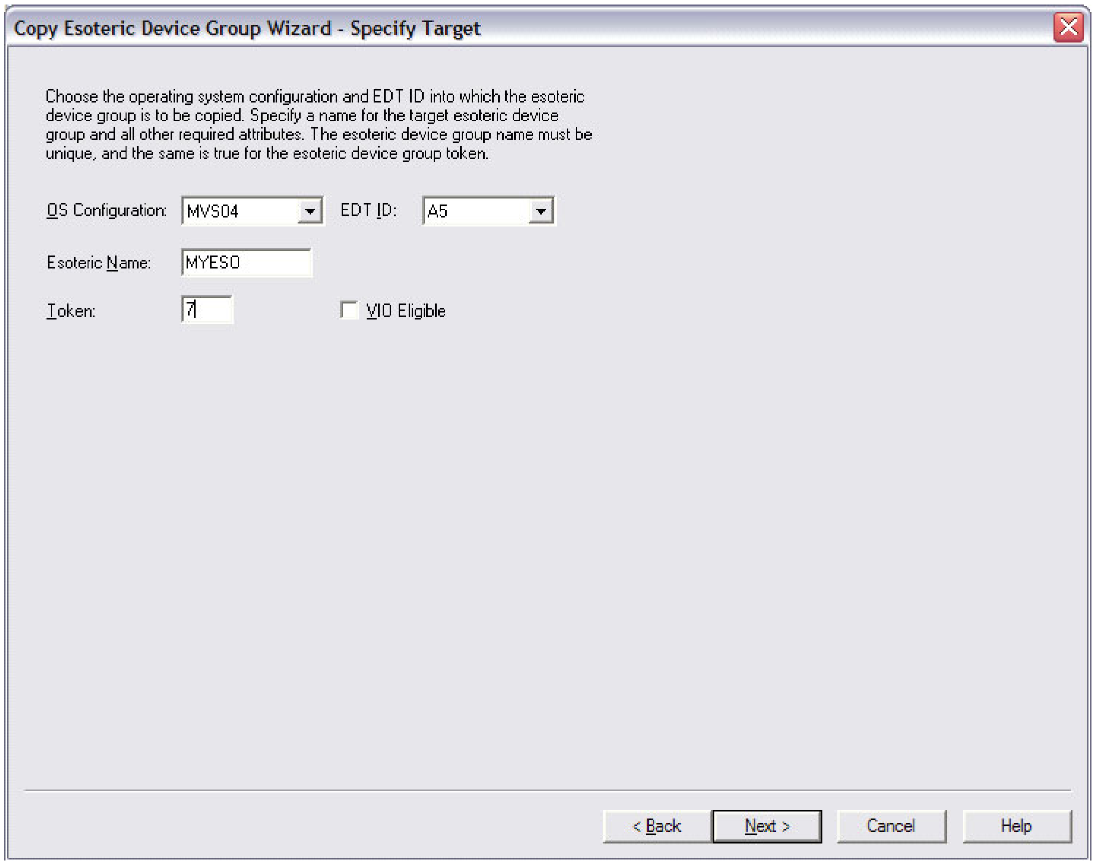
Figure 24-45 The Overview page

On the Specify Target page shown in Figure 24-46 on page 473 you specify the target operating system and target EDT ID as well as the target Esoteric Device Group information.

The default is to copy the Esoteric Device Group within the same EDT.

The target operating system as well as the target EDT must already exist.

Esoteric Device Table specific data must indicate a not yet existing Esoteric Device Group.


Copy Esoteric Device Group Wizard - Specify Target

Choose the operating system configuration and EDT ID into which the esoteric device group is to be copied. Specify a name for the target esoteric device group and all other required attributes. The esoteric device group name must be unique, and the same is true for the esoteric device group token.

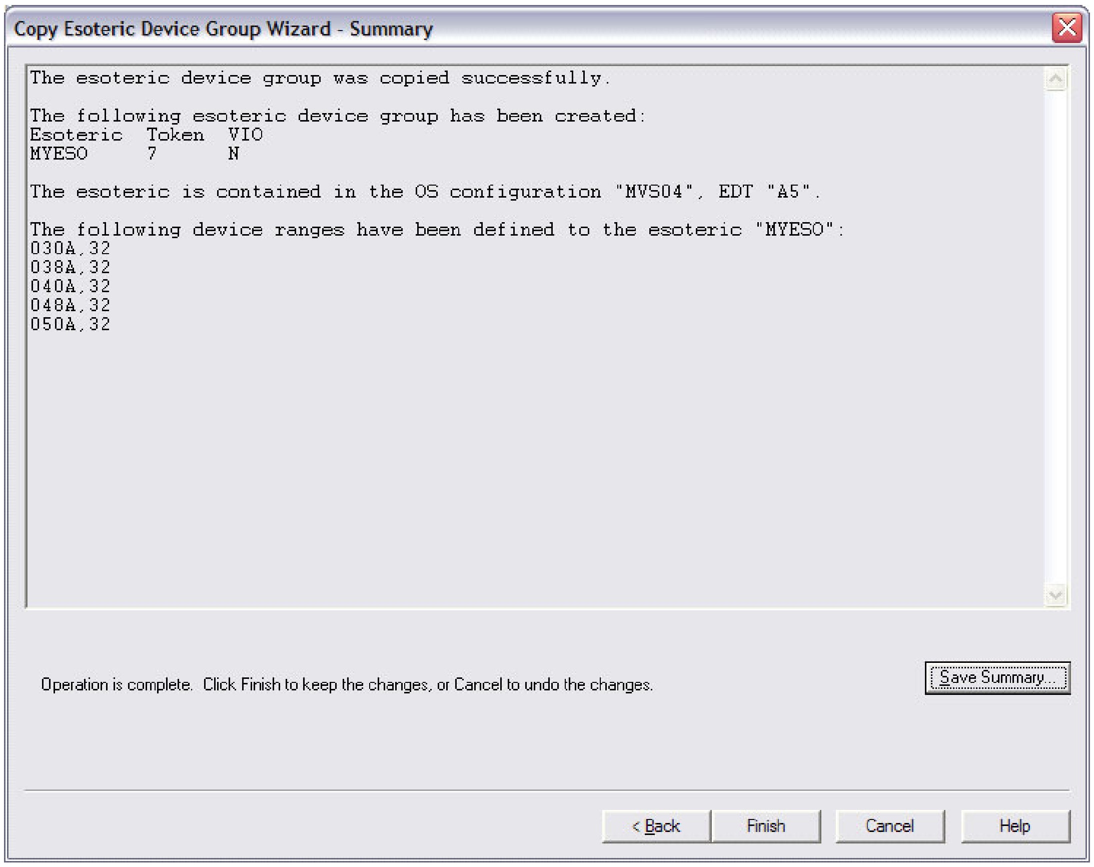
OS Configuration:	MVS04	EDT ID:	A5
Esoteric Name:	MYESO		
Token:	7	☐ VIO Eligible	

Figure 24-46 Specifying a target page

The Summary page shown in Figure 24-47 on page 474 describes the actions that have been performed for the target Esoteric Device Group, as follows:

- The source Esoteric Device Group was copied to the target Esoteric Device Group, and the target Esoteric Device Group was newly created.
- If the source operating system had devices defined to it, the device<->operating system definitions were also copied.

Using the Save Summary button, you can create a detailed report of all changes that have been applied to the configuration.



```
The esoteric device group was copied successfully.

The following esoteric device group has been created:
Esoteric  Token  VIO
MYESO     7      N

The esoteric is contained in the OS configuration "MVS04", EDT "A5".

The following device ranges have been defined to the esoteric "MYESO":
030A,32
038A,32
040A,32
048A,32
050A,32
```

Operation is complete. Click Finish to keep the changes, or Cancel to undo the changes.

Figure 24-47 The Summary page

24.14.6 Create and copy the I/O subsystem

In the previous versions of HCM, the definition of large I/O subsystems can be complex, time-consuming, and error-prone.

z/OS V1R8 HCM provides a wizard to copy an existing I/O subsystem (with included controller, strings of devices, and connections), making possible a much faster, easier, and less error-prone definition.

Compared to the past, today‘s I/O configurations are much bigger and I/O hardware is much more powerful. Likewise, today‘s I/O definition process is much more complex, time-consuming and error-prone.

Implementation of Copy I/O Subsystem

The Copy I/O Subsystem Wizard can be invoked in two ways:

- Via the existing I/O subsystem’s context menu (right context menu), as shown in Figure 24-48 on page 475
- Via the Create menu in the HCM diagram (left bottom context menu), as shown in Figure 24-49 on page 475

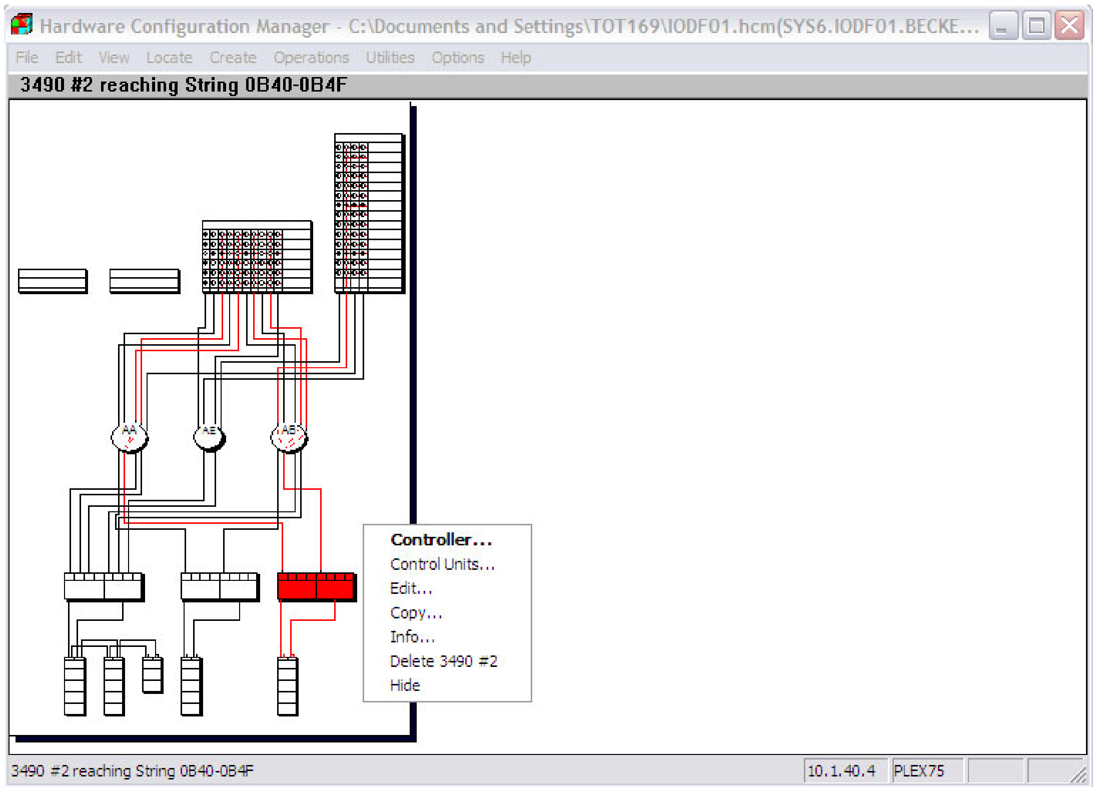

Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKE...
File Edit View Locate Create Operations Utilities Options Help
3490 #2 reaching String 0B40-0B4F
Controller...
Control Units...
Edit...
Copy...
Info...
Delete 3490 #2
Hide
3490 #2 reaching String 0B40-0B4F
10.1.40.4
PLEX75

Figure 24-48 Copying the I/O subsystem page 1

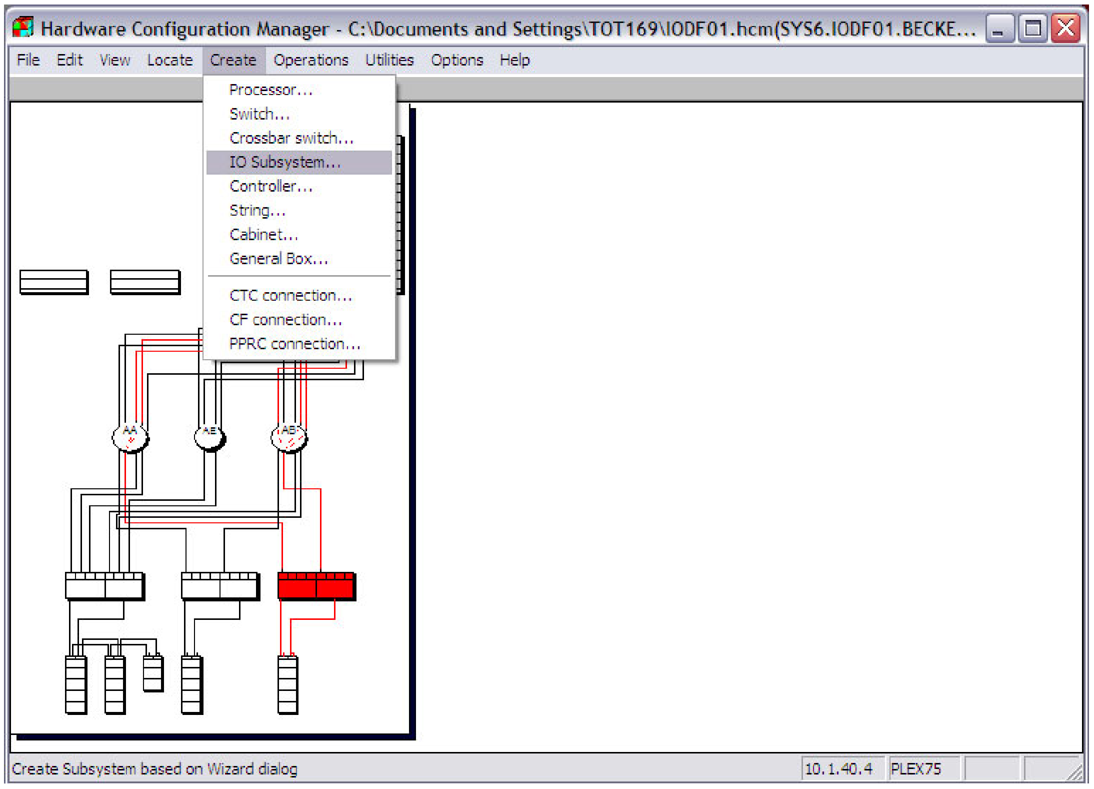

Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKE...
File Edit View Locate Create Operations Utilities Options Help
Processor...
Switch...
Crossbar switch...
IO Subsystem...
Controller...
String...
Cabinet...
General Box...
CTC connection...
CF connection...
PPRC connection...
Create Subsystem based on Wizard dialog
10.1.40.4
PLEX75

Figure 24-49 Copying the I/O subsystem page 2

The HCM Copy I/O Subsystem Wizard has 7 pages, but only on the thirst page do you specify attributes of the new I/O subsystem.

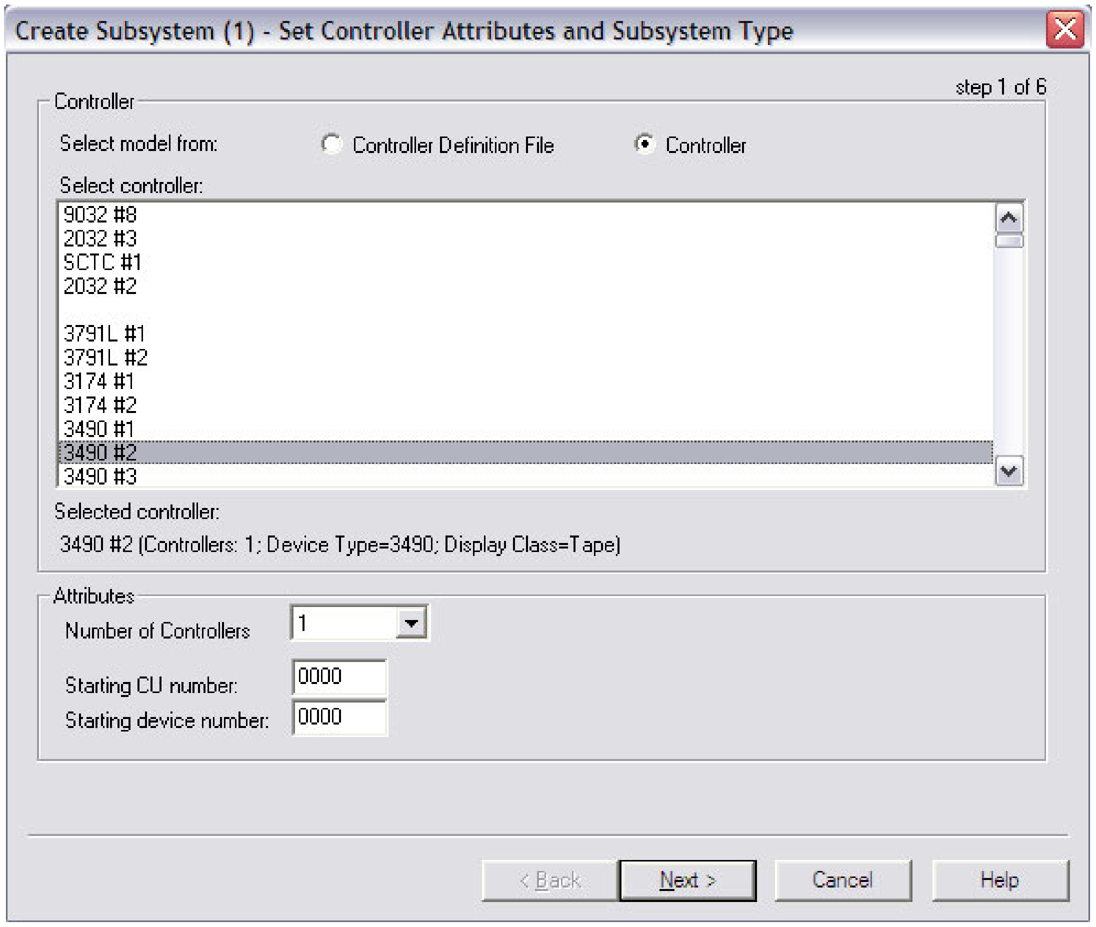
To create a valid new I/O subsystem, as shown in Figure 24-50, you must change the CU numbers, and you can also change other I/O subsystem-specific attributes, such as:

- A controller's serial number
- The subsystem identifier (SSID)
- I/O device IDs (which can be ambiguous, but often follow specific schemes)

The utility helps you change CU numbers and I/O device IDs globally via specification of a new first CU number, and I/O device ID.

