IBM

z/OS Version 1 Release 6 Implementation

- z/OS, ServerPac, WLM, RMF
- DFSMS, SMP/E, z/OS UNIX
- OSA, ISPF, Communication Server

Paul Rogers
Patrick Bruinsma
Olivier Daurces
Robert Kohler
Meganen Naidoo
Miha Petric
Natabar Sahoo

Redbooks

ibm.com/redbooks

International Technical Support Organization

z/OS Version 1 Release 6 Implementation

April 2005

SG24-6377-00

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

First Edition (April 2005)

This edition applies to Version 1 Release 6 of z/OS (5694-A01), to Version 1 Release 6 of z/OS.e (5655-G52), and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Notices

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

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

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

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

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

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

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

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

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

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

Trademarks

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

AnyNet®
CICS®
Domino®
DB2 Universal Database™
DB2®
DFSMS/MVS®
DFSMSdfp™
DFSMSdss™
DFSMShsm™
DFSMSrmm™
Encina®
Enterprise Storage Server®
FlashCopy®
Infoprint®
IBM®
IMS™
Language Environment®
Lotus®
MQSeries®
MVS™
MVS/ESA™
NetView®
OS/390®
Parallel Sysplex®
Processor Resource/Systems Manager™
PR/SM™
RACF®
Redbooks™
Redbooks (logo) ™
RMF™
S/390®
SystemPac®
Tivoli®
VisualAge®
VTAM®
WebSphere®
z/Architecture™
z/OS®
z/VM®
zSeries®

The following terms are trademarks of other companies:

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

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

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

Preface

This IBM® Redbook discusses the many enhancements to z/OS® Version 1 Release 6, and provides information to help you install, tailor, and configure this release.

It first offers a broad overview of z/OS Version 1 Release 6, and then goes into detail on how to install and tailor z/OS and the many components that have been enhanced, such as: the z/OS base control program (BCP), ServerPac, DFSMS, Workload Manager (WLM), RMF™, SMP/E, z/OS UNIX®, ISPF, and Communication Server.

This redbook is intended for systems programmers and administrators responsible for customizing, installing, and migrating to the newest levels of z/OS.

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 JES3, and z/OS UNIX. Before joining the ITSO 17 years ago, Paul worked in the IBM Installation Support Center (ISC) in Greenford, England providing OS/390® and JES support for IBM EMEA and the Washington Systems Center in Gaithersburg, Maryland.

Patrick Bruinsma is an Advisory IT Specialist working for IBM Global Services in The Netherlands. He has six years of experience with z/OS, DB2®, MQSeries®, Websphere MQ Workflow, Blaze Advisor, CICS®, and UNIX System Services. He participated in a previous ITSO residency dealing with UNIX-related topics.

Olivier Daurces is an Advisory IT Specialist working for IBM Technical support in France. He has five years of experience with z/OS. His areas of expertise include RACF® and Parallel Sysplex®.

Robert Kohler is a Certified Consulting Systems Products IT Specialist working for IBM US in Technical Sales Support - Americas Techline. He has more than 23 years of systems programming experience in mainframe environments on MVS™, OS/390 and z/OS platforms. His expertise covers a wide range of hardware and software products. He specializes in installation, implementation, migration, performance tuning, and capacity planning.

Natabar Sahoo is an Advisory IT Specialist working for IBM Singapore in Integrated Technology Services, zSeries® team since 1995. He has 22 years of experience in the IT field and worked for 14 years in Large Systems. He holds a degree in Electrical Engineering from University College of Engineering, Burla, Orissa, India. His areas of expertise include z/OS, Parallel Sysplex, WLM, TCP/IP, RACF and he specializes in problem diagnosis, teaching, system administration, implementation and migration of z/OS and SW products.

Meganen Naidoo is a Technical Architect working for Comparex Africa, the leading provider of competitive, innovative and practical business solutions, based in South Africa. He has more than 20 years of mainframe experience, working on VM, OS/390, z/OS, and Linux® system platforms. His main areas of expertise include a variety of technical topics on z/OS,

CICS, and Storage Management. He specializes in research and development, system installations and migrations, and problem determination and resolutions.

Miha Petric is a system programmer from Slovenia working as IBM subcontractor. His areas of expertise include MVS and its subsystems. He has worked in this field since 1978. He is an IBM Business Partner for education and teaches IBM classes.

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. HYJ Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

1

z/OS Version 1 Release 6 overview

z/OS is an operating system designed to meet the on demand challenges of the e-business world. z/OS delivers very high qualities of service for enterprise transactions and data, and extends these qualities to new applications using the latest software technologies. Some highlights of z/OS are: The 64-bit z/Architecture™ and the IBM zSeries server eliminate bottlenecks associated with the lack of addressable memory. 64-bit real (central) storage support eliminates expanded storage, helps eliminate paging, and may allow you to consolidate your current systems into fewer logical partitions (LPARs), or to a single native image.

This chapter provides an overview of the features and new functional changes offered by z/OS Version 1 Release 6.

1.1 z/OS Version 1 Release 6 enhancements

z/OS V1R6 was designed to deliver continued improvements for running new workloads on z/OS. These workloads include ERP applications with the DB2 database on z/OS, modernizing current IMS™ and CICS applications for the Web, and WebSphere® application serving.

1.1.1 zSeries Application Assist Processor (zAAP)

z/OS V1R6 supports the zSeries Application Assist Processor (zAAP) available on the IBM z990 and z890. zAAP is an attractively priced special processing unit that provides a Java™ execution environment for z/OS with the traditional qualities of service and the integration advantages of the zSeries platform.

1.1.2 24 processors in a single z/OS image

z/OS V1R6 supports up to 24 processors in a single z/OS image. This total can be made up of both Central Processors (CPs) and zAAPs. The sum cannot exceed 24.

1.1.3 64-bit support for C/C++

Application flexibility on z/OS is extended with 64-bit support for C/C++, enabling applications to scale and take advantage of the 64-bit programming model. This improvement is particularly important for new workloads on z/OS that require significantly larger addressability of data. Many customer business applications, Web serving applications, independent software vendor (ISV) applications, and internal IBM componentry is written in both C and C++.

1.1.4 TCP/IP Health Monitor

Additional enhancements include further improvements for TCP/IP availability in a sysplex, with the new TCP/IP Health Monitor.

1.1.5 RRS restart enhancements

There is improved application availability with Resource Recovery Services (RRS) restart enhancements. The RRS enhancements enable a resource manager to restart on any z/OS V1R6 image whenever the resource manager terminates.

z/OS V1R6 has the Program Number 5694-A01. You can order z/OS V1R6 electronically via ShopzSeries. Make sure you order the optional priced and unpriced features that you were using in previous releases of z/OS, as follows:

- The unpriced (relatively new) feature z/OS Security Level 3, which now includes the following:
  - OCSF Security Level 3
  - LDAP 31-bit Client Security Level 3 (a new FMID in V1R6)
  - LDAP 64-bit Client Security Level 3 (a new FMID in V1R6)
  - Network Authentication Service Level 3
  - System SSL Security Level 3, if you desire

- Communications Server Security Level 3

  This function is not part of z/OS Security Level 3.
- The HTTP Server NA Secure feature is now incorporated in the HTTP Server base element.
- Tivoli® NetView® and System Automation users take note as z/OS msys for Operations contains parts of these products.

1.1.6 Ordering z/OS V1R6 electronically

You may order z/OS V1R6 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 access it directly on the ShopzSeries Web site at:

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

Electronic delivery of ServerPac orders

You can now receive ServerPac orders by way of the Internet rather than tape. You can receive electronic delivery of z/OS, z/OS.e, and any of the products that run on them. Ordering is done through ShopzSeries. When the order is ready to download, you get the data required to download it from the ShopzSeries Web site and use it as input to the CustomPac Installation Dialog.

If you order z/OS V1R6 ServerPac, then you would want to use this latest level of SMP/E to process your electronic order. You may order SMP/E V3R3 using the ShopzSeries Web site:

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

You may download this SMP/E level from the download zone when needed; the download zone can be found at:

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

1.2 Base elements and feature changes

Table 1-1provides a summary of element, feature, and component name changes for additions and deletions in V1R6 of z/OS and z/OS.e.

Table 1-1 Changes to base elements and unpriced features in z/OS V1R6

Integrated Security Services	**Base element – New subcomponent LDAP Server 64-bit Client (JRSL362) added.**
z/OS Security Level 3	**Unpriced feature – New feature in z/OS R5 that contains Network Authentication Service Level 3; OCSF Security Level 3; and System SSL Security Level 3 components. New subcomponents LDAP 31-bit Client Security Level 3 (JRSL361); LDAP 64 bit Client Security Level 3 (JRSL363) added. In z/OS V1R6.**
IBM HTTP NA Secure	**Formerly unpriced feature; becomes part of IBM HTTP Server base element.**

1.2.1 Base elements and priced features deleted in z/OS V1R6

Table 1-2 on page 5 lists items that are removed effective with z/OS V1R6. These items were last shipped in z/OS V1R5. Consider these removals when making your plans for system upgrades. These statements represent the current intentions of IBM. IBM development plans are subject to change or withdrawal without further notice.

C/C++ ISPF panels

The C/C++ ISPF panels, which include panels for C/C++ foreground compiles, C/C++ background compiles, and Help panels for these compiles, are removed in z/OS V1R6. The z/OS C/C++ compiler can be invoked through z/OS UNIX, using JCL, and under TSO/E.

Run-time library services (RTLS)

z/OS base element Language Environment®'s use of run-time library services (RTLS) is withdrawn in z/OS V1R6. This function is used primarily in run-time migration. Given the stability and the upward compatibility being provided by the Language Environment run-time library in recent releases of OS/390 and z/OS, this functionality is no longer required.

Dynamic Link Library (DLL) Rename utility

The Dynamic Link Library (DLL) Rename utility, part of z/OS Language Environment, is removed in z/OS V1R6. This utility is used to package and redistribute IBM-supplied DLLs with applications. Since OS/390 V1R3, the C/C++ DLLs have been licensed with the OS/390 and z/OS base operating system. Therefore, the DLL Rename utility is no longer required.

TCP/IP Enterprise-specific MIB module

z/OS Communications Server support for the SMIv1 version of the SNMP IBM MVS TCP/IP Enterprise-specific MIB module is eliminated in z/OS V1R6. Support continues for the SMIv2 version of this MIB module. For customers who want to continue using SMIv1, publicly available tools can be used to convert an SMIv2 MIB module to an SMIv1 MIB module.

System SSL Java class interfaces

The System SSL Java class interfaces and JNI interface are removed in z/OS V1R6. The support within the System SSL Java class interfaces is becoming out of date and lacks the functionality provided by the new set of System SSL APIs. The direction for z/OS Java applications is to use the z/OS JSSE deliverable. JSSE on z/OS provides support for the suite of SSL and TLS protocols and has functional equivalency to System SSL.

DCE Application Support

Effective with z/OS V1R6, IBM removes the base element, Distributed Computing Environment (DCE) Application Support, from z/OS. DCE application support facilitated the interaction between DCE clients and CICS or IMS regions. With the continued evolution of technology and accompanying changes in the marketplace, there is no need for this support. If similar function is required, IBM recommends that customers use IBM WebSphere. The DCE Base Services element, which provides services for developing and running client/server applications, is planned to continue to ship with z/OS and z/OS.e.

Encina® Toolkit Executive

Effective with z/OS V1R6, IBM removes the base element, Encina Toolkit Executive from z/OS. Encina Toolkit Executive provided a set of tools for developing client components of distributed transactional applications. Over time, the marketplace has moved to other technologies. This element, an enabler for DCE Application Support, is another obsolete element of z/OS V1R6 and is no longer provided. There is no replacement.

Text Search

Effective with z/OS V1R6, IBM removes the base element Text Search. There is no replacement.

Table 1-2 Priced feature and base elements deleted in z/OS V1R6

C/C++ ISPF panels (from C/C++)	**Priced Feature - invoke the C/C++ compiler via UNIX, JCL, or TSO/E.**
Run-time Library Services (RTLS) (from Lang Env)	**Base Element - no longer required due to stability and upward compatibility**
Dynamic Link Library (DLL) Rename Utility (from Lang Env)	**Base Element - no longer needed due to C/C++ DLLs being licensed with the z/OS base**
SMIv1 version of the IBM MVS Enterprise-specific MIB module (from Communications Server)	**Base Element - if you want to continue to use SMIv1, publicly available tools can convert SMIv2 to SMIv1**
System SSL Java Class and JNDI support	**Base Element – For Java application, use the IBM Java Secure Sockets Extension (JSSE) product**
DCE Application Support	**Base Element - no replacement necessary. Evaluate WebSphere for similar function**
Encina Toolkit Executive	**Base Element - no replacement offered. Marketplace has moved to other technologies**
Text Search	**Base Element - no replacement offered**

1.2.2 Functions to be withdrawn in the future

The following functions are to be removed in a future release following z/OS V1R6; see Table 1-3 on page 7. When a removal date is determined, IBM will announce it. You are encouraged to consider these removals when making your plans for system upgrades. These statements represent the current intentions of IBM. IBM development plans are subject to change or withdrawal without further notice.

OROUTED

z/OS V1R6 is planned to be the last release in which z/OS Communications Server supports OROUTED. After z/OS V1R6, the function will be removed from the product. Customers should use OMPROUTE as their dynamic routing daemon.

DFSMS ISAM

Due to ISAM's limited functionality and the capabilities of VSAM, particularly VSAM data sets in extended format, z/OS V1R6 is planned to be the last release in which DFSMS ISAM and the utility program, IEABISAM, will be available. IBM has provided the ISAM Compatibility Interface (ISAM CI), which allows users to run an ISAM program against a VSAM KSDS data set. Details on using this interface and procedures for converting ISAM data sets to VSAM data sets can be found in Appendix E of *z/OS DFSMS: Using Data Sets,* SC26-7410. The order numbers for editions of this book are as follows:

- SC26-4922: For DFSMS/MVS® (any release)
- SC26-7339: For OS/390 V2.10
- SC26-7410: For z/OS

This compatibility interface program is planned to continue to be provided as part of DFSMS and will not be discontinued when ISAM is removed from DFSMS.

BIND DNS 4.9.3

z/OS V1R6 is planned to be the last release in which z/OS Communications Server will support BIND DNS 4.9.3. After z/OS V1R6, the function will be removed from the product. 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.

JOBCAT and STEPCAT facilities

IBM intends to remove the DFSMSdfp™ JOBCAT and STEPCAT facilities from base element DFSMSdfp in the future. The JOBCAT and STEPCAT facilities have been in existence for many years, predating the introduction of integrated catalog facility (ICF) catalogs. JOBCAT and STEPCAT were designed to address some of the functional shortcomings of VSAM catalogs, such as:

- VSAM volume ownership, that is, all data sets on a volume having to be in the same VSAM catalog. Multiple catalogs could not point to data sets on the same volume.
- Performance problems resulting from no multilevel alias support, as well as lack of ability to subset catalog data for recovery purposes.
- Restrictions in the definition of the catalog SVC interface.

The introduction of ICF catalogs in the mid-1980s and other catalog enhancements (such as the multilevel alias support) directly addressed those problems. In addition, processes were developed for system build to use system-specific aliases instead of JOBCAT or STEPCAT. CBIPO introduced these processes. They are used today by offerings such as ServerPac to create data set entries in the new master catalog of the system being built.

At the time ICF catalogs were introduced, the JOBCAT and STEPCAT facilities were functionally stabilized. Neither MS-managed data sets nor UCBs above the 16 megabyte line may be used with JOBCAT or STEPCAT. ICF catalogs contain sufficient functional capabilities so that all functions that previously could only be performed with JOBCAT or STEPCAT can now be done without them.

Furthermore, the use of JOBCAT and STEPCAT can actually cause significant problems. Data sets are generally not cataloged according to the normal predictable search order when JOBCAT or STEPCAT is used. This impacts the ability to do comprehensive installation storage management and can increase staff requirements. For example, interval migration and recall using DFSMShsm™ is effectively unusable when the data sets cannot be found using the standard catalog search order.

The use of JOBCAT and STEPCAT can also result in noticeable increases in the time required to perform catalog requests.

C/C++ compilers

From the optional features C/C++ with Debug Tool and C/C++ without Debug Tool, IBM intends to remove the OS/390 V2R10 level of the C/C++ compilers. The OS/390 V2R10 C/C++ compilers are shipped as an aid to migration to the C/C++ compilers that were introduced in z/OS V1R2. The z/OS V1R2 level of the C++ compiler supports the ISO 1998 Standard level of C++.

For information about migrating from the older to the newer level of the compilers, see *z/OS C/C++ Compiler and Run-Time Migration Guide for the Application Programmer,* GC09-4913.

AnyNet

In a future release of z/OS Communications Server, support for AnyNet® is planned to be discontinued and the function will be removed from the product. Customers may implement Enterprise Extender (EE) as the replacement for AnyNet.

Table 1-3 Functions to be withdrawn in the future

OROUTED (from Communications Server)	Base Element - use OMPROUTE as the dynamic routing daemon	After z/OS R6
ISAM and the utility program, IEABISAM (from DFSMS)	Base Element - ISAM Compatibility Interface still be provided (allows you to run an ISAM program against a VSAM KSDS data set)	After z/OS R6
Bind DNS 4.9.3 (from Communications Server)	Base Element - implement BIND 9.2.0 as a replacement (available since z/OS R4)	After z/OS R6
JOBCAT and STEPCAT facilities (from DFSMSdfp)	Base Element	Future
OS/390 R10 level of the C/C++ compilers (from C/C++)	Priced Feature - move to the ISO 1998 Standard level of the compilers (introduced in z/OS R2)	Future
AnyNet (from Communications Server)	Base Element - implement Enterprise Extender as a replacement	Future

1.3 Products related to z/OS v1R6

The following products have changes in them that affect the use of z/OS V1R6:

- IBM SMP/E for z/OS V3R3 (5655-G44) is incorporated in z/OS V1R6 (SMP/E is non-exclusive, which means it can be installed on older levels of z/OS) and is available 9/2004, at no charge to z/OS licensed users.
- IBM Debug Tool for z/OS V4R1 (5655-L24) is no longer incorporated, as of z/OS V1R5.
- ICKDSF R17 (5655-257) is incorporated into z/OS V1R6.
- z/OS V1R6 msys for Operations contains parts of:
  - Tivoli NetView for OS/390 V1R4 (5697-B82)
  - System Automation for z/OS V2R3 (5645-006)
- Tivoli NetView for OS/390 V5R1 (5697-ENV) can be ordered with, and is compatible with, z/OS V1R6 msys for Operations.

Note: Elements and features may be exclusive or nonexclusive, as follows:

- An element or feature is called exclusive to z/OS or z/OS.e if it exists only within z/OS or z/OS.e (not also as a separately orderable, or stand-alone, product) and if future functional enhancements will occur only within z/OS or z/OS.e.
- An element or feature is called nonexclusive if it exists both within z/OS or z/OS.e *and* as a stand-alone product.

1.3.1 SMP/E for z/OS Version 3 Release 3 (5655-G44)

Beginning with z/OS V1R2, SMP/E is nonexclusive because of the introduction of the SMP/E standalone product. SMP/E V3R3 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 or OS/390 to order and install the latest release of SMP/E without having to upgrade their entire operating system.

Note: 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.

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 V3R3 is available at no additional charge to customers and it will run on any currently supported operating system.

SMP/E V3R3 provides the following enhancements for the z/OS V2 Release:

- Utilize z/OS Communications Server FTP Client functionality for SMP/E RECEIVE FROMNETWORK operations.
- Extend GIMZIP and GIMUNZIP functionalities to support VSAM ESDS, KSDS, LDS, RRDS data sets, UNIX directories, and UNIX files.
- Enhance RECEIVE FROMNETWORK service routine to support the internet delivery of ServerPac. GIMTGPKG a new service routine is introduced to transport GIMZIP package from remote FTP server to z/OS host, it ensures integrity of package files and it support secure transmission.
- Utilize IEBCOPY COPYMOD support to reblock load modules to its destination data set's block size, preventing creation of fat blocks in a destination data set. SMP/E uses COPYMOD to copy ++MOD and ++PROGRAM elements.
- Extend RECEIVE sourceid function that it performs as ++ASSIGN function. The goal is to have the specified sourceid assigned to all PTFs in the input stream if they would have been received.

1.3.2 IBM Debug Tool for z/OS Version 4 Release 1 (5655-L24)

The latest available Debug Tool is the IBM Debug Tool for z/OS V4R1. IBM Debug Tool for z/OS Version 4 Release 1 is IBM's interactive source-level debugging tool for compiled applications. It is a program testing and analysis aid that helps you examine, monitor, and control the execution of application programs written in C, C++, COBOL, or PL/I on a z/OS or OS/390 system. By using the disassembly view, Debug Tool also provides support for programs compiled with the NOTEST compiler option, or applications that include other languages.

The Debug Tool supports debugging of application programs that run in the following environments:

- CICS
- IMS
- DB2
- WebSphere
- TSO
- JES batch
- UNIX System Services

Debug Tool for z/OS can be used in conjunction with the following products to debug z/OS and OS/390 applications from the workstation using the remote debug capabilities:

- WebSphere Studio Enterprise Developer
- VisualAge® for Java, Enterprise Edition for OS/390
- VisualAge PL/I for OS/390
- VisualAge COBOL for Windows® NT
- C/C++ Productivity Tools for OS/390

The Debug Tool for z/OS V4R1 provides the following functional and usability improvements:

- You can create a DDNAME to specify the location of the listing or source file.
- You can switch between hexadecimal and decimal display.
- The LIST cursor command has been enhanced.
- The FIND command no longer requires the use of quotes.

The IBM Debug Tool Utilities and Advanced Functions V4R1 has been enhanced to include:

- Debug Utility for IMS
- Usability enhancements
- Support for Language Environment assembler programs

Debug Tool Utilities and Advanced Functions V4R1

Debug Tool Utilities and Advanced Functions V4R1 is a separate optional product that builds on the function in Debug Tool V4.1, providing even more debugging capability for z/OS and OS/390 applications. For more information on Debug Tool Utilities and Advanced Functions V4R1, refer to the software announcement dated September 16, 2003.

1.3.3 IBM Device Support Facilities (ICKDSF) (5655-257)

ICKDSF Release 17 has been considerably enhanced. All previous enhancements, which include many direct access storage device and feature support capabilities, have been rolled up into this release, making it a comprehensive source for device support. So ICKDSF Release 17 now provides broad-based support for ESS, IBM's premier storage offering, with a wealth of copy services and disaster recovery capabilities.

The stand-alone version of ICKDSF is now offered on CD-ROM instead of diskette. Using it from a CD-ROM is much easier, and the enhanced capacity of CD-ROMs supports more ICKDSF function and makes room for expansion as new capabilities are added.

ICKDSF Release 17 also incorporates a broad range of high priority customer requirements, which includes many that have been requested for a number of years. Highlights include:

- All commands can be RACF protected (MVS version only).
- When an uncorrectable data check is detected during an ANALYZE SCAN, the data set name can optionally be printed out if the track in error resides within a data set.
- ICKDSF can check connectivity of devices to help verify, in many cases, whether or not they are cabled correctly.
- ICKDSF checks for the “VTOC full” condition when refreshing a VTOC or building an index.

1.3.4 msys for Operations

Parts of the following two products are included in msys for Operations:

- Tivoli NetView for OS/390 V1R4 (5697-B82)
- Tivoli System Automation for z/OS V2R3 (5645-006)

The System Automation base and Japanese FMIDs are incorporated in z/OS V1R6 if you already have the NetView and System Automation products installed (at the V1R4 and V2R3 levels, respectively).

You can install z/OS V1R6 or z/OS.e V1R6 (including msys for Operations) in the same SMP/E zone as the NetView and System Automation products.

In this case, it is recommended that you order the NetView and System Automation products in your z/OS V1R6 ServerPac. They can be installed in the same zones as z/OS V1R6, and do not require separate maintenance and duplication of service work (which they would if they were in separate zones).

However, if you have an earlier level of either the NetView or System Automation product installed, you have to put the product into a separate zone before installing z/OS V1R6, and maintain its data sets with different names than the z/OS V1R6 msys for Operations data sets. Use BUILDMCS to move the NetView or System Automation product or else you will have to reinstall it. Older levels of NetView and System Automation than what is included in z/OS V1R6 cannot be ordered with a z/OS V1R6 ServerPac.

If you plan on moving from z/OS V1R6 msys for Operations NetView to a full-function NetView V1R4, there is a sample job to assist you. This sample job enlarges the msys for Operations data sets to accommodate the extra space needed for a NetView V1R4 installation. For details, see *Tivoli Netview OS/390 Installation: Migration Guide Version 1,* SC31-8768.

2

ServerPac enhancements for z/OS Release 6

This chapter describes the enhancements to ServerPac in z/OS V1R6.

The following topics are discussed:

- Electronic delivery
- Improved processing for tape-delivered orders
- Reduction of the number of installation jobs
- Process orders in any sequence
- More data collection
- Eliminate unnecessary data entry
- Improved job statement handling
- Automatic reallocation of SCCPWORK data sets
- New and better SMP/E options entries
- RECEIVE dialog panel overview

2.1 Electronic delivery

The installation dialogs can now process electronically delivered orders. This eliminates the requirement for tape processing. Improvements in the Internet infrastructure make electronic delivery a more practical alternative to tape-delivered orders.

2.1.1 Overview of the electronic delivery process

The RECEIVE job uses the new SMP/E GIMGTPKG program to retrieve your order from the IBM FTP server. GIMGTPKG places the data into a file system data set. Later, the RESTORE job uses the SMP/E GIMUNZIP program to load your new system's volumes from the temporary file system. Upon completion, you can delete the temporary file system.

Figure 2-1 on page 12 shows the electronic delivery process.

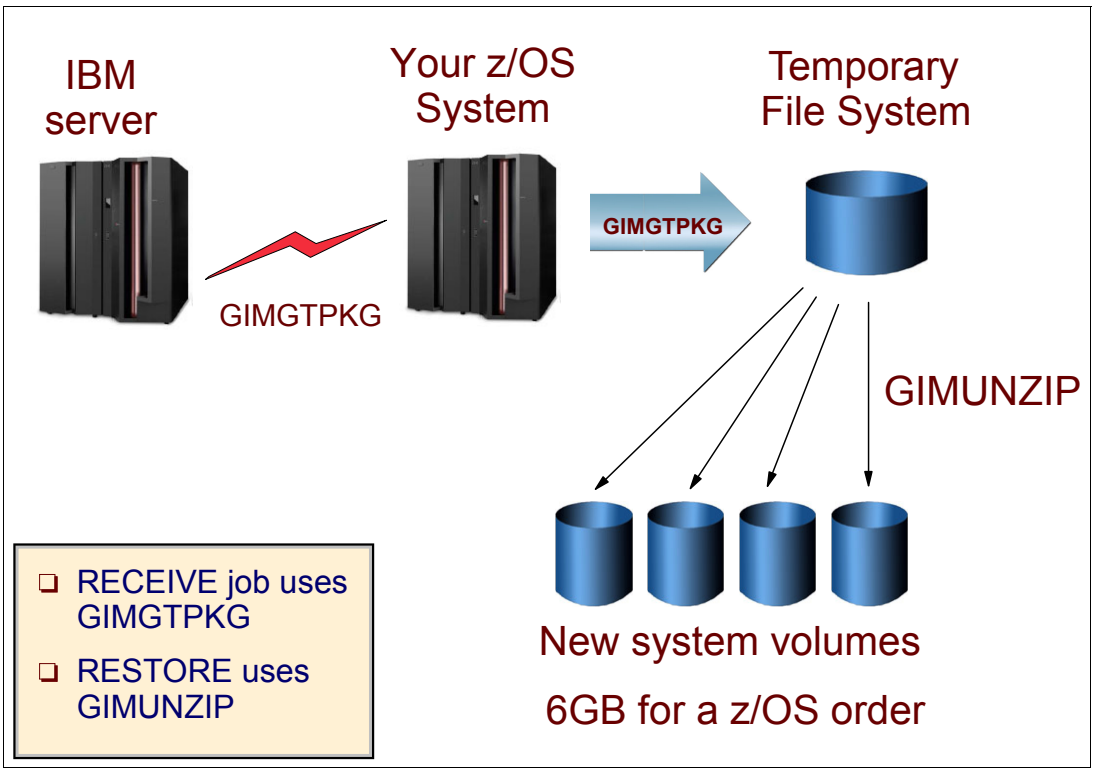

Figure 2-1 Electronic delivery process

2.1.2 Hardware requirements

The ICSF component of the Cryptographic services element of z/OS must be set up to download a ServerPac order. This is because the SMP/E GIMGTPKG utility uses ICSF's SHA-1 hashing to verify that the downloaded files are intact when you get them. ICSF, in turn, requires that cryptographic support be enabled on your processor.

Ensure that you have enough DASD space to download and process the order: about 6 GB for a z/OS order.

If downloading to an intermediate server or workstation, you need enough hard drive space to contain the package: about 5.5 GB for a z/OS order.

Note: Your workstation must be configured as an FTP server.

2.1.3 Software requirements

The software requirements for ServerPac are documented in *z/OS and z/OS.e Planning for Installation,* GA22-7504*1xxx*. However, there are some additional requirements for downloading a ServerPac order. Your system has to be at the following release levels:

- z/OS R3 or higher
- z/OS R4 or higher to download directly to the z/OS system through a firewall using Network Address Translation (NAT)
  - With APAR PQ80281 (PTF UQ82394 or UQ82395)
- SMP/E 3.3

  Provides the new GIMGTPKG utility used to download the packages.
- ICSF

  SHA-1 provides a hashing algorithm to compare the hash value generated at IBM with the one generated on your system and assures that what you receive is what was sent.
- Communications Server IP set up for Secure FTP (FTPS) using Transport Layer Security (TLS)
- RACF set up to support key rings used for Secure FTP with TLS

Note: Once the software requirements are met, ICSF, FTPS, and RACF setup is needed to download a package.

2.1.4 Electronic delivery work flow overview

All ordering for electronic delivery of ServerPac is done using the ShopzSeries Web site at:

`https://www14.software.ibm.com/webapp/ShopzSeries/ShopzSeries.jsp`

Once you make your product selections, place your order as shown in Figure 2-2 on page 14. You should receive an order confirmation within a few hours. After that, you will have to wait for the order to be built by IBM and placed on the download server. Depending on what you order, this might take up to 10 business days. Once the order is ready to be downloaded, you will get another note saying that it is ready. This second note will contain a link to the download page you need to use to get information for proceeding with the next step.

Figure 2-2 Electronic delivery work flow (1 of 5)

Go to the download page shown in Figure 2-3. It contains the information you need to download your order. Record the following data for use in the installation dialog:

- The order number
- The FTP server name
- The source directory
- The FTP user ID you need to download the order
- The FTP password you need to download the order
- The hash value that will be used to check what you download to make sure it's what we sent

Figure 2-3 Electronic delivery work flow (2 of 5)

Using the installation dialog, create the RECEIVE job for the order using the information from the download page as shown in Figure 2-4. When the job starts, it begins phase 1 of the download. Phase 1 downloads the information about your order, and the updated installation dialog.

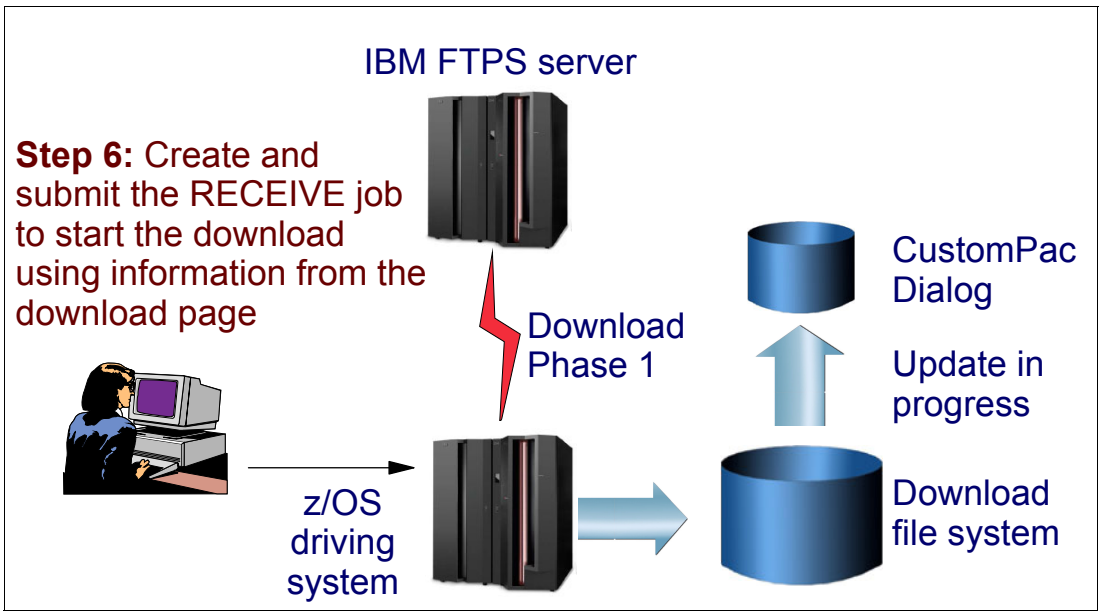

Figure 2-4 Electronic delivery work flow (3 of 5)

For “reasonable” connection speeds, phase 1 should finish fairly quickly. The files in phase 1 total about 30 MB before compression.

When phase 1 finishes, the order shows up in the dialog, and you are then able to select it for installation. Also, the RECEIVE job sends a NOTIFY message to the submitter’s TSO/E user ID as shown in Figure 2-5 on page 16. You need to have INTERCOM set in your TSO/E profile to get the message. Do this by issuing the PROFILE INTERCOM command from ISPF Option 6.

Once phase 1 has finished, you can begin to use the dialog to configure the order. Phase 2 of the download proceeds in the background to get the remainder of the data for your order and place it in the download file system.