All other CU numbers and I/O device IDs are changed according to the original "topology" (that is, the original gaps between the values will remain).

The subsequent wizard pages just document the new I/O subsystem object's contents and connections. No further input is necessary.


Create Subsystem (1) - Set Controller Attributes and Subsystem Type

step 1 of 6

Controller

Select model from: ( ) Controller Definition File [✓ Controller]

Select controller:

9032 #8
2032 #3
SCTC #1
2032 #2

3791L #1
3791L #2
3174 #1
3174 #2
3490 #1
[✓ 3490 #2]
3490 #3

Selected controller:

3490 #2 (Controllers: 1; Device Type=3490; Display Class=Tape)

Attributes

Number of Controllers: 1

Starting CU number: 0000

Starting device number: 0000

Figure 24-50 Creating an I/O subsystem page

24.15 Enhanced OS device group change

Until z/OS V1R7 HCM, you could only change all parameters and features, or the subchannel set ID for device groups, simultaneously.

z/OS V1R8 HCM provides a similar function to HCD, but enhances HCD's function significantly. HCM now allows you to modify user-selected devices, operating system parameters, and features without modifying any other device. The same is true for the subchannel set ID.

Using this function, you can:

- Change selected parameters of a grouping of OS devices
- Change the subchannel set ID of a grouping of OS devices

Implementation

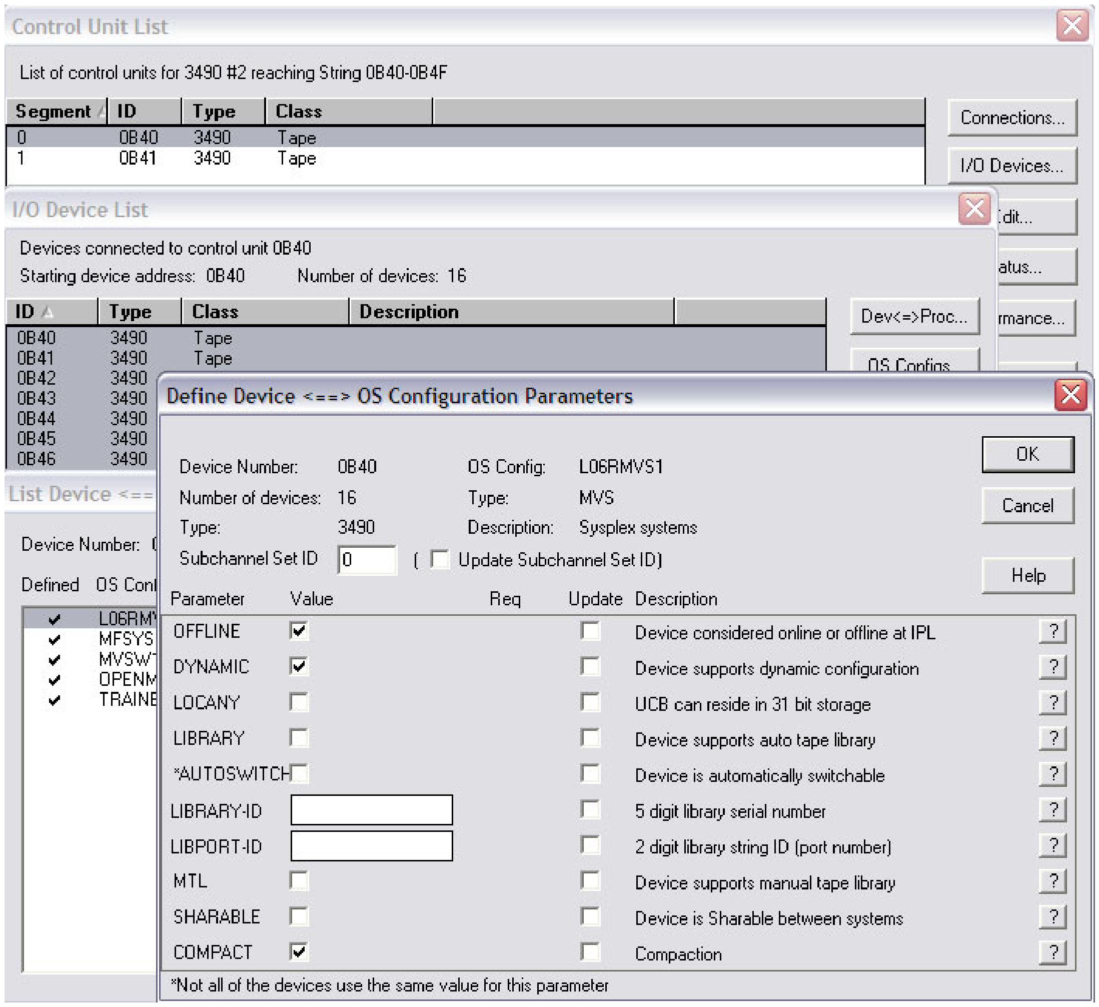
There are two new controls fields:

- Update Subchannel Set ID
- Update column

New controls or checkboxes are selected when:

- The subchannel set ID has been modified
- The subchannel set ID differs for devices of the device group, and the shown subchannel set ID should be applied to all devices of the device range
- A parameter or feature value has been modified
- A parameter or feature value differs for devices of the device group, and the shown parameter or feature value should be applied to all devices of the device range

Figure 24-51 on page 478 shows the new control fields and checkboxes on the Define Device page.



Control Unit List

List of control units for 3490 #2 reaching String 0B40-0B4F

Segment	ID	Type	Class
0	0B40	3490	Tape
1	0B41	3490	Tape

I/O Device List

Devices connected to control unit 0B40

Starting device address: 0B40 Number of devices: 16

ID	Type	Class	Description
0B40	3490	Tape	
0B41	3490	Tape	
0B42	3490		
0B43	3490		
0B44	3490		
0B45	3490		
0B46	3490		

List Device <==

Device Number:

Defined	OS Config
✓	L06RM
✓	MFSYS
✓	MVSW
✓	OPENM
✓	TRAINE

Define Device <==> OS Configuration Parameters

Device Number: 0B40 OS Config: L06RMVS1

Number of devices: 16 Type: MVS

Type: 3490 Description: Sysplex systems

Subchannel Set ID 0 (☐ Update Subchannel Set ID)

Parameter	Value	Req	Update	Description
OFFLINE	✓		☐	Device considered online or offline at IPL
DYNAMIC	✓		☐	Device supports dynamic configuration
LOCANY	☐		☐	UCB can reside in 31 bit storage
LIBRARY	☐		☐	Device supports auto tape library
*AUTOSWITCH	☐		☐	Device is automatically switchable
LIBRARY-ID			☐	5 digit library serial number
LIBPORT-ID			☐	2 digit library string ID (port number)
MTL	☐		☐	Device supports manual tape library
SHARABLE	☐		☐	Device is Sharable between systems
COMPACT	✓		☐	Compaction

*Not all of the devices use the same value for this parameter

Figure 24-51 The Define Device page

24.16 Performance data integration

z/OS V1R8 HCM gives you access to RMF Monitor III performance data for a selected object, as follows:

- The performance data is provided by the RMF Distributed Data Server (DDS) and presented in a Web browser window. This data is only available for objects within the sysplex of the system to which the user is logged on.
- An installed RMF with the DDS set up and running is a prerequisite to this function. Since the DDS is only available on z/OS, you cannot access performance data from z/VM®.
- Describes how hardware-related performance information is provided.
- Shows how detection of performance bottlenecks caused by the hardware configuration is improved.
- Lists the configuration objects for which performance data is provided.
- Identifies the dependencies for getting performance data in HCM.

HCM and RMF have a different perspective on resources, as follows:

- HCM concentrates on hardware objects and the relationship between them.

- RMF organizes I/O-related performance data per MVS image.
- Mapping between HCM objects and RMF resources may require user interaction.

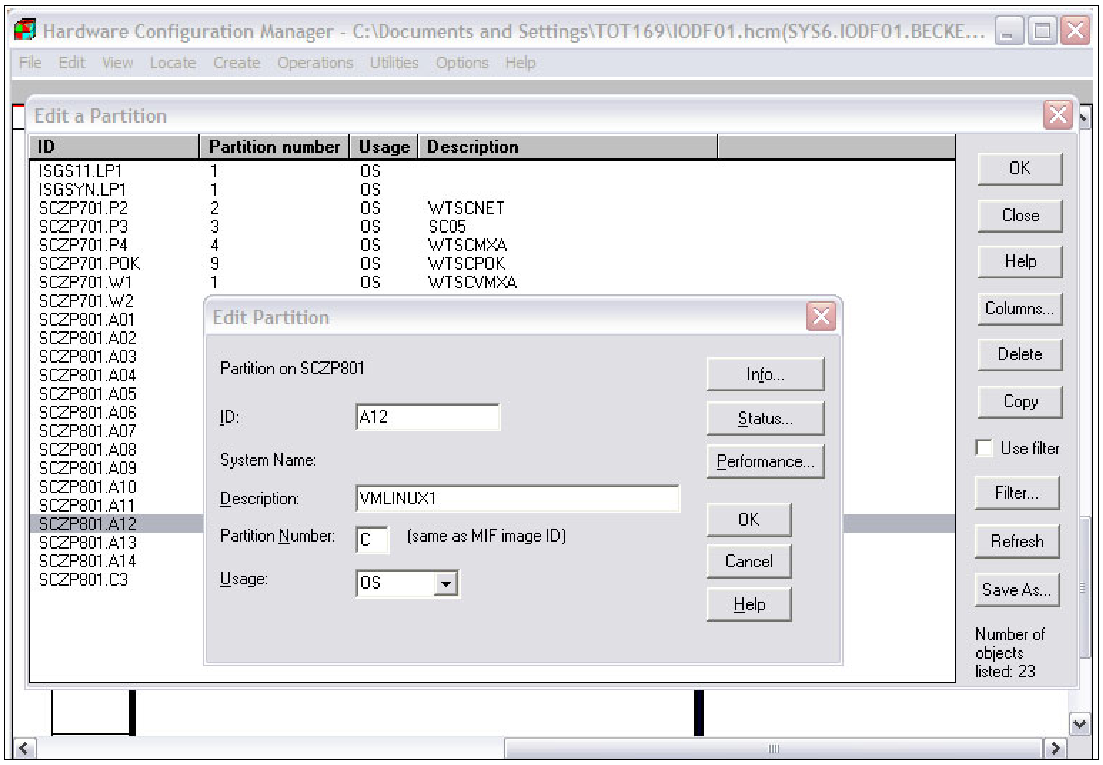
24.16.1 Implementation of performance data integration

z/OS V1R8 HCM calls your preferred Internet browser to connect to the z/OS Resource Measurement Facility (RMF) Distributed Data Server (DDS) server to show performance data for the object under focus.

Performance data integration is invoked by:

- A button in Edit dialogs of objects, as shown in Figure 24-52
- A menu item in the Utilities menu of the menu bar, as shown in Figure 24-53 on page 480
- A menu item in context menus of objects, as shown in Figure 24-54 on page 480
- An entry point to navigate through all performance data provided by DDS, as shown in Figure 24-55 on page 481

Performance data for selected objects (processor, partition, channel, control unit, DASD device), as shown in Figure 24-56 on page 481


Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKE...
File Edit View Locate Create Operations Utilities Options Help
Edit a Partition
OK
Close
Help
Columns...
Delete
Copy
Use filter
Filter...
Refresh
Save As...
Number of objects listed: 23
Edit Partition
Info...
Status...
Performance...
OK
Cancel
Help

ID	Partition number	Usage	Description
ISGS11.LP1	1	OS	
ISGSYN.LP1	1	OS	
SCZP701.P2	2	OS	WTSCNET
SCZP701.P3	3	OS	SC05
SCZP701.P4	4	OS	WTSCMXA
SCZP701.POK	9	OS	WTSCPOK
SCZP701.W1	1	OS	WTSCVMXA
SCZP701.W2			
SCZP801.A01			
SCZP801.A02			
SCZP801.A03			
SCZP801.A04			
SCZP801.A05			
SCZP801.A06			
SCZP801.A07			
SCZP801.A08			
SCZP801.A09			
SCZP801.A10			
SCZP801.A11			
SCZP801.A12			
SCZP801.A13			
SCZP801.A14			
SCZP801.C3			

Partition on SCZP801

ID: A12

System Name:

Description: VMLINUX1

Partition Number: C (same as MIF image ID)

Usage: OS

Figure 24-52 The performance page

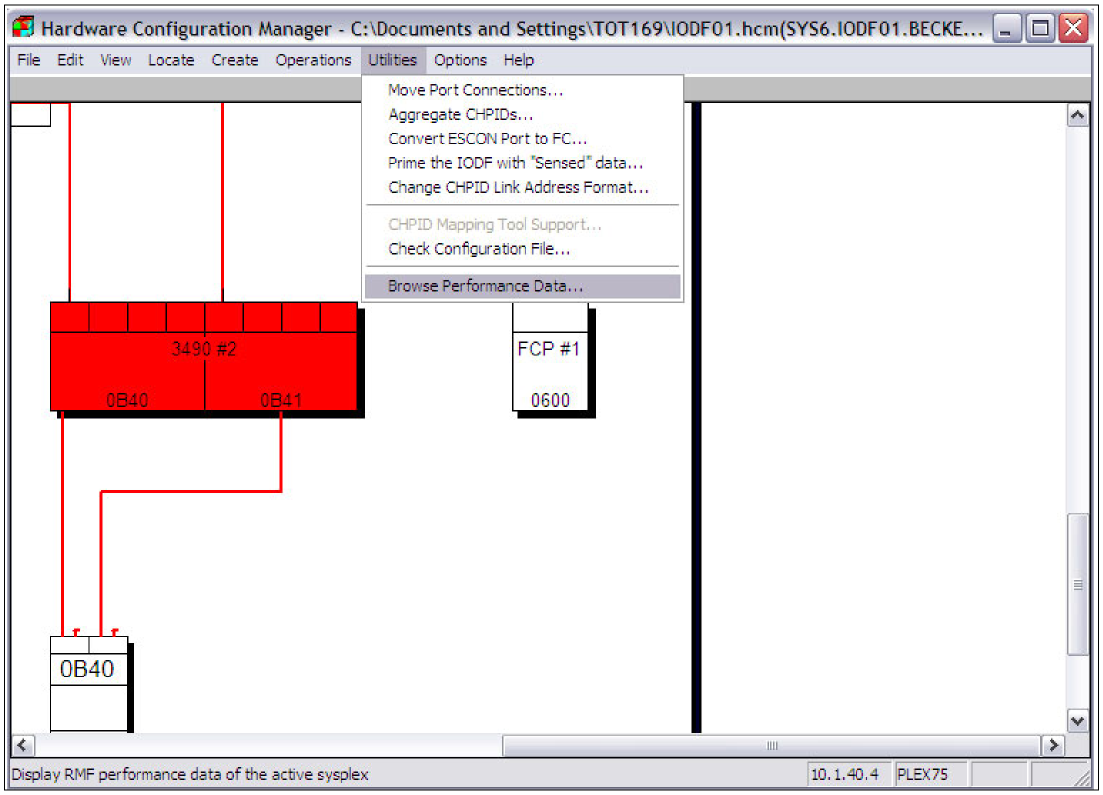

Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKE...
File Edit View Locate Create Operations Utilities Options Help
Move Port Connections...
Aggregate CHPIDs...
Convert ESCON Port to FC...
Prime the IODF with "Sensed" data...
Change CHPID Link Address Format...
CHPID Mapping Tool Support...
Check Configuration File...
Browse Performance Data...
3490 #2
0B40
0B41
FCP #1
0600
0B40
Display RMF performance data of the active sysplex
10.1.40.4
PLEX75

Figure 24-53 Browse performance

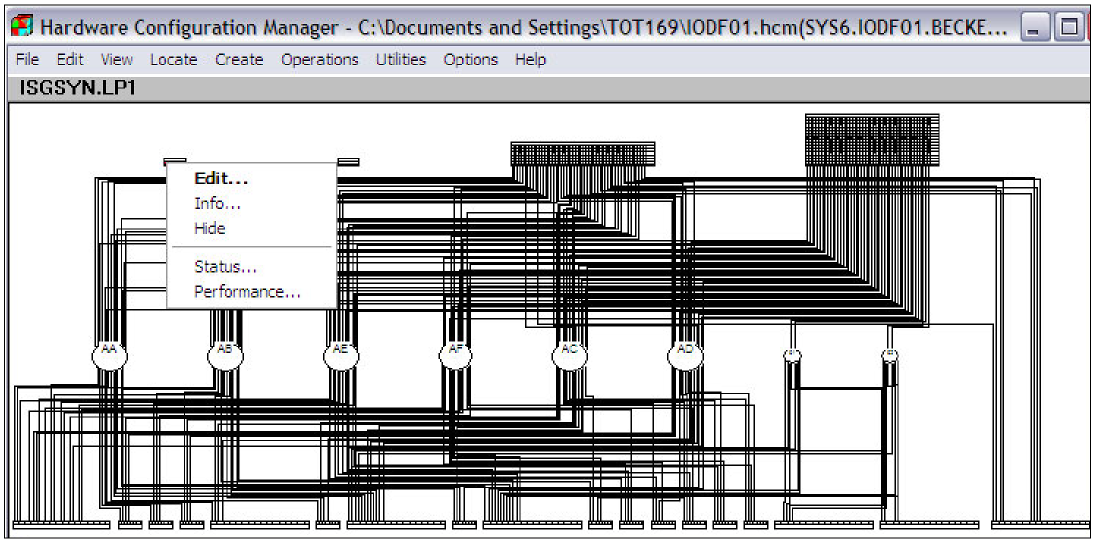
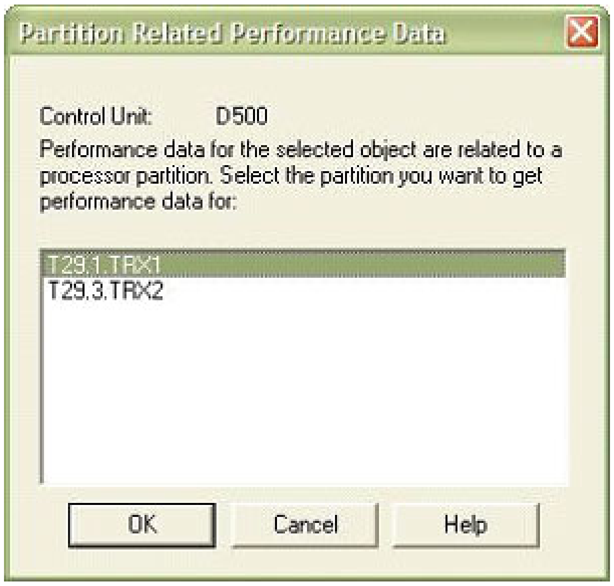