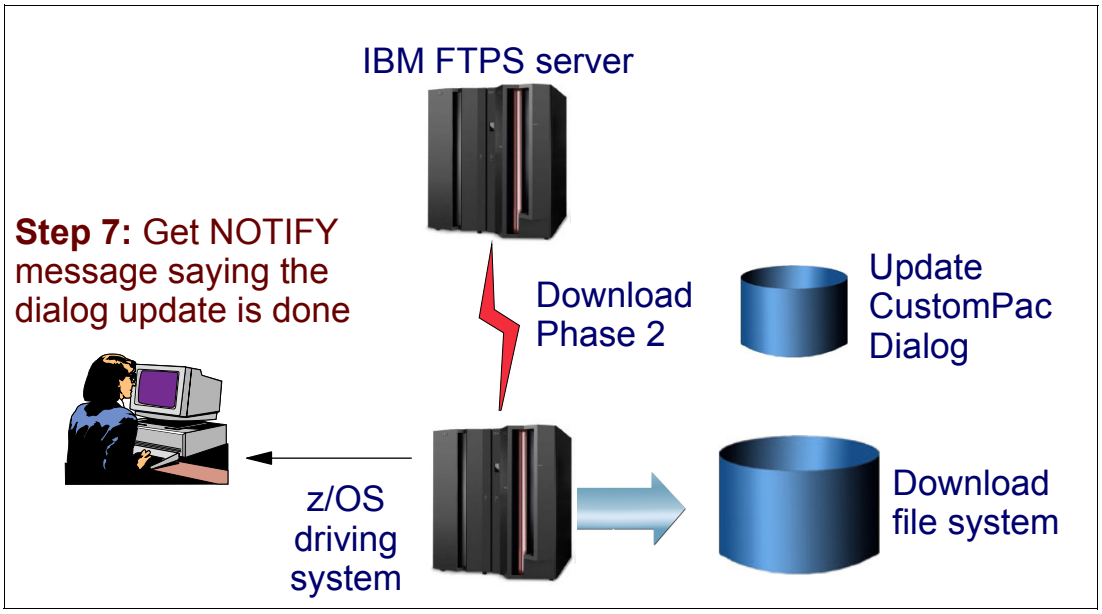

Figure 2-5 Electronic delivery work flow (4 of 5)

You can begin to work with the order while phase 2 completes (refer to Figure 2-6), up to the point where you are ready to generate the installation jobs. However, the installation job option is not enabled in the dialog until phase 2 has finished. Since phase 2 downloads most of the data (about 5-6GB before compression), it is likely to take considerably longer than phase 1 to complete. Once the RECEIVE job has finished and the Installation option has been enabled in the dialog, you can finish installing the order.

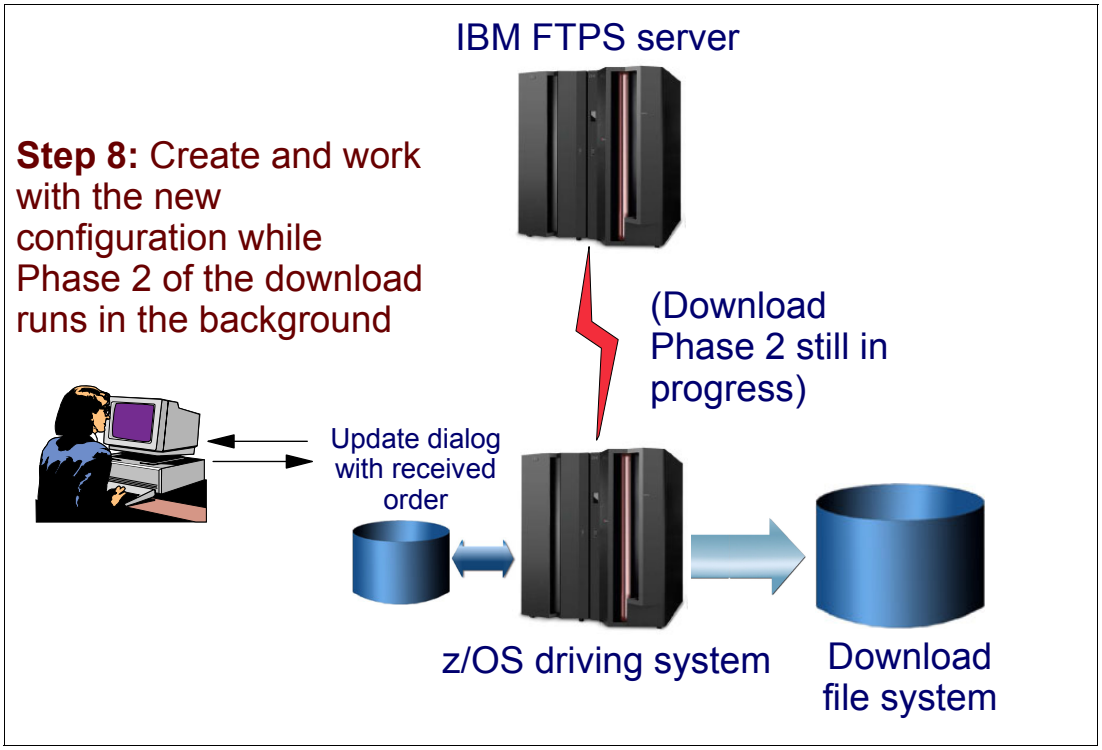

Figure 2-6 Electronic delivery work flow (5 of 5)

2.1.5 ShopzSeries panels for electronic delivery

Figure 2-7 is an example of a download page. There are links to the packing list and publications. Follow the links to download a ServerPac order. There are four ways to do this. We will look at the “download directly to host” way first.

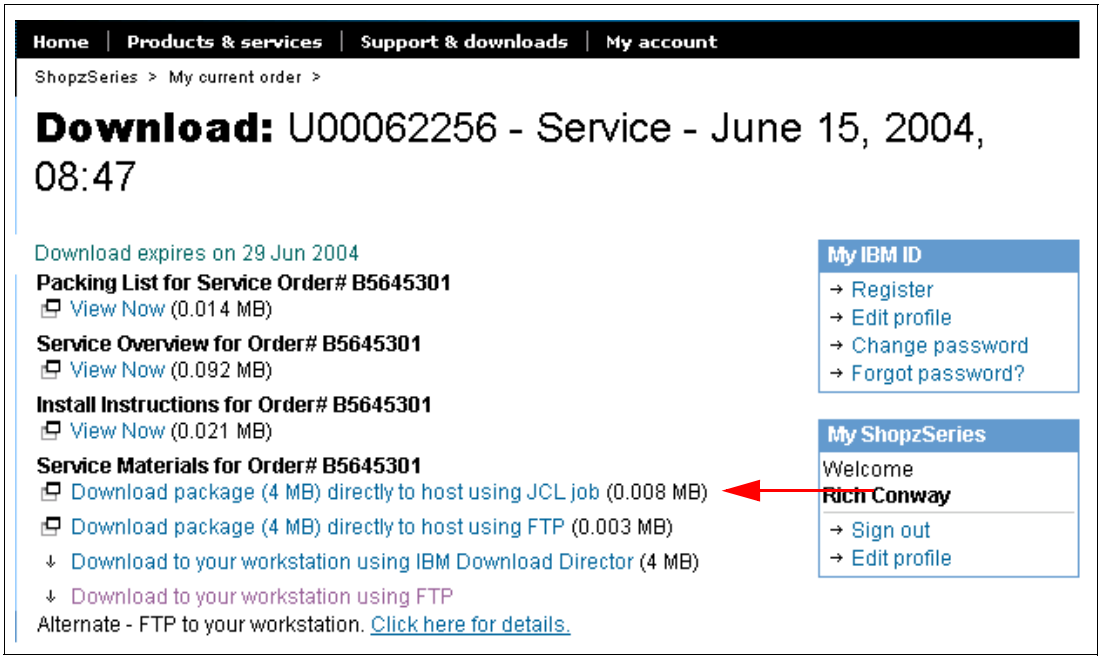

Figure 2-7 ShopzSeries panel (1 of 4)

When you click on the link for downloading directly to your z/OS system, you will see Figure 2-8 on page 18. The second arrow shows the link to the information you need to enter in the CustomPac Installation Dialog to initiate the download. This is a text file that will contain the order number, server name, source directory, FTP user ID and password, and the hash value needed for the download. You should either download this file to your workstation or copy and paste the information to a file on your workstation or on your z/OS system so that it can be copied and pasted into the installation dialog.

The third arrow points to a sample job you can download, edit, and submit to install a new copy of the CustomPac Installation Dialog. Since saved configurations cannot be accessed from a new copy, this job is recommended for new ServerPac customers only.

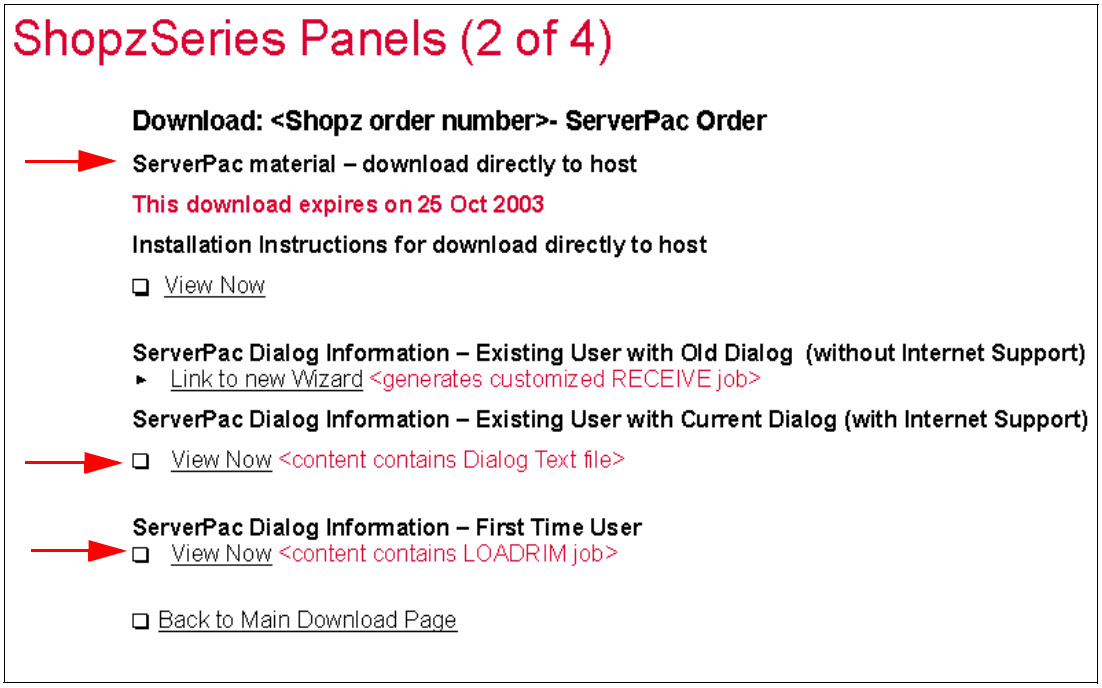

Figure 2-8 ShopzSeries panels (2 of 4)

If you cannot download directly to your z/OS system, you need to have picked the second link on the first download page, which brings you to Figure 2-9. This page links to the Download Director, which you can use to download your ServerPac order to your workstation.

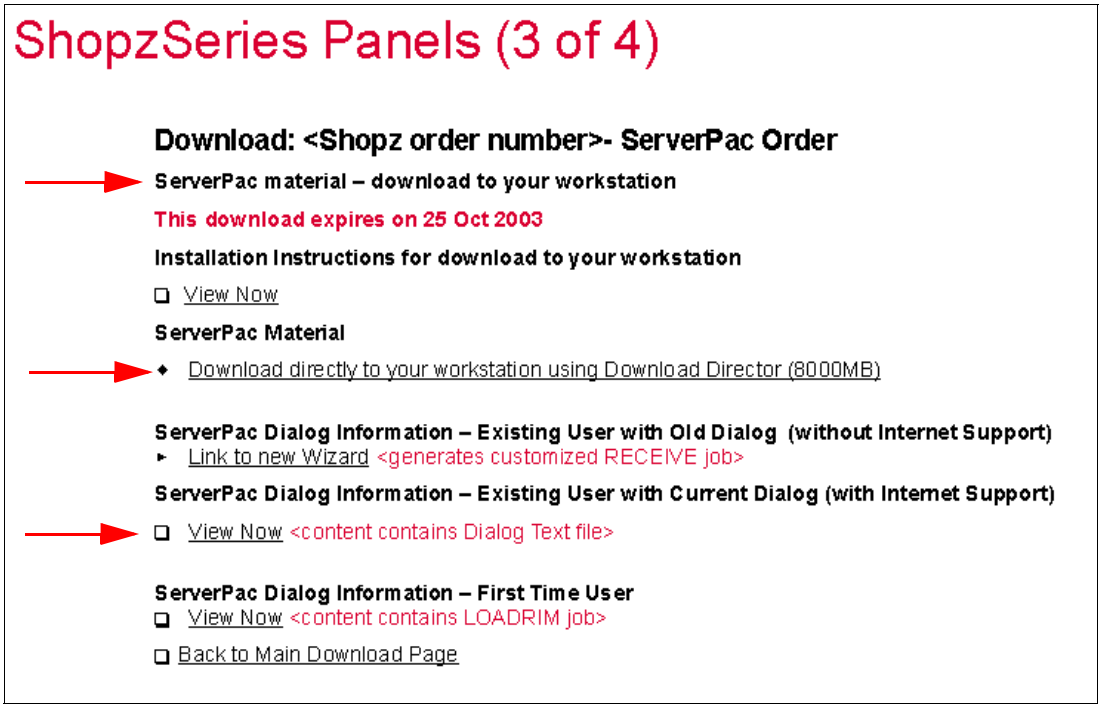

Figure 2-9 ShopzSeries panels (3 of 4)

Figure 2-10 on page 19 shows the information you will need to create the RECEIVE job.

Note: We highly recommend copy/paste to transfer these fields from the download page to the installation dialog.

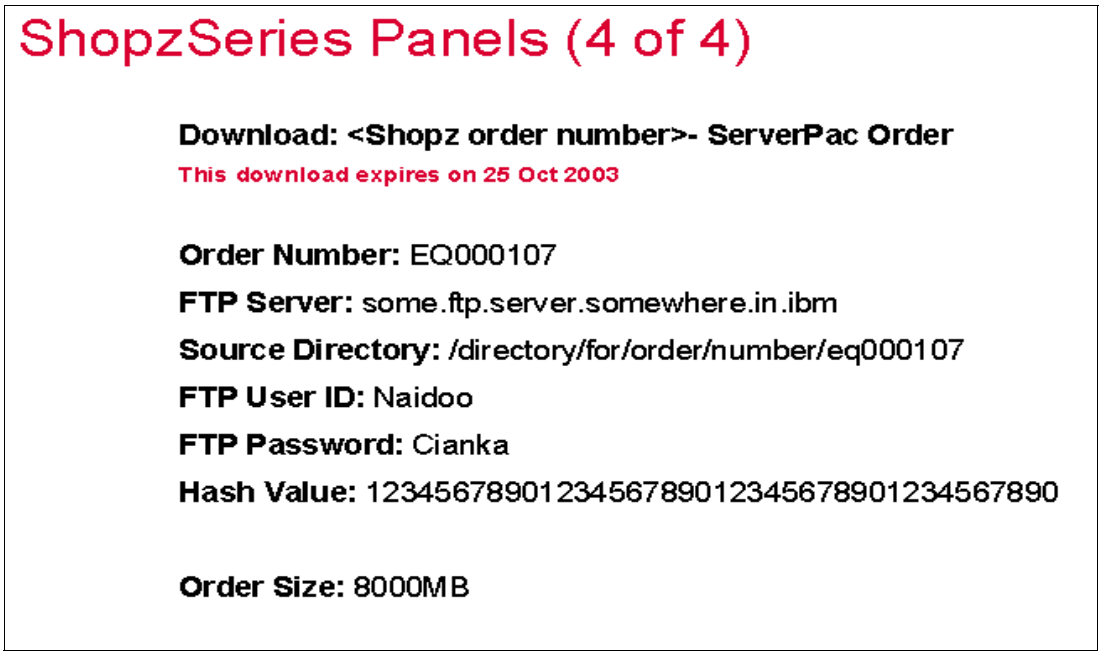

Figure 2-10 ShopzSeries panels (4 of 4)

2.1.6 Overview of store-and-forward download

Use the store-and-forward procedure if your z/OS system cannot connect to the Internet and you have an FTP server that can connect to the Internet. Use the download director (find the link on the download page) to download your order from the IBM server to your server as shown in Figure 2-11.

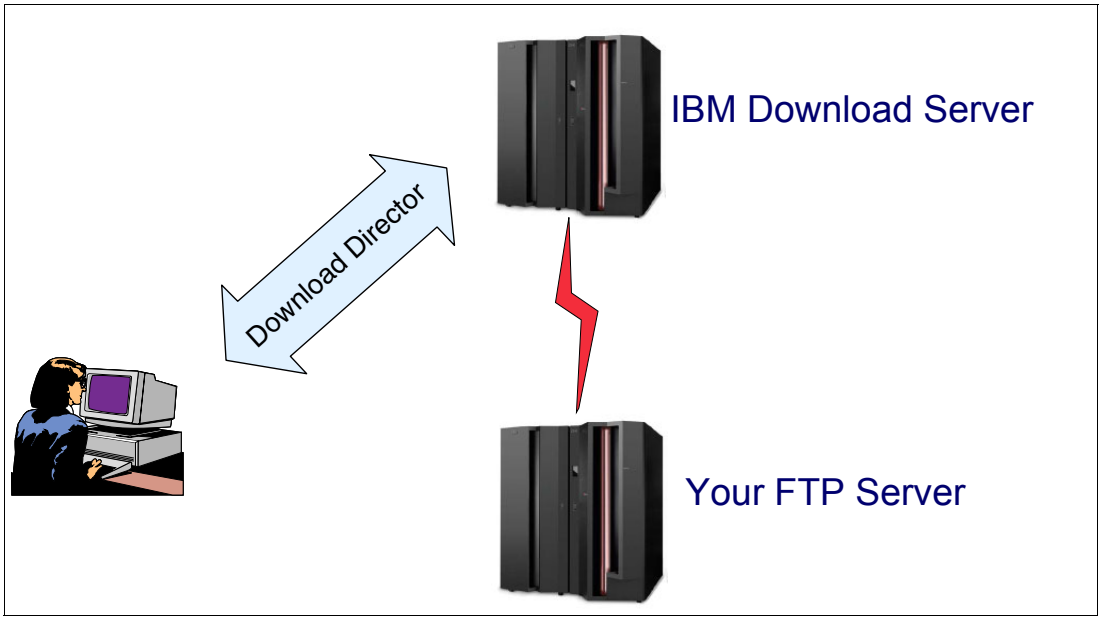

Figure 2-11 Store-and-forward download (1 of 2)

Once your order is on your FTP server, use the CustomPac Installation Dialog to generate the RECEIVE job, as shown in Figure 2-12. Specify your server as the source server, with the appropriate user ID, password, and source directory.

Note: Always specify the hash value from the download page no matter which server is used to do the download to the z/OS system.

From there on, it's the same as if you were using an IBM server. The advantage of using store-and-forward is that your z/OS system need not be connected directly to the Internet. The disadvantage is that you cannot begin to work with your order until the entire package has been placed on your FTP server and Phase 1 of the RECEIVE job has run on your z/OS system.

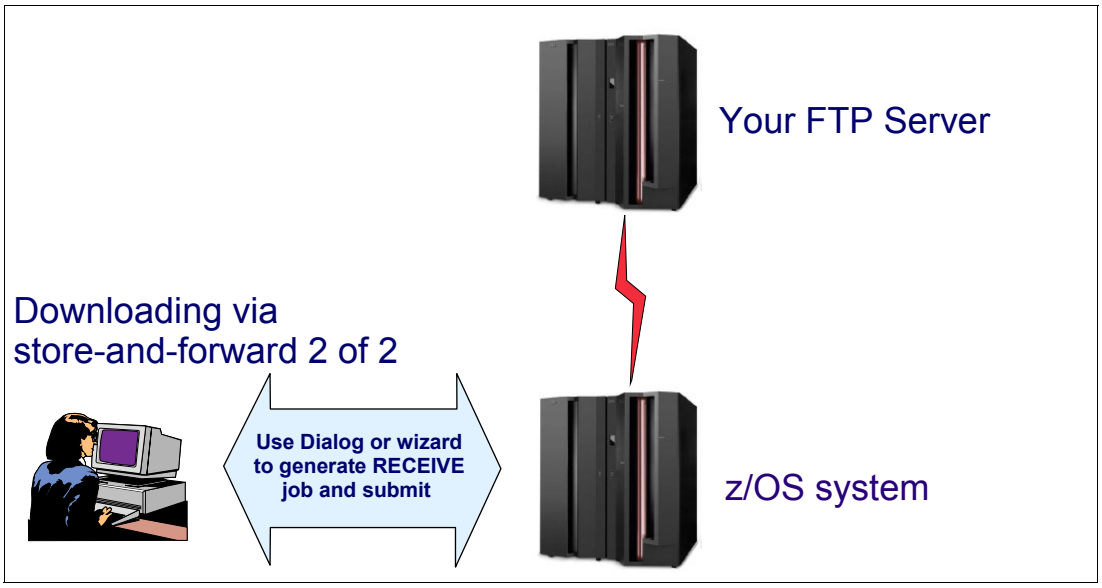

Figure 2-12 Store-and-forward download (2 of 2)

2.1.7 Migration to the new dialog

For new downloaded orders, the dialog is updated by RECEIVE. For new orders delivered on tape, run the UPDATE job (just once) to teach the dialog about the new RIM tape format. It can be obtained from:

- The DOCLIB data set on the RIM tape
- The z/OS Installation Web page

Note: If the first screen says "This dialog supports electronic delivery," then you do *not* need to run the UPDATE job.

To work with older orders, you will have to convert their SCPPEENU and SCPPHENU data sets from sequential to VSAM. There is a sample job in CPAC.SAMPLIB to help with this, named CPPCINV. This can be done right in the middle of installing an older order so that the new order can be installed, or at any time if you want to display saved configurations from older orders.

If you got your order on tape and the first dialog panel does not say “This dialog supports electronic delivery,” as shown in Figure 2-13 on page 21, then:

- Exit the installation dialog.
- Unload the UPDATE job from the DOCLIB data set on the RIM tape.
- Run the UPDATE job.
- Go back into the installation dialog and use it to generate the RECEIVE job.

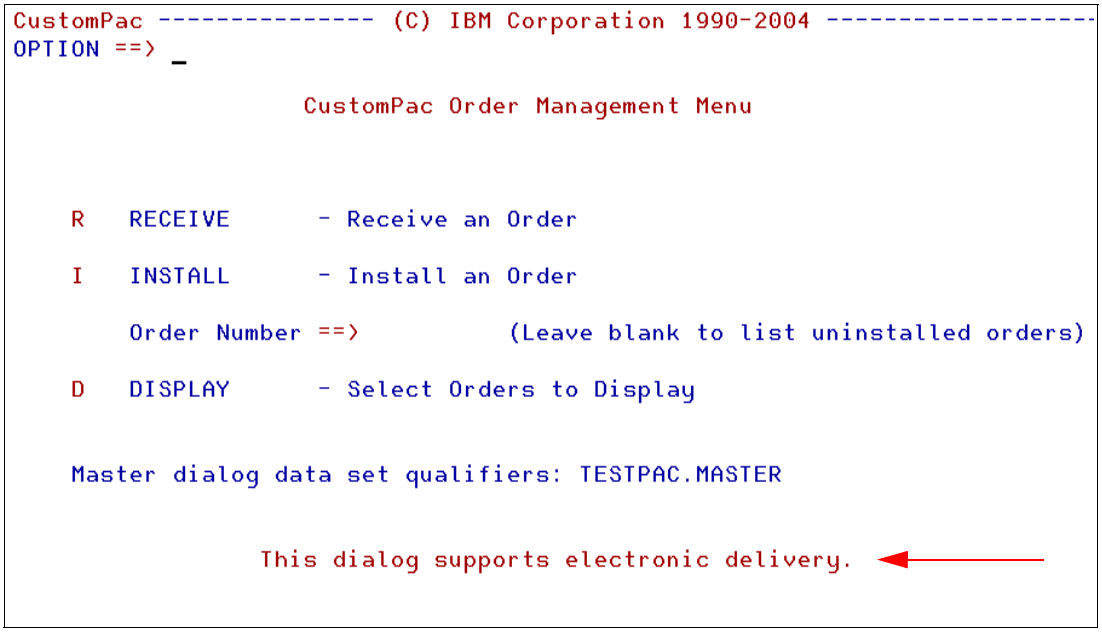

Figure 2-13 Main installation panel

2.1.8 Coexistence

The CustomPac Installation Dialog did not really provide coexistence support prior to z/OS R6. It allowed you to display orders from prior dialog levels, but not install them. Now, the level of the dialog shipped with a ServerPac is always used to install it. The dialog update process updates only the master dialog, which is now used solely for order management functions and not for order installation. Therefore, there are no restrictions on when an order can be installed, and the master dialog update process does not disrupt the installation of an order already in progress.

The CustomPac Installation Dialog now provides full coexistence support:

- ServerPacs can be installed in any order, no matter when they were built.
- The dialog upgrade is nondisruptive even when the installation of other orders is in progress.

Note: If you install z/OS R6 with a ServerPac, you can still finish the installation of a CICS order that was built previously.

2.2 Improved processing for tape-delivered orders

For orders delivered on tape, you now have to mount the RIM tape only one time (rather than six).

2.3 Reduction of the number of installation jobs

The following jobs have been removed:

- Autoupgrade job
  - Now truly automatic.
  - If an upgrade is needed, the master dialog will be updated by the RECEIVE job.
- UNLODOC

  Now part of the RECEIVE job
- UNLDSCPP

  Now part of the RECEIVE job
- UNLDBOOK

  Now part of the RECEIVE job
- LOADCSI

  Now part of the RESTORE job
- RESTFS

  Now part of the RESTORE job
- DELTRANS
  - The CPAC.HFSFILE data set that used to take up space on your DASD is removed.
  - The RESTORE job will restore the pax archive for the UNIX System Services file system data sets directly from tape.
  - This will save about 1,300 cylinders of space during installation and the trouble of deleting it with the above DELTRANS job afterward.

2.4 Process orders in any sequence

Previously, you had to install ServerPacs by package version in ascending order, and installing them by production date was recommended. With the new dialog design, the master dialog is used only for order management functions. Once you select an order to work on, its own dialog data sets are used to process it.

Now you can install ServerPac in any order you want:

- Install a new order and go back to install an older one.
- Install a new order and display an older one without “missing” values in panel fields.

2.5 More data collection

Because ServerPac runs on your z/OS system, the dialog can retrieve your ISPF data set concatenations and your TSO/E user ID, and now it does. The dialog processes each package type appropriately, and no longer asks you whether a package is a ServerPac or SystemPac®.

More data collection is done to eliminate data entry for:

- ISPF concatenation
- TSO/E user ID
- Package type

2.6 Eliminate unnecessary data entry

Already-known-data has been reused to eliminate unnecessary data entry for:

- The Order HLQ
- The Master HLQ

Values entered once to generate the RECEIVE job are saved now and used later so they don't have to be entered twice.

2.7 Improved job statement handling

Everyone usually has a JOB statement available or knows how to write one, so rather than collect all the data needed to build it, a default JOB statement in an ISPF Edit session is displayed so you can copy one of yours into it or modify it when building the RECEIVE job. It is then saved in the skeleton data set for reuse when generating the installation jobs. It's an editable member in the installation option's display of installation jobs, so you can change it later if you want to.

2.8 Automatic reallocation of SCCPWORK data sets

Previously, there were instructions telling you to delete the CPPTEMPx.SCPPWORK data sets so the dialog could reallocate them with the necessary space. Now, they are automatically deleted and reallocated to avoid the manual task or space abends.

2.9 New and better SMP/E OPTIONS entries

The SMP/E OPTIONS entries have also been brought up to date. Now RETRYDDN(ALL), MSGFILTER(YES), MSGWIDTH(80), and COMPACT(YES) are set. Also, a RECZGRP is defined for the zones in the ServerPac to avoid re-receiving already-installed PTFs.

2.10 RECEIVE installation dialog panel overview

When you select "Receive an Order" from the main menu, Figure 2-14 on page 24 is the next panel you will see. This panel collects the information that is common to all orders, whether it is downloaded or delivered on tape.

2.10.1 RECEIVE from server

In this case, we'll receive the order from a server. The rest of the panels will be pretty straightforward. However, note that there is no data entry field for "unit" if you enter a volume

serial. The dialog will retrieve the unit information from the system so that you don't have to enter it.

Note: The volume serial you enter here must be online.

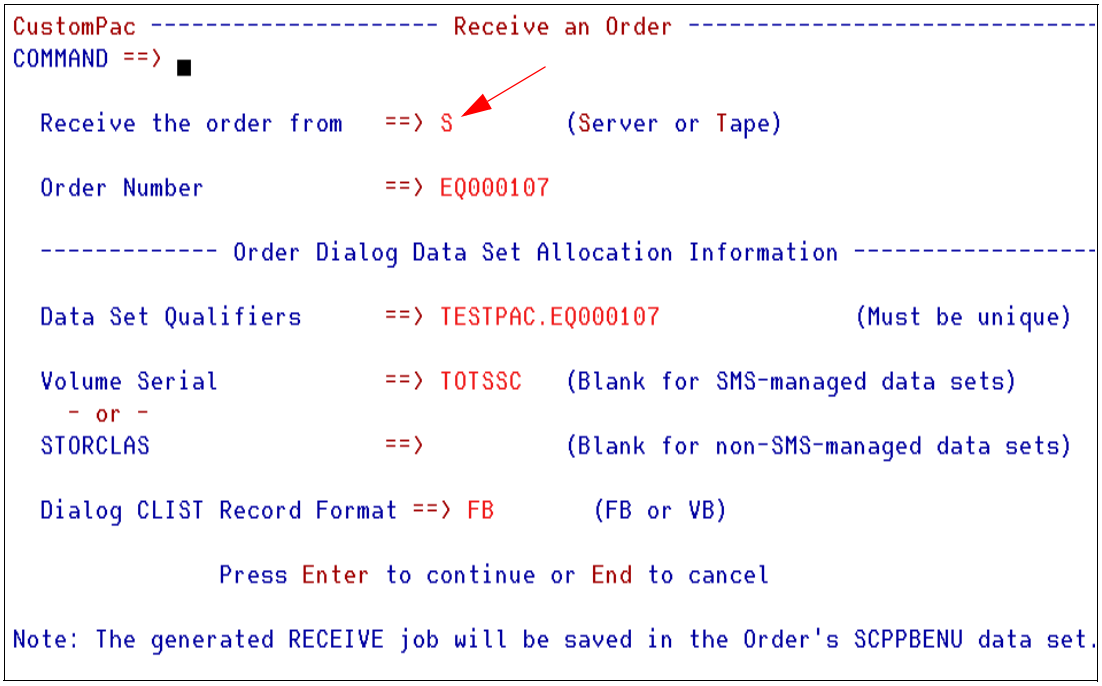

Figure 2-14 RECEIVE installation dialog (1 of 7)

Since we selected to receive the order from a server in the last panel, we need to tell the dialog where the server is and also give it some information it needs to retrieve the order, as shown in Figure 2-15 on page 25. The server name or address can point to an IBM server, if you will download directly to your z/OS system, or to your own server if you used Download Director to get the order from the IBM server, and will download it from there to your z/OS system.

The source directory is the location on the server that contains your ServerPac files. The user ID and password are used to establish the FTP session. The hash value is used during the download to compare the one generated at IBM to the one generated on your system. If they are equal, then what was sent is almost certainly what you received. If they do not match, then the downloaded data is corrupted, and you need to try again.

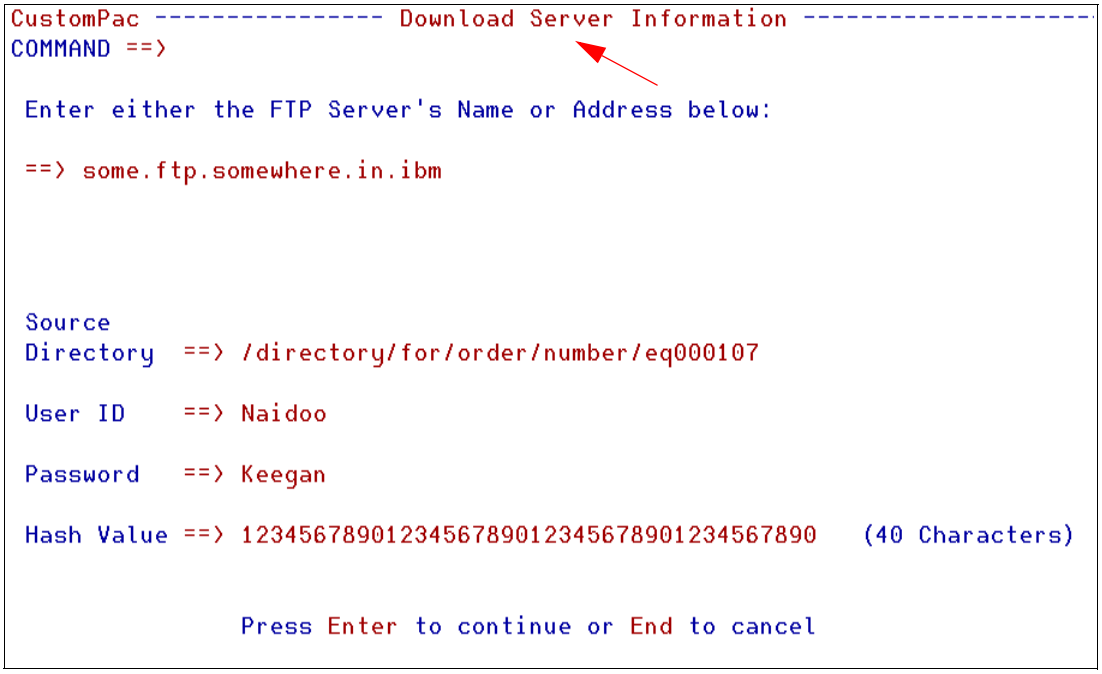

Figure 2-15 RECEIVE installation dialog (2 of 7)

Now that the dialog knows where to get your ServerPac order from, you need to tell it where to store it temporarily while you work with the configuration and generate the installation jobs. On this panel, shown in Figure 2-16 on page 26, specify the destination directory for the download. If you want the dialog to add a job step that creates a new file system data set and mount it, then say “yes” for “Allocate a new file system data set,” and fill in the fields below the line “New File System Data Set Information.”

Make sure that there is enough space for the order, plus enough space to unpack the largest non-VSAM, non-file system data set in the order. Depending on when you order with respect to the RSU cycle, the largest data set might be the SMPPTS. You should add at least 500 cylinders to the amount of space required to download the order to allow the largest data set to be unpacked later.

The download page shows the space required for the order, in megabytes. To convert to 3390 cylinders, multiply the number of MB by 1.4 and add at least 500.

If the download page says the order is 5,000 MB, then:

((5,000 MB) * (1.4 CYL/MB)) + 500 cylinders = 7,500 cylinders

Note: You will need to have defined a volume of 7500 cylinders or larger to download a 5,000 MB order to a single-volume file system data set.

```
 CustomPac ------------ Download File System Information ------------------
 COMMAND ==> _

Target Directory ==> u/naidoo/serverpac/zosr6

Allocate a new file system data set ==> NO   (Yes or No)


--------------------- New File System Data Set Information -----------------

 Data Set Name    ==> naidoo.serverpac.zosr6.download
 Mount Point      ==> /u/naidoo/serverpac/zosr6
 File System Type ==> HFS       (HFS or zFS)
 Primary Space    ==> 7500      (Minimum 500 cylinders)
 Secondary Space  ==> 1000      (Secondary space recommended)

 Volume Serial    ==>           (Leave blank for SMS)
   - or -
 STORCLAS         ==> ussclass  (STORCLAS required for multivolume data set)

              Press Enter to continue or End to cancel
```

Figure 2-16 RECEIVE installation dialog (3 of 7)

Assuming that most z/OS systems that connect to the Internet have firewalls, we'll answer "Yes" to "Do you want to enter firewall commands", as shown in Figure 2-17.

```
CustomPac --------------------- Firewall Commands -------------------
COMMAND ==> _



    Do you want to enter Firewall Commands? ==> YES (Yes or No)



            Press Enter to continue or End to cancel
```

Figure 2-17 RECEIVE installation dialog (4 of 7)

The firewall commands must be entered in a format that the GIMGTPKG program understands, which is in the form of a <FIREWALL> tag and its subtags. A syntactically correct example of a <FIREWALL> tag is provided, as shown in Figure 2-18 on page 27. Remove the XML comment delimiters (<!– and -->) and edit as needed if you have anything that needs to be passed to the firewall.

```
CustomPac - EQ000107 ---------------------------------------- COLUMNS 000 000
COMMAND ==> _                                                 SCROLL ==> PAGE
****** ***************************** Top of Data ******************************
000001 <!--                                                              -->
000002 <!--  Specify needed firewall information and commands below.     -->
000003 <!--  When you have finished, press End to display and submit      -->
000004 <!--  the RECEIVE job.                                            -->
000005 <!--                                                              -->
000006 <!--  Or, use the CANCEL command to return to the prior panel.    -->
000007 <!--                                                              -->
000008 <!--  Notes: 1. All tags below are commented out.  Remove the     -->
000009 <!--            XML comment start and end tags to specify         -->
000010 <!--            firewall information.                             -->
000011 <!--         2. The tags here are defined by SMP/E.  For          -->
000012 <!--            information about them, see SMP/E Commands.       -->
000013 <!--         3. Do not include <CLIENT> tags in this data set.    -->
000014 <!--            They are built automatically by the dialog.       -->
000015 <!--                                                              -->
000016 <!--    <FIREWALL>                                                -->
000017 <!--      <SERVER                                                 -->
000018 <!--        host="host name|host ip address"                      -->
000019 <!--           user="userid"                                      -->
000020 <!--           pw="password"                                      -->
000021 <!--           port="port number"                                 -->
000022 <!--           account="account information">                    -->
000023 <!--      </SERVER>                                               -->
000024 <!--      <FIRECMD>                                               -->
000025 <!--        firewall specific command                             -->
000026 <!--      </FIRECMD>                                              -->
000027 <!--    </FIREWALL>                                               -->
000028 <!--                                                              -->
```

Figure 2-18 RECEIVE installation dialog (5 of 7)

After the panel in Figure 2-19 on page 28, the dialog will put you in an ISPF Edit session where you can edit the default JOB statement, or just copy in your own JOB statement and save it. This JOB statement will also be used to generate the installation jobs later, but it can be changed by selecting it from the installation option's job list and editing it there. Since you've seen an edit session before, we won't show you another one.

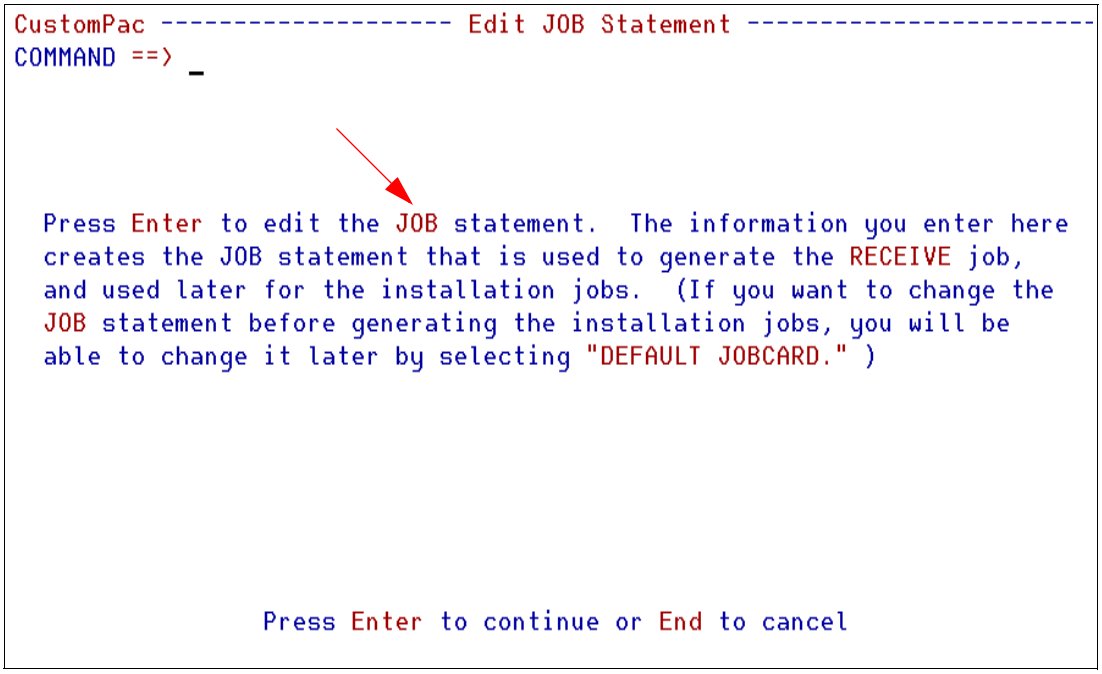

Figure 2-19 RECEIVE installation dialog (6 of 7)

Having modified the JOB statement as needed, the next thing to do is to generate the RECEIVE job so you can submit it. Pressing Enter, as shown in Figure 2-20, will display it in ISPF Edit. One of its steps will save a copy of the job as originally generated in the SCPPBENU data set, so if something goes wrong or you just want to see what was generated, you have a copy without needing to generate it again. If you modify the job before submitting it, though, the modified job will be submitted but it will not be saved.

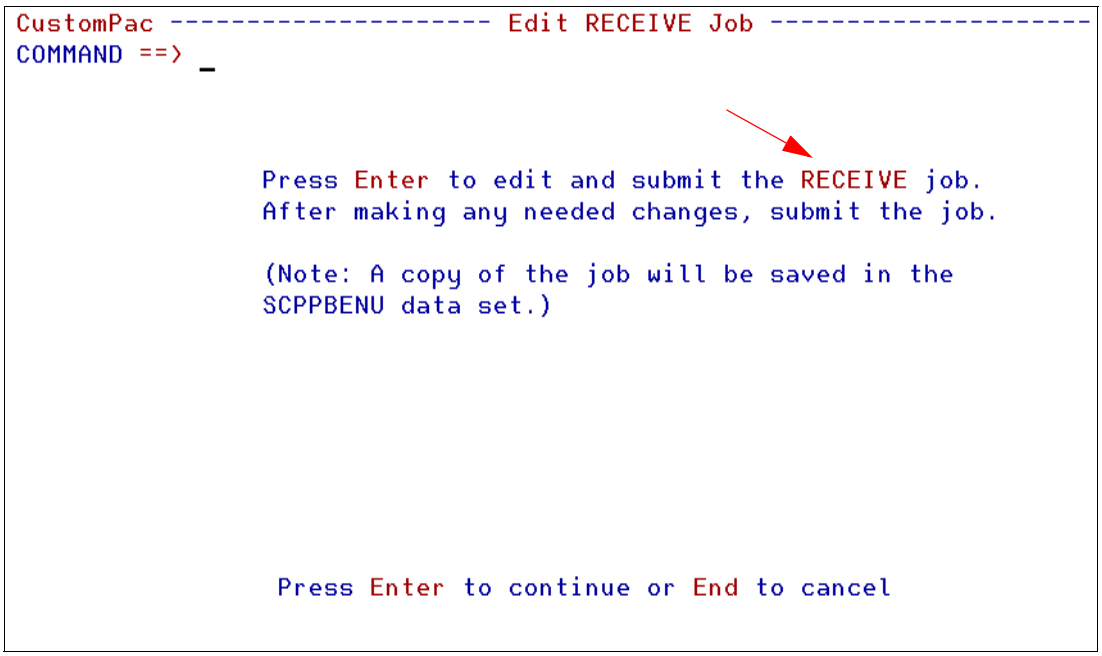

Figure 2-20 RECEIVE installation dialog (7 of7)

2.10.2 RECEIVE from tape

This time, after “Receive the order from,” we’ll say “T” for “Tape”. The rest of this panel (Figure 2-21) is the same, no matter how you receive your ServerPac.

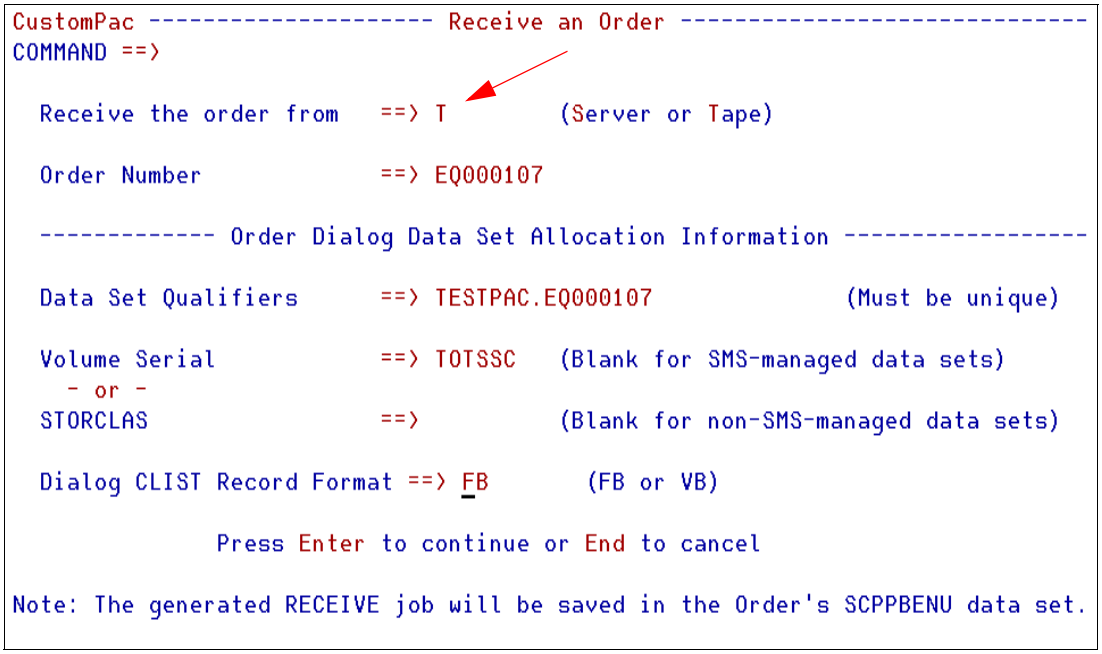

Figure 2-21 RECEIVE from tape (1 of 2)

The panel in Figure 2-22 collects the information we need to run the RECEIVE job from tape, namely the RIM tape volume serial and tape unit. The edit job statement introduction panel, subsequent edit session, RECEIVE job introduction panel, and subsequent edit session are the same as for orders to be downloaded from a server.

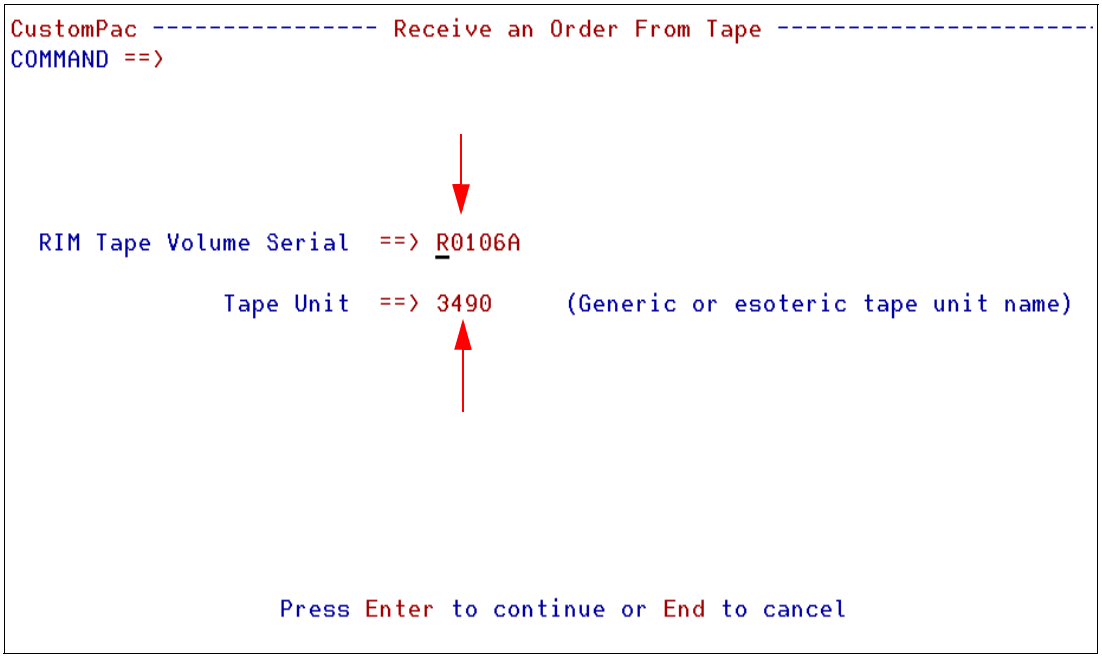

Figure 2-22 RECIEIVE from tape (2 of 2)

3

z/OS V1R6 base control program (BCP) enhancements

The base control program (BCP) provides essential operating system services. It includes the I/O configuration program (IOCP), the workload manager (WLM), system management facilities (SMF), the z/OS UNIX System Services (z/OS UNIX) kernel, and support for Unicode.

As of z/OS V1R3 and z/OS.e V1R3, the BCP also includes the program management binder, which was formerly in the DFSMSdfp base element. Previous versions of z/OS established basic support for 64-bit operation. z/OS V1R6 continues to build on the basic support by enabling additional system components to exploit the new architecture. It also provides expanded capabilities that include a fully functional 64-bit environment for C/C++ applications that facilitates application program and middleware growth into the 64-bit virtual environment. This chapter focuses on the enhancements provided with the z/OS V1R6 base control program (BCP), as follows:

- System logger - 64-bit virtual support
- RRMS 64-bit callable services
- RRS - restart anytime/anywhere
- SMF buffer constraint relief
- GRS enhancements
- 2047 members per XCF group
- Service aids enhancements
- MVS allocation enhancements
- New JCL keywords for PSF
- Unicode enhancements
- Greater than 16 CPU support
- Support for zAAP
- Binder enhancements
- Linkage index reuse
- Relocate structure after CF maintenance

3.1 System logger - 64-bit virtual support

Prior to z/OS V1R6, the System Logger services API could only be called in 31-bit mode. 64-bit exploiters had to operate in bimodal mode. Additionally, there was no support for user data areas above the 2 GB bar.

Under z/OS V1R6, the System Logger services API can now be called in 64-bit mode. 64-bit applications can call logger services directly without switching addressing modes. They no longer need to be bimodal. Also, several of the logger services that handle high volume and large data buffers allow those buffers to utilize storage above the 2 GB bar. However, with one exception for the new BUFFER64 keyword, all addresses for user data areas and parameter lists must still be 31-bit. In order to use these services in 64-bit mode, it is necessary to reassemble the module that invokes them.

New keyword: BUFFER64

The new BUFFER64 keyword has been added to the following system logger services. Each of these APIs deals with high volume buffers containing large amounts of user data.

- IXGBRWSE
- IXGIMPRT
- IXGWRITE

These buffers can now be placed in storage above the 2 GB bar to provide storage constraint relief. Each service has its own length requirement for the buffer and the size of the buffer accepted has not changed.

The existing BUFFER keyword can only be used for 31-bit addressing. The BUFFER64 keyword can contain any valid 64-bit address, whether it be above or below the bar. The two keywords are mutually exclusive.

You can call any of the system logger services in AMODE 64, but the parameter list and all other data addresses, with the exception of BUFFER64, must reside in 31-bit storage.

IXGBRWSE service This service supports the new BUFFER64 keyword. IXGBRWSE is used to browse (read) log data from a log stream. Using IXGBRWSE, a program can read consecutive log blocks in a log stream or search for and read a specific log block in a log stream. Log data is returned into the caller's buffer (BUFFER64) at the specified address, which can be any valid 31- or 64-bit address. The keyword is valid on REQUEST(READCURSOR) and REQUEST(READBLOCK).

IXGIMPRT service This service supports the new BUFFER64 keyword. IXGIMPRT is used to import (similar to write) a log block, specifying block ID and time stamp. It is a logstream recovery service. BUFFER64 specifies the address of the buffer from which the log block is to be imported.

IXGWRITE service This service supports the new BUFFER64 keyword. IXGWRITE is used to write log data to a logstream. BUFFER64 specifies the address of the buffer from which the log block is to be written.

Example of system logger services

SYS1.SAMPLIB(IXGASM64) contains samples of how to invoke logger services in 64-bit mode and the use of the BUFFER64 keyword. Figure 3-1 on page 33 shows an example of the above three services using the BUFFER64 keyword.

```
...
IXGWRITE BUFFER64=(2),BLOCKLEN=50,
      STREAMTOKEN=STREAMTOKEN,ANSAREA=ANSAA,ANSLEN=ANSLEN,
      RETCODE=RETCODE,RSNCODE=RSNCODE,MF=(E,PLIST,COMPLETE)
...
IXGIMPRT BUFFER64=(2),BLOCKLEN=50,BLOCKID=BLOCKID,
      GMT_TIMESTAMP=STCK1,LOCALTIME=STCK1,
      STREAMTOKEN=STREAMTOKEN,ANSAREA=ANSAA,ANSLEN=ANSLEN,
      RETCODE=RETCODE,RSNCODE=RSNCODE,MF=(E,PLIST,COMPLETE)
...
IXGBRWSE REQUEST=READCURSOR,BROWSETOKEN=BROWSETOKEN,
      BUFFER64=(2),BUFFLEN=50,DIRECTION=OLDTOYOUNG,
      BLKSIZE=RETURNSIZE,RETBLOCKID=RETBLOCKID,
      STREAMTOKEN=STREAMTOKEN,ANSAREA=ANSAA,ANSLEN=ANSLEN,
      RETCODE=RETCODE,RSNCODE=RSNCODE,MF=(E,PLIST,COMPLETE)
...
```

Figure 3-1 System logger services using the BUFFER64 keyword

Summary of system logger service changes

Table 3-1 shows the changes made to system logger services for 64-bit support.

Table 3-1 Summary of system logger service changes

Service name	31-bit/64-bit support	BUFFER64 keyword
IXGBRWSE	**Yes**	**Yes**
IXGCONN	Yes	
IXGDELET	Yes	
IXGIMPRT	**Yes**	**Yes**
IXGINVNT	Yes	
IXGOFFLD	Yes	
IXGQUERY	Yes	
IXGUPDAT	Yes	
IXGWRITE	**Yes**	**Yes**

Interaction and dependencies

The 64-bit virtual system logger support requires z-Architecture mode hardware. There are no new software requirements in this release. Any 64-bit system logger application can exploit this functionality.

Migration and coexistence

Logger macros with the BUFFER64 keyword can be compiled or assembled on a z-Architecture mode machine. If you try to execute it on a pre-z/OS V1R6 system, it will fail with return code 8, reason code ‘840’x (Bad version).

3.2 RRMS 64-bit callable services

Prior to z/OS V1R6, resource managers running in AMODE(64) with parameters residing in 64-bit addressable storage could not directly invoke recoverable resource management services (RRMS) callable services, since these services only accept parameters using 31-bit addresses from AMODE(31) callers.

z/OS V1R6 provides RRMS callable services in assembler and C that can be called directly by an AMODE(64) caller, and that accept parameters passed using 64-bit addresses. This avoids switching to AMODE(31) before invoking RRMS callable services and copying parameters to 31-bit addressable storage.

There are new ATR, CTX, and CRG callable services provided for AMODE(64) callers. These services allow parameters to be in 64-bit addressable storage. The names of the 64-bit callable services are different from those for the 31-bit callable services.

Table 3-2 shows a summary of RRMS callable registration services.

Table 3-2 RRMS callable registration services

31-bit name	64-bit name	64-bit parm supported?	31-bit name	64-bit name	64-bit parm supported?
CRGGRM	CRG4GRM	Yes	CRGSEIF	CRG4SEIF	Yes
CRGRRMD	CRG4RRMD	Yes	CRGDRM	CRG4DRM	Yes

Table 3-3 shows a summary of RRMS callable context services.

Table 3-3 RRMS callable context services

31-bit name	64-bit name	64-bit parm supported?	31-bit name	64-bit name	64-bit parm supported?
CTXBEGC	CTX4BEGC	Yes	CTXRCID	CTX4RCID	Yes
CTXDINT	CTX4DINT	Yes	CTXRCC	CTX4RCC	Yes
CTXENDC	CTX4ENDC	Yes	CTXSDTA	CTX4SDTA	Yes
CTXEINT1	CTX4EINT	Yes	CTXSCID2	CTX4SCID	Yes
CTXRDTA	CTX4RDTA	Yes	CTXSWCH	CTX4SWCH	Yes

Table 3-4 shows a summary of RRMS callable unauthorized services.

Table 3-4 RRMS callable unauthorized services

31-bit name	64-bit name	64-bit parm supported?	31-bit name	64-bit name	64-bit parm supported?
ATRBACK	ATR4BACK	Yes	ATRRURD2	ATR4RURD	Yes
ATRBEG	ATR4BEG	Yes	ATRSPSP2	ATR4SPSP	Yes
ATRCCUR3	ATR4CCUR	Yes	ATRSUSI2	ATR4SUSI	Yes
ATRCMIT	ATR4CMIT	Yes	ATRABAK	ATR4ABAK	Yes
ATRDPSP2	ATR4DPSP	Yes	ATRACMT	ATR4ACMT	Yes
ATREND	ATR4END	Yes	ATRADCT1	ATR4ADCT	Yes

31-bit name	64-bit name	64-bit parm supported?	31-bit name	64-bit name	64-bit parm supported?
ATRAFGT	ATR4AFGT	Yes	ATRRUSI2	ATR4RUSI	Yes
ATRAPRP	ATR4APRP	Yes	ATRRWID2	ATR4RWID	Yes
ATRDINT	ATR4DINT	Yes	ATRSIT	ATR4SIT	Yes
ATREINT5	ATR4EINT	Yes	ATRSPID	ATR4SPID	Yes
ATRIBRS	ATR4IBRS	Yes	ATRSROI1	ATR4SROI	Yes
ATRIERS	ATR4IERS	Yes	ATRSSPC	ATR4SSPC	Yes
ATRIRLN	ATR4IRLN	Yes	ATRSWID2	ATR4SWID	Yes
ATRIRNI	ATR4IRNI	Yes	ATRRENV	ATR4RENV	Yes
ATRIRRI	ATR4IRRI	Yes	ATRSENV	ATR4SENV	Yes
ATRISLN	ATR4ISLN	Yes	ATRREIC	ATR4REIC	Yes
ATRPDUE	ATR4PDUE	Yes	ATRRUSF1	ATR4RUSF	Yes
ATRRID	ATR4RID	Yes			

Attention: The SRRCMIT and SRRBACK services have no 64-bit equivalents. Use ATR4CMIT and ATR4BACK services instead.

3.3 RRS - restart anytime/anywhere

Prior to z/OS V1R6, Resource Recovery Services (RRS) did not allow a resource manager (RM) with outstanding interests to move to another system without a recoverable resource services (RRS) failure. This can prevent an installation from restoring a resource manager to the preferred system after recovering from a system outage. Here is an example illustrating this:

1. RM A is running on SY1 and is active with RRS.
2. SY1 comes down because of some problem. So, RRS and RM A are now failed.
3. You move RM A to SY2, which is in the same RRS logging group, and recover the failed transactions.
4. You restart SY1 after the problem is resolved.
5. Now you attempt to move RM A back to SY1 to restore the preferred configuration. But RRS does not allow RM A's restart, since RM A is active with RRS on SY2.
6. The only way to resolve this is to recycle RRS.

z/OS V1R6 allows resource managers to be restarted on a different system within the same RRS logging group, without canceling RRS. RRS will manage any outstanding transactions across the multiple systems internally. This enhancement eliminates an RRS outage simply to move a resource manager to a different system. For this support, resource managers do not have to make any code changes.

3.3.1 Migration and coexistence

Toleration APARs OA04884 and OA05504 are required to be installed on z/OS V1R3, z/OS V1R4, and z/OS V1R5 before any system in the RRS logging group moves to z/OS V1R6.

Without these APARs, the downlevel system cannot properly interpret the z/OS V1R6 changed processing, which will result in improper RRS processing of transactions.

You can only use “restart anytime/anywhere” across z/OS V1R6 systems. Unless all systems in the RRS logging group are at z/OS V1R6, you will not get the full benefit of this support.

When you attempt to restart any resource manager running on z/OS V1R6 system, the downlevel system will fail with a x’F02’ return code from the Begin_Restart service. Also, z/OS V1R6 systems will reject any resource manager attempting to restart from a downlevel system where RRS remained active with a x’F02’ return code from the Begin_Restart service.

3.4 SMF buffer constraint relief

SMF initially allocates 8 MB in its own high private storage for its buffers. These buffers are used to store records that are passed to SMF. SMF then asynchronously writes these records to the SMF data sets (SYS1.MANx) on DASD. If the data in the buffer is not copied to the SMF data sets or if data transfer from the buffers to the data set is slow, SMF continues writing the data it generates to its buffers. When the initial allocation of 8 MB storage is filled, SMF increases this buffer in 8 MB increments up to a maximum of 128 MB. If the buffers are allowed to be filled, it will result in a loss of SMF data.

The message IEE986E is issued when the allocation of buffers in the SMF address space exceeds the warning level (default is 25%). As each additional allocation occurs, this message is redisplayed with an updated percentage value unit all of the buffer space is exhausted. When the buffer is 100% filled, message IEE979W is issued.

SMF data is lost when:

- An SMF data set is not available and the maximum buffer allocation of 128 MB is full.
- SMF data is generated faster than additional SMF buffers can be obtained.

APAR OW56001 provides relief by increasing the initial, incremental, and maximum buffer size limit and the warning level. The maximum buffer allowed with this APAR is 1024 MB.

3.4.1 Buffer constraint relief

z/OS V1R6 provides parmlib support to allow customization of buffer sizing and the buffer usage warning level. The requested maximum buffer size is preallocated in SMF virtual storage. The buffer is dynamically managed such that actual storage used is maintained in a chained queue. As the buffer empties, chain elements are made available for use again. Page release processing releases auxiliary storage used to back the buffer storage.

3.4.2 New SMF parameters

The following new SMF parameters have been added:

- BUFSIZMAX
- BUFUSEWARN

BUFSIZMAX

The BUFSIZMAX in parmlib is specified as:

BUFSIZMAX(nnnnM) Specifies the maximum amount of storage that SMF can use for SMF record data buffering purposes. The value of BUFSIZMAX can range from a minimum of 128 MB to a maximum of 1 GB. The default value is 0128 MB. The command D SMF,O can be used to determine the current setting for the maximum amount of buffer space available for SMF to use.

Note: We recommend to check the “high water mark” value in the Type 23 record (SMF23BFH), and set the BUFSIZMAX value to twice this value.

BUFUSEWARN

The BUFUSEWARN in parmlib is specified as:

BUFUSEWARN (*dd*) Specifies the buffer warning level percentage when SMF will start issuing message IEE986E. When the amount of “in-use” buffer percentage falls below the BUFUSEWARN, message IEE986E is deleted. The value specified is from 10 to 90 (10% to 90%). The default is 25 (25% of BUFSIZMAX value). D SMF,O displays the BUFUSEWARN value.

Implementation

The new SMF parameters are specified in the following ways:

- SYS1.PARMLIB(SMFPRMxx).
- The SET SMF (T SMF) command.
- The SET command specifies a different SMFPRMxx parmlib member or initiates the restart of SMF. The command is SET SMF=xx, where xx specifies the two alphanumeric characters indicating the SMFPRMxx member of the logical parmlib containing the parameters the system is to use.
- The SETSMF (SS) command.
- The SETSMF command allows an installation to add a SUBPARM parameter or replace any previously-specified parameter in the active SMF parmlib member except the ACTIVE, PROMPT, SID, or EXITS parameters. The SETSMF command cannot add a parameter to the active SMF parmlib member. The SETSMF command cannot be used with a SMFPRMxx member that specified NOPROMPT. To avoid possible confusion with the SET SMF command, use the abbreviation SS for the SETSMF command.

  The command is issued as **SS *parameter*(*value*[,*value*]...).** For example:

```
SETSMF SUBSYS(STC,TYPE(0:127),INTERVAL(003000)) or
SS SUBSYS(STC,TYPE(0:127),INTERVAL(003000))
```

Message changes

The following messages are changed to support this enhancement:

- IEE967I

  The new BUFSIZMAX and BUFUSEWARN parameters will be displayed on the IEE967I display SMF options message. Figure 3-2 on page 38 shows the messages generated when the command is issued.

```
D SMF,O
IEE967I 16.16.59 SMF PARAMETERS 754
        MEMBER = SMFPRM00
        MULCFUNC -- DEFAULT
        BUFUSEWARN(25) -- DEFAULT
        BUFSIZMAX(0128M) -- DEFAULT
        SYNCVAL(00) -- DEFAULT
        DUMPABND(RETRY) -- DEFAULT
        SUBSYS(STC,NOINTERVAL) -- SYS
        SUBSYS(STC,NODETAIL) -- SYS
        SUBSYS(STC,EXITS(IEFUS0)) -- PARMLIB
        ...
```

Figure 3-2 D SMF,O command output

- IEE979W SMF DATA LOST - NO BUFFER SPACE AVAILABLE TIME=hh.mm.ss

  Changes: The enhanced SMF buffer logic and parmlib options are discussed in the Explanation and Operator Response sections. Guidance is provided on relieving the data loss condition.

- IEE986E SMF has used nn% of available buffer space

  Changes: The enhanced SMF buffer logic and parmlib options are discussed in the Explanation and Operator Response sections. Guidance is provided on avoiding the data loss condition.

SMF record updates

- SMF record type 23 Statistics Update Section—comment changes only made to the following fields:

  SMF23BFA Amount of each buffer allocation request

  SMF23BFT Total amount of buffer storage currently allocated (and recently used)

  SMF23BFM Buffer storage maximum in affect (BufSizMax binary value)

  SMF23BFL Buffer warning level in effect (BufUseWarn binary value)

- SMF record type 90 “IPL SMF/SET SET/SETSMF Section” header section for subtypes 5, 9, 13, and 15 has been increased in size from 36 bytes to 48 bytes.

  Any programs that use an explicit offset from the beginning of the SMF90 record to a field in any of the sections following this section may need to be updated. Table 3-5 shows the initial Dsect of sections following “IPL SMF/SET SET/SET SMF Section” and their offset fields.

Table 3-5 SMF record type 90 changes

Offset Field	Section	Initial Dsect
SMF90ODA	“SMF Data Set Section”	SMF90DSE
SMF90OWK	“Subsystem Record Section”	SMF90WCH
SMF90OOT	“Subsystem Parameter Section”	SMF90SUB

The new fields added to “IPL SMF/SET SMF/SETSMF Section” are shown in Figure 3-3 on page 39. The new fields follow the existing field, SMF90IDT.

Offsets		Name	Length	Format	Description
32	20	SMF90IDT	4	packed	Date of IPL, in the form 0CyydddF
36	24	**SMF90BFM**	5	EBCDIC	BUFSIZMAX value (ddddu)
41	29	**SMF90BFL**	2	EBCDIC	BUFUSEWARN value (dd)
43	2B	**SMF90RV8**	5	EBCDIC	Reserved

Figure 3-3 Changes to IPL SMF/SET SMF/SET SMF Section

Coexistence

Since SMF is specific to each LPAR, there are no coexistence issues with this support.