Hardware Configuration Manager - C:\Documents and Settings\TOT169\IODF01.hcm(SYS6.IODF01.BECKE...
File Edit View Locate Create Operations Utilities Options Help
ISGSYN.LP1
Edit...
Info...
Hide
Status...
Performance...
AA
AB
AE
AF
AC
AD

Figure 24-54 Performance items

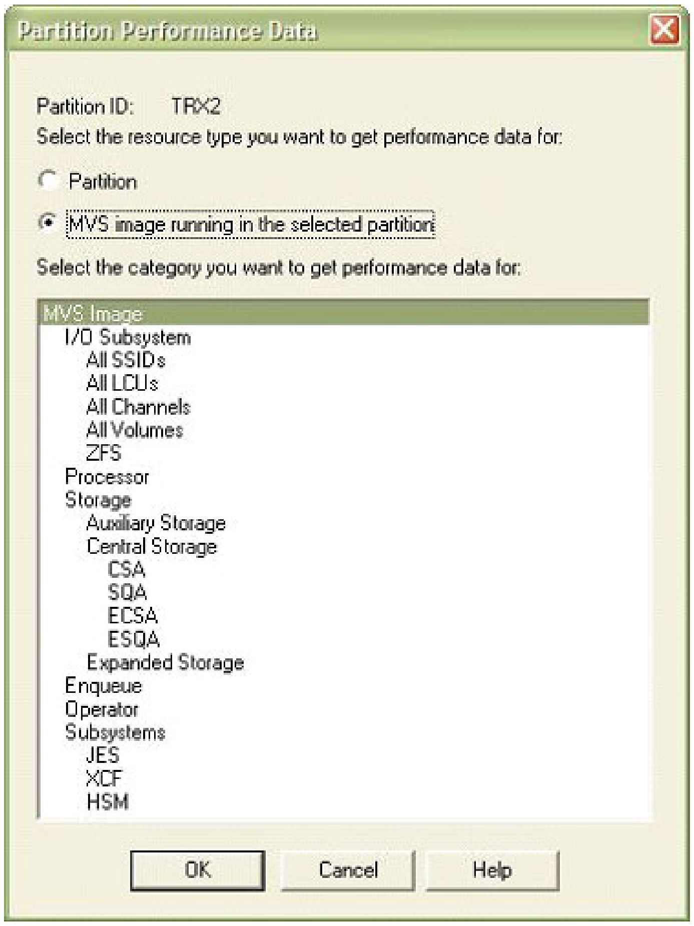


Control Unit: D500

Performance data for the selected object are related to a processor partition. Select the partition you want to get performance data for:

T29.1.TRX1
T29.3.TRX2

Figure 24-55 The partition performance page



Partition ID: TRX2

Select the resource type you want to get performance data for:

( ) Partition
(●) MVS image running in the selected partition

Select the category you want to get performance data for:

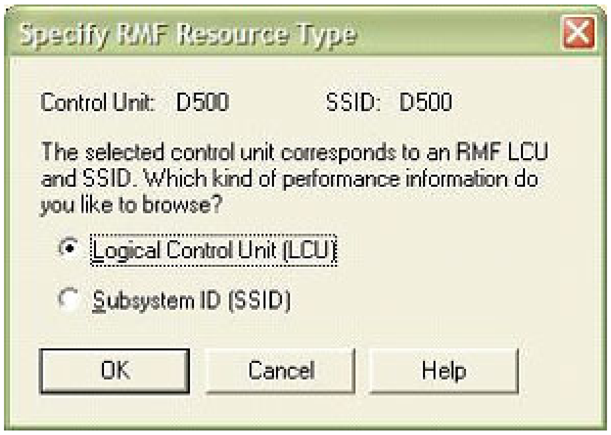
```
MVS Image
  I/O Subsystem
    All SSIDs
    All LCUs
    All Channels
    All Volumes
    ZFS
  Processor
  Storage
    Auxiliary Storage
    Central Storage
      CSA
      SQA
      ECSA
      ESQA
    Expanded Storage
  Enqueue
  Operator
  Subsystems
    JES
    XCF
    HSM
```

Figure 24-56 The partition performance data page

RMF has two different resource types that correspond to a control unit:

- A subsystem identifier (SSID) identifies the control unit as a hardware object.
- A logical control unit (LCU) identifies it to the operating system.

HCM lets you specify which aspect of a control unit to view, as shown in Figure 24-57. If no SSID is specified for the control unit, this dialog is omitted.



Control Unit: D500 SSID: D500

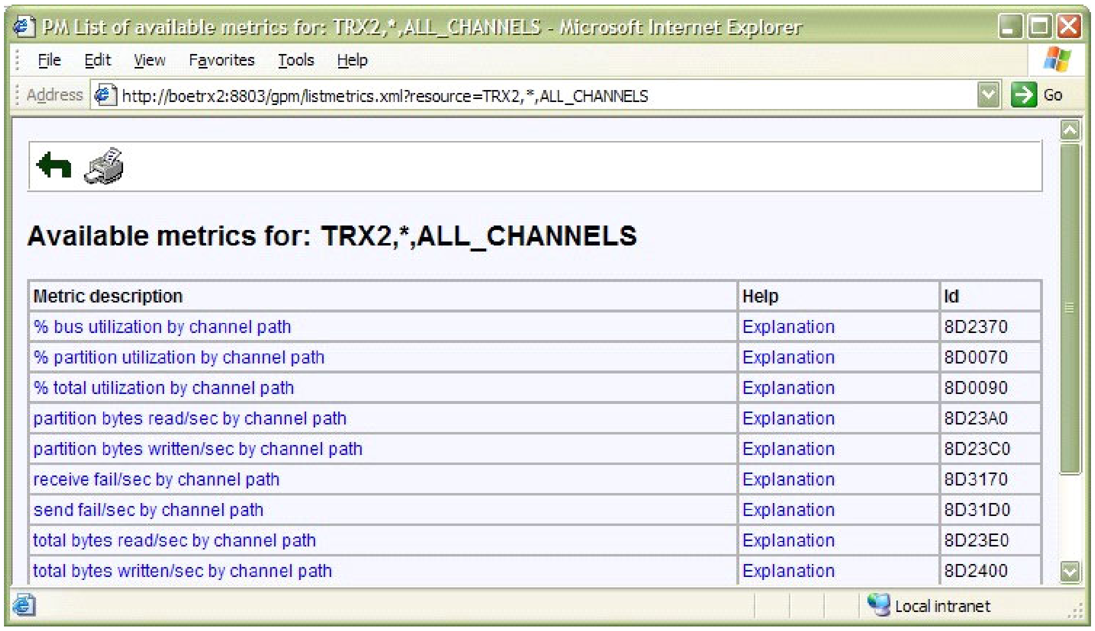
The selected control unit corresponds to an RMF LCU and SSID. Which kind of performance information do you like to browse?

[✓ Logical Control Unit (LCU)]
Subsystem ID (SSID)

Figure 24-57 The RMF resource type option

The system Web browser is launched with a list of performance metrics of the selected object.

Look at performance data by clicking the desired metric, as shown in Figure 24-58.


PM List of available metrics for: TRX2,*,ALL_CHANNELS - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address
Go
Local intranet

http://boetrx2:8803/gpm/listmetrics.xml?resource=TRX2,*,ALL_CHANNELS

Available metrics for: TRX2,*,ALL_CHANNELS

Metric description	Help	Id
% bus utilization by channel path	Explanation	8D2370
% partition utilization by channel path	Explanation	8D0070
% total utilization by channel path	Explanation	8D0090
partition bytes read/sec by channel path	Explanation	8D23A0
partition bytes written/sec by channel path	Explanation	8D23C0
receive fail/sec by channel path	Explanation	8D3170
send fail/sec by channel path	Explanation	8D31D0
total bytes read/sec by channel path	Explanation	8D23E0
total bytes written/sec by channel path	Explanation	8D2400

Figure 24-58 Available metrics in a browser

24.16.2 Interactions and dependencies

Software dependencies

The z/OS V1R8 RMF Distributed Data Server (DDS) has to be installed, set up, and running in the sysplex.

Performance data is only available for the active sysplex. It is recommended to focus the HCM diagram to the active sysplex by checking the Active Sysplex check box in the Change View Attributes dialog.

DDS makes use of WLM naming services, beginning with z/OS V1R8. This feature is required by this line item.

DDS uses XML documents with XSL stylesheets to display performance data in the Web browser. The Web browser must be able to render such documents. All major Web browsers support this (for example, Microsoft Internet Explorer 5.5 or above, Mozilla 1.4 or above, Firefox 1.0 or above, Netscape 7.0 or above).

Performance Data Integration is not available for HCM used with z/VM.

24.16.3 Installation

With z/OS V1R8 HCM.

Install HCD SPE APAR OA14334.

24.17 Compare Reports for IODFs

z/OS V1R8 HCM provides a dialog that allows generating and viewing the HCD Compare Reports on the workstation, as follows:

- From now on HCD Compare Reports do not have to be generated outside of HCM. They can be viewed immediately without the need to access a data set.
- In addition, z/OS V1R8 HCM now adds access to the HCD IODF Compare Reports, which show differences between two IODFs.
- Also, access to the HCD Compare CSS/OS Reports was added: These reports work on a single IODF, comparing the CSS view and the OS view on the defined devices.

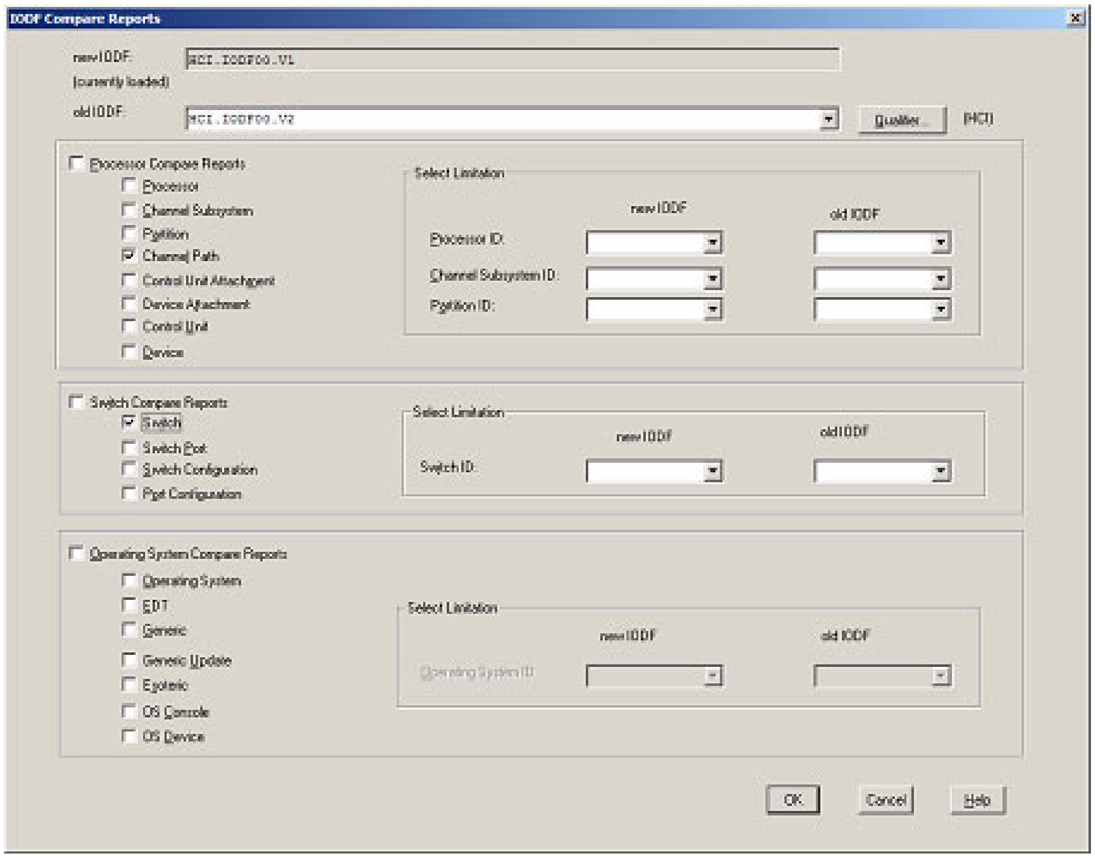
24.17.1 Implementation of IODF Compare Reports

The function View IODF Compare Reports is invoked from the HCM File menu. The menu item is disabled when running HCM in standalone mode (or when no configuration is loaded).

HCM offers this function via a new dialog, shown in Figure 24-59 on page 484. The same report types (Processor Compare Reports, Switch Compare Reports, and OS Compare Reports) and report limitations (for example, specific processor or operating system) as under HCD can be selected from HCM.

The request is sent to HCD. HCD generates the report and provides the output back to HCM where it is displayed in a separate window, as shown in Figure 24-60 on page 484. The report has the same layout as under HCD. It can be saved in a file on the workstation.

HCM does not validate the report input parameters or the report output. All validity checks are done by HCD. Any error messages from HCD are displayed in the Message List window of HCM.



IODF Compare Reports

new IODF: (currently loaded) HCI.IODF00.V1

old IODF: HCI.IODF00.V2 Qualifier... (HCI)

- ☐ Processor Compare Reports
  - ☐ Processor
  - ☐ Channel Subsystem
  - ☐ Partition
  - ☑ Channel Path
  - ☐ Control Unit Attachment
  - ☐ Device Attachment
  - ☐ Control Unit
  - ☐ Device

Select Limitation (new IODF / old IODF): Processor ID: / Channel Subsystem ID: / Partition ID:

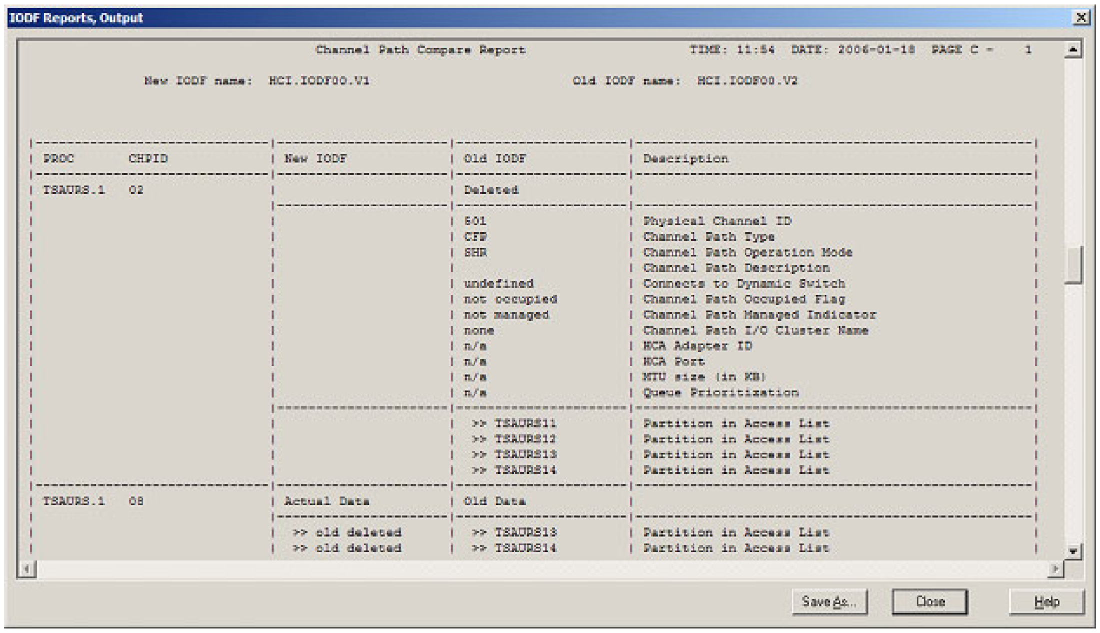
- ☐ Switch Compare Reports
  - ☑ Switch
  - ☐ Switch Port
  - ☐ Switch Configuration
  - ☐ Port Configuration

Select Limitation (new IODF / old IODF): Switch ID:

- ☐ Operating System Compare Reports
  - ☐ Operating System
  - ☐ EDT
  - ☐ Generic
  - ☐ Generic Update
  - ☐ Esoteric
  - ☐ OS Console
  - ☐ OS Device

Select Limitation (new IODF / old IODF): Operating System ID:

Figure 24-59 The IODF Compare Report page



IODF Reports, Output

```
                                Channel Path Compare Report                 TIME: 11:54  DATE: 2006-01-18  PAGE C -    1

                 New IODF name:  HCI.IODF00.V1                        Old IODF name:  HCI.IODF00.V2


|------------------------------|-----------------------|-----------------------|------------------------------------------|
| PROC        CHPID            | New IODF              | Old IODF              | Description                              |
|------------------------------|-----------------------|-----------------------|------------------------------------------|
| TSAURS.1    02               |                       | Deleted               |                                          |
|                              |-----------------------|-----------------------|------------------------------------------|
|                              |                       | 501                   | Physical Channel ID                      |
|                              |                       | CFP                   | Channel Path Type                        |
|                              |                       | SHR                   | Channel Path Operation Mode              |
|                              |                       |                       | Channel Path Description                 |
|                              |                       | undefined             | Connects to Dynamic Switch               |
|                              |                       | not occupied          | Channel Path Occupied Flag               |
|                              |                       | not managed           | Channel Path Managed Indicator           |
|                              |                       | none                  | Channel Path I/O Cluster Name            |
|                              |                       | n/a                   | HCA Adapter ID                           |
|                              |                       | n/a                   | HCA Port                                 |
|                              |                       | n/a                   | MTU size (in KB)                         |
|                              |                       | n/a                   | Queue Prioritization                     |
|                              |-----------------------|-----------------------|------------------------------------------|
|                              |                       |  >> TSAURS11          | Partition in Access List                 |
|                              |                       |  >> TSAURS12          | Partition in Access List                 |
|                              |                       |  >> TSAURS13          | Partition in Access List                 |
|                              |                       |  >> TSAURS14          | Partition in Access List                 |
|------------------------------|-----------------------|-----------------------|------------------------------------------|
| TSAURS.1    08               | Actual Data           | Old Data              |                                          |
|                              |-----------------------|-----------------------|------------------------------------------|
|                              |  >> old deleted       |  >> TSAURS13          | Partition in Access List                 |
|                              |  >> old deleted       |  >> TSAURS14          | Partition in Access List                 |
```

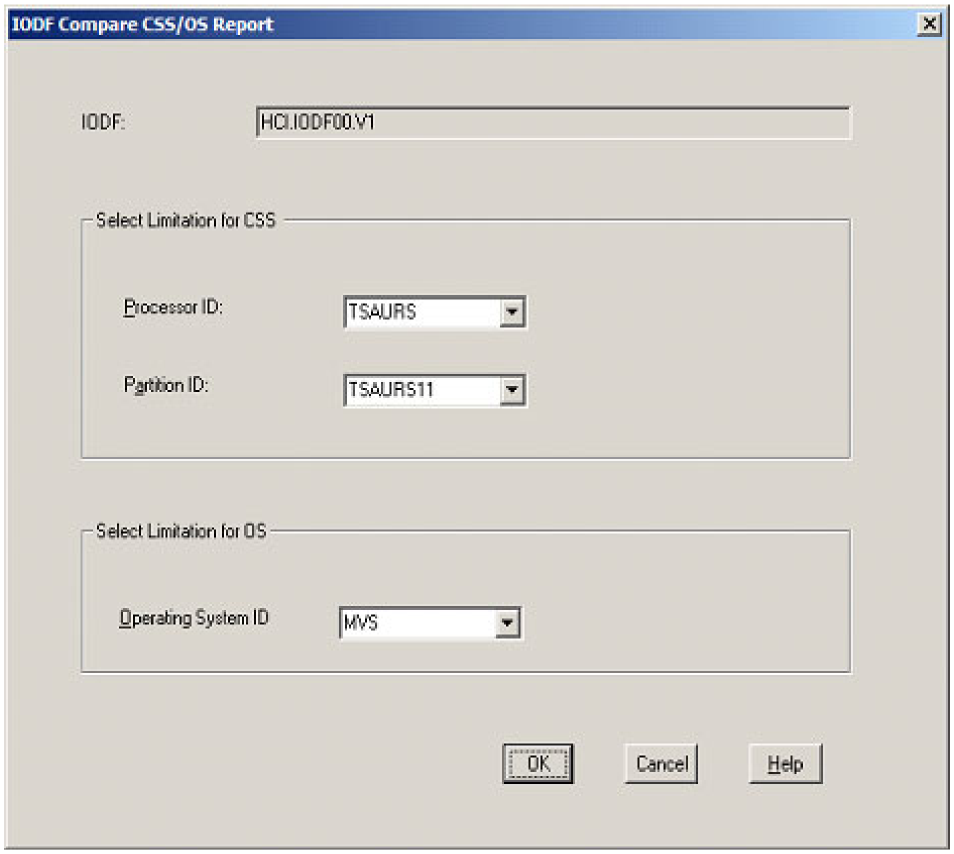
Figure 24-60 The IODF Compare Report output

The function View IODF Compare CSS/OS Reports is invoked from the HCM File menu. The menu item is disabled when running HCM in standalone mode (or when no configuration is loaded).

HCM offers this function via a new dialog, shown in Figure 24-61.

The request is sent to HCD. HCD generates the report and provides the output back to HCM where it is displayed in a separate window, as shown in Figure 24-62 on page 486. The report has the same layout as under HCD. It can be saved in a file on the workstation.

HCM does not validate the report input parameters or the report output. All validity checks are done by HCD. Any error messages from HCD are displayed in the Message List window of HCM.



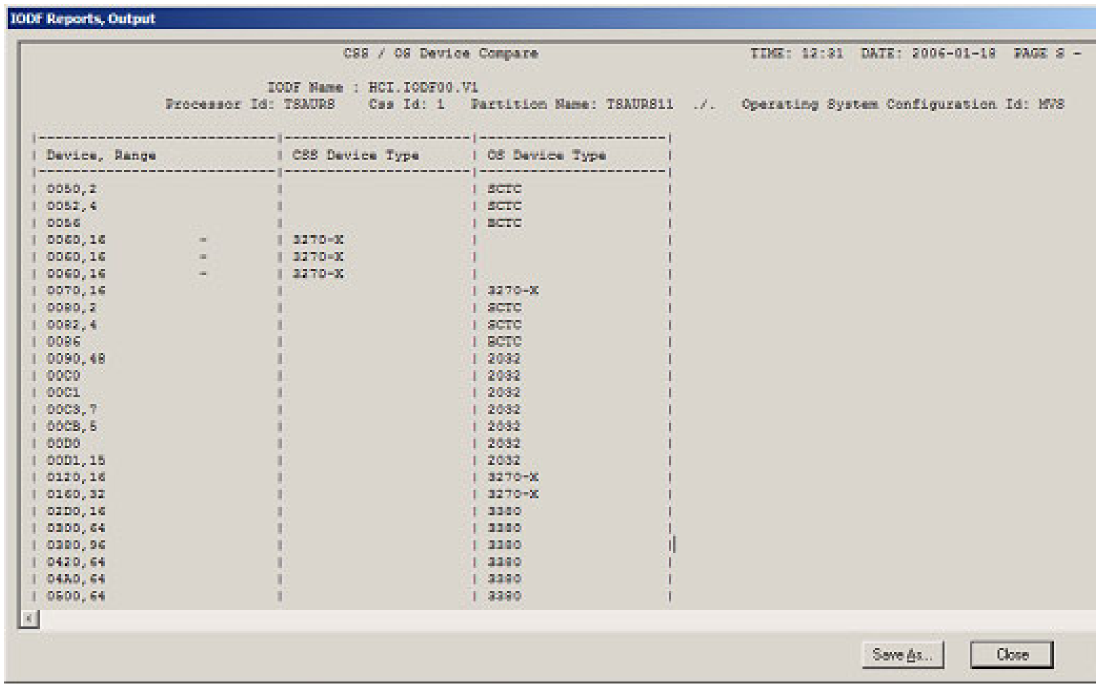
IODF: HCI.IODF00.V1

Processor ID: TSAURS

Partition ID: TSAURS11

Operating System ID MVS

Figure 24-61 The IODF Compare CSS/OS Report page



```
                          CSS / OS Device Compare                     TIME: 12:31  DATE: 2006-01-18  PAGE S -

                     IODF Name : HCI.IODF00.V1
            Processor Id: TSAURS    Css Id: 1   Partition Name: TSAUR811  ./.   Operating System Configuration Id: MVS

 |-----------------------------|----------------------|----------------------|
 | Device, Range               | CSS Device Type      | OS Device Type       |
 |-----------------------------|----------------------|----------------------|
 | 0050,2                      |                      | SCTC                 |
 | 0052,4                      |                      | SCTC                 |
 | 0056                        |                      | BCTC                 |
 | 0060,16           -         | 3270-X               |                      |
 | 0060,16           -         | 3270-X               |                      |
 | 0060,16           -         | 3270-X               |                      |
 | 0070,16                     |                      | 3270-X               |
 | 0080,2                      |                      | SCTC                 |
 | 0082,4                      |                      | SCTC                 |
 | 0086                        |                      | BCTC                 |
 | 0090,48                     |                      | 2032                 |
 | 00C0                        |                      | 2032                 |
 | 00C1                        |                      | 2032                 |
 | 00C3,7                      |                      | 2032                 |
 | 00CB,5                      |                      | 2032                 |
 | 00D0                        |                      | 2032                 |
 | 00D1,15                     |                      | 2032                 |
 | 0120,16                     |                      | 3270-X               |
 | 0160,32                     |                      | 3270-X               |
 | 02D0,16                     |                      | 3380                 |
 | 0300,64                     |                      | 3380                 |
 | 0380,96                     |                      | 3380                 |
 | 0420,64                     |                      | 3380                 |
 | 04A0,64                     |                      | 3380                 |
 | 0600,64                     |                      | 3380                 |
```

Figure 24-62 The IODF Compare CSS/OS Report output

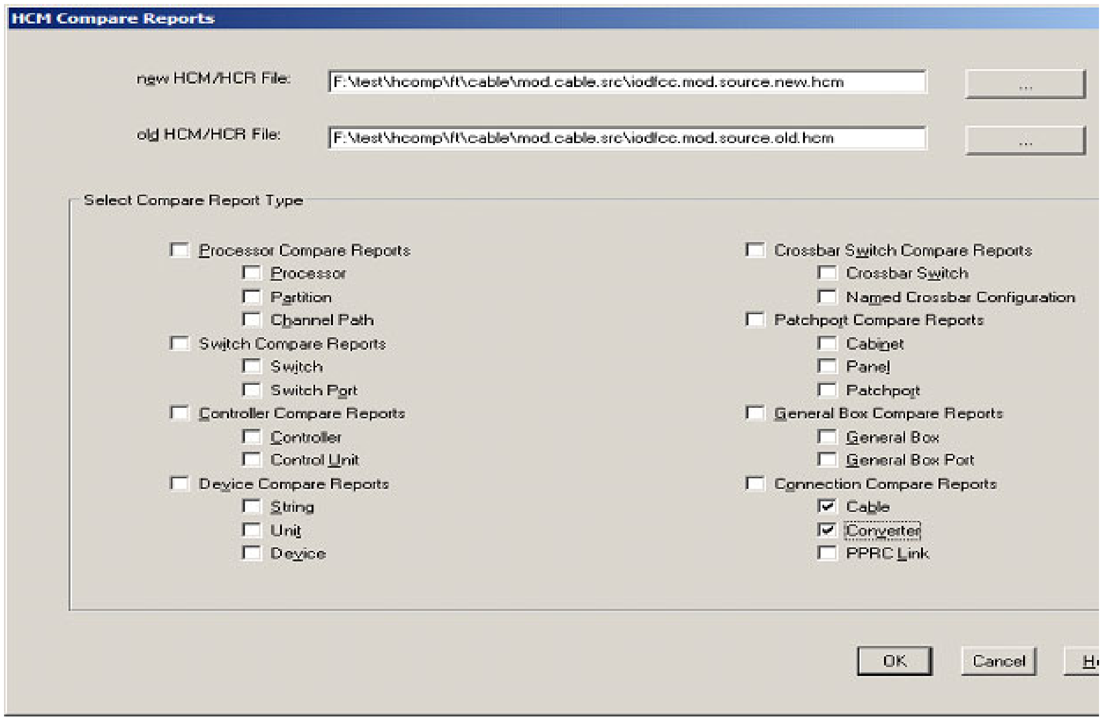
24.18 Compare Reports for HCM configuration files

z/OS V1R8 HCM can compare physical aspects of configurations (like controllers, cables, patchports, etc.) along with the logical configuration objects. Comparison for almost all HCM-native objects is now available. Twenty different report types or objects can be selected by groups or individually.

Output options are available similar to those for HCD Compare Reports (show added, deleted, and modified objects, including or excluding unchanged attributes).

The output is shown in a separate window and can be saved to a text file; see Figure 24-64 on page 487.

The comparison works on HCM-native objects, so no access to the IODFs is needed, allowing reports to be generated very quickly in standalone mode.


HCM Compare Reports

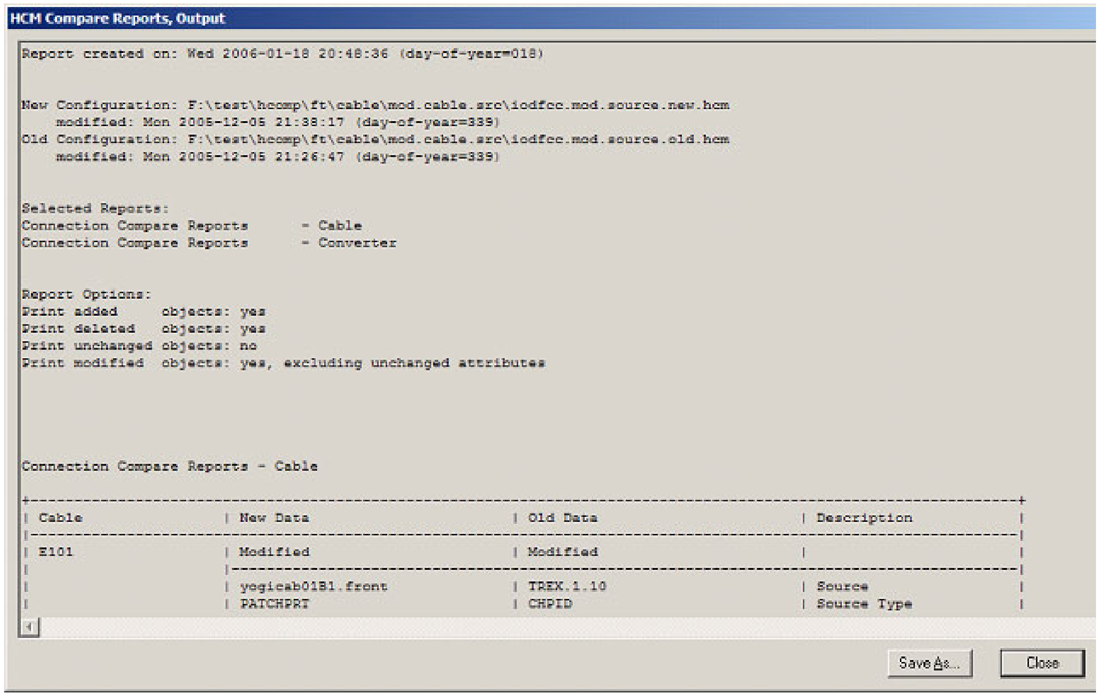
new HCM/HCR File: F:\test\hcomp\ft\cable\mod.cable.src\iodfcc.mod.source.new.hcm

old HCM/HCR File: F:\test\hcomp\ft\cable\mod.cable.src\iodfcc.mod.source.old.hcm

Select Compare Report Type

- ☐ Processor Compare Reports
  - ☐ Processor
  - ☐ Partition
  - ☐ Channel Path
- ☐ Switch Compare Reports
  - ☐ Switch
  - ☐ Switch Port
- ☐ Controller Compare Reports
  - ☐ Controller
  - ☐ Control Unit
- ☐ Device Compare Reports
  - ☐ String
  - ☐ Unit
  - ☐ Device
- ☐ Crossbar Switch Compare Reports
  - ☐ Crossbar Switch
  - ☐ Named Crossbar Configuration
- ☐ Patchport Compare Reports
  - ☐ Cabinet
  - ☐ Panel
  - ☐ Patchport
- ☐ General Box Compare Reports
  - ☐ General Box
  - ☐ General Box Port
- ☐ Connection Compare Reports
  - [✓ Cable]
  - [✓ Converter]
  - ☐ PPRC Link

Figure 24-63   The HCM Compare Reports page


HCM Compare Reports, Output

```
Report created on: Wed 2006-01-18 20:48:36 (day-of-year=018)


New Configuration: F:\test\hcomp\ft\cable\mod.cable.src\iodfcc.mod.source.new.hcm
    modified: Mon 2005-12-05 21:38:17 (day-of-year=339)
Old Configuration: F:\test\hcomp\ft\cable\mod.cable.src\iodfcc.mod.source.old.hcm
    modified: Mon 2005-12-05 21:26:47 (day-of-year=339)


Selected Reports:
Connection Compare Reports      - Cable
Connection Compare Reports      - Converter


Report Options:
Print added     objects: yes
Print deleted   objects: yes
Print unchanged objects: no
Print modified  objects: yes, excluding unchanged attributes




Connection Compare Reports - Cable

+-------------------------------------------------------------------------------------------------------------------+
| Cable                 | New Data                         | Old Data                         | Description             |
|-------------------------------------------------------------------------------------------------------------------|
| E101                  | Modified                         | Modified                         |                         |
|                       |---------------------------------------------------------------------------------------------|
|                       | yogicab01B1.front                | TREX.1.10                        | Source                  |
|                       | PATCHPRT                         | CHPID                            | Source Type             |
```

Figure 24-64   The HCM Compare Report output

24.18.1 Interactions and dependencies

View HCD Compare Reports requires HCD SPE APAR OA14334.

24.19 Restrictions

24.19.1 Label change for objects

Starting with z/OS 1.8 HCM, you can no longer create controllers, crossbar switches. and strings with a label containing a "." (period). Also, if you edit existing objects of these types containing a period in their labels, HCM enforces a label change by issuing a corresponding message.

The reason for this behavior is that objects with a period in their labels cannot be compared in reports and cannot be exported.

24.19.2 Change in exported data

Starting with z/OS 1.8 HCM, the Export Data function is modified to no longer export controllers for switch-internal control units to the CONTRLLR.TXT table. Also, switch-internal control units are no longer exported to the CU.TXT table.

The reason for this behavior is that controllers for switch-internal control units are volatile objects that are deleted and created with each loading of the configuration.

25

Unicode Services with z/OS V1R8

z/OS V1R8 supports the new Unicode standards for integrating new applications on z/OS. These new applications not only include IBM products, but also vendor packages. All this functionality enriches the globalization aspects of z/OS.