3.5 GRS enhancements

Global resource serialization (GRS) provides two critical system serialization services: the ENQ service and the Latch service. When GRS services are not working well, various failures may occur, ranging from poor performance of the system or subsystem to a complete outage.

3.5.1 GRS ENQ service

The GRS ENQ service provides the ability to serialize an abstract resource within a JOBSTEP, a SYSTEM, or a multisystem complex (GRS complex). The GRS complex is usually equivalent to the sysplex. Using the HW reserve function, DASD volumes can be shared between different systems that are not in the same GRS complex or even the same operating system such as between VM, Linux, and z/OS. Enq/Reserve services can be used by authorized or unauthorized users. Almost every component, subsystem, and many applications use ENQ in some shape or form.

This service uses the ENQ, DEQ, and RESERVE macros for serialization processing. The macros identify the resource by its symbolic name, which has three parts: QNAME, RNAME, and SCOPE. The resource is serialized with a scope of STEP, SYSTEM, SYSTEMS, or sysplex. The resource is used by authorized or unauthorized users and ownership is either shared or exclusive. This service is widely used and controlled by installations using RNLs and Exits.

GRS latch service

The GRS latch service provides a high-speed serialization service for authorized callers only. It uses user-provided storage to manage a lock table that is indexed by a user-defined lock number. GRS latch is also widely used.

This service uses Latch_Create, Latch_Obtain, Latch_Release and Latch_Purge services for serialization processing. The resource is associated with a latch number and the latch has a scope of single system. The latch is used by authorized users only; ownership is either shared or exclusive. This service is widely used by systems and subsystems. It cannot be controlled by installations.

GRS complex

GRS is configured either in Ring or Star mode in order to communicate with other instances within the GRS complex. IBM recommends GRS Star for performance and availability reasons.

GRS Ring consists of one or more systems connected to each other using CTCs or XCF connections. The links are used to pass global resource information from one system to

another in the complex. Requests are made by passing a message or token, called the ring system authority message (RSA-message), between systems in a round-robin or ring fashion. The requester can't get the requested ENQ until all other systems have seen the RSA message.

GRS Star uses a coupling facility lock structure, ISGLOCK, so each system can go directly to the coupling facility to get an ENQ. No messages are sent in non-contention cases. In a star complex, when an ISGENQ, ENQ, DEQ, or RESERVE macro is issued for a global resource, MVS uses information in the ISGLOCK structure to coordinate resource allocation across all systems in the sysplex.

Figure 3-4 shows a conceptual view of GRS Star and Ring complexes.

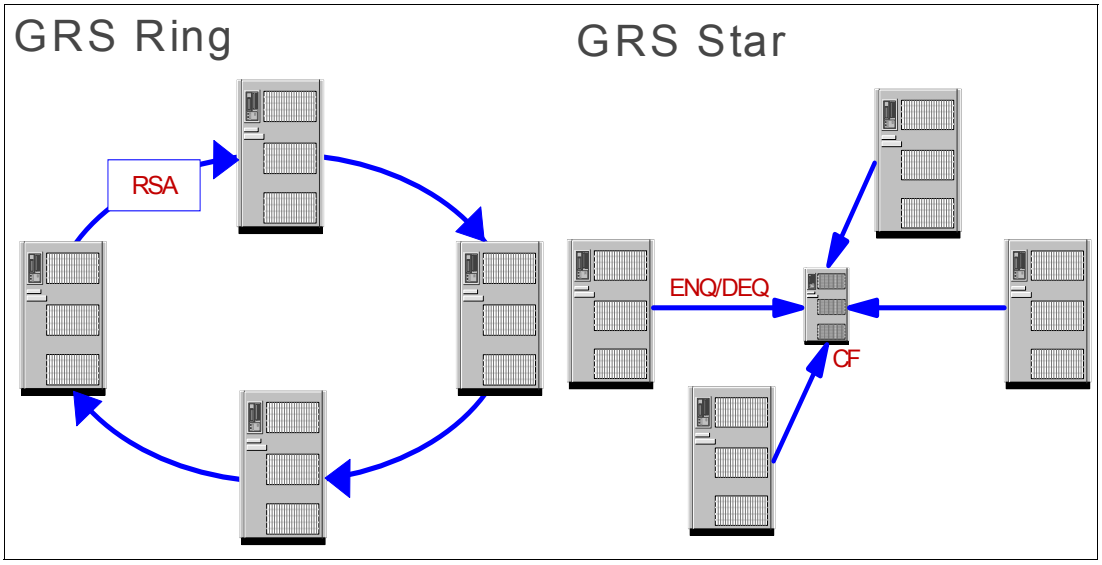

Figure 3-4 A conceptual view of GRS Star and Ring complexes

3.5.2 z/OS V1R6 GRS enhancements

Enhancements have been made to GRS ENQ, RESERVE and LATCH services. This section discusses the changes to these services.

GRS ENQ enhancements

z/OS V1R6 introduces two new services, ISGENQ and ISGQUERY. ISGENQ replaces ENQ, DEQ, and RESERVE and provides this support in one service. The ISGQUERY macro is used to obtain information about the status of each resource identified to Global Resource Serialization (GRS), which includes information about the tasks that have requested the resource. The goal of this enhancement is to provide AMODE 64 support and to provide better RAS (Reliability Availability Serviceability) for GRS users. These 64-bit APIs are available for callers who require it or who prefer to have their control information in storage above the bar.

Many problems are addressed by this enhancement:

- There have been instances where the resource identity (QNAME, RNAME, SCOPE) is changed by installation exits or the application itself by providing different values. As a result, an ENQ does not match a DEQ and this results in severe consequences, depending on who issues the DEQ.

To prevent this, the new ISGENQ service provides a token (ENQTokens) back to the ENQ requester. ISGENQ uses ENQTokens to represent current ENQ requests. These tokens, which are guaranteed to be unique within a sysplex, can be used on subsequent ISGENQ requests to change the request from shared to exclusive control, or to release the ENQ. ENQTokens are created for every ENQ, including those resulting from ENQ and RESERVE macro calls. Installation exits cannot alter the resource identity for an ISGENQ release (DEQ). The identity is the same as what was determined on the obtain.

- ENQ, DEQ, RESERVE LINKAGE=SYSTEM provides a PC rather than SVC interface to GRS ENQ services. This was provided to allow users to perform these functions in a cross-memory environment. However, there are significantly more instructions performed by this function.

  In z/OS 1.6, the instruction path has been decreased by 50% for LINKAGE=SYSTEM. This is important as LINKAGE=SYSTEM is getting more usage and the new ISGENQ service is based on this.

- There is a reoccurring problem where subsystems or applications are configured for a type of shared environment that does not match the resulting action taken by the installation RNLs or installation exits. For example, the application issues SYSTEMS/GLOBAL ENQ and is trying to serialize a multisystem resource. However, the SCOPE of the ENQ is incorrectly being converted (excluded) to a SYSTEM/LOCAL SCOPE resource. This in turn causes integrity errors. In another example, the application issues a RESERVE, but it has been converted to a GLOBAL ENQ in certain environments to prevent deadlocks and/or improve performance.

  You can now use the ISGQUERY RNL and ISGENQ TEST services to obtain information about the status of the resource. This helps GRS API users to verify the resource is being serialized for its intended status.

Table 3-6 shows the new 64-bit GRS ENQ services.

Table 3-6 GRS ENQ services

Service Description	31-bit	64-bit
Request Change/Control of a Serially Reusable Resource	ENQ	ISGENQ REQUEST=OBTAIN… CHANGE
Release Control of a Serially Reusable Resource	DEQ	ISGENQ REQUEST=RELEASE
Reserve a Shared Device	RESERVE	ISGENQ REQUEST=OBTAIN RESERVEVOLUME=YES
Extract Information From Global Resource Queues	GQSCAN	ISGQUERY REQINFO=QSCAN

3.5.3 GRS RESERVE enhancements

The RESERVE macro serializes access to a resource (a data set on a shared DASD volume) by obtaining control of the volume on which the resource resides to prevent jobs on other systems from using any data set on the entire volume.

The synchronous RESERVE feature was added to Global Resource Serialization in OS/390 Release 7. The SYNCHRES option in the GRSCNFxx parmlib member allows an installation to specify whether the system should obtain a hardware RESERVE for a device prior to returning from the RESERVE service. Thus the caller has both the ENQ (recommended to be excluded to local ENQ) and the HW RESERVE. This option might prevent jobs that have a delay between a hardware RESERVE request being issued and the first I/O operation to the

device. Prior to the implementation of the SYNCHRES option, the opportunity for a deadlock situation was more likely to occur.

Under z/OS V1R6, the GRSCNFxx parmlib member SYNCHRES option is active by default. The installation can deactivate it through either the GRSCNFxx parmlib member or the SETGRS operator command.

Having SYNCHRES off can cause a deadlock and/or data integrity problem. Having it on may cause a very minimal negative performance effect. Therefore, z/OS V1R6 has provided you the ability to decide how you want the reserve to be handled, by using the ISGENQ SYNCHRES(YES,NO, SYSTEM) option. If you know that the serialization protocol works fine by getting a local ENQ exclusive and then doing some processing before the physical volume RESERVE completes, then you can choose SYNCHRES(NO) on ISGENQ. If you know that your serialization protocol can result in a deadlock or possible data corruption if the reserve is obtained after the ENQ is obtained but before the physical volume RESERVE completes, then specify SYNCHRES(YES) on ISGENQ to prevent this from occurring.

3.5.4 GRS ENQ and latch enhancements

When there is contention on a resource, the operating system must insure that a resource holder is given the appropriate dispatching priority, a temporary promotion, in order to release the resource and thus reduce/eliminate the contention. Not doing this properly can cause long-term hangs and/or performance bottlenecks.

To manage the ENQ/Latch contention effectively, z/OS V1R6 GRS now exploits the “type 2” system event (Sysevent) services, ENQHOLD and ENQRLSE, which were introduced by WLM in z/OS 1.3. This service allows the caller to pass additional data to SRM describing the request.

GRS latch enhancements

The following enhancements have been made to GRS latch services:

- z/OS V1R6 provides 64-bit APIs for callers who require this or who prefer to have their control information in above-the-bar storage. For example:
  - LE Language Environments (LE) USS 64-bit environments require that the heap (dynamic area) be above the bar.
  - Storage-constrained applications can experience relief by moving data structures to 64-bit above-the-bar storage.
- Under z/OS V1R6, GRS provides constraint relief for instances where there is a large amount of contention for latches. The latch user who obtains latches exclusively, very frequently, and for short periods of time will benefit most from this enhancement.

  z/OS 1.3 provided some relief by removing the CMSEQDQ lock intersect between latch processing and GRS ENQ processing.

  z/OS V1R6 decreased the intersect during latch contention processing by:
  - Creating four internal latches per latch set. There can be many latch sets per subsystem.
  - Not having to acquire the local lock when obtaining these fast locks.
  - Removing the need to obtain the CMSLATCH lock or any lock when resuming a suspended waiter. The CMSLATCH lock is still used for certain non-frequent processing.

Table 3-7 on page 43 shows the new GRS Latch services.

Table 3-7 GRS latch services

Service description	31-bit	61-bit
Create a Latch Set	ISGLCRT	ISGLCR64
Obtain a Latch	ISGLOBT	ISGLOB64
Release a Latch	ISGLREL	ISGLRE64
Purge a Requestor from a Latch Set	ISGLPRG	ISGLPR64
Purge a Group of Requestors from a Group of Latch Sets	ISGLPBA	ISGLPB64

New messages

The following new messages are added:

- **ISG355I** IARV64 service-name SERVICE FAILED, RC=return-code, RSN=reason-code START@=64-bit starting address END@=64-bit ending address DIAG=x DETECTING MODULE=name of the detecting module

  Explanation: An IARV64 service was issued, but failed with an error return and reason code. Refer to the IARV64 documentation for information on return and reason codes.

  System action: The system continues processing.

- **ISG356E** SYSTEM system-name DOES NOT SUPPORT ISGQUERY. SYSPLEX WIDE REQUESTS MAY CONTAIN INCOMPLETE DATA.

  Explanation: GRS detected a system in the GRS complex that is incapable of handling ISGQUERY requests from other systems.

  System action: The issuing system continues processing. However, it will not send any sysplex-wide ISGQUERY requests to the named system. Data returned on all sysplex-wide ISGQUERY requests may be incomplete.

Migration and Coexistence

GRS does require the compatibility APAR OA02469 for down-level systems that are in the same sysplex as a z/OS V1R6 system.

3.6 2047 members per XCF group

XCF groups are a sysplex-wide resource used for communication within a sysplex. Tasks join groups to become a member. A member resides on one system and can communicate with other members of the same group across the sysplex.

XCF originally supported between 8 and 511 members per group. This number was sufficient for most groups, as they typically have a small number of members. But this value was too small for those installations that run large numbers of CICS regions.

The member limit was increased to the maximum value of 1023 in OS/390 V2R5.

Large customers are now running with 800+ CICS regions. They will not be able to start more than 1023 CICS regions due to the XCF member limit. This limit was derived from the 2 GB size of the IXCDSMEM data space and a group limit of 2045.

XCF has been enhanced to provide support for up to 2047 members per XCF group. The new couple data set formatting utility (IXCL1DSU) now supports formatting a sysplex CDS for up

to 2047 members. Formatting a sysplex CDS for more than 1023 members causes a version 4 CDS to be created. XCF now creates a new data space, IXCDSMEX, to store information for more members.

Table 3-8 shows the summary of CDS versions and their supported OS release.

Table 3-8 Summary of CDS versions

Version	Feature	Where supported?
1	Base XCF support	MVS/ESA™ SP 4.1.0
2	32 system limit	MVS/ESA SP 5.1.0
3	1023 member limit	OS/390 V2R5.0 (OW21511 enables down to MVS/ESA SP 5.2.0)
4	2047 member limit	z/OS V1R6.0 (OA04034 enables down to z/OS V1R4.0)

Migration and coexistence

APAR OA04034 enables 2047-member support for z/OS V1R4 and z/OS V1R5. To determine if the system supports 2047 members or not, issue the command D A,XCFAS and verify the existence of the IXCDSMEX data space. The command output will be similar to what is shown in Figure 3-5.

```
D A,XCFAS
IEE115I 14.20.31 2004.105 ACTIVITY 268
 JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
00004    00025    00005      00031    00043    00005/00010       00012
 XCFAS    XCFAS    IEFPROC  NSW  *   A=0006   PER=NO   SMC=000
                                     PGN=N/A  DMN=N/A  AFF=NONE
                                     CT=00.54.28  ET=00165.48
                                     WKL=SYSTEM   SCL=SYSTEM   P=1
                                     RGP=N/A      SRVR=NO  QSC=NO
                                     ADDR SPACE ASTE=03969180
                                     DSPNAME=IXCDSMUS ASTE=030D6000
                                     DSPNAME=IXCDSMEX ASTE=036DE200
                                     DSPNAME=IXCDSMEM ASTE=036DE180
```

Figure 3-5 D A,XCFAS showing existence of the IXCDSMEX data space

Messages

- The old version of IXCL1DSU will not format a sysplex CDS for more than 1023 members. If you try to format the CDS with more than 1023 members, you will receive the message:

  ```
  IXC291I INVALID NUMBER VALUE, xxxx
  ```

- A sysplex will not use a version 4 sysplex CDS unless all systems support 2047 members. Any attempt to do so will receive the message:

  ```
  IXC255I UNABLE TO USE DATA SET dsname AS THE ALTERNATE FOR SYSPLEX: IT WAS
  CREATED AT A FORMAT LEVEL HIGHER THAN THIS SYSTEM CAN USE
  ```

- When using a version 4 sysplex CDS, systems without support for 2047 members cannot join the sysplex and will receive the message:

  ```
  IXC258I COUPLE DATA SET dsname WAS CREATED AT A FORMAT LEVEL HIGHER THAN THIS
  SYSTEM CAN USE
  ```

Note:

- Formatting the sysplex CDS with more members than needed wastes space and degrades the performance of XCF group services.
- A sysplex-wide IPL is needed to decrease the number of members.

3.7 Service aids enhancements

Tools and service aids are provided by MVS for problem diagnosis. The tools include dumps and traces, while service aids include the facilities provided for diagnosis. For example, an SVC dump and a system trace are tools and logrec data set; AMBLIST is a service aid.

The following topics describe the enhancements for the z/OS V1R6 service aid component.

3.7.1 Stand-alone dump

The stand-alone dump (SADMP) program produces a stand-alone dump of storage that is occupied by a system that failed. The term stand-alone means that the dump is performed separately from normal system operation and does not require the system to be in a condition for normal operation. You must create the stand-alone dump program on DASD or tape and use it to IPL. The stand-alone dump program produces a high-speed, unformatted dump of central storage and parts of paged-out virtual storage on a tape device or DASD. This dump supplies information that is needed to determine why the system failed.

A conventional MVS data set may occupy a maximum of 64 K tracks on a single volume. Volume capacity in bytes is dependent on track size, but it yields an effective ability to record less than 4 GB of data on existing DASD devices.

Many dump data sets, both system dumps and SADMPs, occupy multiple volumes. System dumps are generated during normal system operation. The system dump service takes the help of DFSMS while writing the dumps. The system dump supports extended PS data sets and also uses the striping and compression technologies to store the dumps. This helps to write the dump to a volume with more than 64 K tracks.

Before z/OS V1R6, SADMP had the restriction of writing the dump into a data set with DSORG=PS. SADMP supports DASD data sets occupying up to 16 volumes. A single data set can hold less than 64 GB of data (16 times 4 GB per volume). SADMP does allow the system operator to specify a secondary dump data set as a target, but the transition from one 16-volume data set to another includes an undesirable period of time as all of the volumes fill, and the pace of recording falls to a small fraction of what was occurring initially.

This problem has been addressed in z/OS V1R6. Now, z/OS V1R6 SADMP can use extended format non-VSAM data sets (DSORG=PS-E). These data sets are not limited to 64 K tracks per volume and can occupy a maximum of 32 volumes. So, it gives the facility the opportunity to plan ahead for the DASD space required for a SADMP. You may change your SMS DATACLASS for this support. However, the following restriction still applies:

- The SADMP data set does not support striping.
- The SADMP data set does not support compression.

AMDSADDD is a REXX utility for the system programmer to allocate and initialize the DASD SADMP data set. This utility is supplied in SYS1.SAMPLIB. When you plan to take a stand-alone dump to DASD, you must allocate and initialize the dump data set using this utility.

The z/OS V1R6 AMDSADDD REXX exec supports extended PS data sets for SADMP. For this, you must ensure that all DSORG=PS-E data sets are SMS-managed, have a DATACLAS that specifies no compression, and have a STORCLAS that specifies a sustained data rate of zero (suppress striping).You can explicitly request a STORCLAS or DATACLAS when AMDSADDD is invoked, or your installation's automatic class selection (ACS) routines can assign an acceptable STORCLAS or DATACLAS value. AMDSADDD now expands the *type* keyword to support SMS classes, as shown in Figure 3-6.

```
(Type[,[Storclas][,[Dataclas][,[Mgmtclas]]]])
where
Type is the device type as in earlier releases.
Storclas is the SMS storage class.
Dataclas is the SMS data class.
Mgmtclas is the SMS management class.
```

Figure 3-6 AMDSADDD utility type keyword

Note: APAR OA04140 will roll back this support up to z/OS V1R4.

3.7.2 SYSMDUMP enhancements

A SYSMDUMP is an unformatted dump, used for the diagnosis of system problems when the dump is requested in a program. A SYSMDUMP is the preferred type of ABEND dump and is processed using IPCS. Unformatted dumping is also much more efficient because it writes data faster, which allows the application to capture diagnostic data and be brought back online faster. SYSMDUMP is collected using the SYSMDUMP DD statement in the JCL.

Many customer applications specify a SYSMDUMP DD in their JCL to capture SYSMDUMP. Before z/OS V1R6, SYSMDUMP was not blocked. z/OS V1R6 allows SYSMDUMPs to be blocked, thereby accelerating the dumping process and reducing the period between the time that a problem is encountered and the time that the application is fully operational again.

z/OS V1R6 SYSMDUMP writes RECFM=FBS data sets. System-determined BLKSIZE may be requested. Multiple dumps may be written using DISP=MOD to a SYSMDUMP data set. System support for SYSMDUMPs anticipates this and pads all blocks written to BLKSIZE to ensure that the RECFM=FBS attribute is retained.

3.7.3 GTFTRACE enhancements

The generalized trace facility (GTF) is a service aid used to record and diagnose system and program problems. GTF is a part of the MVS system product. It is explicitly activated by entering a START GTF command. GTF is used to record a variety of system events and program events on all of the processors. If you use the IBM-supplied defaults, GTF lists many of the events. However, because GTF uses more resources and processor time than system trace, IBM recommends that you use GTF when you experience a problem, selecting one or two events that you think might point to the source of your problem. This will give you detailed information that can help you diagnose the problem. You can trace combinations of events, specific incidences of one type of event, or user-defined program events that the GTRACE macro generates.

GTF has been enhanced in many ways for z/OS V1R6. The most significant change allows multiple instances of GTF to be active concurrently.

Start multiple instances of GTF

Before z/OS V1R6, only one instance of GTF could be active. This restriction prevented more than one trace being taken concurrently. Sometimes, an installation is asked by two or more vendors to run a GTF trace at the same time. Combining several requests and collecting all the data via a single GTF instance not only generates a large volume of data to the same trace data set, but also makes it difficult for vendors to process the trace data. Also, you may be required to run the GTF trace multiple times for the same problem to collect different sets of trace information required by different vendors.

z/OS V1R6 allows multiple instances of GTF to be active concurrently. This is a functional enhancement and not directed toward enhancing performance. When you activate multiple GTF instances, each instance operates as a system task in its own address space. The only way to activate GTF is to enter a START GTF command from a console with master authority. Using this command, select either IBM's procedure or your cataloged procedure for GTF. The cataloged procedure defines the GTF operation; you can accept the defaults that the procedure establishes, or change the defaults by specifying certain parameters on the START GTF command.

Each instance of GTF can be assigned a unique identifier that is specified on the START GTF command after the GTF keyword. This will allow you to recognize and control specific instances of GTF. If a unique identifier is not specified, the operating system assigns a default identifier with the device number of the volume on which the trace data set resides. This makes it difficult to differentiate between multiple instances of GTF.

Figure 3-7 shows an extract from the SDSF DA panel showing multiple instances of GTF active at the same time. We have issued the command S GTF twice and S GTF.GTF once. You will notice that two instances of GTF have the same identifier, 2517. This is the DASD device number where the trace data set resides. This is because we issued the S GTF command twice without an identifier. However, you will also notice that they are running in different address spaces (different ASID and different JobID).

```
Cmd JobName  StepName ProcStep JobID    ASID
-------------------------------------------
    GTF      2517     IEFPROC  STC30283 0021
    GTF      2517     IEFPROC  STC30282 0069
    GTF      GTF      IEFPROC  STC30284 0077
```

Figure 3-7 Multiple instances of GTF

Other GTF changes

Other miscellaneous changes were made in GTF:

- The GTF SYS and DSP options have traced the return from SVCs for a long time. But the SVC trace has not done this, and this trace has not provided enough information for problem diagnosis. SYS and DSP traces are more intrusive on system operation than an SVC trace, so it has not been easy to run the trace with SYS or DSP options. The z/OS V1R6 GTF trace includes both SVC entry and return data, addressing this concern.
- GTF, like other MVS traces, discards trace entries rather than seriously impacting the parts of the system being monitored. z/OS V1R6 GTF adds a SIZE parameter that can be employed to tell GTF to use more private area buffers. This prevents the loss of trace entries during brief bursts of high activity if the output medium happens to be slow to record all trace entries being produced. The SIZE parameter is specified as:

  ```
  SIZE = {nnnnnnK|nnnnnnM|1024K}
  ```

This specifies the buffer size in bytes (nnnnnK or nnnnnM). The range for the size keyword is 1 MB to 2046 MB. The default is 1024 KB. If the amount is less than 1024 KB, GTF uses the default.

- A number of years ago MVS contents supervisor enhancements were implemented that caused LINK, LOAD, and XCTL services to use multifunction SVC 122 entries to support some options. But GTF did not support the display of the program names involved for this service. z/OS V1R6 GTF now supports both the older dedicated contents supervisor SVCs for these services as well as the SVC 122 entry points.

3.7.4 IPCS enhancements

IPCS makes better dump analysis available to offset the increasing complexity of the product environments being run on the system. Following are the changes that have occurred in IPCS.

EPTRACE subcommand

A new EPTRACE subcommand has been added. This subcommand is used to generate reports on the control flow between programs as indicated by 72-byte save areas. The syntax is shown in Figure 3-8.

```
Syntax -
   EPTRACE     KEYFIELD/SAVEAREA DATA('data') ORDER(ENTRY/RETURN)
               ACTIVE/DATASET('dsn')/FILE('ddn')/PATH('path')
               FLAG(INFORMATIONAL/WARNING/ERROR/
                   SERIOUS/SEVERE/TERMINATING)
               PRINT/NOPRINT TERMINAL/NOTERMINAL TEST/NOTEST

   Defaults  - KEYFIELD, DATA(TCBCURRENT), ORDER(ENTRY)
```

Figure 3-8 New IPCS EPTRACE subcommand

KEYFIELD Requests a high-level report that leaves symbols to permit you to subsequently “zoom in” on interesting data.

SAVEAREA Requests a report similar to the save area trace produced by SNAP/ABDUMP.

DATA('data') Specifies a symbol associated with one of the following groups of structures: TCB, RB, REGSAVE.

ORDER(ENTRY) Processes the save area chain in the same order that programs were entered.

ORDER(RETURN) Processes the save area chain in return order.

FINDSWA subcommand

A new FINDSWA subcommand has been added. This is used to locate a scheduler work area (SWA) block, including a SWA block prefix, in a dump. The syntax is shown in Figure 3-9 on page 49.

```
Syntax -
   FINDSWA     'addr' CONTEXT('symbol')
               ACTIVE/DATASET('dsn')/FILE('ddn')/PATH('path')
               DISPLAY PRINT/NOPRINT TERMINAL/NOTERMINAL TEST/NOTEST
               VERIFY/NOVERIFY
   Alias - FSWA
   Default - CONTEXT(JSCBACTIVE)
   Required - 'addr'
```

Figure 3-9 New IPCS FINDSWA subcommand

where:

'addr' Describes a 3-byte SWA virtual address (SVA).

CONTEXT('symbol')) Specifies a symbol describing a STRUCTURE(JSCB) or a STRUCTURE(TCB) that provides context for the search.

3.8 MVS allocation enhancements

Historically, UNABLE TO ALLOCATE errors have been extremely hard to debug. It indicates that while no errors occurred, no allocation was possible. Usually, this means that there was a request for several devices but the requested number of devices was not available. In this case recovery allocation would attempt to alleviate the problem by varying offline devices online, and/or by prompting the operator to wait for devices that are allocated elsewhere. If, after exhausting all recovery attempts, allocation was still unable to satisfy the request, the UNABLE TO SATISFY error was issued.

The potential for this error exists in several places in allocation. At those points, an error code is set but no error message is generated until allocation is ready to return to the requester. Further, the error messages generated are queued internally and not issued via WTO. This means that a SLIP trap on the message ID is impossible, as the error message is issued long after the error is detected.

Setting a SLIP trap on the locations that set the error reason code is difficult because it is required to know the service level of several modules. To trap the error code, a dump of the current system needs to be analyzed before a SLIP trap is created. This may include up to 20 individual traps to catch the exact error. Sometimes, it is difficult to easily recreate the problem and so the problem diagnosis takes a long time.

z/OS V1R6 allocation has been enhanced to create a message when the error is detected (versus doing it later, as is done with existing messages) and queue it with the existing message. Further, it provides a mechanism for IBM to request a dump of the problem without using SLIP.

3.8.1 New allocation message

A new message, IEF705I, has been added. Using this message, you can:

- See the progression of refinements made by Allocation to determine what esoteric or device was chosen but failed to allocate.
- See the counts of devices that contributed to the error—total number, allocated, offline, available.
- Provide detailed information about the error, including which instance it is.

- Obtain instructions from IBM to create a dump of the problem.

This eliminates the iterative process previously necessary to determine module service levels.

Usage and invocation

The issuance of message IEF705I depends on the type of allocation request. For batch allocation, IEF705I will appear in JOBLOG whenever message IEF702I appears. For dynamic allocation, IEF705I will appear wherever message IKJ56241I appears in JOBLOG, system log, an application error file, or not at all. You may consult dynamic allocation exploiter documentation for how messages returned by dynamic allocation are externalized. Also, refer to *z/OS MVS Programming: Authorized Assembler Services Guide,* SA22-7608, for information on requesting messages from dynamic allocation.

You may act or ignore the message, depending on the condition in which it is issued. Some applications, such as DFSMShsm, expect message IEF702I. In this case, you may simply ignore the message. If the error is not expected, use the information provided in the message to ensure that the system is configured properly and contact IBM service.

Figure 3-10 shows the new message IEF705I along with an example.

```
New message IEF705I
IEF705I DIAGNOSTIC INFORMATION FOR UNSUCCESSFUL ALLOCATION
DETECTED AT &time BY &CallerMod INSTANCE &instance
&jobname &stepname &procname &ddname &relpos
DEVICES FOR &unitname -
 TOTAL: &tot / OFFLN: &off / ALLOC: &alc / AVAIL: &avl
DIAGNOSTIC UNIT/DEVICE TYPE INFO:
 &U1 &U2 &U3 &U3 &U4 &U5
DYNAMIC ALLOCATION REQUEST FLAGS: &flg1 &flg2

Examples
 IEF705I DIAGNOSTIC INFORMATION FOR UNSUCCESSFUL ALLOCATION
DETECTED AT 02/04/2004 15:21:09.874581 BY IEFAB486 INSTANCE 00000002
JOB702IS STEP2 SBJTAPE
DEVICES FOR 3390M -
TOTAL: 00000004 / OFFLN: 00000000 / ALLOC: 00000000 / AVAIL: 00000000
DIAGNOSTIC UNIT/DEVICE TYPE INFO:
 3390M 3390M 3010200F 00132000 00132000
DYNAMIC ALLOCATION REQUEST FLAGS: 0000 51100000
```

Figure 3-10 New message IEF705I and example

3.9 New JCL keywords for PSF

PSF 3.4.0 provides new function that required the addition of new PSF JCL keywords. This topic describes the JCL keyword support provided for PSF in support of the new PRMODE, FONTPATH, and USERPATH keywords.

3.9.1 PRMODE keyword

PRMODE has been added to the PRINTDEV statement. It is used to indicate the default processing mode PSF uses to print data sets containing both single-byte and double-byte fonts. This works the same way as PRMODE on the OUTPUT statement. The valid values are

SOSI1, SOSI2, SOSI3, and SOSI4. All other values are ignored by PSF. Figure 3-11 shows an example of the PRMODE keyword syntax.

SOSI1 Specifies that each shift-out, shift-in code is converted to a blank and a Set Coded Font Local text control.

SOSI2 Specifies that each shift-out, shift-in code is converted to a Set Coded Font Local text control.

SOSI3 Specifies that each shift-in code is converted to a set coded font local text control and two blanks. A shift-out code is converted to a set coded font local text control.

SOSI4 Specifies that each shift-out, shift-in code is to be skipped and not counted when calculating offsets for the print data set. SOSI4 is used when double-byte character set text is converted from ASCII to EBCDIC.

```
//PRT619   CNTL
//PRT619   PRINTDEV FONTDD=*.FONT03,
//         FONTPATH=*.TTFONT01,
//             :
//             :
//             :
//         DISCINTV=0,
//         DATACK=UNBLOCK,
//         MGMTMODE=IMMED,
//         TIMEOUT=REDRIVE,
//         PRMODE=SOSI4,
//         CHARS=(60DB),
//         IPADDR=9.99.98.40
//PRT619   ENDCNTL
```

Figure 3-11 New PRMODE JCL keyword example

3.9.2 FONTPATH keyword

The FONTPATH keyword is used to specify the DDNAME to the font path library. The font path libraries contain True Type and Open Type fonts. These fonts are stored in HFS or zFS files. Figure 3-12 on page 52 shows an example of the FONTPATH JCL keyword syntax along with the paths for font path libraries.

Note: PSF ignores the FONTPATH keyword unless PSF has been Unicode-enabled.

```
//TTFONT01 DD PATH='/usr/lpp/PSF/fonts/ttf/'
//          DD PATH='/usr/lpp/PSF/fonts/ttfa_g/'
//          DD PATH='/usr/lpp/PSF/fonts/ttfh_m/'
//          DD PATH='/usr/lpp/PSF/fonts/ttfn_s/'
//          DD PATH='/usr/lpp/PSF/fonts/ttft_z/'
//          DD PATH='/usr/lpp/PSF/fonts/otfa_g/'
//          DD PATH='/usr/lpp/PSF/fonts/otfh_m/'
//          DD PATH='/usr/lpp/PSF/fonts/otfn_s/'
//          DD PATH='/usr/lpp/PSF/fonts/otft_z/'
...
//PRT619   CNTL
//PRT619   PRINTDEV FONTDD=*.FONT03,
//         FONTPATH=*.TTFONT01,
//         FONT300=*.FONT03,
//         FONT240=*.FONT02,
//         OVLYDD=*.OLAY01,
//         PSEGDD=*.PSEG01,
//         PDEFDD=*.PDEF01,
//         FDEFDD=*.FDEF01,
//               :
//               :
//               :
//         PRMODE=SOSI4,
//         CHARS=(60DB),
//         IPADDR=9.99.98.40
//PRT619   ENDCNTL
```

Figure 3-12 New FONTPATH JCL keyword example

3.9.3 USERPATH keyword

USERPATH is used to specify one to eight path names to be used in the user path library. The user path library contains True Type and Open Type fonts. These fonts are stored in HFS and zFS files. Figure 3-13 shows an example of the USERPATH keyword syntax.