In addition, Unicode on Demand, which was introduced in z/OS V1R7, is also enhanced.

The following topics are described in this chapter:

- Pause/Release and page fix support
- Upgrade Unicode Services to Unicode 4.0
- BiDi and character shaping service
- Unicode system services for stringprep
- Collation® requirements for ERP

25.1 Pause/release and page fix support

In z/OS V1R7, Unicode on Demand introduced support for the dynamic loading of tables when running in TCB mode. Environments that run in SRB mode cannot dynamically add tables to storage, as described here:

- Unicode Services uses WAIT/POST macros to schedule the loading of tables. WAIT/POST cannot be used in SRB mode, such as when executing DB2 queries. Thus, DB2 cannot take advantage of dynamic loading of tables and Unicode Services returns RC=8, RS=3 when tables are not in storage. In addition, all tables are page fixed, thereby occupying real storage.
- The WAIT/POST macros have been replaced with the pause/release mechanism. The only difference between these services is that pause/release can be invoked when running in either TCB or SRB modes. To the existing end user, running in TCB mode, the behavior of Unicode on Demand will look the same. However, when callers are running in SRB mode and a table needs to be dynamically loaded into storage, they will no longer encounter a failure message.

z/OS V1R8 enhancement

z/OS V1R8 changed the internal services, to allow adding tables when running both in TCB mode and SRB mode. The changes provide an option to page fix the tables when loaded into storage, as follows:

- Now all tables are loaded in non-page fixed storage. However, a keyword is provided on the ADD, REPLACE and IMAGE statements and also on all Unicode callable services to allow the option of page fixing. Only page fixed tables are governed by the threshold of the REALSTORAGE parameter of the CUNUNIxx parmlib member
- The restriction for Unicode on Demand is thereby removed. Now you can dynamically add tables to storage when running in any type of environment. This reduces the use of real storage. And given that you no longer page fix our tables in storage, the tables are now eligible for paging.

25.1.1 Implementation of page fixing tables

An enhanced CUNUNIxx parmlib member and a new operator command, SETUNI, are provided to establish and update the environment by adding, deleting, and replacing tables. The new Unicode Services interfaces provided starting in z/OS V1R7 are an expanded CUNUNIxx parmlib member and a SETUNI operator command that accomplish the same function as the parmlib member. With either of these interfaces you can do the following:

- Add, replace, or delete tables in a conversion image, specifying the From-CCSID, the To-CCSID, and optionally, the techniques required.
- Add, replace, or delete case conversion tables.
- Add, replace, or delete normalization tables.
- Add, replace, or delete collation tables.
- Add, replace, or delete Stringprep profiles.
- Add an image, without requiring that it be in the parmlib concatenation.

Operator command examples

Figure 25-1 on page 491 shows examples of the SETUNI ADD commands.

```
SETUNI ADD,FROM=xxxx,TO=yyyy [,TECHNIQUE=] [,DSNAME=] [,VOLSER=] [,PAGEFIX=YES
| NO]
SETUNI ADD,CASE [=LOCAL | [,SPECIAL] | [,NORMAL] ] [,DSNAME=] [,VOLSER=]
[,PAGEFIX=YES | NO]
SETUNI ADD,NORMALIZE [,DSNAME=] [,VOLSER=] [,PAGEFIX=YES | NO]
SETUNI ADD,COLLATE [,DSNAME=] [,VOLSER=] [,PAGEFIX=YES | NO]
SETUNI ADD,IMAGE=zzzzzzzz [,DSNAME=] [,VOLSER=] [,PAGEFIX=YES | NO]
```

Figure 25-1 SETUNI ADD commands

Figure 25-2 on page 491 shows examples of the **SETUNI REPLACE** commands.

```
SETUNI REPLACE,FROM=xxxx,TO=yyyy[,TECHNIQUE=] [,DSNAME=] [,VOLSER=] [,FREE=NO |
(YES,FORCE)] [,PAGEFIX=YES | NO]
SETUNI REPLACE,CASE [=LOCAL | [,SPECIAL] | [,NORMAL] ] [,DSNAME=] [,VOLSER=]
[,FREE=NO | (YES,FORCE)] [,PAGEFIX=YES | NO]
SETUNI REPLACE,NORMALIZE [,DSNAME=] [,VOLSER=] [,FREE=NO | (YES,FORCE)]
[,PAGEFIX=YES | NO]
SETUNI REPLACE,COLLATE [,DSNAME=] [,VOLSER=] [,FREE=NO | (YES,FORCE)]
[,PAGEFIX=YES | NO]
```

Figure 25-2 SETUNI REPLACE commands

CUNUNIxx parmlib member

There are new operands in the CUNUNIxx parmlib member, as shown in Figure 25-3 on page 491.

```
ADD FROM(xxxx) TO(yyyy) [TECHNIQUE(…..)] [ DSNAME(….)] [VOLSER(…)] [PAGEFIX(YES
| NO)]
ADD CASE [(LOCAL | [ SPECIAL] | [ NORMAL]) ] [ DSNAME(….)] [VOLSER(….)]
[PAGEFIX(YES | NO)]
ADD NORMALIZE [ DSNAME(….)] [ VOLSER(….)] [PAGEFIX(YES | NO)]
ADD COLLATE [ DSNAME(….)] [ VOLSER(….)] [PAGEFIX(YES | NO)]
ADD IMAGE(zzzzzzzz) [ DSNAME(….)] [ VOLSER(….)] [PAGEFIX(YES | NO)]
REPLACE FROM(xxxx) TO(yyyy) [TECHNIQUE(…..)] [DSNAME(….)] [VOLSER(….)] [FREE(NO
| YES,FORCE)] [PAGEFIX(YES | NO)]
REPLACE CASE [(LOCAL | [ SPECIAL] | [ NORMAL]) ] [DSNAME(….)] [VOLSER(….)]
[FREE(NO | YES,FORCE)] [PAGEFIX(YES | NO)]
REPLACE NORMALIZE [DSNAME(….)] [VOLSER(….)] [FREE(NO | YES,FORCE)] [PAGEFIX(YES
| NO)]
REPLACE COLLATE [DSNAME(….)] [VOLSER(….)] [FREE(NO | YES,FORCE)] [PAGEFIX(YES |
NO)]
IMAGE [(]member[)] [PAGEFIX(YES | NO)] [;]
```

Figure 25-3 CUNUNIxx parmlib member examples

25.1.2 Migration considerations

Pause/release support is being rolled back to z/OS V1R7 via APAR OA14231. However, the page fix support will not be rolled back. Existing CUNUNIxx parmlib members will continue to work.

This new functionality allows you to remove the UNI=xx statement from the IEASYSxx parmlib member and then allow Unicode on Demand load the necessary tables into storage.

25.2 Upgrade Unicode Services to Unicode 4.0

In the previous versions, the support in Unicode Services for normalization is based on the Unicode 3.0.1 standard, and the later Unicode standards provide support for additional sets of characters like surrogates that are not recognized during normalization.

Normalization is a process of removing alternate representations of equivalent sequences from textual data, to convert the data into a form that can be binary-compared for equivalence. In the Unicode standard, normalization refers specifically to processing to ensure that canonical-equivalent (and/or compatibility-equivalent) strings have unique representations.

z/OS V1R8 provides support for later Unicode standards in normalization service, as follows:

- Unicode 3.2.0
- Unicode 4.0.1
- Unicode 4.1.0

Because applications are required to process and manipulate data beyond the customer's local codeset, support such as this will be necessary. As part of collating strings, the characters may need to be normalized. This will be done automatically. Unicode Services exploits z/Architecture when using normalization tables for improved performance.

25.2.1 Normalization implementation

For normalization conversion, use the ADD, REPLACE, or DELETE NORMALIZATION statement. The normalization versions that can be specified are:

- UNI301 - default
- UNI320
- UNI401
- UNI410

Operator commands for normalization

Using the operator commands, the term `normver` specifies the Unicode standard table version, as shown in Figure 25-4 on page 492.

```
SETUNI ADD,NORMALIZE [=normver] [,DSNAME=] [,VOLSER=] [,PAGEFIX=YES | NO]
SETUNI REPLACE,NORMALIZE [=normver] [,DSNAME=] [,VOLSER=] [,FREE=NO |
(YES,FORCE)] [,PAGEFIX=YES | NO]
SETUNI DELETE,NORMALIZE [=normver] [,FREE=NO | (YES,FORCE)]
```

Figure 25-4 Operator commands to normalize tables

CUNUNIxx parmlib member

There are new value specifications in the CUNUNIxx parmlib member for normalization conversion, as shown in Figure 25-5 on page 493.

Where normver - Specifies the Unicode standard table version and possible values are:

- UNI301 - default
- UNI320
- UNI401
- UNI410

```
ADD NORMALIZE[(normver)] [DSNAME(….)] [VOLSER(….)] [PAGEFIX(YES | NO)]
REPLACE NORMALIZE[(normver)] [DSNAME(….)] [VOLSER(….)] [FREE(NO | YES,FORCE)]
[PAGEFIX(YES | NO)]
DELETE NORMALIZE[(normver)] [FREE(NO | YES,FORCE)]
```

Figure 25-5 CUNUNIxx parmlib member specifications for normalization

Implementation considerations

The following callable services are required in HLASM and C/C++ to exploit this function. In order to exploit this new behavior, a new version of the parameter area is required (version 2). The tables will be loaded dynamically into storage (Unicode on Demand) if not already there.

- CUNLNORM for 31-bit callers

  The parameter CUNBNPRM_Version (31-bit) is to be set by the caller and must be set with CUNBNPRM_Ver2 or CUN4BNPR_Ver2 (value of 2) in order to load any of the new tables. The default value will be CUNBNPRM_Ver.

- CUN4LNOR for 64-bit callers

  The parameter CUN4BNPR_Version (64-bit) is to be set by the caller.

  - The default value will be CUN4BNPR_Ver (value of 1) and will use the UNI301 table (for upward compatibility).

When CUNBNPRM_UniVersion (31-bit) or CUN4BNPR_UniVersion (64-bit) is set by the caller, this specifies the Unicode Normalization Data version, and possible values are:

- CUNBNPRM_NONE or CUN4BPR_NONE (DEFAULT value), means that Unicode 3.0.1 is to be used
- CUNBNPRM_UNI301 or CUN4BNPR_UNI301
- CUNBNPRM_UNI320 or CUN4BNPR_UNI320
- CUNBNPRM_UNI401 or CUN4BNPR_UNI401
- CUNBNPRM_UNI410 or CUN4BNPR_UNI410

Figure 25-6 on page 494 shows the DISPLAY output (D UNI,ALL).

```
D UNI,ALL
CUN3000I 02.57.05 UNI DISPLAY 581
 ENVIRONMENT: CREATED       03/20/2006 AT 02.53.49
               MODIFIED      03/20/2006 AT 02.56.46
             IMAGE CREATED --/--/---- AT --.--.–
    SERVICE: CHARACTER      CASE          NORMALIZATION  COLLATION
             STRINGPREP     BIDI
    STORAGE: ACTIVE       1026 PAGES
             LIMIT      524287 PAGES
   CASECONV: NONE
  NORMALIZE: ENABLED
   NORM VER: UNI401
    COLLATE: ENABLED
 COLL RULES: UCA400R1_CUNENUS
STRPROFILES: NONE
 CONVERSION: NO CONVERSIONS FOUND
```

Figure 25-6 DISPLAY output (D UNI,ALL)

Note: Existing CUNUNIxx parmlib members will continue to work. Existing applications using the Unicode normalization service will also continue to work.

25.3 BiDi and character shaping service

The Unicode standard prescribes a memory representation order known as *logical order*. When text is presented in horizontal lines, most scripts display characters from left to right. However, there are several scripts (such as Arabic or Hebrew) where the natural ordering of horizontal text in display is from right to left. If all of the text has the same horizontal direction, then the ordering of the display text is unambiguous. However, when bidirectional text (a mixture of left-to-right and right-to-left horizontal text) is present, some ambiguities can arise in determining the ordering of the displayed characters.

z/OS V1R8 Unicode Services provides a set of APIs that allows the transformation and the display order of strings such as Arabic or Hebrew where the natural ordering of horizontal text in display is from right to left, as shown in Figure 25-7 on page 494.

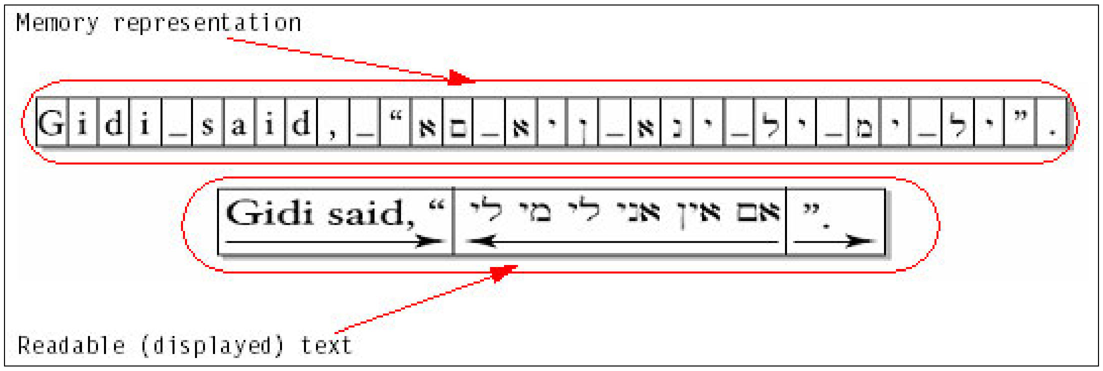

Memory representation
G i d i _ s a i d , _ “ א ם _ א י ן _ א נ י _ ל י _ מ י _ ל י ” .
Gidi said, “ אם אין אני לי מי לי ”.
Readable (displayed) text

Figure 25-7 The character shaping service

The Unicode standard defines the conversion of Unicode text from the memory representation to readable (displayed) text for example, between logical order and display order).

The Bidi transformation service is based on the Unicode standard Annex #9:

```
http://www.unicode.org/reports/tr9
```

This service allows the transformation and conversion of bidirectional strings.

25.3.1 Implementation considerations

Following are the new callable services in HLASM or C/C++:

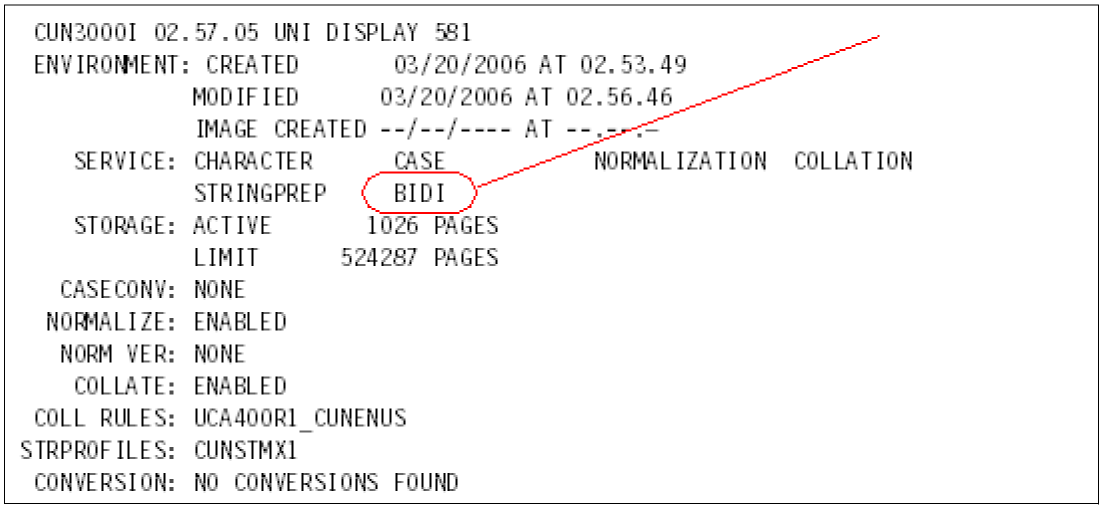
- CUNLBIDI for 31-bit callers
- CUN4LBID for 64-bit callers

In the parameter list, prefix the following parameters with CUNBBPRM_ (31-bit) or CUN4BBPR_ (64-bit) and use the following keywords:

- Version - set by caller.
  - Specifies the version of the parameter area. This field must be initialized for the first call to the stub routine CUNLBIDI/CUN4LBID using the constant CUNBBPRM_Ver/CUN4BBPR_Ver which is supplied by the interface definition file CUNBPIDF.
- Length - set by caller.
  - It is the length of the parameter area. HLASM users must initialize this field for the first call to CUNLBIDI/CUN4LBID using the constant CUNBBPRM_Len/CUN4BBPR_Len which is supplied by the interface definition file CUNBPIDF.
- Src_Buf_Ptr - set by caller, updated by service.
- Specifies the beginning address of a string of text characters with a length specified in the Src_Buf_Len parameter.
- Src_Buf_ALET - set by caller
  - Specifies the ALET to be used to access the source buffer addressed by Src_Buf_Ptr. Use an ALET value of 0 to designate the primary address space.
- Src_Buf_Len - set by caller, updated by service
  - Specifies the length in bytes of the source buffer addressed by Src_Buf_Ptr.
- Targ_Buf_Ptr - set by caller.
  - Specifies the beginning address of an area of storage to be used to store the final string layout.
- Targ_Buf_ALET - set by caller.
  - Specifies the ALET to be used to access the target buffer addressed by Targ_Buf_Ptr. Use an ALET value of 0 to designate the primary address space.
- Targ_Buf_Len - set by caller, updated by service.
  - Specifies the length in bytes of the target buffer addressed by Targ_Buf_Ptr. It should be at least the same size of Src_Buf_Len.
- Wrk_Buf_Ptr - set by caller, used by service for conversion purposes.
  - Specifies the beginning address of an area of storage that the Stringprep service can use to store immediate results.

- Wrk_Buf_ALET - set by caller
  - Specifies the ALET to be used to access the work buffer addressed by Wrk_Buf_Ptr. Use an ALET value of 0 to designate the primary address space.
- Wrk_Buf_Len - set by caller, updated by service.
  - Specifies the length in bytes of the work buffer addressed by Wrk_Buf_Ptr. It should be at least the same size of the Src_Buf_Len.
- CCSID_Src – set by caller.
  - Specifies the CCSID of the source.
- CCSID_Trg – set by caller.
  - Specifies the CCSID of the target.
- Flag1 – set by caller.
  - Bit position Name
  - 1XXX XXXX Bidi_context
  - X1XX XXXX Bidi_impalg
  - Bidi_Context – Specifies whether the context is left to right or right to left.
    - 0: Context left-to-right
    - 1: Context right-to-left
  - Bidi_impalg – Specifies whether the algorithm will transform, depending on each character's characteristic.
    - 0: Basic algorithm
    - 1: Implicit algorithm
  - Return_Code - set by service,
    - Specifies the return code.
  - Reason_Code - set by service,
    - Specifies the reason code

Figure 25-8 on page 496 shows BIDI in command response to a D UNI,ALL command.



```
 CUN3000I 02.57.05 UNI DISPLAY 581
 ENVIRONMENT: CREATED       03/20/2006 AT 02.53.49
              MODIFIED      03/20/2006 AT 02.56.46
              IMAGE CREATED --/--/---- AT --.--.--
    SERVICE: CHARACTER      CASE           NORMALIZATION  COLLATION
             STRINGPREP     BIDI
    STORAGE: ACTIVE       1026 PAGES
             LIMIT      524287 PAGES
   CASECONV: NONE
  NORMALIZE: ENABLED
   NORM VER: NONE
    COLLATE: ENABLED
 COLL RULES: UCA400R1_CUNENUS
STRPROFILES: CUNSTMX1
 CONVERSION: NO CONVERSIONS FOUND
```

Figure 25-8 Command shows BIDI in the D UNI,ALL output

25.4 Unicode system services for stringprep

Application programs can display text in many different ways. Similarly, a user can enter text into an application program in a myriad of fashions. Internationalized text (that is, text that is not restricted to the narrow set of US-ASCII characters) has many input and display behaviors that make it difficult to compare text in a consistent fashion.

The z/OS V1R8 Unicode Services provides a set of APIs that will help you apply some processing rules for Unicode text using profiles of these rules.

Note: The need for this support originated from NFS V4 protocol, which is based on the Internet Engineering Task Force (IETF) and Internet Engineering Steering Group (IESG) Request for Comments (RFC) 3454. The preparation of internationalized strings specifies that Unicode strings need to be prepared (“string preped”) before being passed on the network. A profile of string prep converts a single string of input characters to a string of output characters, or returns an error if the output string would contain a prohibited character. Stringprep profiles cannot both emit a string and return an error.

These profiles allow you to enter internationalized text strings in applications and have the highest chance of getting the content of the strings correct. In this case, “correct” means that if two different users enter what they think is the same string into two different input mechanisms, the strings should match on a character-by-character basis.

Steps for preparing strings

The steps for preparing strings are as following:

- Map - For each character in the input, check if it has a mapping and, if so, replace it with its mapping.
- Normalize - Possibly normalize the result of step 1 using Unicode normalization with form KC. (Normalization Version 3.2)
- Prohibit - Check for any characters that are not allowed in the output. If any are found, return an error.
- Check bidi - Possibly check for right-to-left characters, and if any are found, make sure that the whole string satisfies the requirements for bidirectional strings.

The intention of string prep is to define tables and have the profiles of string prep select among those defined tables. The character repertoire that is used in the input and output to string prep is based on Unicode 3.2.

Note that the string prep is based on Unicode 3.2 and should not be considered to automatically apply to later Unicode versions. The IETF, through an explicit standards action, may update this document as appropriate to handle later Unicode versions.

The z/OS Unicode Services provides four profiles that can be used by customers to prepare their strings.

25.4.1 Implementation considerations

The table CUNSTCSP is used by NFS for UNIX System Services path and filenames, as shown in Figure 25-9 on page 498.

```
Case: Mixed Case
Map Table:B.1 Commonly mapped to nothing
Normalization:No
Prohibited tablesC.3 Private use
           C.4 Non-character code points
           C.5 Surrogate Codes
           C.6 Inappropriate for plain text
           C.7 Inappropriate for canonical representation
           C.8 Change display properties or are deprecated
           C.9 Tagging characters
Bidi  No
```

Figure 25-9 Table CUNSTCSP

The table CUNSTCIS is used by NFS for MVS Data Set names, as shown in Figure 25-10 on page 498.

```
Case: Insensitive
Map Table:B.1 Commonly mapped to nothing
Normalization:KC
Prohibited tablesC.2.1 ASCII control charactersC.2.2 Non-ASCII control
characters
           C.3 Private use
           C.4 Non-character code points
           C.5 Surrogate Codes
           C.6 Inappropriate for plain text
           C.7 Inappropriate for canonical representation
           C.8 Change display properties or are deprecated
           C.9 Tagging characters
Bidi  Yes
```

Figure 25-10 Table CUNSTCIS

The table CUNSTMX1is used by NFS for user name in name@domain, as shown in Figure 25-11 on page 498.

```
Case: Mixed Case
Map Table:B.1 Commonly mapped to nothing
Normalization:No
Prohibited tablesC.2.1 ASCII control charactersC.2.2 Non-ASCII control
characters
           C.3 Private use
           C.4 Non-character code points
           C.5 Surrogate Codes
           C.6 Inappropriate for plain text
           C.7 Inappropriate for canonical representation
           C.8 Change display properties or are deprecated
           C.9 Tagging characters
Bidi  Yes
```

Figure 25-11 Table CUNSTMX1

The table CUNSTMX2 is used by NFS for domain name in name@domain, as shown in Figure 25-12 on page 499.

```
Case: Mixed Case
Map Table:B.1 Commonly mapped to nothing
          B.2 Mapping for case-folding used with NFKC
Normalization:KC
Prohibited tablesC.2.1 ASCII control charactersC.2.2 Non-ASCII control
characters
          C.3 Private use
          C.4 Non-character code points
          C.5 Surrogate Codes
          C.6 Inappropriate for plain text
          C.7 Inappropriate for canonical representation
          C.8 Change display properties or are deprecated
          C.9 Tagging characters
Bidi  Yes
```

Figure 25-12 Table CUNSTMX2

Note: These tables and their nomenclature are defined in RFC 3454.

Operator commands

Figure 25-13 on page 499 shows new operator commands used for string preps.

Where `name` specifies one of the profiles previously described in “Implementation considerations” on page 497, as follows:

- CUNSTCSP
- CUNSTCIS
- CUNSTMX1
- CUNSTMX2

```
SETUNI ADD,STRPROFILE= name [,DSNAME=] [,VOLSER=] [,PAGEFIX=YES | NO]
SETUNI REPLACE, STRPROFILE= name [,DSNAME=] [,VOLSER=] [,FREE=NO | (YES,FORCE)]
[,PAGEFIX=YES | NO]
SETUNI DELETE,STRPROFILE= name [,FREE=NO | (YES,FORCE)]
```

Figure 25-13 Operator commands for string prep

New operands in CUNUNIxx parmlib member

The CUNUNIxx parmlib member has new operands.

Where name specifies one of the profiles described in “Implementation considerations” on page 497, as follows:

- CUNSTCSP
- CUNSTCIS
- CUNSTMX1
- CUNSTMX2

```
ADD STRPROFILE(name) [DSNAME(….)] [VOLSER(….)] [PAGEFIX(YES | NO)]
REPLACE STRPROFILE(name) [DSNAME(….)] [VOLSER(….)] [FREE(NO | YES,FORCE)]
[PAGEFIX(YES | NO)]
DELETE STRPROFILE(name) [FREE(NO | YES,FORCE)]
```

Figure 25-14 CUNUNIxx parmlib member specifications

New callable services in HLASM or C/C++

Following are the new callable services:

- CUNLSTRP for 31-bit callers
- CUN4LSTR for 64-bit callers

In the parameter list, prefix the following parameters with CUNBPPRM_ (31-bit) or CUN4BPPR_ (64-bit), as follows:

- Version - set by caller.
  - Specifies the version of the parameter area. This field must be initialized for the first call to stub routine CUNLSTRP/CUN4LSTP using the constant CUNBPPRM_Ver/CUN4BPPR_Ver which is supplied by the interface definition file CUNBPIDF.
- Length - set by caller.
  - It is the length of the parameter area. HLASM users must initialize this field for the first call to CUNLSTRP/CUN4LSTP using the constant CUNBPPRM_Len/CUN4BPPR_Len which is supplied by the interface definition file CUNBPIDF.
- Prof_Name - set by caller.
  - Name of the profile to be applied on the Source buffer. Possible values are CUNSTCSP, CUNSTCIS, CUNSTMX1, CUNSTMX2.
- Src_Buf_Ptr - set by caller, updated by service.
  - Specifies the beginning address of a string of text characters. At the completion of the Stringprep, Src_Buf_Ptr will be updated to point just past the last character that was successfully Prepared. If all bytes are normalized, Src_Buf_Len will be zero.
- Src_Buf_ALET - set by caller.
  - Specifies the ALET to be used to access the source buffer addressed by Src_Buf_Ptr. Use an ALET value of 0 to designate the primary address space.
- Src_Buf_Len - set by caller, updated by service.
  - Specifies the length in bytes of the source buffer addressed by Src_Buf_Ptr. The source buffer length may be zero. In this case, nothing is prepared.
- Targ_Buf_Ptr - set by caller.
  - Specifies the beginning address of an area of storage where the string text to be prepared will be stored. At the completion of the normalization, Targ_Buf_Ptr will point just past the last character stored, and Targ_Buf_Len will be updated to indicate the number of bytes not yet consumed in the buffer.
- Targ_Buf_ALET - set by caller.
  - Specifies the ALET to be used to access the target buffer addressed by Targ_Buf_Ptr. Use an ALET value of 0 to designate the primary address space.
- Targ_Buf_Len - set by caller, updated by service.

  - Specifies the length in bytes of the target buffer addressed by Targ_Buf_Ptr. It is strongly suggested this length be at least four times the size as Src_Buf_Len.
- Wrk_Buf_Ptr - set by caller, updated by service.
  - Specifies the beginning address of an area of storage that the Stringprep service can use to store immediate results.
- Wrk_Buf_ALET - set by caller.
  - Specifies the ALET to be used to access the work buffer addressed by Wrk_Buf_Ptr. Use an ALET value of 0 to designate the primary address space.
- Wrk_Buf_Len - set by caller, updated by service.
  - Specifies the length in bytes of the work buffer addressed by Wrk_Buf_Ptr. It is strongly suggested this length to be the same size as Targ_Buf_Len.

The following flags should be set by the caller:

- Bit position Name
- 000X XXXX Reserved
  - These flag bits are reserved for internal service use and should be set to 0.
- XXX1 XXXX UTF_Version
  - UTF_Version: Specifies UTF version source buffer is being passed to the service.
  - 0: UTF-8
  - 1: UTF-16. Default
- XXXX 1XXX UnAssignedEr
  - UnAssignedEr: According to RFC 3454
  - 0: Indicates that the string prep is to be terminated with an error. This is the default.
  - 1: Indicates that the string prep is to give a warning and continue processing.
- Return_Code - set by service.
  - Specifies the return code.
- Reason_Code - set by service.
  - Specifies the reason code.

Figure 25-15 on page 502 shows STRINGPREP in the DISPLAY output (D UNI,ALL).

```
D UNI,ALL
CUN3000I 02.57.05 UNI DISPLAY 581
 ENVIRONMENT: CREATED      03/20/2006 AT 02.53.49
             MODIFIED     03/20/2006 AT 02.56.46
             IMAGE CREATED --/--/---- AT --.--.–
    SERVICE: CHARACTER     CASE         NORMALIZATION  COLLATION
             STRINGPREP    BIDI
    STORAGE: ACTIVE      1026 PAGES
             LIMIT     524287 PAGES
   CASECONV: NONE
  NORMALIZE: ENABLED
   NORM VER: NONE
    COLLATE: ENABLED
 COLL RULES: UCA400R1_CUNENUS
STRPROFILES: CUNSTMX1
 CONVERSION: NO CONVERSIONS FOUND
```

Figure 25-15 Command shows the STRINGPRERP and STRPROFILES

25.5 Collation requirements for ERP

The current support in Unicode Services for Collation is based on the Unicode 3.0.1 standard. ERP applications such as PeopleSoft and Siebel employ the Unicode Collation Algorithm (UCA) from the Unicode 4.0.0 and 4.1.0 Standards. DB2 wants to provide collation support and use the "collation keywords" commonly used across the industry.

There is a need to enable string comparison according to collation rules based on the locale's language, region, and variant. In the previous z/OS versions, z/OS Unicode collation service callers used to compare UTF-16BE strings with UCA rules only, based on Unicode standard 3.0.1, so no tailoring or customizing was enabled.

Now z/OS V1R8 is providing, improving, and implementing functions through the following means:

- Support for later Unicode standards in collation service, such as:
  - Unicode 4.0.0 (Unicode 4.0.1)
  - Unicode 4.1.0
- Unicode collation algorithm (UCA) to use "collation keywords" as an easy way to call Unicode Collation services
- Collation rules mixer engine (CRME) in order to customize default UCA tables from desired locale
- 237 locales in a new data set, SYS1.SCUNLOCL, specifically for collation purposes
- A mechanism to allow you to supply your own collation rules to cover customized string comparisons

Benefits of collation service

This function provides you with the following benefits:

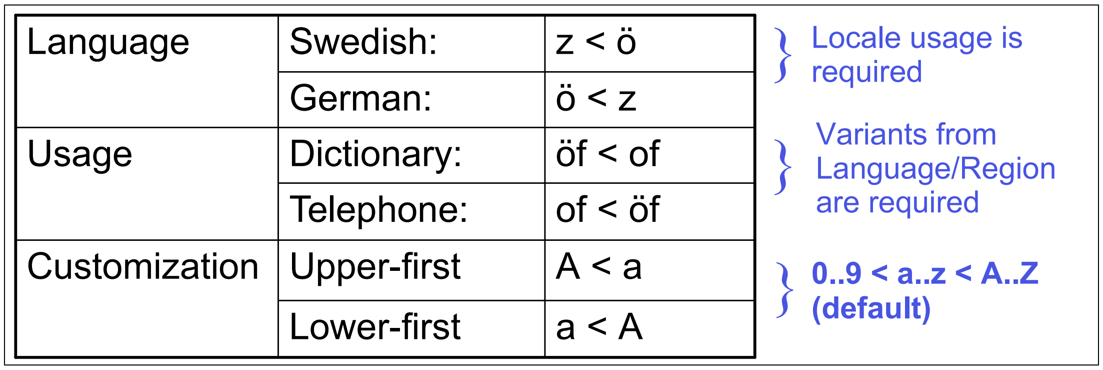
- It enables UTF-16BE (big endian) strings to be collated (sorted) according to desired language properties or user collation rules (UCR).

- It allows IBM applications such as DB2 and other vendor applications such as PeopleSoft and Siebel to provide consistent functionality across platforms.
- Unicode Services will exploit z/Architecture when using collation tables for improved performance.

25.5.1 Implementation considerations

A multilevel comparison algorithm is employed to address the complexities of language-sensitive sorting. However, collation is not uniform; it varies according to language and culture. For example, the German, French, and Swedish languages sort the same characters differently. Collation may also vary by specific application. That is, even within the same language, dictionaries may sort differently than phonebooks or book indices.

As shown in Figure 25-16 on page 503, each subsequent level is applied if no differences are found at the previous level. If a..z < A..Z < 0..9 or A..Z < a..z < 0..9 is needed, then customized collation rules would need to be provided.



Language	Swedish:	z < ö
	German:	ö < z
Usage	Dictionary:	öf < of
	Telephone:	of < öf
Customization	Upper-first	A < a
	Lower-first	a < A

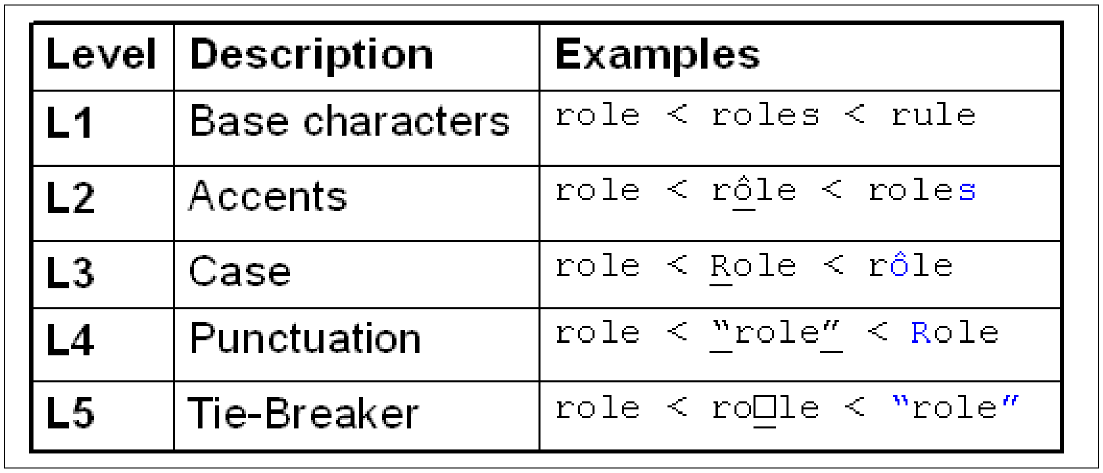
Locale usage is required
Variants from Language/Region are required
0..9 < a..z < A..Z (default)

Figure 25-16 An example of collation

Using collation basics

Many users see the Unicode code charts and expect the characters in their language to be in the “correct” order in the code charts. However, because collation varies by language—and not just by script—it is not possible to arrange code points for characters so that a simple binary string comparison produces the desired collation order for all languages. And because multi-level sorting is a requirement, it is not even possible to arrange code points for characters so that simple binary string comparison produces the desired collation order for any particular language. Separate data tables are required for correct sorting order.