Note: PSF will ignore the USERPATH keyword unless PSF has been Unicode-enabled.

```
//STEP1    EXEC PGM=IEBGENER,REGION=4096K
//OUTPUT1  OUTPUT DATACK=UNBLOCK,PAGEDEF=029A,
//         USERPATH=('//usr/lpp/PSF/fonts/ttfa_g',
//         '/usr/lpp/PSF/fonts/ttfh_m',
//         '//usr/lpp/PSF/fonts/ttfn_s',
//         '/usr/lpp/PSF/fonts/ttft_z'),
//         USERLIB=('PSFMVS.FCT.R340.RESOURCE')
//SYSUT2   DD SYSOUT=K,OUTPUT=*.OUTPUT1
//SYSIN    DD DUMMY
//SYSPRINT DD SYSOUT=T
//SYSUT1   DD DSN=PSFMVS.FCT.R340.LINEDATA(TO05A),DISP=SHR
```

Figure 3-13 New USERPATH JCL keyword syntax

Note: This support has been rolled down to z/OS 1.2 via APARs OA02478 and OA05130.

3.10 Unicode enhancements

z/OS V1R6 support for Unicode offers character conversion, case conversion normalization, and collation. Within character conversion, characters are converted from one coded character set identifier (CCSID) to another. Case conversion allows conversion to upper or lower case based on the file's UnicodeData.txt and SpecialCasing.txt provided by the Unicode consortium. Normalization allows the decomposition or composition of Unicode encoding input to any of four normalization forms.

The following enhancements have been made to z/OS V1R6 Unicode support:

- Loading of pre-built image for DB2
- Performance Improvement for Unicode Conversion Service

3.10.1 Loading of pre-built image for DB2

With DB2 V8, all data, including DB2 catalogs, can be stored in Unicode. DB2 catalogs allow SQL to contain Unicode literals and names. This provides better integration with Java and Microsoft® technologies, which use Unicode to allow communication across the globe without errors in character conversion. DB2 data Unicode support allows for re-engineering of applications for international business. The Unicode encoding scheme addresses the problems that are encountered when users who live in different geographies and speak many different languages interact with the same DB2 server.

DB2 interfaces with z/OS Unicode services to provide code page conversions. Therefore, during the initialization of DB2 V8, it is required that:

- The z/OS Unicode services are properly enabled by the customer. If this is not the case, all conversion requests will fail with RC=12, RS=1, and IPL will be necessary to enable the Unicode environment. In addition, DB2 will not initialize.
- The required code page conversions that will be requested by a DB2 user are available in the Unicode environment. This means that the correct code page conversion tables are loaded in the Unicode environment (specifically into the "active conversion image"). If this is not the case, the conversion request will fail with RC=8, RS=3 (unsupported CCSID was specified).

Again, during DB2 initialization, a conversion request is issued in order to check if z/OS Unicode services is active. If the requested code page conversion is not part of the conversion image, then DB2 V8 will not initialize. To recover, it is necessary that a conversion image with the required conversion tables be built and loaded.

Enhancement

The objective of this enhancement is to create an image that contains all possible conversion tables that DB2 supports and load it during DB2 initialization, when z/OS Unicode conversions services has not been set up by the customer (UNI=xx not specified in IEASYSxx parmlib member). This eliminates the need for DB2 V8 customers to customize z/OS Unicode services. System programmers will no longer need to create an image with the necessary conversion tables and customize the CUNUNIxx parmlib member.

Note: This enhancement is implemented in the base code of z/OS V1R6. However, this has been rolled down to z/OS V1R2 via APAR OA04069.

Loading of a Unicode prebuilt image is only supported when the caller is in PSW Key=7 and TCB mode. DB2 initialization uses this state of execution. When UNI=xx is not specified in the

system parameters at IPL time, Unicode services automatically loads a prebuilt image during DB2 initialization that contains all the EBCDIC and ASCII conversion tables. Subsequent Unicode conversion calls use the tables provided in this image. The prebuilt image must be page-fixed in storage and requires 39 MB (9862 pages) of real storage. This automatic loading of the image is illustrated in Figure 3-14.

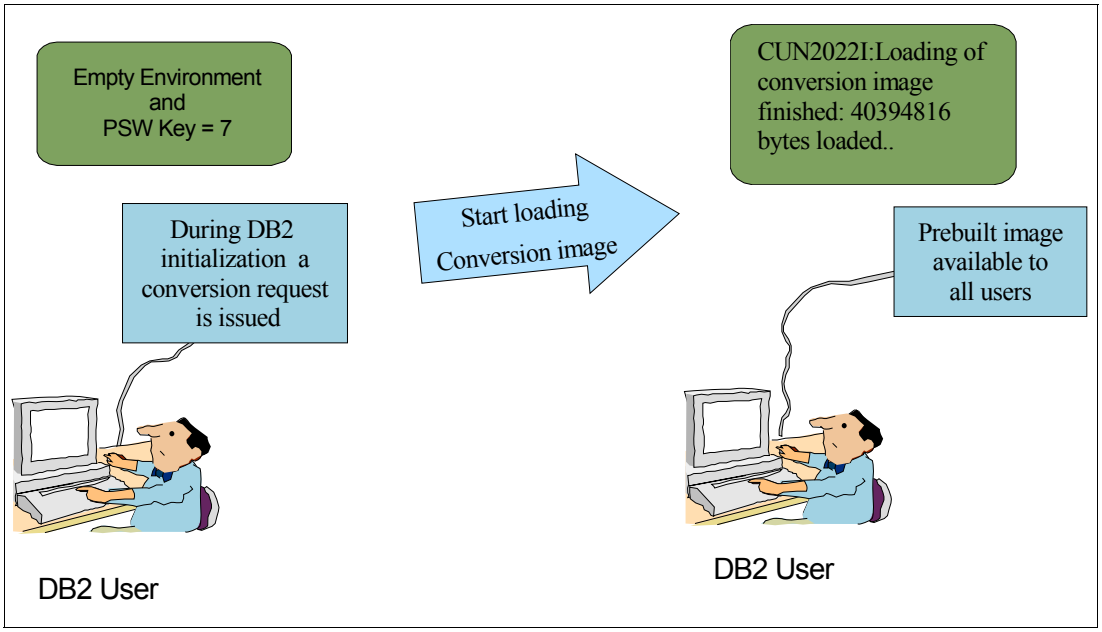

Figure 3-14 Unicode prebuilt loading during DB2 initialization

A quick way to verify the successful loading and page fixing of the prebuilt image is by issuing the console command DISPLAY UNI,STORAGE. The result should show 9862 pages of storage utilized by the active image.

Once the prebuilt image has been loaded and page-fixed in storage, issuing the SET UNI=xx command will replace it with the image specified in the corresponding CUNUNIxx parmlib member. Subsequently, the prebuilt image cannot be reloaded using another SET UNI command. Rather, an IPL without specifying UNI=xx in the system parameters and a Unicode conversion call from a Key=7, TCB mode environment is required to load the prebuilt image.

Prebuilt image - CUNIDHC2

The prebuilt image contains all the ASCII to/from UTF8 and EBCDIC to/from UTF8 conversion tables with the ER (enforced subset and roundtrip) technique. In z/OS V1R6 a new sample job, SYS1.SAMPLIB(CUNSISM6), provides the conversion statements that were used to build this image.

The prebuilt image contains the CCSIDs, as shown in Figure 3-15 on page 55.

```
EBCDIC CCSIDs
00037 00256 00259 00273 00275 00277 00278 00280 00282 00284 00285 00286 00290 00293
00297 00300 00420 00421 00423 00424 00500 00803 00833 00834 00835 00836 00837 00838
00870 00871 00875 00880 00905 00918 00924 00930 00931 00933 00935 00937 00939 01025
01026 01027 01047 01097 01112 01122 01123 01140 01141 01142 01143 01144 01145 01146
01147 01148 01149 01364 01388 01390 01399 04133 04369 04370 04371 04373 04374 04376
04378 04380 04381 04386 04393 04396 04516 04519 04520 04596 04930 04931 04932 04933
04934 04966 04967 04976 05014 05026 05029 05031 05033 05035 05123 05143 08229 08448
08482 08492 08612 08692 09025 09026 09028 09030 09122 09125 09127 09131 12544 12588
12788 13121 13124 13218 13219 13221 13223 16421 16684 16884 17314 20517 20980 24613
25076 28709 29172 32805 33058 33268 33698 33699 41460 45556 49652 53748 61696 61711
61712
ASCII CCSIDs
00301 00367 00437 00813 00819 00850 00851 00852 00853 00855 00856 00857 00858 00860
00861 00862 00863 00864 00865 00866 00868 00869 00874 00891 00895 00896 00897 00903
00904 00912 00915 00916 00920 00921 00922 00923 00926 00927 00928 00932 00934 00936
00938 00941 00942 00943 00944 00946 00947 00948 00949 00950 00951 00954 00956 00957
00958 00959 00965 00970 00971 01004 01006 01008 01009 01010 01011 01012 01013 01014
01015 01016 01017 01018 01019 01020 01021 01023 01040 01041 01042 01043 01046 01051
01088 01089 01098 01100 01101 01102 01103 01104 01105 01106 01107 01114 01115 01124
01125 01126 01131 01250 01251 01252 01253 01254 01255 01256 01257 01275 01276 01277
01280 01281 01282 01283 01350 01351 01362 01363 01380 01381 01383 01385 01386 04397
04533 04946 04947 04948 04949 04951 04952 04953 04960 04964 04965 04970 04992 04993
05023 05028 05038 05043 05045 05046 05047 05050 05052 05053 05054 05055 05100 05137
05142 05210 05211 05346 05347 05348 05349 05350 05351 05352 05353 05354 05476 05477
05479 08493 08629 09047 09056 09060 09066 09089 09124 09139 09142 09146 09572 09575
12725 13152 13235 13238 13242 16821 17331 17354 20917 21450 24877 25013 25426 25427
25428 25429 25431 25432 25433 25436 25437 25438 25439 25440 25441 25442 25444 25445
25450 25467 25473 25479 25480 25502 25503 25504 25508 25510 25512 25514 25518 25520
25522 25524 25525 25527 25580 25616 25617 25618 25619 25664 25690 25691 29109 29522
29523 29524 29525 29527 29528 29529 29532 29533 29534 29535 29536 29537 29540 29541
29546 29614 29616 29618 29620 29621 29623 29712 29713 29714 29715 29760 33205 33618
33619 33620 33621 33623 33624 33632 33636 33637 33665 33700 33717 33722 37301 37719
37728 37732 37761 37796 37813 41397 41824 41828 45493 45920 49589 61697 61698 61699
61700 61710
```

Figure 3-15 Prebuilt image CCSIDs

Note: The above CCSID list represents the conversions supported by the prebuilt image. Each CCSID can be converted to and from UTF-8 (CCSID 1208), to and from UCS2 (CCSID 1200), and to and from CCSID 0367. All tables were built using the ER technique.

New messages

Three new messages were introduced with this support:

- **CUN2046I** AN EMPTY UNICODE ENVIRONMENT HAS BEEN ESTABLISHED
- **CUN2047I** UNICODE CONVERSION ENVIRONMENT NOT ACTIVE. UNICODE DYNAMIC LOAD CAPABILITY IS NOT AVAILABLE
- **CUN3004I** IMAGE img_name WAS NOT FOUND

Migration consideration

Customers who have already created and loaded an image are not affected by this support.

3.10.2 Performance improvement for Unicode conversion service

Currently, the EBCDIC->UTF8 and UTF8->EBCDIC conversions use an intermediate conversion to CCSID 1200, hence are being treated as an “any-to-any” conversion. The any-to-any conversion involves many external module calls, which increases the execution path length.

A performance improvement has been made to the Unicode character conversion service. When certain new requirements are met, it no longer considers EBCDIC <-> UTF8 conversions as any-to-any and uses a new and shorter code path. In addition, it takes advantage of specific hardware instructions intended for EBCDIC<-> UTF8 conversion. This performance improvement encompasses all EBCDIC character sets, including Single-Byte (SBCS), Double-Byte (DBCS), and Multi-Byte (MBCS).

In order to activate the performance improvements in the character conversion between EBCDIC <-> UTF8 conversions, the following two new requirements have to be met. If both conditions are satisfied, the improved code path is used.

- During the image generation process, an existing reserved field in the control blocks is used to flag which tables can qualify for the performance improvements.

  Customers that have an existing image that contains EBCDIC <-> UTF8 conversion tables need to rebuild their conversion image in order to take advantage of this performance improvement. Images that are not rebuilt will continue to work but will not exploit the new performance improvement. Newly built images will automatically pick up the changes needed for this performance enhancement.

- During execution this flag is checked and a validation check is done to ensure that the work area buffer length is 3 times the source buffer length.

  Application developers who use z/OS Unicode conversion services must provide a larger WorkBuffer and/or TargetBuffer when calling the conversion services. Specifically, the target buffer or the work buffer must be 3 times the size of the source buffer, expressed mathematically as:

  target buffer len >= work buffer len >= source buffer len * 3

  Existing applications that don't change the target or work buffer lengths will continue to work using the longer code path.

Both changes need to be made in order to take advantage of the performance improvement.

3.11 Greater than 16 CPU support

Prior to z/OS V1R6, a single z/OS image supported a maximum of 16 processors, addressed from x'0' to x'F'. In many cases, new workloads are designed to be more CPU intensive in comparison to traditional workloads. Even the existing workloads running in an image may need more processing power. If a workload needs more CPU power, the simple solution is to split the workload into multiple images. But sometimes it may not be possible to do so.

As a solution to this, z/OS V1R6 has been enhanced to support more than 16 CPUs in a single z/OS image. As per the current change, z/OS V1R6 will support up to 24 CPUs (literally, CPUs with addresses up to x'17'). The greater than 16 CPU support is totally transparent to all personnel that interact with the system. However, the level of efficiency achieved is highly dependent upon workload.

This support is available with z990 and z890 processors. Review the PSP buckets for any necessary PTFs required for this support. The system messages, which contained a

single-digit CPU address, have been changed to include a 2-digit CPU address. Affinity scheduling is not supported for processors with addresses larger then x'F'.

3.12 Support for zAAPs

A new assist processor, called zSeries Application Assist Processor (zAAP) has been introduced for Java Applications on z/OS. zAAP is available and takes advantage of the z990 and z890 processor-rich environment. This is built upon z/OS increased processor scaling capabilities in z/OS 1.6, which supports more than 16 processors within a logical partition. zAAP may contribute to lowering the overall cost of computing for z/OS Java technology-based applications.

3.12.1 Overview of zAAPs

Strategic Web-based applications are increasing at exponential rates and many of them are driven by Java technology-based applications. Java applications require more IT resources than traditional applications. Web-based application workloads are also often unpredictable. The objective of zAAP is to move Java processing cycles to a lower-cost, fully integrated zSeries operating environment.

When configured with general purpose processors within logical partitions running z/OS, zAAPs may help increase general purpose processor productivity and may contribute to lowering the overall cost of computing for z/OS Java technology-based applications. zAAPs are designed to operate asynchronously with the general processors to execute Java programming under control of the IBM Java Virtual Machine (JVM). This can help reduce the demands and capacity requirements on general purpose processors, which may then be available for reallocation to other zSeries workloads. Figure 3-16 on page 58 illustrates this behavior. Without zAAP, the Java applications run in the general purpose CP, which consumes, for example, 1000 MIPs, 500 MIPs for Web application execution and 500 MIPs for Java code execution. Now, in the zAAP integrated environment, the Java code runs in the zAAP, thereby reducing the CPU utilization to 500 MIPs (in this example).

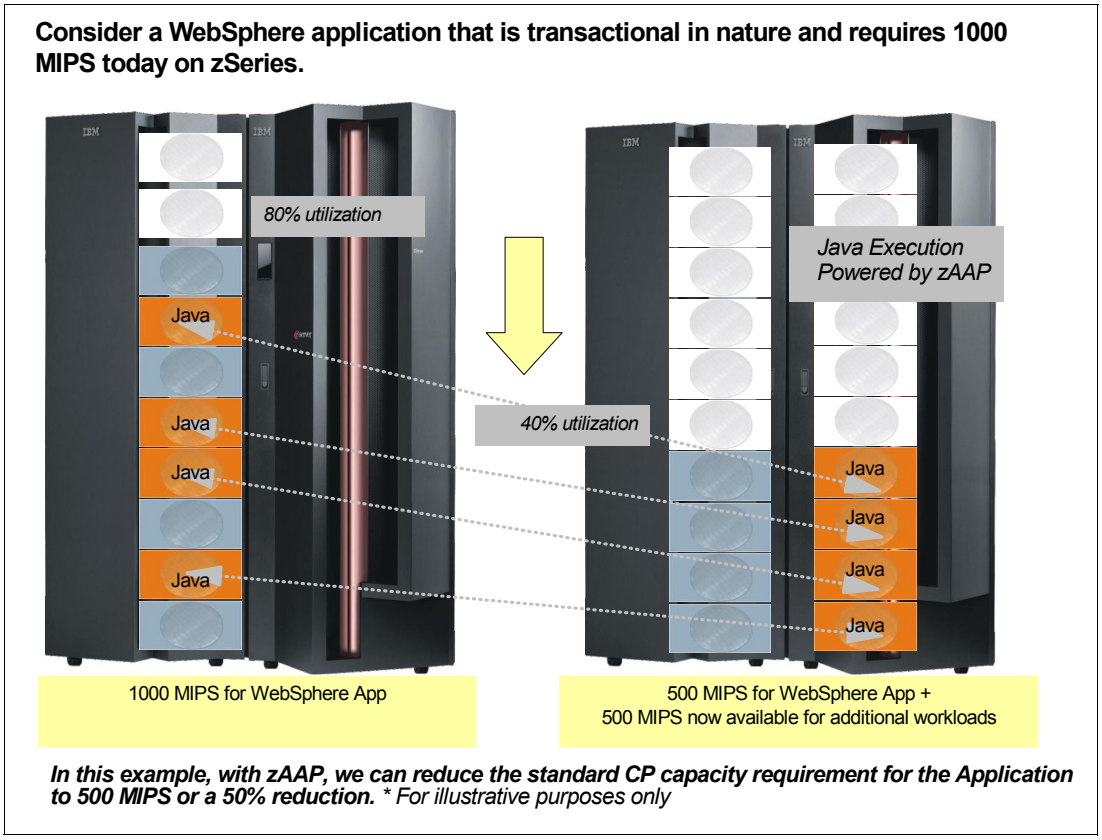

Figure 3-16 Concept of zAAP (for illustration only)

The amount of general purpose processor savings varies based on the amount of Java application code executed by zAAPs. This is dependent upon the amount of Java cycles used by the relevant applications and on the zAAP execution mode selected by the customer.

3.12.2 z/OS zAAP partition

When a z/OS logical partition is configured, both CPs and zAAPs are defined as necessary to support the planned Java and non-Java workloads. zAAPs may be configured as initially online or reserved for subsequent use by z/OS as necessary. Since zAAPs cannot be IPLed, at least one central processor is required for each z/OS partition.

zAAPs may be defined as either shared by other logical partitions or dedicated to a specific partition. However, both the CPs and zAAPs for each partition will have the same shared or dedicated attribute. For a given partition, you cannot define shared central processors and dedicated zAAPs, or vice versa. PR/SM™ configures shared zAAPs from the same pool of shared special purpose processors as the Internal Coupling Facility (ICF) and Integrated Facility for Linux (IFL).

Collectively, all shared ICFs, IFLs and zAAPs will also dynamically share in the processing requirements for all three special purpose processor types, as controlled by PR/SM.

Figure 3-17 on page 59 shows a z/OS logical partition with zAAP.

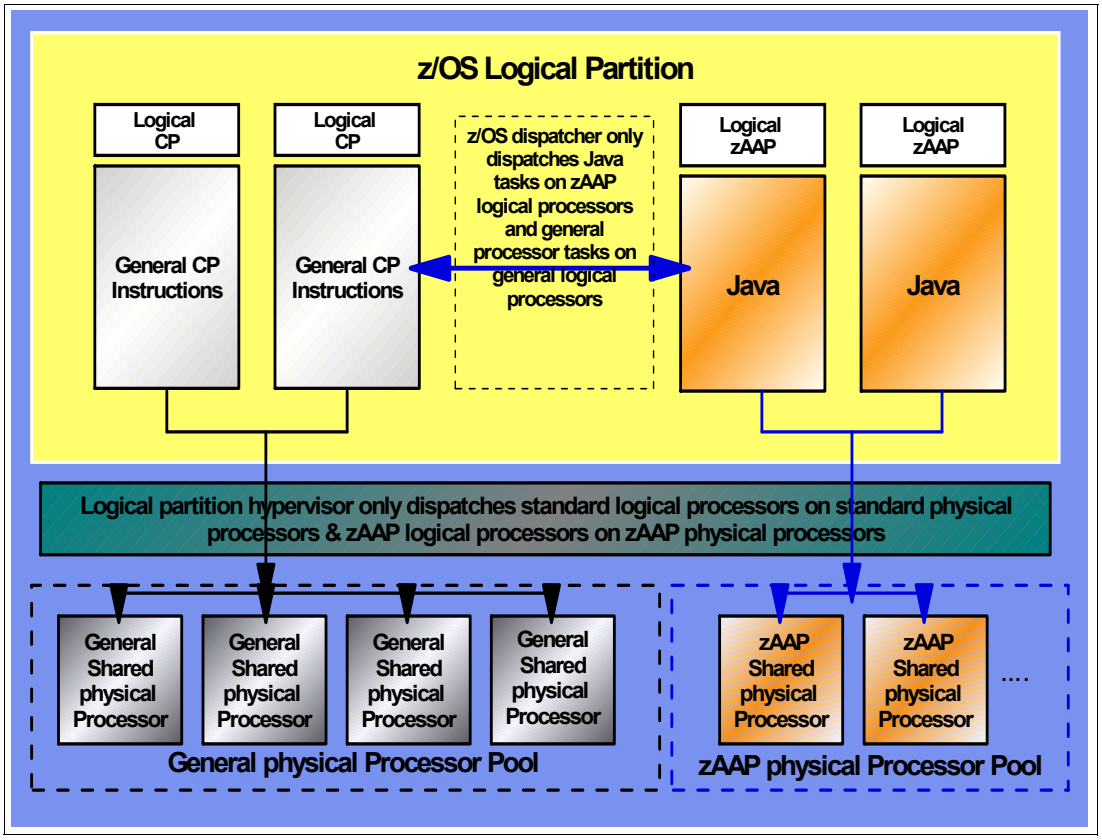

Figure 3-17 z/OS zAAP partition

3.12.3 zAAP workflow with z/OS V1R6

z/OS V1R6 provides the infrastructure to manage work units on the various processors. It recognizes the processor type and partitions work based on that type and whether the work unit is eligible for that type. Java provides the identification mechanism to mark a work unit as “zAAP-eligible” or “not zAAP-eligible”.

zAAP only executes z/Architecture mode instructions and IBM´s JVM code, Java code, and associated z/OS infrastructure code (for example, z/OS dispatcher, supervisor services, etc.). IBM´s JVM is the only authorized zAAP exploiter.

The IBM JVM processing cycles can be executed on the configured zAAPS with no anticipated modifications to the Java applications. Execution of the JVM processing cycles on a zAAP is a function of the Software Developer's Kit (SDK) 1.4.1 for zSeries, z/OS 1.6, and the Processor Resource/Systems Manager™ (PR/SM).

When z/OS is IPLed, it determines how many zAAPs are configured and subsequently assigns Java programming to the zAAPs when requested by the JVM. The JVM communicates with the z/OS supervisor to enable the JVM as eligible for execution on a zAAP. The next time the zAAP is available for dispatching, the JVM task is selected for execution. The z/OS dispatcher, operating in conjunction with the z990 PR/SM facility, dispatches zAAP eligible tasks on available zAAPs. During execution of the Java programming, when there is a Java Native Interface (JNI) call that executes code outside the JVM (for example, DB2), the JVM again communicates with the z/OS dispatcher to switch execution back to the next available central processor. On return from the JNI call, the JVM again enables itself as zAAP-eligible and the above process is repeated.

3.12.4 Java execution flow

Figure 3-18 shows the Java execution flow with zAAP, which depicts the IBM JVM requesting z/OS to switch execution to a zAAP, and then to switch back to a general purpose processor when the Java application ends, or when the application requests a non-Java programming function such as a database request. The z/OS dispatcher performs the switching to the appropriate logical processors. PR/SM then performs the appropriate logical processor switching to an associated physical processor.

Integrated Facility for Applications (IFA)-eligible work can run on both IFAs and standard processors. There are options that specify how the Java execution cycles are dispatched. You can specify that standard processors will not run any IFA-eligible work unless there are no IFAs operational in the partition. You can also prevent IFA-eligible work from running on standard processors because of software licensing considerations.

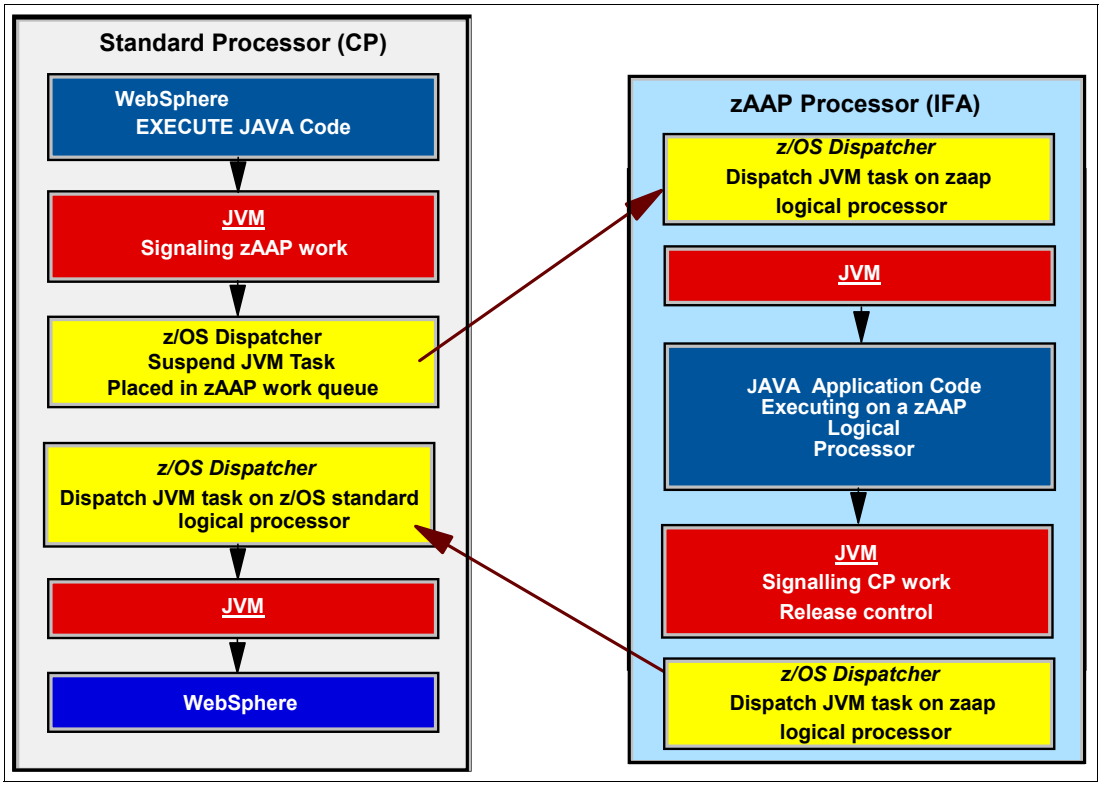

Figure 3-18 Java execution flow with zAAP

The example in Figure 3-18 shows the execution of a Java unit of work, as follows:

- Initially the Java code is dispatched on a standard processor (CP) and any other unit of work.
- Before the Java code gets executed on a Java machine (JVM), JVM signals to the dispatcher the current unit of work is zAAP-eligible work.
- When the current unit of work releases control, it is placed by the dispatcher in the zAAP dispatcher work queue. When a zAAP processor becomes available, the dispatcher selects the highest priority work from the zAAP work queue and dispatches it on the zAAP processor.
- A zAAP-eligible unit of work can be executed on a zAAP (if a zAAP is available). Work executing on the zAAP inherits the dispatching priority from the execution on the regular CP. The Java application code executes on this zAAP (also called an IFA, or Integrated

Facility for Application) processor. The MVS dispatcher dispatches the Java code to the zAAP processor unit.

- When the Java machine finishes processing (the unit of work finishes the execution of the Java code) it signals to the dispatcher that the current unit of work is not eligible for zAAP processing anymore. When the unit of work releases control, it is placed in the dispatcher “standard logical processor” work queue.
- The JVM returns to the WebSphere code running on the standard processor CP.

3.12.5 Java application calling DB2

Figure 3-19 illustrates a WebSphere server address space executing the IBM JVM and associated Java programming. In this example, the Java application program uses a Java Native Interface (JNI) to request a z/OS DB2 data base access. The JNI calls the Java JDBC API method, which in turn calls the associated JDBC Dynamic Link Library (DLL) routine. The JDBC DLL then uses a database connector (RRSAF) to interface with DB2, operating in its own address space, in order to access the application-requested data. All Java programs (the JVM) are eligible to execute on a zAAP. All programming outside the JVM will execute only on general purpose processors.

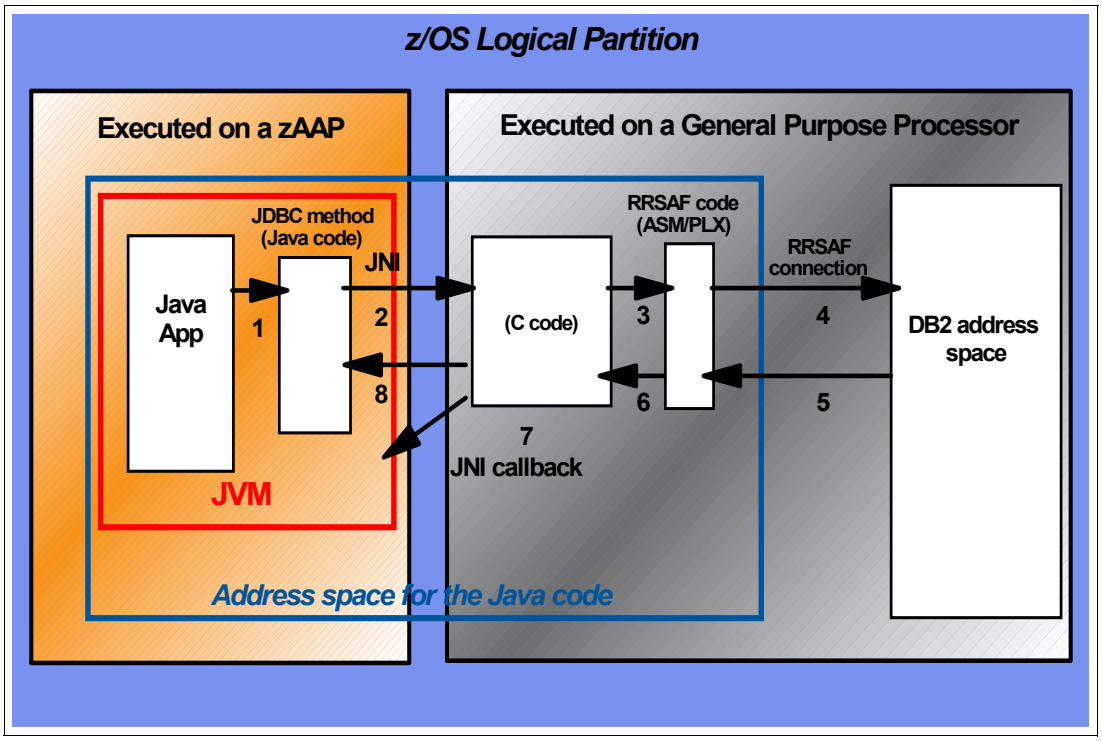

Figure 3-19 Java application calling DB2

3.12.6 Advantages of using zAAPs

Execution of the Java applications on zAAPs within the same z/OS LPAR as their associated database subsystems can also help simplify the server infrastructures and improve operational efficiencies. For example, use of zAAPs could reduce the number of TCP/IP programming stacks, firewalls, and physical interconnections (and their associated processing latencies) that might otherwise be required when the application servers and their database servers are deployed on separate physical server platforms.

zAAP integrated environments provide a cost-competitive single tier integrated container for:

- Application deployment (for example, WebSphere Application Server Java development platform)
- Direct, integrated DB access within the same logical partition without all the overhead implied by a TCP/IP communications stack
- Security and transaction contexts, network stack efficiencies (no wires, fewer TCP/IP stacks, firewalls, etc.)

Additionally, the following benefits are also provided:

- All of the zSeries and z/OS quality of service within the same logical partition.
- Takes advantage of the z990 multi-book scaling capabilities.
- Builds upon the increased processor scaling capabilities in z/OS 1.6, that is, more than 16 processors within a logical partition.
- Introduces a new special purpose engine built upon existing zSeries specialized engine technologies ICFs, IFLs, SAPs.