To address the complexities of language-sensitive sorting, a multilevel comparison algorithm is employed, as shown in Figure 25-17 on page 504. Each subsequent level is applied if no differences are found at previous level. In the L5 example listed in Figure 25-17 on page 504, the □ represents a format character, which is otherwise completely ignorable.



Level	Description	Examples
L1	Base characters	role < roles < rule
L2	Accents	role < rôle < roles
L3	Case	role < Role < rôle
L4	Punctuation	role < “role” < Role
L5	Tie-Breaker	role < ro□le < “role”

Figure 25-17 Collation levels

Collation process overview

The collation process overview, illustrated in Figure 25-18 on page 505, proceeds as follows:

1. The process starts using the Default Unicode Collation Table (DUCET) from the Unicode standard. This is a text file which contains the *base* rules for Collation without tailoring.
2. The DUCET file is processed by us, and we generate binary Collation tables which are part of SYS1.SCUNTBL.
3. The names of the collation tables use the following nomenclature:
   - CUNO400x (where x goes from A to Q)
   - CUNO410x (where x goes from A to Q)
4. SYS1.SCUNLOCL contains a list of 236 Locales and an Locale index file name CUNLOCLI.
5. As part of the user application:
   - The collation parameter area CUN4BOPR is set to CUN4BOPR_Ver2 in order to exploit this new behavior.
   - Buffers are set.
   - A collation keyword is set.
6. When Collation Service detects that a requested collation table is not in storage, it will be loaded dynamically.
7. The Collation Rules Mixer Engine (CRME) applies rules from locale and modify tables, customizing them according to Locale sets.
8. After the collation tables are customized, they are written into the Unicode Data Space.
9. The dynamic process returns control to the Unicode Collation Service and then a HANDLE is set according Collation Rules, which is used by the Collation Service.
10. After the HANDLE is obtained, collation tables are known and then source buffers are processed and generate either binary string comparison (A < B or A = B or A > B) or sort keys (a sequence of bytes that can be compared later among other sort keys).

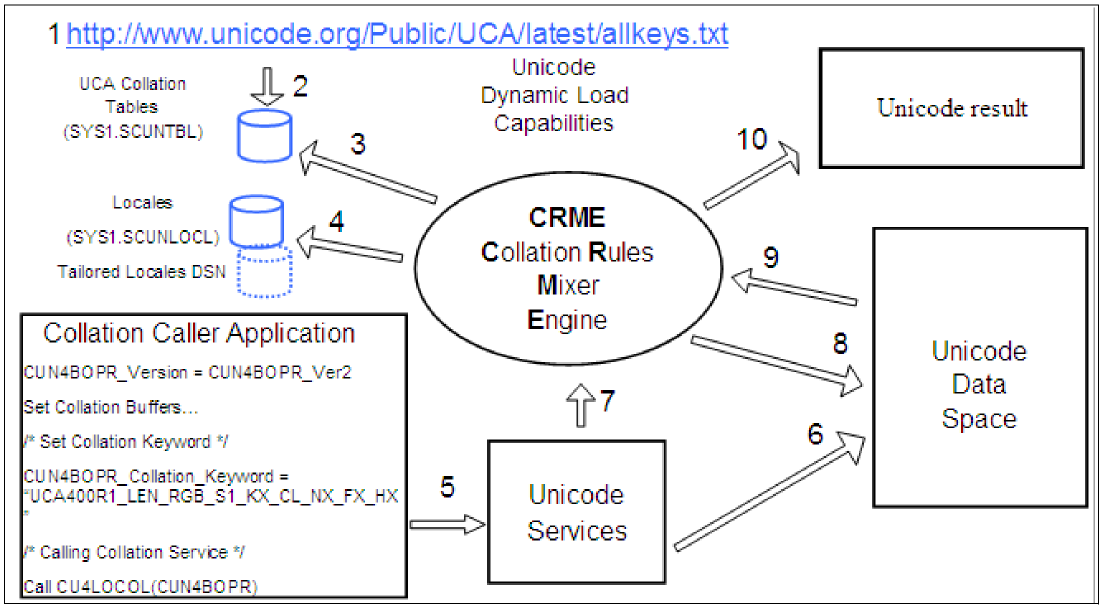

1 http://www.unicode.org/Public/UCA/latest/allkeys.txt
UCA Collation Tables (SYS1.SCUNTBL)
2
Unicode Dynamic Load Capabilities
3
10
Unicode result
Locales (SYS1.SCUNLOCL)
Tailored Locales DSN
4
CRME Collation Rules Mixer Engine
9
Collation Caller Application
CUN4BOPR_Version = CUN4BOPR_Ver2
Set Collation Buffers...
/* Set Collation Keyword */
CUN4BOPR_Collation_Keyword = "UCA400R1_LEN_RGB_S1_KX_CL_NX_FX_HX"
/* Calling Collation Service */
Call CU4LOCOL(CUN4BOPR)
8
Unicode Data Space
7
6
5
Unicode Services

Figure 25-18 Overview of the collation process

Operator commands

Figure 25-19 on page 505 shows the collation operator commands.

```
SETUNI ADD,COLLATE[=UCAver] [,DSNAME=] [,VOLSER=] [[,LOCALE=locale [,DSNAME=]
[,VOLUME=]] | [,COLRULES=colrules [,DSNAME=] [,VOLUME=]]] [,PAGEFIX=YES | NO]
SETUNI REPLACE, COLLATE[=UCAver] [,DSNAME=] [,VOLSER=] [,FREE=NO | (YES,FORCE)]
[[,LOCALE=locale [,DSNAME=] [,VOLUME=]] | [,COLRULES=colrules [,DSNAME=]
[,VOLUME=]]] [,PAGEFIX=YES | NO]
SETUNI DELETE, COLLATE[=UCAver] [,FREE=NO | (YES,FORCE)] [[,LOCALE=locale] |
[,COLRULES=colrules]]
```

Figure 25-19 Operator commands for collation

New operands in CUNUNIxx parmlib member

Figure 25-20 on page 506 shows the CUNUNIxx parmlib member examples for collation.

Where:

UCAver Specifies the Unicode Collation Algorithm (UCA) versions. Possible values are:

- UCA301
- UCA400R1
- UCA410

locale Specifies the locale member name where collation rules are to be loaded.

colrules Specifies the User Collation Rules member name where collation rules are to be loaded.

```
ADD,COLLATE[(UCAver)] [,DSNAME(…)] [,VOLSER(…)] [[,LOCALE(locale) [,DSNAME(…)]
[,VOLUME(…)]] | [,COLRULES(colrules) [,DSNAME(…)] [,VOLUME(…)]]] [,PAGEFIX(YES
| NO)]
REPLACE, COLLATE[(UCAver)] [,DSNAME(…)] [,VOLSER(…)] [,FREE(NO | YES,FORCE)]
[[,LOCALE(locale) [,DSNAME(…)] [,VOLUME(…)]] | [,COLRULES(colrules)
[,DSNAME(…)] [,VOLUME(…)]]] [,PAGEFIX=YES | NO]
SETUNI DELETE, COLLATE[(UCAver)] [,FREE(NO | YES,FORCE)] [[,LOCALE(locale)] |
[,COLRULES(colrules)]]
```

Figure 25-20 CUNUNIxx parmlib member examples

Callable services in HLASM or C/C++

Following are the new callable services:

- CUNLOCOL for 31-bit callers
- CUN4LCOL for 64-bit callers

The parameter list is prefixed with the parameters CUNBOPRM_ (31-bit) or CUN4BOPR_ (64-bit). Extensive changes and multiple new fields have been added to the new version of the parameter list. For more detailed information, refer to *z/OS Support for Unicode: Using Unicode Services*, SA22-7649.

Figure 25-21 on page 506 shows UCA400R1_CUNENUS in the DISPLAY output for the D UNI,ALL command.

```
D UNI,ALL
CUN3000I 02.57.05 UNI DISPLAY 581
 ENVIRONMENT: CREATED       03/20/2006 AT 02.53.49
             MODIFIED      03/20/2006 AT 02.56.46
             IMAGE CREATED --/--/---- AT --.--.–
    SERVICE: CHARACTER      CASE          NORMALIZATION  COLLATION
             STRINGPREP     BIDI
    STORAGE: ACTIVE       1026 PAGES
             LIMIT      524287 PAGES
   CASECONV: NONE
  NORMALIZE: ENABLED
   NORM VER: NONE
    COLLATE: ENABLED
 COLL RULES: UCA400R1_CUNENUS
STRPROFILES: CUNSTMX1
 CONVERSION: NO CONVERSIONS FOUND
```

Figure 25-21 UCA400R1_CUNENUS in the Display output

Installation considerations

There is a new target library, SYS1.SCUNLOCL, and it should be cataloged. There is also a new distribution library, SYS1.ACUNLOCL.

26

REXX/CLIST variable storage constraint relief

This chapter explains how to adapt to today’s high data usage environments more easily and seamlessly by using REXX/CLIST, especially when gathering and processing command and program output. The following topics are discussed:

- The purpose of the changes on REXX/CLIST storage (RXSTOR)
- How this new function is enabled and used
- How applications that might not be tolerant of the new TSO/E PROFILE command to create a variable storage setting can be run
- How the REXX OUTTRAP change complements the new PROFILE command variable storage setting

26.1 REXX/CLIST enhancements

In the previous versions of z/OS, during normal execution, REXX variables are kept in 31-bit addressable storage and a CLIST pool had been kept in storage below the 16 MB line. This severely limited the number of lines of output that could be trapped from authorized commands and programs.

z/OS V1R8 provides enhancements to REXX/CLIST support, as follows:

- There is a new operand to the PROFILE command variable (VARSTORAGE(HIGH/LOW)), to control where variable pool storage resides.
- The variable pool, used by CLIST and by REXX output trapping of output from authorized command/program, can be (optionally) moved above the 16 MB line.
- REXX OUTTRAP is enhanced with the SKIPAMT argument to allow skipping of a specified number of lines before OUTTRAP trapping begins.

Using the these enhancements, you can do the following:

- Use the PROFILE command to tell TSO whether REXX/CLIST should use storage below or above 16 MB.
- Force CLIST variables to be placed in storage above the 16 MB line, thus increasing the number of variables and amount of information that can be kept in CLIST variables.
- Write REXX EXECs that can trap much larger amounts of output generated by authorized commands or programs invoked from REXX.

The complete syntax of the TSO/E PROFILE command is shown in Figure 26-1 on page 508.

```
PROFILE   LINE('CHARACTER')/LINE(ATTN)/LINE(CTLX)/NOLINE
                CHAR('CHARACTER')/CHAR(BS)/NOCHAR
                PROMPT/NOPROMPT  INTERCOM/NOINTERCOM
                PAUSE/NOPAUSE  MSGID/NOMSGID
                MODE/NOMODE  WTPMSG/NOWTPMSG
                PREFIX('PREFIX')/NOPREFIX  LIST
                RECOVER/NORECOVER
                PLANGUAGE('LANGUAGE')
                SLANGUAGE('LANGUAGE')
                VARSTORAGE(HIGH/LOW)
```

Figure 26-1 Syntax of the TSO/E PROFILE command

To use the new support, a TSO/E user must first issue PROFILE VARSTORAGE(HIGH).

REXX EXECs or CLISTs that need to exploit the new larger variable space should issue the PROFILE VARSTORAGE(HIGH) command. However, in the event that some existing application is unable to use variables that reside in 31-bit addressable storage (usually because some program of the application invokes the IKJCT441 CLIST/REXX variable interface routine while in AMODE24), you can continue to run that application by first resetting your profile to VARSTORAGE(LOW), which is the default. Then invoke the application. This provides relief until the application invoking IKJCT441 can be updated to perform the call in AMODE31.

It is reasonable to assume that just one user or group of users might use an application that cannot tolerate variables in 31-bit storage. In this case, it is not necessary to require that all users disable use of variable storage high just because one user or group of users could not use this feature.

Further, it is possible that a given user might need to use high variable storage to run some particular application, but might need low variable storage for another. Placing the control in the TSO/E profile level provides for a better level of control.

26.2 Implementation considerations

REXX/CLIST variable constraint relief is enabled by the TSO/E PROFILE VARSTORAGE(HIGH/LOW) command, and is used when a CLIST or REXX EXEC is invoked. It looks at and remembers the PROFILE VARSTORAGE setting at the moment the procedure begins.

In order to enable use of storage above the 16 MB line for the variables used by CLIST and output trapped from authorized commands or programs invoked by REXX, the user should specify the following:

```
PROFILE VARSTORAGE(HIGH)
```

In order to view the current profile setting, issue the TSO/E PROFILE command without operands, as shown:

```
PROFILE
```

Figure 26-2 on page 509 shows the command response to the PROFILE command.

```
IKJ56688I CHAR(0) LINE(0) PROMPT INTERCOM NOPAUSE MSGID MODE WTPMSG NORECOVER
PREFIX(TUSER01) PLANGUAGE(ENU) SLANGUAGE(ENU) VARSTORAGE(LOW)
IKJ56689I DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL
```

Figure 26-2 Message IKJ56688I

The current PROFILE VARSTORAGE setting that is active at the time a CLIST or REXX EXEC begins remains in effect for the life of that CLIST or EXEC. If the CLIST or EXEC issues another PROFILE command to change the current VARSTORAGE setting, the new setting will not take effect until a new CLIST or REXX EXEC is started. This includes any CLIST or REXX EXEC that is invoked from the currently executing procedure.

26.2.1 Message IKJ56688I changed

Existing message IKJ56688I that is returned as output for the PROFILE command now returns a current setting of the VARSTORAGE operand, as shown in Figure 26-3 on page 509.

```
IKJ56688I CHAR(0)  LINE(0)    PROMPT   INTERCOM   NOPAUSE MSGID   MODE
WTPMSG   RECOVER   PREFIX(TUSER01) PLANGUAGE(ENU) SLANGUAGE(ENU)
VARSTORAGE(HIGH)
```

Figure 26-3 Message IKJ56688I

26.3 CLIST/REXX variable interface routine

Application programs can use the IKJCT441 interface to read or write REXX or CLIST variables. Callers requesting REXX variables have always been required to be AMODE31, but requestors for CLIST variables previously were allowed to be AMODE24.

z/OS V1R8 callers requesting CLIST or REXX variables who are not AMODE31 now get a return code x'58' if the PROFILE VARSTORAGE(HIGH) is set.

Return code X'58'

A new return code may be returned to callers of IKJCT441 that invoke it to retrieve variables (for example, for entry codes TSVERETR, TSVNOIMO, or TSVELOC), as follows:

- The caller of IKJCT441 is running AMODE24, but the variables requested may be in 31-bit addressable storage. The caller should change to AMODE31 before calling IKJCT441 so that variables returned can be addressed by the caller.
- If a program calling IKJCT441 to read a variable (entry codes TSVERETR, TSVNOIMP, or TSVELOC) gets a return code x'58', the program should be changed, if possible, to invoke IKJCT441 in AMODE31.

  As a circumvention until this can be done, the user can force CLIST and authorized REXX variables to be kept in 24-bit addressable storage by changing the user's profile setting to PROFILE VARSTORAGE(LOW). In this case, variables are kept in 24-bit storage and, therefore, will be addressable by AMODE24 callers. Return code x'58' is only possible while running with PROFILE VARSTORAGE(HIGH).

The IKJCT441 program interface, along with the new return code, is documented in *z/OS TSO/E Programming Services*, SA22-7789.

26.4 REXX OUTTRAP function

Within a REXX EXEC, an optional SKIPAMT tells OUTTRAP to skip the specified number of lines before beginning to trap output. This is useful for trapping both authorized and unauthorized output.

REXX EXECs doing OUTTRAP can use a new optional fourth argument on the OUTTRAP function to limit the amount of output trapped, as follows:

```
X=OUTTRAP(LINE.,MAXCNT,'CONCAT',SKIPAMT)
If no SKIPAMT arg is specified, it defaults to 0
```

The enhancement to OUTTRAP was designed to complement and extend its existing functionality. The new skip amount argument, SKIPAMT, is similar in format and use to the second argument MAXCNT. However, instead of telling output trapping to disregard all lines of output *after* some MAXCNT lines have been processed, SKIPAMT tells output trapping to disregard all lines of output from being trapped *until* the specified SKIPAMT count has been satisfied.

Also new variables, varname.SKIPPED and varname.SKIPAMT, are provided so that the REXX user can determine the SKIPAMT, in much the way they use varname.MAX to determine the current maximum, or varnameTRAPPED to determine the total number of lines processed.

For clarity, note that varname.TRAPPED returns the total number of lines *processed*, regardless of whether they were actually trapped. Varname.0 returns the number of lines actually *trapped*.

OUTTRAP parameter

Use of the new OUTTRAP parameter is completely optional. If not specified, OUTTRAP behaves exactly as it did before this operand was introduced; namely, as if SKIPAMT were zero (0).

Refer to *z/OS TSO/E REXX Reference*, SA22-7790 for more information about the REXX OUTTRAP function and use of the new SKIPAMT argument.

For the skip amount, you may specify an integer in the range 0 to 999,999,999. Leading zeros are ignored, as long as the total length of the argument does not exceed 36 characters. This argument is optional. If not specified, the default skip amount is zero (0).

26.4.1 New message IRX0218E

The new REXX message IRX0218E may be issued when using OUTTRAP with a SKIPAMT, as shown in Figure 26-4 on page 511.

```
IRX0218E The skip amount argument passed to OUTTRAP is invalid.
```

Figure 26-4 Message IRX0218E

This message is issued in response to a skip amount argument used in the OUTTRAP function that is incorrect for one or more of the following reasons:

- Some of the characters are not valid.
- The skip amount is out of the allowed range.
- The skip amount is longer than 36 characters.

26.4.2 Implementation considerations

OUTTRAP traps output from commands, including those written in REXX. A command written in REXX cannot turn output trapping off on behalf of its invoker. Output trapping should be turned on and off at the same EXEC level. Therefore, a command written in REXX should only turn output trapping off if that command turned it on.

Note the following, as illustrated in Figure 26-5 on page 512:

- The first example illustrates correct usage of OUTTRAP.
- The second illustrates incorrect usage. Note that the placement of the y = OUTTRAP('OFF') statement must be within the REXX1 EXEC, not the REXX2 command.

```
Correct usage of OUTTRAP

        x = OUTTRAP('VAR.')
        "%REXX2"
        y = OUTTRAP('OFF')
        EXIT
        /* REXX2 command */
        SAY "This is output from the REXX2 command " /* This will be trapped */
        RETURN

Incorrect usage of OUTTRAP

        /* REXX1 */
        x = OUTTRAP('VAR.')
        "%REXX2"
        EXIT
        /* REXX2 command */
        SAY "This is output from the REXX2 command " /* This will be trapped */
        y = OUTTRAP('OFF')
        RETURN
```

Figure 26-5 Examples of the correct and the incorrect use of OUTTRAP

Figure 26-6 on page 512 shows the resulting values in variables after the following OUTTRAP function is performed, using the SKIP argument of OUTTRAP to skip over lines of output before trapping begins.

This is an example of OUTTRAP using the SKIPAMT argument. Notice that new variables xyz.skipped and xyz.skipamt are now returned, in addition to the other set of OUTTRAP variables that have always been set.

The SKIPAMT was 50, therefore 40 lines of output produced by the CMD1 invocation and 10 lines from the CMD2 invocation are skipped before trapping begins. (Because the “CONCAT” operand was specified, output spanning multiple command invocations accumulate into a single, ever-increasing trapping variable.)

```
 x = OUTTRAP("XYZ.",10,"CONCAT",50)
   “CMD1”                  /* CMD1 produces 40 lines of output     */

  /* CMD1 has 40 lines of output. Because 40 is less than 50, no     */
  /* lines of CMD1 are trapped. After CMD1, OUTTRAP variables have   */
  /* the following values:                                           */
    XYZ.0                  --> 0           No lines trapped yet
    XYZ.1                  --> XYZ.1
    XYZ.2                  --> XYZ.2
     :         :
    XYZ.10                 --> XYZ.10
    XYZ.MAX                --> 10
    XYZ.TRAPPED     --> 40         40 lines of output handled so far
    XYZ.CON         --> CONCAT
    XYZ.SKIPPED     --> 40         Number of lines skipped so far
    XYZ.SKIPAMT     --> 50         Total number to skip before trapping
```

Figure 26-6 Resulting values in variables after CMD1

Note that after CMD2 completes, 50 lines have been skipped, and 5 have been actually trapped. This is represented by XYZ.0=5, as shown in Figure 26-7.

```
"CMD2"                  /* CMD2 produces 15 more lines of output   */
 /* CMD2 has 15 lines of output. 10 additional lines will be skipped, */
 /* and trapping will begin with the 11th line (51st line overall).   */
   XYZ.0                  --> 5           5 lines trapped
   XYZ.1                  --> cmd2 output line 11 (output line 51 overall)
   XYZ.2                  --> cmd2 output line 12 (output line 52 overall)
   XYZ.3                  --> cmd2 output line 13 (output line 53 overall)
   XYZ.4                  --> cmd2 output line 14 (output line 54 overall)
   XYZ.5                  --> cmd2 output line 15 (output line 55 overall)
   XYZ.6                  --> XYZ.6
    :          :
   XYZ.10                 --> XYZ.10
   XYZ.MAX        --> 10
   XYZ.TRAPPED    --> 55          55 lines of output handled so far
   XYZ.CON        --> CONCAT
   XYZ.SKIPPED    --> 50          Number of lines skipped so far
   XYZ.SKIPAMT    --> 50          Total number to skip before trapping
```

Figure 26-7 Resulting values in variables after CMD2

Note that CMD3, as shown in Figure 26-8 on page 513, produces 10 more lines of output, but 5 lines have already been trapped into XYZ.1 through XYZ.5. Because a MAXNT of 10 was specified in argument 2 of OUTTRAP, only the first 5 lines of CMD3 output are trapped.

At this point the maximum number of lines (10) have been trapped in XYZ.1 through XYZ.10. Variable XYZ.0 will contain `10`, to indicate 10 lines have been trapped. XYZ.trapped contains `65`, to show that a total of 65 lines of output have been processed.

```
"CMD3"                /* CMD3 produces 10 more lines of output         */
 /* CMD3 has 10 lines of output. 5 additional lines will be trapped     */
 /* until MAX has been reached, and the remaining lines are discarded.  */
   XYZ.0              --> 10          10 lines trapped
   XYZ.1              --> cmd2 output line 11 (output line 51 overall)
   XYZ.2              --> cmd2 output line 12 (output line 52 overall)
   XYZ.3              --> cmd2 output line 13 (output line 53 overall)
   XYZ.4              --> cmd2 output line 14 (output line 54 overall)
   XYZ.5              --> cmd2 output line 15 (output line 55 overall)
   XYZ.6              --> cmd3 output line 1  (output line 56 overall)
   XYZ.7              --> cmd3 output line 2  (output line 57 overall)
    :          :
   XYZ.10             --> cmd3 output line 5  (output line 60 overall)
   XYZ.MAX            --> 10
   XYZ.TRAPPED        --> 65          65 lines of output handled so far
   XYZ.CON            --> CONCAT
   XYZ.SKIPPED        --> 50          Number of lines skipped so far
   XYZ.SKIPAMT        --> 50          Total number to skip before trapping
   x = OUTTRAP('OFF')  --> turn off outtrap.
```

Figure 26-8 Resulting values in variables after CMD3

REXX EXEC to invoke an authorized command

In Figure 26-11 on page 515, an EXEC is divided into two steps, A and B, and a user is about to use a REXX EXEC to invoke an authorized command, MYACMD.

The EXEC uses OUTTRAP to trap and process the output. Although the command may produce 500,000 or more lines of output, the user is only interested in the last 70,000 lines. The user wants to capture these lines and write them to a data set to be processed later.

As shown in Figure 26-9 on page 514 for step A, the user issues the PROFILE command without operands to see current settings, and to ensure that the VARSTORAGE setting is HIGH.

```
 PROFILE
IKJ56688I CHAR(0)  LINE(0)    PROMPT   INTERCOM   NOPAUSE MSGID  MODE    WTPMSG
NORECOVER PREFIX(TUSER01) PLANGUAGE(ENU) SLANGUAGE(ENU) VARSTORAGE(LOW)
IKJ56689I DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL
```

Figure 26-9 The VARSTORAGE setting is LOW

Because the VARSTORAGE setting is LOW, the EXEC changes it to HIGH issuing the PROFILE command, as shown:

```
PROFILE VARSTORAGE(HIGH)
```

This example demonstrates how you can interrogate your current PROFILE setting to ensure that the PROFILE VARSTORAGE(HIGH) setting is active before the REXX EXEC is invoked.

For step B, the user then invokes the following EXEC, as shown in Figure 26-10 on page 514. This EXEC attempts to trap partial output from an authorized command, as follows:

- First, the function call ‘x = OUTTRAP(line.,0)’ is issued, with a MAXCNT=0 (second operand) so that no output gets trapped when the command MYACMD is subsequently executed. However, line.trapped will still get updated to contain the number of lines of output processed (that is, generated) by the command, while line.0 will be set to zero (0), showing that no lines were actually trapped.

```
/* REXX */
"ALLOC FI(OUTDD) DA(...my output dataset...) SHR REUSE“
/*    ...  other processing ... */

/*************************************************************/
/* First determine how much output the command produces.        */
/* Specify a MAXAMT of 0 (2nd operand on OUTTRAP) to            */
/* indicate that we don't want to actually trap any output      */
/* yet. We just want to determine the number of lines of        */
/* output produced.                                             */
/*************************************************************/
X = OUTTRAP(line.,0)
"MYACMD"
No_of_lines = line.trapped   /* Total number of lines of output */
```

Figure 26-10 The REXX EXEC output showing lines of output processed

- After determining the total number of lines of output generated by the command, SKIPAMT is computed as (No_of_lines-70000). OUTTRAP is then issued again, with the computed SKIPAMT specified, so that everything but the last 70,000 lines will be skipped before trapping begins when the next command is issued. If less than 70,000 lines of output are generated by the command, then all lines are trapped by the EXEC, as shown in Figure 26-11 on page 515.

```
/**************************************************************/
/*  Throw away everything except the last 70,000 lines by      */
/*  setting a SKIPAMT on the OUTTRAP function call.            */
/**************************************************************/
Skipamt = MAX(No_of_lines - 70000,0)
X= OUTTRAP(line.,'*','CONCAT',skipamt)
"MYACMD"
say ‘Total skipped=‘line.skipped ‘Total processed=’line.trapped
n = line.0    /* number actually trapped - expecting 70,000     */
X= OUTTRAP(‘OFF’)   /* Turn off further trapping                */
/**************************************************************/
/* Last 70,000 lines are trapped in line.1 to line.n where n is */
/* the line.0 value. If fewer than 70,000 lines of output were  */
/* produced, then all of them are trapped.                      */
/**************************************************************/
"execio" n" diskw outdd (stem line. finis" /*Write trapped lines*/
If rc=0 then
  Say n ' lines of data have been written to file OUTDD'
/* ...more processing   ...  */
"FREE FI(OUTDD)"
exit 0
```

Figure 26-11 The REXX EXEC output showing lines processed, skipped, trapped or captured

Notice that line.trapped contains the number of lines actually processed, not the number of lines captured. Furthermore, line.skipped contains the number of lines actually skipped, while line.0 contains the number of lines actually trapped or captured. Observe that if the SKIPAMT had been zero (0), as it would be on pre-V1R8 releases, then line.trapped and line.0 would have been equal.

Finally, the trapped information is written (by execio) to some output data set for later analysis.

Note: The example illustrated in Figure 26-10 and Figure 26-11 assumes that the output generated by two consecutive invocations of the command MYACMD will be the same. That is, the output is assumed *not* to have changed between the two back-to-back invocations.

26.5 Migration considerations

By default, all TSO/E users begin running with PROFILE VARSTORAGE(LOW). To exploit 31-bit variable storage, users must set PROFILE VARSTORAGE(HIGH).

If you need to run an application that invokes IKJCT441 to read CLIST or REXX variables and that application invokes IKJCT441 in AMODE24, a return code x’58’ may be returned and the request fails, as previously discussed. Note the following:

- The caller of IKJCT441 should be changed to invoke it in AMODE31. However, as a circumvention until this change can be made, the application can continue to be run with PROFILE VARSTORAGE(LOW).

- In addition, any installation which has modified the REXX parameter modules IRXPARMS, IRXTSPRM, or IRXISPRM, may want to update the installation copy of these modules to include the SYSCALL host environment and EZAFTPKR function package routine. These are now included in the new IBM version of these parameter modules shipped in z/OS V1R8.

TSO/E REXX considerations

The following considerations affect TSO/E REXX:

- As of V1R8, the TSO/E REXX default parameter modules (IRXPARMS, IRXISPRM, and IRXTSPRM) have been updated to contain SYSCALL as a new host command environment for USS REXX commands.
- A new function package, EZAFTPKR, has been added in support of the FTP client API.
- Information about the use of this new host command environment and the new FTPAPI function can be found in publications associated with those products:
  - *Using REXX and z/OS UNIX Systems Services*, SA22-7806, provides more information about the SYSCALL REXX host command environment.
  - *IP Programmer's Guide and Reference*, SC31-8787, provides information about the FTP Client Application Programming Interface.

If your installation has modified any of the three REXX parameter modules mentioned, you may want to update your modified copies to contain the new SYSCALL host environment, and the new EZAFTPKR function package. You can refer to the updated versions of the REXX parameter modules as shipped in SYS1.SAMPLIB to see how the new SYSCALL host environment and EZAFTPKR function package routine should be included.

26.5.1 Coexistence support

Coexistence applies to lower-level systems which coexist (share resources) with latest z/OS systems. The new SKIPAMT argument of the OUTTRAP function is only supported on z/OS V1R8 or later. On earlier levels a statement such as:

```
x=outtrap(line.,1000,'CONCAT,10000)
```

Would fail with errors as follows:

- IRX0217E Too many arguments were passed to OUTTRAP
- 39 +++ x=outtrap(line.,1000,'CONCAT',10000)
- IRX0040I Error running AWSY05T1, line39: Incorrect call to routine

If you use an EXEC with the SKIPAMT argument, and the EXEC is in a library such that it might be invoked from both z/OS V1R8 and pre-V1R8 systems, you can use the REXX SYSVAR function to test the TSO/E level of TSO/E before invoking OUTTRAP with the SKIPAMT argument, as shown in Figure 26-12 on page 517.

```
tsolvl = SYSVAR('SYSTSOE')       /* ver/rel/mod of TSO/E, as: VRRM
                                    Release V1R8 is '3080'           */
skipamt = 1000                   /* Number of lines to skip past before
                                    reaching lines of interest       */
If tsolvl >= '3080' then         /* If Release 8 or later            */
  x = outtrap(line.,10,'noconcat',skipamt) /* trap just 10 lines starting
                                              at 1001 in variables line.1
                                              thru line.10           */
else                                       /* else release 7 or earlier */
  x = outtrap(line.,1010,'noconcat') /* trap first 1010 lines, even
                                        though we are just interested  in
                                        in lines 1001 to 1010        */
"LIST 'userid.MY.INPUTDS' "      /* List mydata set, but trap output
                                    since OUTTRAP is active         */
```

Figure 26-12 The use of OUTTRAP to capture LIST dsn

As illustrated in Figure 26-12, OUTTRAP is used to capture "LIST 'dsn' " output from listing some sequential data set that might contain thousands of lines of data. For example, suppose we are only interested in lines 1001 through 1010. We could use the SKIPAMT argument to trap just the lines of interest and avoid using storage to trap the unneeded first 1000 lines, providing that we are running on V1R8 or later.

If running on V1R8, just the 10 lines (1001 to 1010) will be trapped. If running on an earlier release, lines 1 to 1010 are trapped, but just lines 1001 to 1010 are the lines of interest.

26.6 Installation considerations

There are no changes to the installation considerations, unless the installation has modified the default REXX parameter modules (IRXPARMS, IRXTSPRM, IRXISPRM).

If your installation has modified (customized) its own version of REXX parameter the modules (IRXPARMS, IRXTSPRM, or IRXISPRM), then you may want to update the installation copy of these modules to include the SYSCALL host environment and EZAFTPKR function package routine. As mentioned, these are now included in the new IBM version of these parameter modules shipped in z/OS V1R8.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see "How to get IBM Redbooks" on page 520. Note that some of the documents referenced here may be available in softcopy only.

- *z/OS Version 1 Release 3 and 4 Implementation,* SG24-6581
- *z/OS Version 1 Release 5 Implementation,* SG24-6326
- *z/OS Version 1 Release 6 Implementatio*n, SG24-6377
- *z/OS Version 1 Release 7 Implementatio*n, SG24-6755

Other publications

These publications are also relevant as further information sources:

- *z/OS MVS Initialization and Tuning Guide*, SA22-7591
- *z/OS MVS Programming: Workload Management Services*, SA22-7919
- *z/OS V1R8 MVS: Planning for Workload Manager*, SA22-7602-12
- *z/OS MVS System Commands*, SA22-7627
- *z/OS MVS Using the Subsystem Interface*, SA22-7642
- *z/OS Resource Measurement Facility User's Guide*, SC33-7990
- *IBM Health Checker for z/OS User's Guide*, SA22-7994
- *z/OS Distributed File Service Messages and Codes*, SC24-5917
- *z/OS Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS Communications Server: IP Configuration Reference*, SC31-8776
- *z/OS DFSMSdfp Diagnosis*, GY27-7618
- *z/OS MVS Programming: Workload Management Services*, SA22-7619
- *z/OS Language Environment Programming Reference*, SA22-7562
- *z/OS XML System Services User's Guide and Reference*, SA23-1350
- *z/OS Support for Unicode: Using Unicode Services*, SA22-7649
- *z/OS TSO/E Programming Services*, SA22-7789
- *z/OS TSO/E REXX Reference*, SA22-7790

Online resources

These Web sites and URLs are also relevant as further information sources:

- RMF homepage:

http://www.ibm.com/servers/eserver/zseries/zos/rmf

- Capacity table of processors

  http://www.ibm.com/servers/eserver/zseries/srm/

- z/OS Web deliverables are available from:

  http://www.ibm.com/eserver/zseries/zos/downloads/

- For more information, and for all previously announced statements of direction affecting z/OS V1R7 and future releases, visit:

  http://www.ibm.com/servers/eserver/zseries/zos/zos_sods.html

- Files for CIM 2.9.0 can be downloaded from the DMTF Web site at:

  http://www.dmtf.org/standards/cim

- In the Pegasus enhancement proposal #140, a specification is described in Section 3.6.1 in the following URL:

  http://www.ietf.org/rfc/rfc2616.txt

- The Bidi transformation service is based on the Unicode standard Annex #9

  http://www.unicode.org/reports/tr9

-

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

z/OS Version 1 Release 8 Implementation

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

z/OS Version 1 Release 8 Implementation

Consoles, CFRM, RMF, WLM, Health Checker, DFSMS, zAAPs

zIIPs, RSM and SRM, z/OS UNIX, zFS, z/OS XML, Unicode

JES2, JES3, LE, GRS, HCD, HCM

This IBM Redbook describes the functional enhancements to z/OS for Version 1 Release 8 (z/OS V1R8). These enhancements are designed to help installations install, tailor, migrate, and configure z/OS V1R8.
This redbook describes the following enhancements:
z/OS Version 1 Release 8 Overview
Installation and migration to z/OS V1R8
Console restructure
CFRM performance enhancements
RMF enhancements
WLM enhancements
zSeries Integrated Information Processors (zIIPs)
RSM and SRM
Miscellaneous BCP changes
z/OS UNIX System Services
zFS enhancements
JES2 z/OS V1R8 enhancements
JES3 z/OS V1R8 enhancements
DFSMS enhancements
Program management enhancements
z/OS XL C/C++ enhancements
Language Environment enhancements
GRS enhancements
z/OS XML (eXtensible Markup Language)
z/OS V1R8 Common Information Model enhancements
Enterprise Workload Manager (EWLM)
IBM Health Checker for z/OS in z/OS V1R8
Storage clear for PL/I applications
HCM/HCD enhancements
Unicode services
REXX/CLIST variable storage constraint relief

SG24-7265-00 ISBN 0738494658