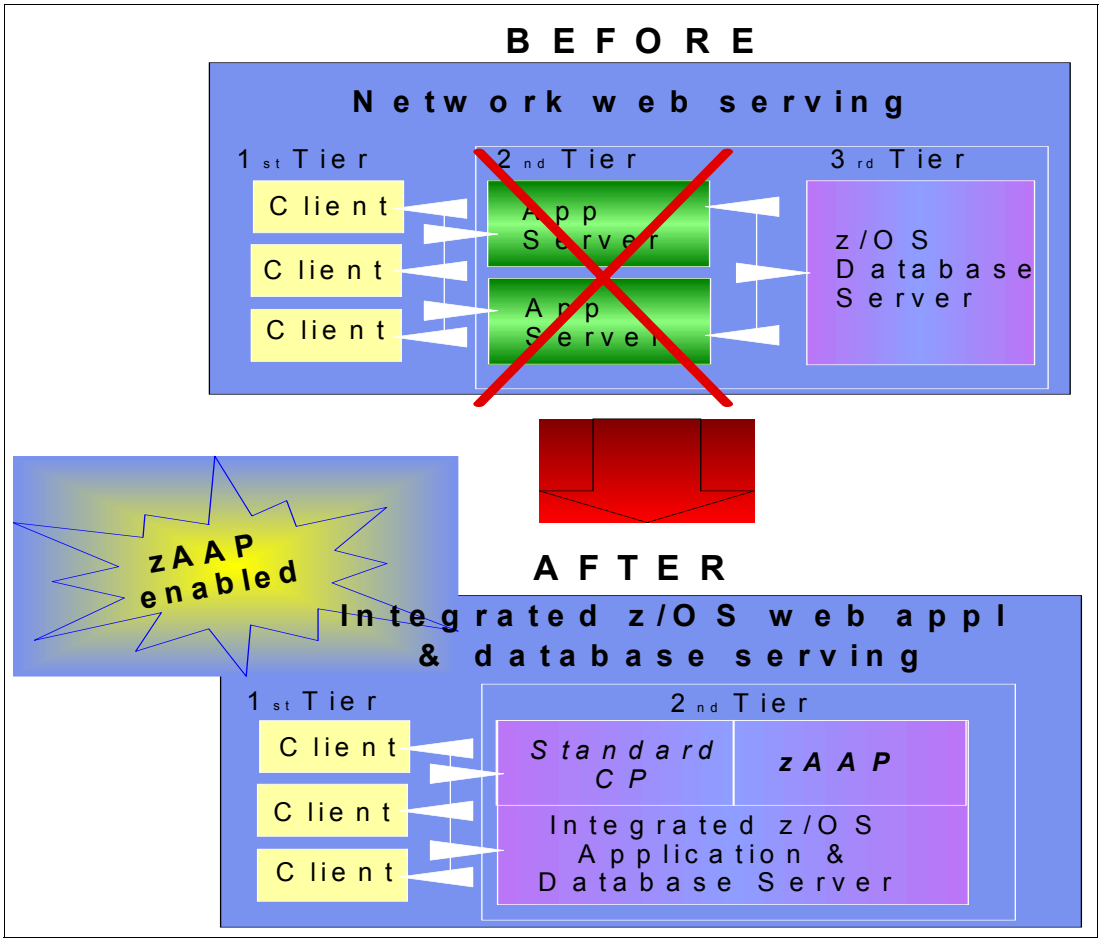

Figure 3-20 zAAP single-tier environment

zAAPs limitations

The following restrictions apply to zAAP processors:

- The system cannot be IPLed on a zAAP processor.

- zAAPs only execute z/Architecture mode instructions.
- zAAPs do not support all manual operator controls such as PSW Restart, LOAD, or LOAD derivative (load from file, CDROM, or server).
- zAAPs do not respond to SIGP requests unless enabled by a z/OS that supports zAAPs.

3.12.7 zAAP exploitation and Projection Tool

A zAAP Projection Tool for Java 2 Technology Edition, SDK 1.3.1, along with an accompanying Excel Workbook tool for reading, organizing, and analyzing data, allows customers who are considering using zAAPs to learn the potential for Java execution on zAAPs for their existing applications. This tool gathers usage information about how much CPU time is spent executing Java code that could potentially execute on zAAPs.

By running a Java workload that is representative of the production system operations, it reports, via the Java log, how much of that workload could be eligible for execution on zAAPs. This information is also useful for predicting the number of zAAPs that might be necessary in order to provide an optimum zAAP configuration.

Eligible subsystems for zAAP exploitation

Table 3-9 outlines the subsystems and minimum Java level dependencies for determining the zAAP Java execution and exploitation potential, or it can be used to determine the Java execution potential to use zAAPs. Use the table as follows:

- The items with a YES in the middle column, while not able to exploit the benefits of zAAPs, show where the zAAP Projection Tool for Java 2 Technology Edition, SDK1.3.1 can be used to assist in zAAP capacity planning. For instance, WebSphere V5.0.2 cannot exploit a zAAP processor. However, the zAAP Projection Tool can be used to determine zAAP Java execution potential if the WebSphere workloads were migrated to the required level for zAAP exploitation.
- The items with a YES in the last column can fully exploit zAAPs with the appropriate PTF level.

Table 3-9 IMinimum Java levels for zAAP exploitation or determining zAAP execution potential

Subsystem version	zAAP Projection Tool for Java 2 Technology Edition, SDK 1.3.1 - Used to Determine zAAP Java execution potential	IBM SDK for z/OS, Java 2 Technology Edition, V1.4, with PTF for APAR PQ86689
WAS 5.0.2	YES	
WAS 5.1		YES
IMS V7	YES	YES
IMS V8	YES	YES
IMS V9		YES
CICS 2.2	YES	
CICS 2.3		YES
DB2 V7	YES	YES
DB2 V8	YES	YES

Projection tool output

The tool is designed to create a print line in the STDOUT file every five minutes, as follows:

- This interval is not related to the RMF or SMF intervals.
- The time is reported as Java time eligible to execute on the zAAP.
- The remaining time is identified as standard CP time.
- In addition, the total CPU time for all TCBs, SRBs and enclaves is reported as address space time.

Figure 3-21 is an example of the format and contents of this print line. The key fields of interest for performance analysis, which are highlighted in this figure, are the following:

Switches To/From IFA	State changes in processing of zAAP-eligible work versus not eligible work for all Java threads in the JVM
Java IFA	Time accumulated for Java threads processing zAAP-eligible work
Java Standard CPU	Time accumulated for Java threads processing zAAP non-eligible work
Interval address space CPU	Time for all work in the address space across all dispatchable units, including the Java threads

```
IFA Projection data for system id=<SYSD.50594238> Starting at: 18:31:30- Current address
space CPU: 0.008068 sec.
<SYSD.50594238> Interval at: 18:36:30 Switches To/From IFA: 3717251 Java IFA: 99.3 sec.
Java Standard CPU 101.86 sec. Interval address space CPU: 208.66 sec.
<SYSD.50594238> Interval at: 18:41:30 Switches To/From IFA: 3903114 Java IFA: 104.27
sec. Java Standard CPU 106.95 sec. Interval address space CPU: 219.09 sec.
<SYSD.50594238> Interval at: 18:46:30 Switches To/From IFA: 4176332 Java IFA: 111.57
sec. Java Standard CPU 114.44 sec. Interval address space CPU: 234.43 sec.
<SYSD.50594238> Interval at: 18:51:30 Switches To/From IFA: 3842225 Java IFA: 102.64
sec. Java Standard CPU 105.28 sec. Interval address space CPU: 215.68 sec.
<SYSD.50594238> TOTAL at: 18:51:30 Switches To/From IFA: 15638922 Java IFA: 418 sec.
Java Standard CPU 429 sec. Total address space CPU: 878 sec.
```

Figure 3-21 Output from the projection tool

Microsoft Excel workbook

A Microsoft Excel workbook is available, with associated programming, that processes the output shown in Figure 3-21 and stores the zAAP processing information in a spreadsheet. The workbook can help a capacity planner to manipulate the data in the spreadsheet. Figure 3-22 is a sample of the contents of the spreadsheet after execution within the workbook.

SMF name	Instance or Group	RMF interval start	zAAP	CP	Space	%Time zAAP eligible	zAAP% engine eligible	Other Java% engine	Appl% engine	zAAP% w/capt ratio	ZAAPs w/wait
			Service Class		newwork		all LPARS			85%	75%
SYSD	test1	18:31:00	99	102	209	48%	33%	34%	70%	39%	52%
SYSD	test1	18:36:00	104	107	219	48%	35%	36%	73%	41%	55%
SYSD	test1	18:41:00	112	114	234	48%	37%	38%	78%	44%	58%
SYSD	test1	18:46:00	103	105	216	48%	34%	35%	72%	40%	54%

Figure 3-22 Contents of a spreadsheet following processing in Excel workbook

zAAP Projection Tool workbook

The zAAP Projection Tool workbook can be downloaded from the same Web site as the zAAP Projection Tool. This site is:

```
www6.software.ibm.com/dl/zosjava2/zosjava2-p
```

The Projection Tool workbook is capable of doing the following:

- Combining data from multiple JVMs.
- Collecting seconds of zAAP-eligible processing, non zAAP-eligible (standard CP) processing, and total address space time for the Java spaces.
- Combining data from multiple address spaces, service classes and LPARs.
- Combining the data and aligning it to intervals such as the RMF interval used.
- Adjusting zAAP utilization by factoring in z/OS capture ratios.
- Expressing zAAP and standard CP time as a percent of the engine (single CP) that the data was collected on. This can be used as input to the projected number of zAAPs needed, factoring in a target maximum utilization to ensure workload responsiveness.

3.13 Binder enhancements

z/OS provides program management services that let you create, load, modify, list, read, and copy executable programs. With the program management binder, you can create executable modules in either of two formats and store them (depending on the format) in PDS or PDSE libraries, or in z/OS UNIX files. The two types of executable modules are load modules and program objects that collectively may be referred to as “program modules.” Of these two formats, program objects are the newer. They remove many of the restrictions of the load module format and support new functionality. You can use the z/OS loader to load saved program modules into virtual memory for execution. You can also use the program management binder to build and execute a program in virtual storage in a single step.

The existing keywords (CODE and DATA) in the IMPORT statement cannot distinguish between 32-bit vs. 64-bit DLLs. z/OS V1R6 provides two new keywords, CODE64 and DATA64, to support 64-bit CODE and DATA in the IMPORT statement.

3.13.1 Binder control statements

Control statements are provided to the binder to specify editing operations and identify additional input. Also, you can provide entry and module names and specify the authorization code of a program module. Each binder control statement specifies an operation and one or more operands. Changes have been made to the IMPORT and INCLUDE control statements.

IMPORT statement

The IMPORT statement specifies an external symbol name to be imported and the library member or z/OS UNIX file name where it can be found. An imported symbol is one that is expected to be dynamically resolved. Two new keywords, CODE64 and DATA64, have been added to this control statement.

IMPORT {CODE | DATA | CODE64 | DATA64}, *dllname, import_name*, [,*offset*]

CODE or CODE64 If specified, the *import_name* must represent the name of a code section or entry point. CODE64 is specified when using 64-bit addressing mode and CODE is specified for any other addressing mode.

DATA or DATA64 If specified, the *import_name* must represent the name of a variable or data type definition to be imported. DATA64 is specified when using 64-bit addressing mode and DATA is specified for any other addressing mode.

dllname The name of the DLL module that contains the import_name to be imported. If it is a member of a PDS or PDSE, it must be a primary name or an alias. The length is limited to eight bytes unless it is an alias name in a PDSE directory. In that case, the limit is 1024 bytes. If it is a z/OS UNIX file, the file name is limited to 255 bytes.

import_name Specifies the symbol name to be imported. It can be up to 32767 bytes in length.

offset Contains up to 8 hexadecimal characters.

Figure 3-23 shows an example of an IMPORT control statement.

```
//         EXEC PGM=IEWL,PARM='MAP,XREF,CASE=MIXED'
//LOADMOD  DD DSNAME=PROJECT.LOADLIB,DISP=SHR
//OBJECT1  DD PATH='/sl/app1/pm3d3/dlla01',PATHDISP=(KEEP,KEEP)
//SYSLIN   DD *
  IMPORT CODE TAXES97,Compute_97_Taxes_Schedule1
  IMPORT CODE TAXES97,Compute_97_Taxes_Schedule2
  IMPORT CODE64 TAXES03,Compute_03_Taxes_Schedule1
  IMPORT CODE64 TAXES03,Compute_03_Taxes_Schedule2
  IMPORT DATA REVENUE,TotalRevenue
  IMPORT DATA64 REVENUE03,TotalRevenue03
  INCLUDE OBJECT1
  ...
/*
```

Figure 3-23 Example of an IMPORT control statement

INCLUDE statement

The INCLUDE control statement specifies sequential data sets, library members, or z/OS UNIX files that are to be sources of additional input for the binder. Starting with z/OS V1R6, the IMPORTS option has been made the default and three new options, NOIMPORTS, NOATTR, NOALIASES, have been added.

INCLUDE [{-ATTR, | -IMPORTS, | -ALIASES, | -NOATTR, | -NOIMPORTS, | -NOALIASES}...] {*ddname*[(*membername*[,...])][,...] | *pathname*[,...]}

IMPORTS Specifies that dynamic resolution information (if any) will be copied from the input module. Starting in z/OS V1.6 this option is no longer required, as the INCLUDE statement will always bring in any available dynamic resolution information unless it is suppressed by -NOIMPORTS. This option is still supported for compatibility reasons.

NOIMPORTS Specifies that dynamic resolution information (if any) will not be copied from the input module.

NOATTR Specifies that module attributes will not be copied from the input module.

NOALIASES Specifies that the aliases of the input will not be copied from the input module.

Note: If both IMPORTS and NOIMPORTS options are specified, the last valid option will be used.

Figure 3-24 shows an example of using the INCLUDE control statement.

```
//LOADMOD  DD DSNAME=PROJECT.LOADLIB,DISP=SHR
//OBJECT2  DD PATH='/sl/app1/pm3d3/dllaO2',PATHDISP=(KEEP,KEEP)
//SYSLIN   DD *
  INCLUDE LOADMOD(TESTMOD,READMOD)
  INCLUDE '/ml/app1/pm3d3/dlla01'
  INCLUDE OBJECT2
  ...
/*
```

Figure 3-24 Example of the INCLUDE control statement

3.13.2 Binder API

The IMPORT API can be called by the IEWBIND macro or by placing the address of the import parameter list in general purpose register 1. The IMPORT API accepts new ITYPE options CODE64 or DATA64. If the IEWBIND macro is not used, “E” is used for CODE64 and “F” is used for DATA64 when using the IMPORT parameter list.

The INCLUDE API IMPORTS option value has been changed from NO to YES.

SYSDEFSD DD statement—side file

Side file has been changed to mark exported function and data that are AMODE 64 using CODE64 and DATA64.

When the DYNAM(DLL) option is used to build a DLL module, a side file might be generated along with it. The side file is saved in the data set represented by the SYSDEFSD ddname. The side file contains a collection of IMPORT control statements that can be used by other DLLs in order to resolve their own external references during dynamic linking. If a DLL does not export any symbols, no side file is generated for it. When the side file is used as input to the bind, any statement not explicitly specifying CODE64 or DATA64 will be interpreted as 31-bit (AMODE=31) DLLs. When an application that wishes to use exported symbols from a DLL is linked, the binder issues an error message if the AMODE of the referencing ESD does not match that specified on the import statement.

Imported and Exported Symbol Table

The Imported and Exported Symbol Table is part of the Module Summary Report. This table is printed if binder options XREF and DYNAM(DLL) are specified and there are symbols to import or export. This table has been changed to print DATA64 and CODE64 for data and functions that are AMODE 64. Figure 3-25 shows an example.

```
 *** IMPORTED         AND        EXPORTED         SYMBOLS ***
IMPORT/EXPORT   TYPE       NAME                          MEMBER
--------------------------------------------------------------------------------
EXPORT          CODE64     __dt__9strstreamFv
EXPORT          DATA64     __instance_5Locks
```

Figure 3-25 Example of Imported and Exported Symbol Table

3.14 Linkage index reuse

The program call (PC) instruction has an operand called PC number, which consists of 12-bit Linkage Index (LX) and 8-bit Entry Index (EX). Although the architectural limit for LX is 4096, z/OS supports only up to 2048 LXs. The NSYSLX parameter in the IEASYSxx parmlib member increases or decreases the number of system LXs available for system use.

There are system LXs and non-system LXs. The LXRES macro is used with SYSTEM=YES to obtain a system LX and with SYSTEM=NO to obtain a non-system LX. A system LX is usable by all address spaces with no overt action on their part. When the owner terminates, the system LX is never made available for someone else to reuse. The original requester, however, can choose to reconnect to the LX when the address space terminates and restarts. A non-system LX is usable only by address spaces that connect through it. When the owner terminates, the LX is made available for someone else to use when all connectors have disconnected.

Each instance of DB2 or MQ uses multiple LXs. A customer site often has multiple instances of DB2 and MQ running in the system. When these address spaces are recycled, some of their LXs remain unavailable for reuse. Over a period of time, the system runs out of LXs.

Enhancements have been made to reuse LXs in circumstances where previously they could not be reused and by increasing the number of LXs, by using:

- New hardware architecture
- z/OS infrastructure exploitation
- Application exploitation

3.14.1 New hardware architecture

HW uses the high 12 bits of the PC number that had been ignored previously. Instead of 12 bits of LX, there can now be 24 bits of LX, called “big LX”. The additional 12 bits are called the “sequence number”. As a result, the PC instruction now takes both a PC number and a sequence number to build the LX, as follows:

- When the LX has the 2048 bit off, LX is 0-2047 (12 bits) and it works as it does today.
- When LX has the 2048 bit on and a sequence number is not associated, the PC instruction needs only the PC number.
- When LX has the 2048 bit on and a sequence number is associated, the PC instruction needs both PC number and sequence number.

z/OS infrastructure

z/OS V1R6 supports up to 32 K LXs and provides support for associating a sequence number with the PC number.

Application exploitation

To create and use an LX that can be > 2047, create both the PC number and sequence number before issuing the PC instruction. To reuse an LX that can be > 2047, issue the PC with a PC number and a sequence number.

3.14.2 z/OS V1R6 support

The following changes have been made to z/OS V1R6 for this support:

- z/OS V1R6 supports 32K LXs.

- A bit in the PSA (PSAALR, byte PSAMISCF offset x'A7D'), which is the same as a bit in the CVT (CVTALR, byte CVTFLAG2, offset x'179'), will indicate if new architecture is both available and enabled. When this bit is “1”, the new architecture is available and usable. If the bit is “0”, the old architecture is in force.
- The SYSSTATE OSREL parameter indicates the release this code path will be running on. For example, SYSSTATE OSREL=ZOSV1R6 indicates that the macro expansions can assume that the code will be running on ZOSV1R6.
- Basic PC services (for example, ETCRE, ETCON, and LXRES) are sensitive to SYSSTATE OSREL. Produce stacking-PC expansion only when the z/OS release supports the code path.
- Changes have been made to the ELXLIST parameter on LXRES, ETCON, LXFRE, and ASCRE.
- The LXLIST parameter now consists of a 4-byte count followed by 4-byte entries.
- The ELXLIST parameter now consists of a 4-byte count followed by 8-byte entries. The high 4 bytes of each entry is the sequence number associated with the LX.
- LXRES LXSIZE=12 | 16 | 23 | 24, specify the largest LX size.
- There are new messages added for Big LX shortages:
  - `IEA070E SYSTEM BIG LX SHORTAGE HAS BEEN DETECTED`
  - `IEA071I SYSTEM BIG LX SHORTAGE HAS BEEN RELIEVED`
  - `IEA072E NON-SYSTEM BIG LX SHORTAGE HAS BEEN DETECTED`
  - `IEA073I NON-SYSTEM BIG LX SHORTAGE HAS BEEN RELIEVED`
- There are new reason codes added for a 052/053 abend:
  - 052-051B: bad LX sequence number on ETCON
  - 052-0216: bad LX sequence number on LXFRE
  - 053-1416: more than 16383 system big LX's requested
- There are new reason codes added for the 0E0 completion code:
  - Linkage First Table exception
  - Linkage Second Table exception

3.15 Reallocate structure after CF maintenance

After a CF structure is initially allocated, the system makes no attempt to optimize the placement of that structure. There are many reasons why structures can and do move around and over time get into suboptimal locations. Specifically, there is no easy means for restoring structures to their desired location. In cases where structures support user-managed or system-managed duplexing rebuild and the installation has three or more CFs, the use of existing commands to restore structures to their desired locations is a cumbersome process.

The situations in which a structure moves around include moving structure instances to accommodate CF maintenance or needing to move to facilitate activating a new CFRM administrative policy. Afterwards, the structure instances do not reside in their desired CF locations, and performance may be impacted. The series of operator commands needed to relocate the instances are complex and prone to operator error.

A method is needed to facilitate the movement of structure instances such as DB2 GBPs to their desired CFs. APAR OA03481 provides a new REALLOCATE parameter with commands SETXCF START and SETXCF STOP to address this issue.

3.15.1 REALLOCATE process

The REALLOCATE process provides a means to evaluate each allocated structure and to adjust as needed the location of structure instances with a single operator command. This command works like POPULATECF (whose primary usage is to populate a Coupling Facility that has been brought into use in a sysplex). Unlike POPULATECF, the REALLOCATE process can adjust the location of both simplex and duplexed structures. The advantage of the REALLOCATE process is that it uses structure allocation criteria to determine whether to initiate Structure Rebuild processing to move allocated instances into CFs listed in the preference list from either the active or pending CFRM policy. If the structure instances already reside in the “more suitable” coupling facility, the system will not initiate Structure Rebuild processing.

For example, assume that a CF structure is duplexed with the old instance in the third CF listed in the preference list and the new instance in the first CF listed in the preference list. If evaluation using the structure allocation criteria determines that the “more suitable” location for the structure instances should have the old instance in the first CF listed and the new instance in the second CF listed, then the REALLOCATE process would take two steps to adjust the location by stopping duplexing, keeping the new instance followed by reduplexing the structure. As a result, the old instance will be in the first CF listed in the preference list and the new instance will be in the second CF listed.

3.15.2 Example of operator actions

The following example outlines the steps to be taken by an operator to remove a Coupling Facility from the configuration for maintenance and then subsequently returning it to service. In the example, CF1 is to be removed, leaving CF2 and CF3 in the configuration. The structures in CF1 must be moved to one of the other Coupling Facilities before CF1 can be removed from the configuration.

- Removing CF1 from the configuration

  The operator can move structures in CF1 to another Coupling Facility with the commands:

  - SETXCF START,REBUILD,CFNAME=CF1,LOCATION=OTHER

    The LOCATION=OTHER specification indicates that the structures are to be rebuilt in another Coupling Facility in each structure's preference list other than the Coupling Facility in which the structure now resides. (If no other suitable Coupling Facility exists in the CFRM policy for a particular structure, it might be necessary to update the CFRM policy so that a Coupling Facility can be found.)

  - SETXCF START,REBUILD,STRNAME=strname,LOCATION=OTHER

    Using this command, the XCF signalling structure is rebuilt one at a time.

  - SETXCF STOP,REBUILD,DUPLEX,CFNAME=CF1

    This command is used to stop structure duplexing. When a duplexing rebuild is stopped with CFNAME specified, the instance allocated in the specified Coupling Facility is *not* kept.

  Once all structures have been removed from CF1, it can be removed from the configuration for maintenance.

- Moving structures back into CF1

  The user can take any of the following two actions to move the structure back after CF1 is brought back to the configuration.

  a. After CF1 has been returned to the sysplex configuration, the operator can issue the SETXCF START,REBUILD,POPULATECF=CF1 command to move structures back

into CF1. The POPULATECF keyword ensures that only those structures that have CF1 as a higher Coupling Facility in their preference list than where the structure is currently allocated will be rebuilt in CF1. Thus, the entire set of allocated simplex structures in the CFRM active policy are eligible to be rebuilt in CF1, even if they were not part of the original CF1 evacuation.

As each structure in the CFRM policy is examined, messages are issued to the operator indicating:

- Whether a structure is eligible to be rebuilt
- Whether a structure is not eligible to be rebuilt because CF1 is not in the structure's preference list
- Whether the structure is to remain in its current location because, although it might be eligible to be rebuilt, CF1 is not higher in the structure's preference list than that in which the structure already resides or there are other reasons why CF1 is not more suitable
- The successful rebuild of a structure into CF1

Note: After the population of CF1, if it is necessary to redistribute the structures for a better sysplex balance, the preference lists in the CFRM policy could be updated. Once the updated CFRM policy is activated, the POPULATECF function could again be used for CF1 or any other Coupling Facility to redistribute the structures.

b. After CF1 has been returned to the sysplex configuration, the operator can issue the SETXCF START,REALLOCATE command to move structures back into CF1 as appropriate, based on the current CFRM policy and structure allocation criteria. The REALLOCATE process ensures that only those structure instances allocated in a "less suitable" Coupling Facility have structure rebuild processing initiated to adjust the location or activate a pending policy change.

   Thus, the entire set of allocated structures (simplex or duplexed) in the CFRM active policy are eligible to be evaluated for placement in CF1 and any other Coupling Facility in use by the sysplex, even it they were not part of the original CF1 evacuation. As each structure in the CFRM policy is examined, messages are issued to the operator indicating:

   - IXC574I the current location of instances, the policy information used, and the results of applying the XCF allocation algorithm
   - IXC544I when a REALLOCATE processing is not attempted for the structure
   - Messages for structure rebuild processing when a structure needs to be reallocated
   - IXC545I and IXC543I when the REALLOCATE process completes

   Note: The POPULATECF function and the REALLOCATE process are mutually exclusive.

3.15.3 Comparison of POPULATECF and REALLOCATE

Table 3-10 shows a comparison between the POPULATECF function and REALLOCATE process.

Table 3-10 Comparison of the POPULATECF function and REALLOCATE process

POPULATE function	REALLOCATE process
Use SETXCF commands or the IXLREBLD macro to start or stop.	Use SETXCF commands to start or stop.
Only supports simplex structures.	Supports both simplex and duplexed structures.
Uses the position of the specified coupling facility in the preference list of a structure to select as a candidate structure.	Evaluates current location of structure instances based on structure allocation criteria to select as a target structure.
Predetermines the set of simplex structures that are pending rebuild and lists in message IXC540I.	Does not predetermine the set of structures but selects a target structure, takes necessary steps, then proceeds to examine the next allocated structure.
Only a single step (rebuild that can be user-managed or system-managed) is used to relocate the structure.	The number of steps used to adjust the location is based on whether the structure is simplex or duplexed.
Only the specified CF is considered for allocating the new instance.	The new instance may be in any in-use CF and is selected based on the structure allocation criteria.

3.15.4 New keywords

A new keyword, REALLOCATE, has been added to the SETXCF START|STOP command.

SETXCF START *command*

The command is: SETXCF START,REALLOCATE

REALLOCATE or REALLOC Specifies that the REALLOCATE process is to be initiated. The status of the REALLOCATE process will be “IN PROGRESS” as shown by DISPLAY XCF,STR or CF.

The REALLOCATE process

The REALLOCATE process uses existing XCF structure allocation algorithms to recognize the need to relocate structure instances from the set of allocated structures in the CFRM active policy. It does this by comparing the current location with the location selected by allocation criteria using either the active or pending CFRM policy. Message IXC574I is issued to show the current location of instances allocated in CFs, the policy information used, and the results of applying the XCF allocation criteria.

When the locations differ or a policy change is pending, the REALLOCATE process uses the structure rebuild process to accomplish the needed adjustments. Structure rebuild processing supports:

- User-managed rebuild
- User-managed duplexing rebuild
- System-managed rebuild
- System-managed duplexing rebuild

Multiple steps may need to be taken to complete the relocation of a given structure. The steps are accomplished using structure rebuild processing (for example, user-managed rebuild) to adjust the location and/or activate a pending policy change for the structure which is the target of the REALLOCATE process. Messages to the operator document the steps being taken for each structure that is examined.

For a simplex structure, one step (rebuild) is used to adjust the location and/or to activate a pending policy change.

For a duplexed structure, two or three steps are used, with step one to stop duplexing and subsequent steps used to rebuild as needed to adjust the location and/or activate a pending policy change and to reduplex the structure.

When the REALLOCATE process does not select an allocated structure, message IXC544I is issued with explanatory text.

Use the DISPLAY XCF,STR,... command to show the current state of processing for allocated structures. An allocated structure may be pending evaluation by the REALLOCATE process or be the current target of the REALLOCATE process. When a structure is not allocated or already examined by the REALLOCATE process, no additional status is displayed.

When the entire process completes, for all structures, the processing provides a report (message IXC545I) summarizing the actions that were taken as a whole.

The REALLOCATE process evaluates all allocated structures, in a serial (one structure at a time) fashion. Each selected structure is processed to completion before the next structure is evaluated. The serial nature of this processing allows even XCF signalling structures to be selected for relocation.

REALLOCATE processing evaluates a structure based on the CFRM policy and on the current conditions, for example available CFs, CF attributes, and connection attributes. For each structure that is not optimally located, it takes the necessary steps to adjust the location of the structure's allocated instances.

It is possible that the conditions of the structure have changed between the time it is evaluated and the time when the steps using structure rebuild processing cause a new instance to be allocated. But the current conditions are used when the structure allocation algorithm is applied. The REALLOCATE process does not validate the resulting location of the allocated instances, but relies on the result of applying the XCF allocation criteria. Because of this, when the necessary steps finish, it is possible that the preferred CFs shown in the message IXC574I, issued with the evaluation information, are not the CFs containing the allocated instances.

Note:

1. The REALLOCATE process is mutually exclusive with the POPULATECF function, which is started either by the SETXCF operator command or the IXLREBLD programming interface.
2. The REALLOCATE process can only be started or stopped using the SETXCF operator commands.
3. Support for the REALLOCATE process is provided by APAR OA03481.
   - The REALLOCATE process cannot be started if an active system exists in the sysplex that does *not* have the APAR installed. If such a system exists, the SETXCF START,REALLOCATE command is rejected.
   - An “in progress REALLOCATE process” is stopped immediately when an active system without the APAR installed is discovered in the sysplex. That means, the SETXCF START,REALLOCATE command is accepted but subsequently an active system without the APAR installed is discovered by an uplevel system, which immediately stops the process.

   In both cases, message IXC543I is issued with explanatory text.

SETXCF STOP command

The command is: SETXCF STOP,REALLOCATEY,[FORCE]

REALLOCATE or REALLOCY,FORCE Specifies that an “in progress REALLOCATE process” is to be stopped.

When stopping without specifying FORCE, REALLOCATE processing completes the steps for the current target structure and then finishes. The status of the REALLOCATE processing will be “STOPPING” as shown by DISPLAY XCF,STR or CF.

When stopping with FORCE specified, REALLOCATE processing finishes immediately *and* the steps for the current target structure may *not* be completed. The FORCE option should be used when structure rebuild processing for the structure that is the target of the REALLOCATE process is not making progress.

When the process finishes, the processing provides a report (message IXC545I) for the selected structures, summarizing the actions that were taken up to the time that processing was stopped. To stop the REALLOCATE process does *not* require issuing the command without FORCE specified before issuing with FORCE specified.

3.15.5 New and changed messages

The following messages are changed for this support:

- IXC359I
- IXC360I
- IXC361I
- IXC362I
- IXC574I

The following new messages are added for this support:

- IXC543I
- IXC544I
- IXC545I
- IXC546I

Note: Install the PTF on all systems in the sysplex prior to issuing (from any system in the sysplex) the new operator command to initiate the REALLOCATE process. A rolling IPL is sufficient to activate the PTF.

4

z/OS V1R6 DFSMS enhancements

DFSMS is an exclusive element of the z/OS operating system. It is a software suite consisting of five elements: DFSMSdfp, DFSMSdss™, DFSMShsm, DFSMSrmm™, and DFSMStvs. These automatically manage data from creation to expiration. This chapter describes the various enhancements to z/OS V1R6 DFSMS.

The following topics are covered:

- DFSMSdfp enhancements
  - SMS volume selection based upon PAV
  - PDSE restartable address space
  - Multilevel security (MLS) SECLABEL in ACS routines
  - Miscellaneous changes
- DFSMSdss enhancements
  - REPLACEUnconditional keyword
  - Record level sharing (RLS) considerations
- DFSMShsm enhancements
  - Multiple secondary space management (SSM) tasks
  - Additional SSM enhancements
- DFSMSrmm enhancements
  - DFSMSrmm client/server support
  - RMM API command classes
  - RMM ISPF usability

4.1 DFSMSdfp enhancements

DFSMSdfp provides various functions: storage management, tape mount management, data management, device management, distributed data access, advanced copy services, and object access method services. z/OS V1R6 DFSMSdfp has been enhanced to support:

- SMS volume selection based upon Parallel Access Volume (PAV)
- PDSE restartable address space
- Multilevel security (MLS) SECLABEL in automatic class selection (ACS) routines

4.1.1 SMS volume selection based upon PAV

Parallel Access Volume (PAV) allows the host system to access the same logical volume using alternative device address unit control blocks (UCBs). The ability to do multiple I/O requests to the same volume nearly eliminates IOSQ time, one of the major components in z/OS response time.

DFSMSdfp provides new support that allows allocations to be automatically directed to high performance devices. This is intended to give the user additional control over the volume selection process for an SMS-managed data set depending on whether a base volume has a PAV or PAVs associated with it.

For this new support:

- A new field, named PAV Capability, has been provided in the Storage Class definition.
- Changes have been made to the volume selection algorithm to take PAV into consideration.
- ISMF and NaviQuest have been updated to support the new function.

Current volume selection algorithm

When a data set is created, SMS follows a sequence of verification steps to determine placement on a volume. It attempts to spread initial allocations evenly across similar volumes and storage groups provided by the ACS storage group selection routine.

Before the volume selection process begins, SMS categorizes each volume in a storage group into one of four lists from which volumes are selected for data set allocation:

Primary list All the volumes in all the specified storage groups are candidates for the primary list, which consists of online volumes that meet all the specified criteria in the storage class and data class, are below their threshold, and whose volume status and storage group status are enabled. All volumes on this list are considered equally qualified to satisfy the data set allocation request. Volume selection starts from this list.

Secondary list Volumes that do not meet all the criteria for the primary volume list are placed on the secondary list. If there are no primary volumes, SMS selects from the secondary volumes.

Tertiary list Volumes are marked for the tertiary list if the number of volumes in the storage group is less than the number of volumes requested. If there are no secondary volumes available, SMS selects from the tertiary candidates.

Rejected list Volumes that do not meet the required specifications (ACCESSIBILITY = CONTINUOUS, AVAILABILITY = STANDARD or CONTINUOUS,

ENABLED or QUIESCED, ONLINE...) are marked rejected and are not candidates for selection.

After the system selects the primary space allocation volume, that volume's associated storage group is used to select any remaining volumes requested for the data set. If you specify an extend storage group, the data may be extended to the specified extend storage group.

Volume selection algorithm changes

Now, you can use the new PAV option of the storage class to improve performance. This option allows you to specify PAV capability as one of the volume selection criteria for the SMS-managed data sets assigned to a storage class. With this, you can ensure that the data sets requiring high performance are allocated to volumes using the parallel access volume feature of the IBM Enterprise Storage Server® (ESS).

Figure 4-1 shows the new PAV capability option to be specified to a storage class.

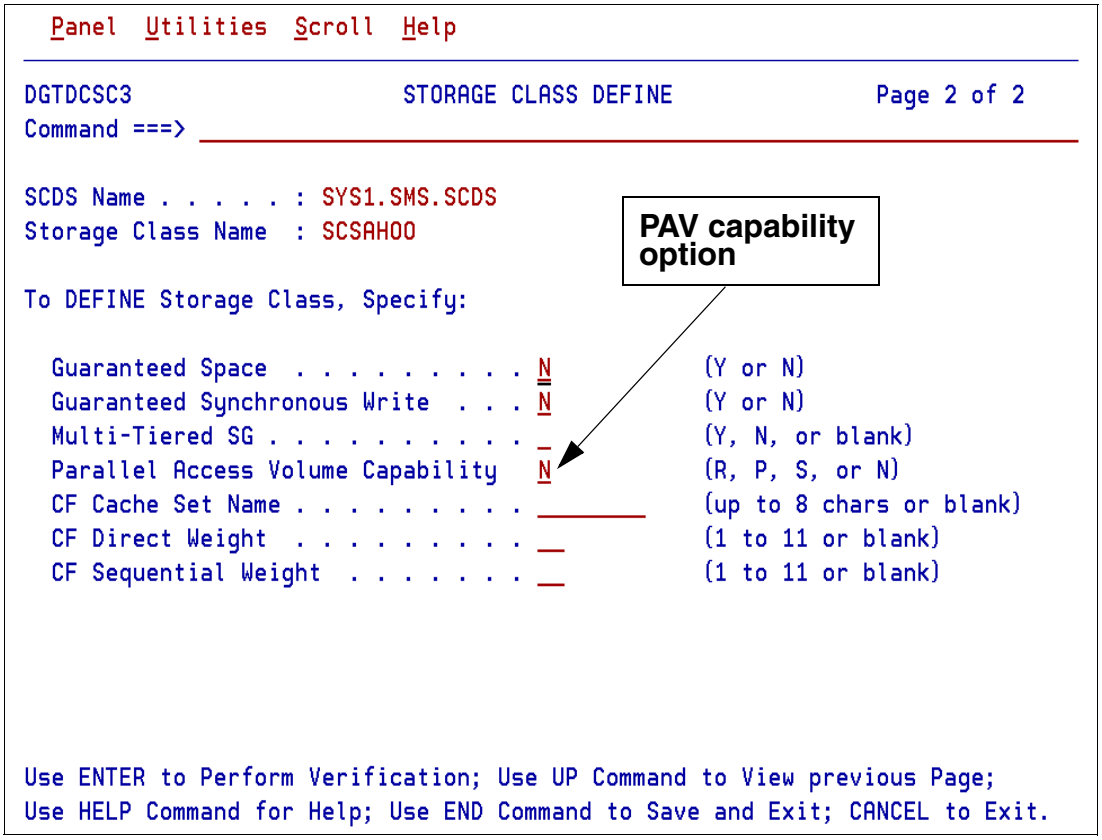

Figure 4-1 Storage class define panel showing PAV capability

The PAV capability field is specified to enable and facilitate volume selection algorithms. The possible values are:

R (REQUIRED) PAV capability is required. Only those volumes that support this capability will be eligible for the allocation request. All other volumes will be rejected from consideration.

P (PREFERRED) PAV capability is preferred. Volumes with this capability will be preferred over volumes that do not have this capability.

S (STANDARD) Volumes without PAV capability will be preferred over volumes that have this capability.

N (NOPREF) Volumes with or without PAV capability will be equally considered for volume selection. This is the default.

The volume selection preferences list has been updated to include PAV capability in the following order:

- VIO
- Data set separation
- Volume count
- High threshold
- SMS status
- END-OF-VOLUME extend
- Non-overflow
- IART (initial access response time)
- Snapshot
- Accessibility
- PAV capability
- Availability
- Extended format
- MSR (millisecond response)

ISMF support

The following ISMF functions have been updated to support the PAV capability:

- Storage class define/alter

  A sample panel is shown in Figure 4-1 on page 79.

- Storage class display

  A sample panel is shown in Figure 4-2 on page 81.

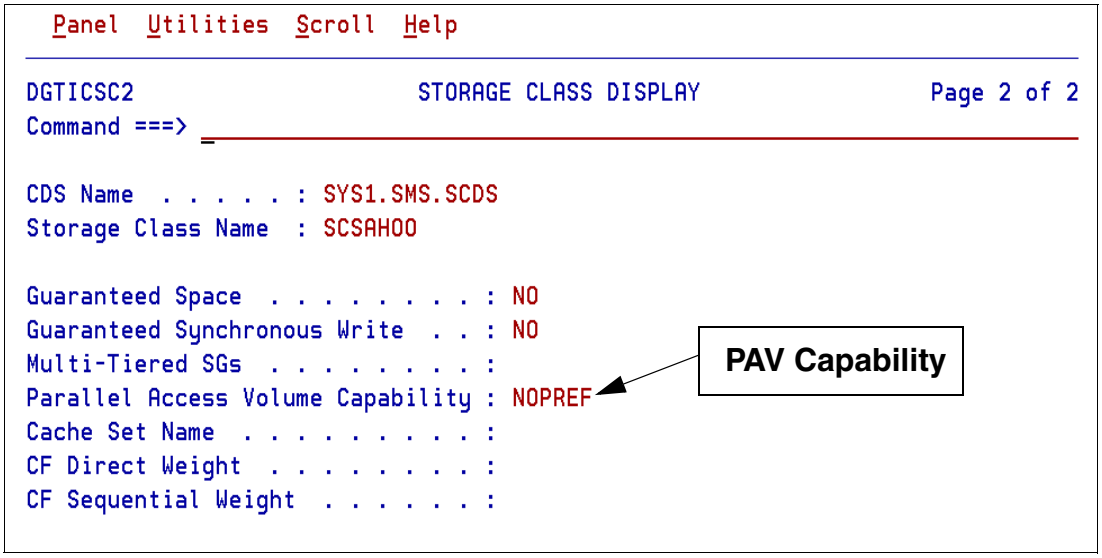

Figure 4-2 Storage class display panel showing PAV capability

- Storage class list

  A sample panel is shown in Figure 4-3.

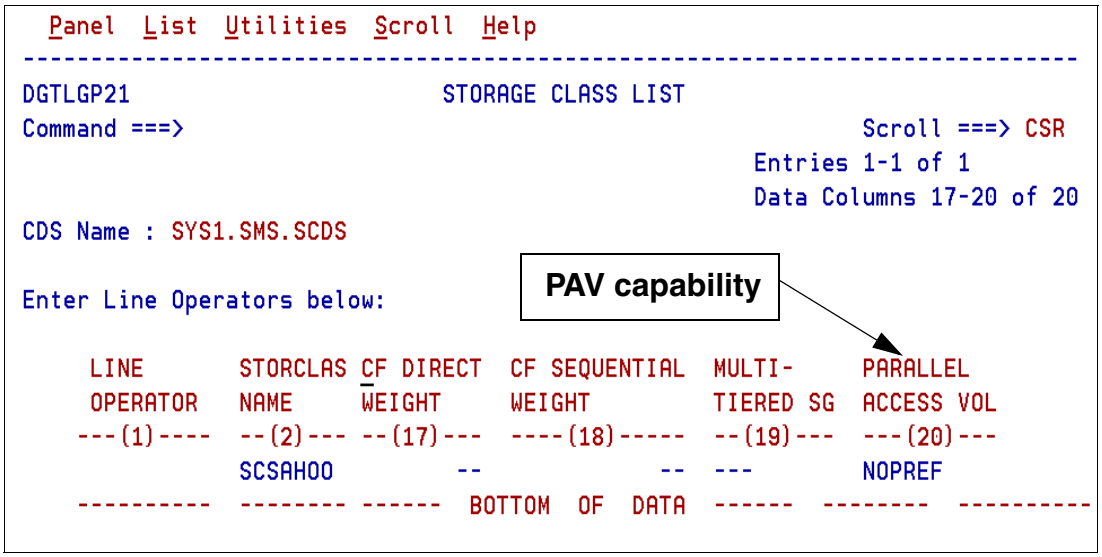

Figure 4-3 Storage class list panel showing PAV capability

- Storage class list print
- Storage class list sort
- Storage class list view

Naviquest support

A new NaviQuest field, PAVCAP(), has been included in sample JCL SYS1.SACBCNTL(ACBJBAS1) to enable users to define, alter, or display the storage class. This is shown in Figure 4-4 on page 82.

```
...
//*********************************************************************
//*                                                                   *
//* SAMPLE JCL TO DEFINE/ALTER/DISPLAY STORAGE CLASSES IN BATCH       *
//*                                                                   *
...
//*   PAVCAP   : Use the PARALLEL ACCESS VOLUME CAPABILITY field      *
//*              to modify the volume preferencing.                   *
//*                                                                   *
//*              Possible Values : R/P/S/N                            *
...
//SYSTSIN  DD *
...
MULTITSG() +
PAVCAP() +
CFCACSTN() +
...
```

Figure 4-4 Sample JCL SYS1.SACBCNTL(ACBJBAS1)

Migration/coexistence consideration

There are no specific migration and coexistence considerations for this enhancement. Installations should be aware that a storage class may be defined under z/OS V1R6 to specify PAV capability. If done, lower level systems, below z/OS V1R6, will ignore this field. There are no toleration PTFs required for this.

4.1.2 PDSE restartable address space

This enhancement is intended to improve PDSE reliability and availability by eliminating a need to re-IPL a system due to a hang, deadlock, or out-of-storage condition.

Introduction

In July 2002, APAR OW53245 provided the new function to combine the PDSE address spaces SMXC and SYSBMAS to a single address space called SMSPDSE. The introduction of the SMSPDSE address space improved overall PDSE usability and reliability by:

- Reducing excessive ECSA usage (by moving control blocks into the SMSPDSE address space)
- Reducing re-IPLs due to system hangs (in failure or CANCEL situations)
- Providing storage administrator's tools for monitoring and diagnosis (for example, determining which systems are using a particular PDSE)

To further improve reliability and availability, a PDSE address space restart capability has been provided in z/OS V1R6 DFSMS. The purpose of the restartable PDSE address space is to eliminate the need to re-IPL a system or systems due to a hang or deadlock condition, or an out-of-storage condition. Even though the SMSPDSE address space incorporates guaranteed recovery during a failure or cancel situation, a PDSE hang or deadlock is still possible. The restartable PDSE address space eliminates the need to re-IPL in order to recover from a PDSE error.

PDSE address spaces

With z/OS V1R6, DFSMSdfp provides a new restartable PDSE address space named SMSPDSE1. As a result, there will be two address spaces for processing PDSEs: SMSPDSE and SMSPDSE1. A z/OS system can have only the SMSPDSE address space, or both the

SMSPDSE and SMSPDSE1 address spaces. Some control blocks that are associated with reading, writing, and loading PDSE members are still located in the extended common service area (ECSA).

SMSPDSE

SMSPDSE is a non-restartable address space for PDSE data sets that are in the LNKLST concatenation. The linklist and other system functions use global connections. The SMSPDSE address space cannot be restarted because global connections cannot handle the interruption and reconnection that are part of an address space restart operation. SMSPDSE is the only PDSE address space for the z/OS system when one of the following conditions exists:

- The IGDSMSxx initialization parameter, PDSESHARING, is set to NORMAL.
- The IGDSMSxx initialization parameters in a sysplex coupled systems environment are set as follows:

  PDSESHARING(EXTENDED)

  PDSE_RESTARTABLE_AS(NO)

SMSPDSE1

This is the new restartable address space that provides connections to and processes requests for those PDSEs that are not part of the LNKLST concatenation. To create the SMSPDSE1 address space during IPL in a sysplex coupled systems environment, set the IGDSMSxx initialization parameters as follows:

PDSESHARING(EXTENDED)

PDSE_RESTARTABLE_AS(YES)

User programs will maintain the connection to the PDSE and its members during and after SMSPDSE1 restart. Also, the restart will not cause failure of a user job, TSO session, an edit or browse of a PDSE member, or LISTPDS of a PDSE.

SMS initialization parameters

This section provides a description of the changed and new SMS initialization parameters.

Enhanced parameters

The following existing IGDSMSxx parameters have been enhanced with synonym parameters for the global PDSE address space (SMSPDSE) initialization:

LRUCYCLES synonym PDSE_LRUCYCLES

LRUTIME synonym PDSE_LRUTIME

HSP_SIZE synonym PDSE_HSP_SIZE

BMFTIME synonym PDSE_BMFTIME

New parameters

The following new IGDSMSxx parameters have been added for the restartable PDSE address space initialization:

- PDSE_RESTARTABLE_AS(YES|NO)

  Specifies whether PDSE initialization during IPL NIP processing will bring up a second restartable PDSE address space. If specified as PDSE_RESTARTABLE_AS(YES), along with a specification of PDSESHARING(EXTENDED), allows PDSE initialization to create a second restartable PDSE address space, SMSPDSE1.

The PDSE_RESTARTABLE_AS specification is set in the IGDSMSxx member of SYS1.PARMLIB. This option cannot be modified via an operator command. Therefore, the PDSE_RESTARTABLE_AS option can only be changed with a system IPL.

Default: NO

- PDSE1_BMFTIME(nnn|3600)

  Specifies the number of seconds that SMS is to wait between recording SMF records for buffer manager facility (BMF) cache use for the SMSPDSE1 address space. You can specify a value from 1 to 86399 (23 hours, 59 minutes, 59 seconds).

  The SMF_TIME keyword, if set to YES, overrides the PDSE1_BMFTIME keyword.

  Default: 3600 (one hour)

- PDSE1_HSP_SIZE({nnn|256})

  On systems that have also specified PDSESHARING(EXTENDED) and RESTARTABLE_PDSE_AS(YES), this parameter specifies the size of the hiperspace that is used for the restartable SMSPDSE1 member caching.

  By default, the PDSE1 hiperspace uses either 256 MB of expanded storage, or in combination with the value specified in the PDSE1_HSP_SIZE parameter and half of the system's available expanded storage (whichever amount is lower).

  By default, on systems that are running in z-Architecture mode, the PDSE hiperspace uses either 256 MB of real storage, or one quarter of the available real storage (whichever amount is lower).

  Note: If the amount of available real storage is 64 MB or less, the amount of real storage used is limited to one eighth of the available real storage.

  The PDSE1_HSP_SIZE parameter can be used to request up to 512 MB for the PDSE1 hiperspace. Or you can indicate that the hiperspace is not to be created (by setting PDSE1_HSP_SIZE to 0).

  Note: Regardless of the value specified in the PDSE1_HSP_SIZE parameter, this parameter will have no effect if RESTARTABLE_PDSE_AS(NO) is either specified or allowed to default.

  If a valid value is specified for PDSE1_ HSP_SIZE, the system uses it to create the PDSE1 hiperspace at IPL time. The PDSE1_HSP_SIZE value remains in effect for the duration of the IPL. This value cannot be changed via an operator command.

  If not enough expanded storage is available to satisfy the PDSE_HSP_SIZE value and PDSE1_HSP_SIZE value, the system uses some portion of the available expanded storage (up to the full amount) for the PDSE hiperspace and the PDSE1 hiperspace, depending on the amount of caching activity in the system. The system will stop caching PDSE1 members if the available expanded storage becomes full.

  On systems not running in z/Architecture mode, if no expanded storage is online to the system, the hiperspace cannot be created.

  Important: Use the PDSE1_HSP_SIZE parameter with care. If the PDSE1_HSP_SIZE value is too low for normal PDSE1 hiperspace usage, non-global PDSE performance can be degraded. If the PDSE1_HSP_SIZE is too large, and there is contention for storage on the system, then the performance of other components or applications could be degraded.

To determine the current PDSE1_HSP_SIZE value of the PDSE1 hiperspace, use the DISPLAY SMS,OPTIONS command, or review the messages that are written to syslog when SMS is started.

To evaluate the effectiveness of a particular PDSE1_HSP_SIZE value, you can examine SMF type 42, subtype 1 records.

Default: 256M

- PDSE1_LRUCYCLES(nnn|240)

  Specifies the maximum number of times (5 to 240 cycles) that the buffer management facility (BMF) least recently used (LRU) routine will pass over inactive buffers before making them available for reuse. (While this parameter sets the maximum value, BMF dynamically changes the actual number of times it passes over inactive buffers.)

  PDSE1_LRUCYCLES is related to PDSE_LRUTIME. A change to the PDSE1_LRUCYCLES value introduced by this parameter takes effect on the next execution of the LRU routine. Most installations should use the default value. In some very high data rate situations, this value may be changed for tuning purposes. Monitor the SMF 42 type 1 record to determine the amount of caching activity in the BMF data space.

  Default: 240

- PDSE1_LRUTIME(nnn|15)

  Specifies the number of seconds (5 to 60) that the buffer management facility (BMF) will wait between calls to the BMF data space cache LRU routine. Most installations should use the default value. In some very high data rate situations this value may be changed for tuning purposes. Monitor the SMF type 1 record to determine the amount of caching activity in the BMF data space.

  Default: 15

- PDSE1_MONITOR({YES|NO}[,interval|60[,duration|15]])

  Specifies how the processing for the PDSE1 monitor should be started or modified. YES turns on monitor processing, NO turns off monitor processing. If this parameter is omitted and the restartable PDSE address space is created, the monitor is started with default values of 60 seconds for *interval* and 15 seconds for *duration*.

  interval specifies the number of seconds between successive scans of the monitor. If the PDSE1_MONITOR keyword is specified, but *interval* is omitted, the interval will be unchanged from its initial value, except at IPL time when the interval will be set to 60.

  duration specifies the number of seconds a possible error condition must exist before it is treated as an error. If the PDSE1_MONITOR keyword is specified, but *duration* is omitted, the duration will be unchanged from its initial value, except at IPL time when the interval will be set to 15.

  Default: YES

Planning and migration considerations

You should consider the following SW and HW requirements for implementing the PDSE restartable address space.

Software requirements: The restartable PDSE address space feature requires z/OS V1R6 and the supplied PTFs, if any.

Coexistence requirements: If one system in your sysplex has the restartable PDSE address space code installed, then all systems in the sysplex must have the restartable PDSE address space code installed, or the required PTFs. This step is required even if you do not attempt a restart on those systems.

Setting up the SMSPDSE1 address space

Before you start SMSPDSE1, the system already has the nonrestartable SMSPDSE address space. The restartable PDSE address space (SMSPDSE1) is optional and available only for systems that use PDSESHARING(EXTENDED). If you decide to start SMSPDSE1, there will be two PDSE address spaces in the system:

- The nonrestartable SMSPDSE address space is used for PDSEs that are contained in the LNKLIST.
- The restartable SMSPDSE1 address space is used for all other PDSEs in the system.

Perform the following steps to set up the SMSPDSE1 address space:

1. Specify PDSESHARING(EXTENDED) in the IGDSMSxx parmlib member.

   Example: PDSESHARING(EXTENDED)

2. Specify PDSE_RESTARTABLE_AS(YES) in the IGDSMSxx parmlib member.

   Example: PDSE_RESTARTABLE_AS(YES)

3. Optionally, specify the values for the following IGDSMSxx parmlib parameters to tune the SMSPDSE1 address space:

   PDSE1_LRUCYCLES

   PDSE1_LRUTIME

   PDSE1_HSP_SIZE

   PDSE1_BMFTIME

   PDSE1_MONITOR

   Example:

   PDSE_RESTARTABLE_AS(YES)

   PDSE1_MONITOR(YES)

   PDSE1_LRUCYCLES(200)

   PDSE1_LRUTIME(50)

   PDSE1_HSP_SIZE(256)

   PDSE1_BMFTIME(3600)

   This example brings up the SMSPDSE1 address space with the monitor turned on. The SMSPDSE1 address space uses a hiperspace of 256 MB for PDSE member caching. SMS waits 3600 seconds before recording SMF records for BMF caching for the SMSPDSE1 address space. The BMF waits 200 cycles before reusing inactive buffers and 50 seconds before calling the BMF data space cache. Monitor the SMF 42 type 1 record to determine the amount of caching activity in the BMF data space and tune the PDSE1 parameters.

4. Optionally, specify the values for the following IGDSMSxx parmlib parameters to tune the nonrestartable SMSPDSE address space:

   PDSE_LRUCYCLES

   PDSE_LRUTIME

   PDSE_HSP_SIZE

   PDSE_BMFTIME

   PDSE_MONITOR

Example:

PDSE_MONITOR(YES)

PDSE_LRUCYCLES(250)

PDSE_LRUTIME(15)

PDSE_HSP_SIZE(256)

PDSE_BMFTIME(3600)

5. IPL the z/OS V1R6 system to create the SMSPDSE and SMSPDSE1 address spaces. To verify that both the SMSPDSE and SMSPDSE1 address spaces exist after you IPL z/OS V1R6, issue the following commands:

   D A,SMSPDSE

   D A,SMSPDSE1

Figure 4-5 shows a sample of the IGDSMSxx parmlib member we used in our system to start the SMSPDSE1 address space.

```
SMS ACDS(SYS1.SMS.ACDS)
    COMMDS(SYS1.SMS.COMMDS)
    INTERVAL(15)
    DINTERVAL(150)
    ...
    ACSDEFAULTS(NO)
    PDSESHARING(EXTENDED)
    PDSE_RESTARTABLE_AS(YES)
    PDSE_MONITOR(YES)
    PDSE1_BMFTIME(3600)
    PDSE1_HSP_SIZE(256)
    PDSE1_LRUCYCLES(200)
    PDSE1_LRUTIME(50)
    PDSE1_MONITOR(YES)
    TRACE(ON)
    SIZE(128K)
    TYPE(ALL)
    JOBNAME(*)
    ...
```

Figure 4-5 Sample IGDSMSxx parmlib member

System IPL

When we IPL our system, we see two address spaces, SMSPDSE and SMSPDSE1. When we issue the D A command against the PDSE address spaces, we receive the message shown in Figure 4-6 on page 88.

```
IGW061I SMSPDSE1 INITIALIZATION COMPLETE.

D A,SMSPDSE
IEE115I 17.15.55 2004.098 ACTIVITY 153
 JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
00004    00025    00002      00031    00043    00002/00010       00012
 SMSPDSE  SMSPDSE           NSW  *   A=0008   PER=NO   SMC=000
                                     PGN=N/A  DMN=N/A  AFF=NONE
                                     CT=000.326S  ET=00.43.51
                                     WKL=SYSTEM   SCL=SYSTEM   P=1
                                     RGP=N/A      SRVR=NO  QSC=NO
                                     ADDR SPACE ASTE=03969200
                                     DSPNAME=SYSBMFDS ASTE=036DE800
                                     DSPNAME=SYSBMFHS ASTE=0200D000
D A,SMSPDSE1
IEE115I 17.18.00 2004.098 ACTIVITY 155
 JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
00004    00025    00002      00031    00043    00002/00010       00012
 SMSPDSE1 SMSPDSE1          NSW  *   A=0009   PER=NO   SMC=000
                                     PGN=N/A  DMN=N/A  AFF=NONE
                                     CT=000.260S  ET=00.45.56
                                     WKL=SYSTEM   SCL=SYSTEM   P=1
                                     RGP=N/A      SRVR=NO  QSC=NO
                                     ADDR SPACE ASTE=03969240
                                     DSPNAME=SYSBMFDS ASTE=036DE880
                                     DSPNAME=SYSBMFHS ASTE=0200D080
```

Figure 4-6 Sample messages in SYSLOG

Operator commands

SMS command processing has been modified to provide support for the new restartable PDSE1 address space.

Display SMS options

The following command is issued to display the SMS options:

```
D SMS,OPTIONS
```

This command is used to verify the SMS options set or defaulted for PDSE address spaces. Figure 4-7 on page 89 shows the command output in our system.

```
D SMS,OPTIONS
IGD002I 08:12:40 DISPLAY SMS 526
ACDS     = SYS1.SMS.ACDS
COMMDS   = SYS1.SMS.COMMDS
INTERVAL = 15           DINTERVAL = 150
SMF_TIME = YES          CACHETIME = 3600
CF_TIME = 1800          PDSE_RESTARTABLE_AS = YES
PDSE_BMFTIME = 3600     PDSE1_BMFTIME = 3600
PDSE_LRUTIME = 15       PDSE1_LRUTIME = 50
PDSE_LRUCYCLES = 240    PDSE1_LRUCYCLES = 200
LOCAL_DEADLOCK = 15     GLOBAL_DEADLOCK = 4
REVERIFY = NO           DSNTYPE = PDS
ACSDEFAULTS = NO        PDSESHARING = EXTENDED
OVRD_EXPDT = NO         SYSTEMS = 8
PDSE_HSP_SIZE = 256MB   PDSE1_HSP_SIZE   = 256MB
USE_RESOWNER = YES      RLS_MAX_POOL_SIZE = 100MB
RLSINIT = YES           RLSTMOUT = 0
CICSVR_INIT = YES       COMPRESS = GENERIC
CICSVR_DSNAME_PREFIX = DWWUSER.V3R1M0
Rls_MaxCfFeatureLevel = Z
PDSE_MONITOR = (YES,0,0)  PDSE1_MONITOR = (YES,0,0)
...
```

Figure 4-7 D SMS,OPTIONS command output

Restart the SMSPDSE1 address space

Once you initialize the restartable PDSE address space with parameters PDSESHARING(EXTENDED) and PDSE_RESTARTABLE_AS(YES), you can use operator commands to restart the SMSPDSE1 address space. This type of operator intervention occurs only in extreme failure situations; for example, when PDSEs are hung because a canceled task is holding a latch.

Before restarting the SMSPDSE address space, we strongly recommend to take the following actions:

- Issue the VARY SMS,PDSE1,ANALYSIS operator command to determine which PDSE is causing the hang and which latch is being used.
- Cancel the related task to attempt to free the latch that is used.
- If the system is still hung, issue the VARY SMS,PDSE1,FREELATCH command to force the release of the latch.

If the system is still hung, then restart the SMSPDSE1 address space using this command:

Vary SMS,PDSE1,RESTART[,QUIESCE(duration|3),COMMONPOOLS(NEW|REUSE)]

This command recycles the restartable PDSE address space (SMSPDSE1); it is valid if the system was IPLed with a restartable PDSE address space. The operands are as follows:

Restriction: PDSERESTART(YES) and PDSESHARING(EXTENDED) must be up and running to use this command.

PDSE1 Allows you to select the restartable SMSPDSE1 address space.

RESTART Use this operand to terminate the SMSPDSE1 address space and immediately activate a new instance of the SMSPDSE1 address space.

QUIESCE Specifies the maximum time interval, in seconds, that existing in-flight operations quiesce before the current instance of the SMSPDSE1 address space is terminated. This is an attempt to get all current work out of the address space before SMSPDSE1 stops. Also, during this interval, new requests coming to the SMSPDSE1 address space are held until a new instance of the SMSPDSE1 address space completes initialization.

Attention: If the interval chosen is too long and SMSPDSE1 address requests do not complete, the address space restart is delayed. This delay affects processing on this system, and can also affect PDSE processing on other systems in the sysplex, because this system holds the serialization on PDSEs until the address space terminates. For this reason, caution should be used when selecting a quiesce interval that is too long in duration.

The Quiesce duration can be a number from 0 to 900 seconds. The default, if not specified, is 3 seconds. Quiesce applies to the system where the command is issued.

COMMONPOOLS(*NEW*)
SMSPDSE1 abandons the old common storage cell pools and creates a new set of cell pools in ECSA.

COMMONPOOLS(*REUSE*)
This is the default. SMSPDSE1 reuses the existing common storage cell pools that were created by the previous instance of SMSPDSE1.

Attention: Part of SMSPDSE1 startup processing creates storage cell pools in ECSA. When SMSPDSE1 restarts, the old storage cell pools are not deleted. Normally, it is preferable to reuse the existing ECSA cell pools in order to avoid ECSA depletion. If the reason for the SMSPDSE1 restart is related to a problem with the existing ECSA cell pools, then it is preferable to create new ones by specifying COMMONPOOLS(NEW).

Figure 4-8 on page 91 shows the V SMS,PDSE,RESTART command outputs when issued in our system.

```
V SMS,PDSE1,RESTART

IGWO36I VARY SMS,PDSE1,RESTART COMMAND ACCEPTED.
IGWO57I WAITING FOR SMSPDSE1 SHUTDOWN.
IGWO55I SMSPDSE1 SHUTDOWN IN PROGRESS.
IGW999I XQUIESCE Started
IGWO62I SMSPDSE1 IS QUIESCING.
IGWO65I SMSPDSE1 QUIESCE COMPLETE.
IGWO58I SMSPDSE1 SHUTDOWN COMPLETE.
IGWO59I SMSPDSE1 IS BEING ACTIVATED.
IGWO40I PDSE IGWLGEDC Connected
IGWO40I PDSE Connecting to XCF for Signaling
IGWO40I PDSE Connected to XCF for Signaling
IGWO40I PDSE Posting initialization
IGWO43I PDSE MONITOR IS ACTIVE 571
++ INVOCATION INTERVAL:60 SECONDS
++ SAMPLE DURATION:15 SECONDS
IGWO61I SMSPDSE1 INITIALIZATION COMPLETE.
IGWO66I SMSPDSE1 IS RECONNECTING ALL USERS.
IGWO69I SMSPDSE1 RECONNECT PHASE COMPLETE.
IGWO70I SMSPDSE1 WILL RESUME ALL USER TASKS.
IGW999I Reconnect Completed Normally
IGW999I Reconnect Completed Normally
IGW999I XQUIESCE Stopping
IGW999I Reconnect Completed Normally
IGW999I Reconnect Completed Normally
```

Figure 4-8 V SMS,PDSE,RESTART command output

Activate the SMSPDSE1 address space

Activate the PDSE1 address space with the following command (valid after the operator forces the SMSPDSE1 address space from the system):

VARY SMS,PDSE1,ACTIVATE[,COMMONPOOLS(NEW|REUSE)]

where:

PDSE1 This allows the operator to select the restartable SMSPDSE1 address space.

ACTIVATE This causes the SMSPDSE1 address space to start up after a prior instance of the SMSPDSE1 address space has been terminated.

COMMONPOOLS(*NEW*) A specification of NEW will cause SMSPDSE1 to abandon the old common storage cell pools and to create a new set of cell pools in ECSA.

COMMONPOOLS(*REUSE*) This is the default. A specification of REUSE will cause SMSPDSE1 to reuse the existing common storage cell pools that were created by the previous instance of SMSPDSE1.

Attention: Part of SMSPDSE1 startup processing creates storage cell pools in ECSA. When SMSPDSE1 restarts the old storage cell pools are not deleted. Normally, it is preferable to reuse the existing ECSA cell pools in order to avoid ECSA depletion. If the reason for the SMSPDSE1 restart is related to a problem with the existing ECSA cell pools, then it is preferable to create new ones by specifying COMMONPOOLS(NEW).

Figure 4-9 shows the messages in our system, when the SMSPDSE1 address space is activated after forcing.

```
V SMS,PDSE1,ACTIVATE

IGW038I VARY SMS,PDSE1,ACTIVATE COMMAND ACCEPTED.
IGW059I SMSPDSE1 IS BEING ACTIVATED.
IGW040I PDSE IGWLGEDC Connected
IGW040I PDSE Connecting to XCF for Signaling
IGW040I PDSE Connected to XCF for Signaling
IGW040I PDSE Posting initialization
IGW043I PDSE MONITOR IS ACTIVE 700
++ INVOCATION INTERVAL:60 SECONDS
++ SAMPLE DURATION:15 SECONDS
IGW061I SMSPDSE1 INITIALIZATION COMPLETE.
IGW066I SMSPDSE1 IS RECONNECTING ALL USERS.
IGW069I SMSPDSE1 RECONNECT PHASE COMPLETE.
IGW070I SMSPDSE1 WILL RESUME ALL USER TASKS.
IGW999I XQUIESCE Stopping
```

Figure 4-9 V SMS,PDSE1,ACTIVATE command output

Analyze PDSE address space

The ANALYSIS command detects a number of the most common problems that result in PDSE or PDSE1 breakage. The analysis is performed for a single system. If information for more than one system is required, the ROUTE command should be used.

Issue the ANALYSIS command only when you suspect that one or more users or jobs are having problems accessing a PDSE or PDSE1. This command and the FREELATCH command use a sampling algorithm that interrogates the state of the PDSE or PDSE1 every hundredth of a second for the number of retries that the user specifies (the default is 1500 retries, which is approximately 15 seconds). Errors are reported only if the state of the PDSE or PDSE1 does not change. The command syntax is:

```
VARY SMS,PDSE | PDSE1,
ANALYSIS[,DSNAME(dsname)[,VOLSER(volser)]][,RETRIES(retries|1500)]
```

where:

dsname When specified, causes the analysis to be performed for a particular PDSE or PDSE1. If the volser is omitted, the data set is found using the default system catalog.

volser Allows you to specify an uncataloged PDSE or PDSE1.

retries Allows you to control the amount of time for which the particular PDSE or PDSE1 situation must remain static. The PDSE or PDSE1 control blocks are examined every hundredth of a second for the number of retries specified. By default, the data set is examined 1500 times or for approximately 15 seconds

before reporting any exceptional conditions. If no exceptional conditions are found, the command returns immediately after the first examination of the control blocks.

Figure 4-10 shows the messages received when the ANALYSIS command is issued in no exception conditions.

```
V SMS,PDSE1,ANALYSIS
IGW031I PDSE ANALYSIS  Start of Report(SMSPDSE1) 116
++ no exceptional data set conditions detected
PDSE ANALYSIS  End of Report(SMSPDSE1)
IGWLHA10:20607200
```

Figure 4-10 V SMS,PDSE1,ANALYSIS command output

Release a latch

The VARY SMS,PDSE|PDSE1,FREELATCH command releases any latch that the ANALYSIS command indicates is held. If this command is used to release a latch held by a process that is still running, it could result in the breakage of the PDSE or PDSE1. The latch is not released unless it is held by the ASID and tcbaddr, indicated in the command. The latch is released only if it is held by the same user for each of the retries. The command syntax is:

VARY SMS,PDSE | PDSE1, FREELATCH(latchaddr,ASID,tcbaddr)[,Retries(retries|1500)]

where:

latchaddr Address of the latch that is to be released.

ASID ASID of the holder of the latch.

tcbaddr Address of TCB that holds the latch. When an SRB holds the latch, the address for the TCB actually points to a control block that represents the active SRB. The value will be above the 16 MB line.

retries Allows you to control the amount of time for which the particular PDSE or PDSE1 situation must remain static. The PDSE or PDSE1 control blocks are examined every hundredth of a second. By default, the data set is examined 1500 times or for approximately 15 seconds, before the latch is released.

Figure 4-11 shows an example of the ANALYSIS command output in an exception condition.

```
V SMS,PDSE,ANALYSIS
IGW031I PDSE ANALYSIS  Start of Report 634
---------data set name------------------ -----vsgt-------
SAHU.MNS.LOD.CMPRM.GN991                 01-STM166-000104
++ Unable to latch DIB:19CF6D40
   Latch:19CF6D50 Holder(012E:009FD078)
   Holding Job:AHUSEPOD
PDSE ANALYSIS  End of Report
```

Figure 4-11 V SMS,PDSE,ANALYSIS output in an exception condition

In this example, the latch is 19CF6D50, ASID is 012E and tcbaddr is 009FD078. You can issue the command V SMS,PDSE,FREELATCH(19CF6D50,012E,009FD078) to free the latch. This is shown in Figure 4-12.

```
V SMS,PDSE,FREELATCH(19CF6D50,012E,009FD078)
IGW032I PDSE FREELATCH  Start of Report 232
++ latch:19CF6D50 released
PDSE FREELATCH  End of Report
```

Figure 4-12 V SMS,PDSE,FREELATCH command output

Monitor SMSPDSE1 address space

The PDSE MONITOR command has been enhanced with additional operands to provide information about the restartable SMSPDSE1 address space. The command syntax is:

Vary SMS[,PDSE|PDSE1],MONITOR[,ON|OFF|RESTART][,interval,duration]

where:

PDSE | PDSE1 This allows the operator to select either the non-restartable SMSPDSE address space by specifying PDSE, or the restartable SMSPDSE1 address space by specifying PDSE1.

ON | OFF | RESTART ON allows the MONITOR to be started. OFF allows the MONITOR to be stopped. RESTART allows the monitor to be stopped and restarted.

interval This specifies the number of seconds between successive scans of the monitor. If not specified, it defaults to the value specified in the IGDSMSxx parmlib member. If it is not specified there, it defaults to 60 seconds.

duration This specifies the number of seconds for which a possible error condition must exist before it is treated as an error. If not specified, it defaults to the value specified in the IGDSMSxx parmlib member. If it is not specified there, it defaults to 15 seconds.

Note: The *interval* and *duration* parameters can only be specified with the ON option, but not with the OFF and RESTART options.

Figure 4-13 shows the MONITOR command output when issued in our system.

```
V SMS,PDSE1,MONITOR,RESTART
IGW043I PDSE MONITOR IS ACTIVE 137
++ INVOCATION INTERVAL:60 SECONDS
++ SAMPLE DURATION:15 SECONDS

V SMS,PDSE1,MONITOR,OFF
IGW043I PDSE MONITOR IS INACTIVE

V SMS,PDSE1,MONITOR,ON,100,30
IGW043I PDSE MONITOR IS ACTIVE 141
++ INVOCATION INTERVAL:100 SECONDS
++ SAMPLE DURATION:30 SECONDS
```

Figure 4-13 V SMS,PDSE1,MONITOR command output

Considerations for restarting the SMSPDSE1 address space

When you face a problem with a system that is not running correctly due to a problem in PDSE processing, you may decide to restart the SMSPDSE1 address space. Before you

attempt to terminate the SMSPDSE1 address space, it is necessary to understand the effect this action will have on the system. This could result in failures of currently running jobs and TSO sessions that are accessing PDSEs. Some jobs won't survive because not all connects are restartable. The following restrictions and limitations still apply:

- The PDSE1 address space restart might not work correctly if more than one SMSPDSE1 address space restart is attempted concurrently. If a SMSPDSE1 restart is performed on more than one system at the same time, then these restarting systems will not be able to benefit from the other systems' xQUIESCE time duration interval.
- If a member is deleted while SMSPDSE1 is down and SMSPDSE1 remains down beyond the quiesce time interval, then a reconnect to that member may fail.
- If a member is deleted while SMSPDSE1 is down, then a reconnect to that member will fail if the quiesce doesn't work, or the SMSPDSE1 address space is forced.
- If callers that are not using the standard PDSE interface are connected, then PDSE will not know that the connection is active. At this time there are no known callers of PDSE that fit this criteria.
- If a user on another system opens for update when the restartable connection had a data set open for read/write, the restart connection will fail.
- Dump and Restore jobs for PDSEs will fail.
- If the PDSE address space terminates as a result of a hard failure or because of the FORCE command, then the results of a PDSE address space restart are unpredictable.
- If all the systems in the PDSE extended sharing sysplex do not have the support for the restartable PDSE address space, then restart results are unpredictable. The support for the PDSE restartable address space should be installed on all systems within the sysplex before activating the SMSPDSE1 address space on any system.
- If one system in a sysplex has the PDSE restartable address space code installed, then all the systems in the sysplex must have the PDSE restartable address space code, or toleration PTFs, installed—even if a restart is not attempted on those systems.
- If the user address space fails to reconnect after the SMSPDSE1 address space restarts, then the user address space should be forced off the system.
- SMF I/O counts can become inconsistent because of an SMSPDSE1 restart operation. The restart of the SMSPDSE1 address space could result in either a loss of some SMF I/O counts, or duplicate counts.
- If the operator has changed the PDSE MONITOR values through the use of the V SMS,PDSE1,MONITOR command, and the SMSPDSE1 address space is restarted, then the values provided either by system defaults or by the MONITOR values specified in the IGDSMSxx member of SYS1.PARMLIB will be used to restart the PDSE1 MONITOR.

PDSE effects during SMSPDSE1 restart

When the operator enters the VARY SMS,PDSE1,RESTART,QUIESCE(x) command, there will be a quiesce time interval as specified in the command, or defaulted if the quiesce parameter is not specified. The quiesce time interval applies to the system where the command is issued. This operand specifies the maximum interval of time in seconds that will allow existing in-flight operations to quiesce before the current instance of the SMSPDSE1 address space is terminated. This is an attempt to get all current work out of the address space before SMSPDSE1 is terminated. During this interval, new requests coming to the SMSPDSE1 address space are held until the SMSPDSE1 address space restarts.

If the interval chosen is too long and there are requests to the SMSPDSE1 address space which do not complete, the restart of the address space is delayed. This not only affects processing on this system, but can affect PDSE processing on other systems in the sysplex,

because this system holds the serialization on PDSEs until the address space terminates. For this reason, caution should be used when selecting a quiesce interval that is too long in duration. The quiesce duration can be a number from 0 to 900 seconds. The default, if not specified, is 3 seconds.

In addition to the quiesce on the system that is doing the SMSPDSE1 address space restart, there will be a partial quiesce of some shared PDSE function on other systems that are participating in PDSESHARING(EXTENDED) within the sysplex. This partial quiesce is referred to as xQuiesce. XQuiesce is a state that exists from the time the SMSPDSE1 address space is restarted and lasts until SMSPDSE1 completes reconnecting the PDSEs. The purpose of xQuiesce is to prevent certain new PDSE operations on the other systems while the SMSPDSE1 address space is being terminated and restarted. With xQuiesce, there should be no post restart failures caused by members deleted on other systems nor Opens for Updates on other systems. If the SMSPDSE1 address space does not successfully start up, then the xQuiesce duration is infinite.

New and changed messages

To support the restartable PDSE address space, some messages have been changed and some new messages have been added.

The message IGW031I has been changed. Figure 4-14 shows the messages when the PDSE ANALYSIS command is issued for PDSE and PDSE1.

```
V SMS,PDSE,ANALYSIS

IGW031I PDSE ANALYSIS  Start of Report(SMSPDSE ) 171
++ no exceptional data set conditions detected
PDSE ANALYSIS  End of Report(SMSPDSE )
IGWLHA10:20607200
V SMS,PDSE1,ANALYSIS
IGW031I PDSE ANALYSIS  Start of Report(SMSPDSE1) 174
++ no exceptional data set conditions detected
PDSE ANALYSIS  End of Report(SMSPDSE1)
IGWLHA10:20607200
```

Figure 4-14 Message IGW031 displayed for PDSE and PDSE1

The following new messages have been added.

- IGW035I SMSPDSE1 IS NOT ENABLED
- IGW036I VARY SMS,PDSE1,RESTART COMMAND ACCEPTED
- IGW044I <SMSPDSE|SMSPDSE1> BMF LRU FAILED. RC:xxxxxxxx RSN:xxxxxxxx
- IGW055I SMSPDSE1 SHUTDOWN IN PROGRESS
- IGW056S SMSPDSE1 SHUTDOWN FAILED, RSN=nnnnnnnn
- IGW057I WAITING FOR SMSPDSE1 SHUTDOWN
- IGW058I SMSPDSE1 SHUTDOWN COMPLETE
- IGW059I SMSPDSE1 IS BEING ACTIVATED
- IGW061I [SMSPDSE | SMSPDSE1] INITIALIZATION COMPLETE
- IGW062I SMSPDSE1 IS QUIESING
- IGW063S SMSPDSE1 IGNORING MUST-COMPLETE TASK ASID:JOBNAME, TCB=nnnnnnnnX
- IGW064I SMSPDSE1 IGNORING IN-PROGRESS TASK ASID:JOBNAME, TCB=nnnnnnnnX
- IGW065I SMSPDSE1 QUIESCE COMPLETE

- IGW066I SMSPDSE1 IS RECONNECTING ALL USERS
- IGW067I SMSPDSE1 RECONNECT TIMEOUT FOR ASID:JOBNAME
- IGW068D SMSPDSE1 IGNORE RECONNECT TIMEOUT(S)? (Y/N)
- IGW069I SMSPDSE1 RECONNECT PHASE COMPLETE
- IGW070I SMSPDSE1 IS ATTEMPTING TO RESUME ALL USER TASKS
- IGW071I SMSPDSE1 IS NOT ACTIVE
- IGW072S CREATION OF SMSPDSE1 FAILED DUE TO STORAGE SHORTAGE
- IGW073S CREATION OF SMSPDSE1 FAILED. MAXUSER EXCEEDED
- IGW074D SMSPDSE1 RETRY QUIESCE? (Y/N)
- IGW075S SMSPDSE1 ADDRESS SPACE LIST HELD BY asid:jobname,TCB=nnnnnnnnX
- IGW076I SMSPDSE1 TASK LIST FOR asid:jobname HELD BY TCB=nnnnnnnnX

4.1.3 Multilevel security (MLS) SECLABEL in ACS routines

The security label is a name used to represent the association between a particular security level and a set of security categories. It indicates the minimum level of security required to access a data set protected by this profile. With z/OS V1R6 DFSMS, you can use a new automatic class selection (ACS) routine read-only security label variable, &SECLABL, as input to the ACS routine. You can set this variable by entering it in the RACF profile of the data set. Or, you can specify the DD SECMODEL or DD PROTECT parameter in the JCL or dynamic allocation. This new ACS read-only variable helps to make allocation decisions rather than using an allocation exit, as done before.

&SECLABL variable

The &SECLABL variable specifies the default security label in the RACF profile of the user or data set if the SECLABEL class is active. Otherwise, the read-only variable contains a null value.

The &SECLABL variable is set from:

- User's profile (discrete profile)
- Data set profile (generic profile)
- ACEE pointing to a discrete profile if DD SECMODEL or PROTECT=YES parameter is specified in JCL or dynamic allocation

Restrictions:

- &SECLABL is set to "null" if the resource class SECLABEL is not active.
- If you define overflow or extended storage groups, ensure that security levels do not conflict.
- You cannot use SECLABEL in ACS routines if you are using automatic data set protection (ADSP). Issue SETROPTS NOADSP to disable ADSP.

Installing MLS labels in ACS routines

You may perform the following steps to implement multilevel security labels in ACS routines:

1. Activate the SECLABEL resource class and define profiles. Following is an example of defining SECLABEL EAGLE and activating the SECLABEL resource class.

Example:

RDEFINE SECDATA SECLEVEL UACC(NONE)

RALTER SECDATA SECLEVEL ADDMEM(CONFIDENTIAL/20, GENERAL/10))

RDEFINE SECDATA CATEGORY UACC(NONE)

RALTER SECDATA CATEGORY ADDMEM(TEAMA, TEAMB, TEAMC)

RDEFINE SECLABEL EAGLE SECLEVEL(GENERAL) ADDCATEGORY(TEAMA)

SETROPTS CLASSACT(SECLABEL) RACLIST(SECLABEL)

2. Create the security label by setting the SECLABEL variable in the RACF data set profile or user's profile.

   Example:

   ALTUSER USER05 SECLABEL(SPARROW)

   PERMIT EAGLE CLASS(SECLABEL) ACCESS(READ) ID(SAHOO)

   ADDSD 'SAHOO.**' UACC(NONE) SECLABEL(SYSLOW)

3. Specify the DD SECMODEL or DD PROTECT=YES parameter in the JCL or dynamic allocation, which creates a discrete profile for the data set; the security label is extracted from this profile. Otherwise, the security label is extracted from the generic data set profile. Figure 4-15 shows JCL examples of using SECMODEL and PROTECT parameters.

► Example using the SECMODEL parameter:

```
//STEP20 EXEC PGM=IEFBR14
//SYSPRINT DD SYSOUT=A
//DD3 DD DSN=USER#1.S16SL001.DATASET3,
//    DISP=(NEW,CATLG),SPACE=(TRK,(2,5)),
//    STORCLAS=S1PO1SO1,UNIT=3390,
//    SECMODEL=USER#1.S16SL001.MODEL.DATASET
The above example specifies the DD SECMODEL parameter in JCL to extract a security label
from the discrete profile.
```

► Example using the PROTECT parameter:

```
//STEP16 EXEC PGM=IEFBR14
//SYSPRINT DD SYSOUT=A
//DD1 DD DSN=USER#1.S16SL002.DATASET1,
//    DISP=(NEW,CATLG),SPACE=(TRK,(2,5)),
//    STORCLAS=S1PO1SO1,UNIT-3390,PROTECT=YES
The above example specifies the DD PROTECT=YES parameter in JCL to extract a security
label from the discrete profile.
```

Figure 4-15 JCL examples showing use of SECMODEL and PROTECT parameters

4. Update the storage group ACS routine with the &SECLABL read-only variable.

   Figure 4-16 on page 99 shows an example of an ACS routine using &SECLABL. This example assumes that a RACF security label ALERT is already defined to the system.

```
PROC &STORGRP

SELECT

WHEN (&SECLABL = 'ALERT')
   DO
      SET &STORGRP = 'S1P01'
      WRITE 'ASSIGN DATA SETS WITH SECLABEL ALERT TO STORAGE GROUP:
      S1P01'
      EXIT CODE(0)
   END

OTHERWISE
   DO
      SET &STORGRP = 'S1P02'
      WRITE 'ASSIGN DATA SETS WITHOUT SECLABEL ALERT TO STORAGE
      GROUP: S1P02'
      EXIT CODE(0)
   END

END

END
```

Figure 4-16 ACS routine using the &SECLABL variable

5. Use the ISMF ACS test sase define/alter application to test the security labels in the storage group ACS routines.
6. Validate and activate the SCDS.

ISMF support

The ACS test application has been updated to support all specifications of SECLABEL. Figure 4-17 on page 100 shows the SECLABEL field added in the ACS TEST CASE DEFINE/ALTER panel (ISMF option 7.4.1 and 7.4.2).

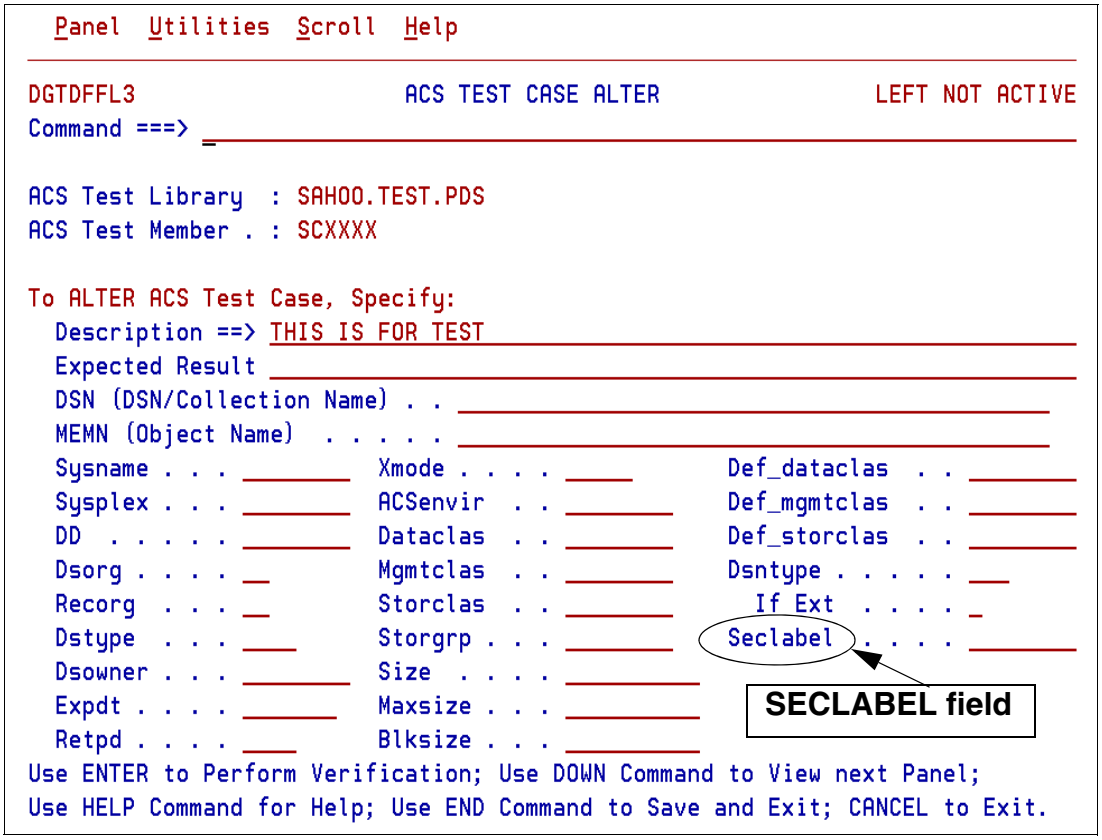

Figure 4-17 ISMF: ACS test case alter panel

NaviQuest support

The test case generation from ISMF saved lists now contains a new SECLABEL field. Figure 4-18 on page 101 shows the new SECLABEL field (ISMF option 11.1.1).

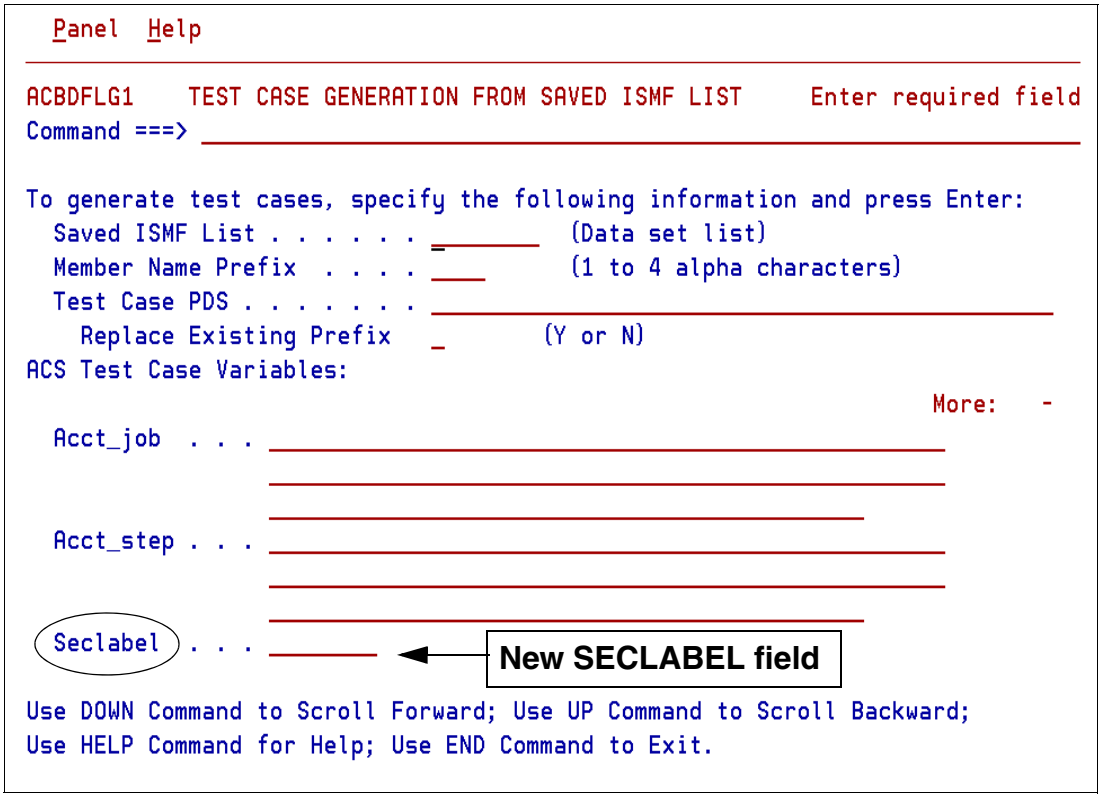

Figure 4-18 New SECLABEL field in ISMF option 11.1.1 panel

Migration and coexistence considerations

- ACS routines using the &SECLABEL read-only variable cannot be translated on systems earlier than z/OS V1R6.
- No specific coexistence considerations pertain except that on lower level systems the &SECLABEL has a “null” value.

4.1.4 Miscellaneous changes

There are some miscellaneous changes to the DFSMSdfp component.

Catalog serviceability

Some new parameters have been added to the F CATALOG command.

TAKEDUMP This parameter causes the Catalog Address SPACE (CAS) to issue an SVCDUMP using the proper options to ensure that all data needed for diagnosis is available.

RESTART This parameter prompts the user for additional information with the following messages:

`IEC363D IS THIS RESTART RELATED TO AN EXISTING CATALOG PROBLEM (Y OR N)?`

If the response to message IEC363D is `N`, the restart continues; if the response is `Y`, another prompt is issued.

`IEC364D HAS AN SVC DUMP OF THE CATALOG ADDRESS SPACE ALREADY BEEN TAKEN (Y OR N)?`

If the response to message IEC364D is N, an SVC dump is taken before restart; if the response is Y, the restart continues.

4.2 DFSMSdss enhancements

DFSMSdss provides various functions such as data movement and replication, space management in the DASD, data backup and recovery, and data set and volume conversion. The following enhancements have been implemented to DFSMSdss components.

4.2.1 REPLACEUnconditional keyword

Currently, we can rename a data set by specifying source data sets and corresponding new names, or rename based upon high-level qualifier (HLQ), during the COPY DATASET or RESTORE DATASET operation.

- RENAMEUnconditional for COPY processing
- RENAME or RENAMEUnconditional for RESTORE processing

During the COPY or RESTORE operation, if a data set exists with the same name as the renamed data set on the target volume or in the standard order of search and is SMS-managed, the operation will fail even with the REPLACE operation. This is because RENAME or RENAMEU take precedence over REPLACE.

To address this problem, a new keyword, REPLACEUnconditional, has been added. REPLACEUnconditional specifies that DFSMSdss is to search the target volumes for usable preallocated data sets. If one is found, it is replaced with the data set from the source volume. When used with the RENAME or RENAMEUnconditional keywords, usable preallocated data sets with the new name are replaced. When used without the RENAME or RENAMEUnconditional keywords, usable preallocated data sets with the same name as the source data set are replaced. If no preallocated target is found, DFSMSdss attempts to allocate a data set. The REPLACE and REPLACEUnconditional keywords cannot be specified together.

Note: This new function has been shipped with APARs OA05249 and OA05874 for DFSMSdss R1F0, R1G0, and R1H0.

COPY or RESTORE to preallocated target data sets

In order to use a preallocated data set, the REPLACE or REPLACEUNCONDITIONAL keyword must be specified. If the REPLACE keyword is specified, the preallocated target data set name must be identical to the source data set name. If the REPLACEUNCONDITIONAL keyword is specified and the RENAME or RENAMEUNCONDITIONAL keyword is also specified, the preallocated target data set name must match the new name filter criteria.

If a target data set is preallocated, it is scratched and reallocated if it is not large enough to contain the dumped data set. VSAM preallocated target data sets are also scratched and reallocated when:

- Any of the following source and target data set attributes do not match:
  - CI size
  - Record length
  - IMBED (only KSDS and key range data sets)
  - Key length (only KSDS and key range data sets)

  - REPLICATE (only KSDS and key range data sets)
  - SPANNED
- The preallocated target is multivolume and the space of the target data set on the first volume is not large enough to contain all of the dumped data.
- The data set was not defined as reusable and the high-used relative byte address (RBA) of a target VSAM KSDS is not 0. (In a reusable data set, you can reset this high-used RBA field to zero at OPEN by specifying MACRF=RST in the ACB at OPEN. VSAM can use this reusable data set like a newly defined data set.)

Figure 4-19 shows an example of the COPY operation with RENAMEU and REPLACE. The COPY operation failed with message ADR472E, RC8 indicating a duplicate data set in the target volume.

```
COPY DS(INCLUDE(ITSO.DEV1.ARCHDEF)) -
INDDNAME(DASD1) -
OUTDDNAME(DASD2) -
REPLACE -
TOL(ENQF) -
RENAMEU(SAHOO)
...
TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'COPY '
2004.103 15:36:19 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED.
RACF LOGGING OPTION IN EFFECT FOR THIS TASK
2004.103 15:36:19 EXECUTION BEGINS
ADR472E UNABLE TO SELECT A TARGET VOLUME FOR DATA SET ITSO.DEV1.ARCHDEF, 08
DATA SET FILTERING IS COMPLETE. 1 OF 1 DATA SETS WERE SELECTED: 0 FAILED SERIALIZATION
       AND 0 FAILED FOR OTHER REASONS.
THE FOLLOWING DATA SETS WERE NOT SUCCESSFULLY PROCESSED
 ITSO.DEV1.ARCHDEF
2004.103 15:36:22 EXECUTION ENDS
2004.103 15:36:22 TASK COMPLETED WITH RETURN CODE 0008
2004.103 15:36:22 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0008 FROM
       TASK    001
```

Figure 4-19 DFDSS COPY with REPLACE

Figure 4-20 on page 104 shows the same example for a COPY operation with RENAMEU and REPLACEU. Now, the COPY operation is successful and the renamed data set replaces the preallocated data set in the target volume.

```
COPY DS(INCLUDE(ITSO.DEV1.ARCHDEF)) -
INDDNAME(DASD1) -
OUTDDNAME(DASD2) -
REPLACEU -
TOL(ENQF) -
RENAMEU(SAHOO)
...
TASKID 001 HAS BEEN ASSIGNED TO COMMAND 'COPY '
2004.103 15:39:10 INITIAL SCAN OF USER CONTROL STATEMENTS COMPLETED.
RACF LOGGING OPTION IN EFFECT FOR THIS TASK
2004.103 15:39:10 EXECUTION BEGINS
DATA SET SAHOO.DEV1.ARCHDEF WILL BE SCRATCHED FROM SBOXB6 BECAUSE OF UNMATCHED SIZE. IT
       WILL BE REALLOCATED
DATA SET SAHOO.DEV1.ARCHDEF HAS BEEN DELETED
DATA SET ITSO.DEV1.ARCHDEF ALLOCATED WITH NEWNAME SAHOO.DEV1.ARCHDEF, ON VOLUME(S):
       SBOXB6
DATA SET FILTERING IS COMPLETE. 1 OF 1 DATA SETS WERE SELECTED: 0 FAILED SERIALIZATION
       AND 0 FAILED FOR OTHER REASONS.
THE FOLLOWING DATA SETS WERE SUCCESSFULLY PROCESSED
 ITSO.DEV1.ARCHDEF
2004.103 15:39:13 EXECUTION ENDS
2004.103 15:39:13 TASK COMPLETED WITH RETURN CODE 0000
2004.103 15:39:13 DFSMSDSS PROCESSING COMPLETE. HIGHEST RETURN CODE IS 0000
```

Figure 4-20 DFDSS COPY operation with REPLACEU

4.2.2 Record level sharing (RLS) considerations

Currently DFSMSdss does not save all of the information in the RLS cell in the VVDS when performing a COPY operation to a preallocated data set. But RESTORE processing does save some RLS information.

COPY processing has been enhanced and now works like RESTORE processing. RLS information of the preallocated target data set is saved when checking the VVRs of the preallocated target. The LOG, LOGSTREAMID, and BWO information is saved. The recovery required bit of the source data set will continue to be carried forward regardless of the preallocated target status.

After data movement, if the preallocated target data set had RLS information, it is put back into the RLS cell, otherwise the target will have no RLS information. If the target data set was not preallocated, the RLS information from the source data set is propagated to the target.

Exit 22, which allows the use of the DFSMSdss API to control what gets placed in the RLS cell, is used by RESTORE processing but will not be invoked for COPY processing.

DFSMSdss, when using FlashCopy®, issues a deleted data space withdraw (DDSW) for the tracks of the preallocated target data set during the REPLACE or REPLACEU operation.

4.2.3 Migration and coexistence considerations

There are no specific migration and coexistence considerations for this enhancement.

4.3 DFSMShsm enhancements

DFSMShsm complements DFSMSdss by providing various functions such as storage management, space management, tape mount management, and availability management. DFSMShsm has been enhanced for performance improvement.

4.3.1 Multiple secondary space management (SSM) tasks

DFSMShsm now allows multiple secondary space management tasks to run concurrently. This enhancement allows all secondary space management functions to process in multiple tasks except from ML1 DASD to ML2 DASD.

Current operation

Secondary space management does the following:

- Statistics cleanup.
  - Restart tape copy
  - Daily statistics records (DSR) and volume statistics record (VSR) records that expired and require cleanup from the migration control data set (MCDS)
- Migration level cleanup performs several scans starting from where it previously stopped:
  - Deletion of expired migrated data sets
    - MCDS records (MCD) that are expired or recalled and require cleanup
    - MCD records that require ML1-to-ML2 data set movement
    - Migration control data set alias entry record (MCA) and Migrated CDS VSAM associations record (MCO) records that are expired
  - One scan of the offline control data set (OCDS)
    - Tape copy needed (TCN) records that require TAPECOPY restart
    - Performed before a DSR/VSR scan in MCDS
- Deletes records from small data set packing (SDSP).
- Moves migration copies from ML1 to ML2.
- Expired ML1 and ML2 processing.
- Runs as a single task.

SSM enhancement

SSM operation has been enhanced to be multithreaded. A new SETSYS MAXSSMTASKS command has been added, which specifies the maximum number of concurrent automatic secondary space management tasks that will run. The syntax of MAXSSMTASKS is shown in Figure 4-21.

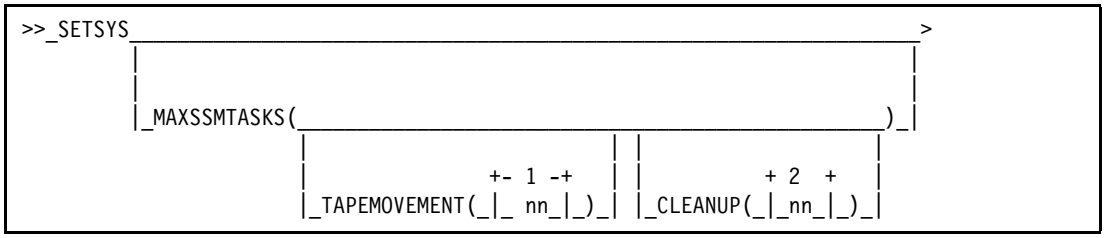

Figure 4-21 MAXSSMTASKS command syntax

CLEANUP MAXSSMTASKS(CLEANUP) specifies the maximum number of secondary space management migration cleanup tasks that can run concurrently. For nn, substitute a decimal number from 0 to 15 for the maximum number of concurrent migration cleanup tasks. A value of 0 indicates that there are no cleanup tasks, and therefore the secondary space management window includes no cleanup processing. The default value is 1. The various cleanup functions are:

- Expiration of migrated data sets
- Deletion of MCDS records that are no longer needed
- Deletion of SDSP records that are no longer needed
- Deletion of DSR, VSR, and TCN records

TAPEMOVEMENT MAXSSMTASKS(TAPEMOVEMENT) specifies the maximum number of automatic secondary space management tape migration tasks that can run concurrently. This parameter applies to migration of data sets from level 1 volumes to level 2 tape volumes. For nn, substitute a decimal number from 0 to 15 for the maximum number of concurrent tape migration tasks. A value of 0 indicates that there are no tape movement tasks, and therefore DFSMShsm performs no ML1-to-ML2 data movement. This setting is useful when no tape drives are available. The default value is 2.

Note: There is still only one task for ML1-to-ML2 DASD data movement.

4.3.2 Additional SSM enhancements

The following additional enhancements have been made to SSM:

- Before z/OS V1R6, SDSP data sets were sometimes open for the entire duration of SSM processing, preventing reorganization of the SDSP data set. SDSP will now be freed and closed if they have not been used for three minutes.
- Expiration of deleted data set processing has been enhanced.
  - Once a migrated data set is identified by SSM as eligible for expiration, a delete management work element (MWE) is built and placed on the recall/delete queue. The queue is processed outside of SSM function by the recall task structure. Therefore, it is possible that the requests are processed after the SSM window has ended.
  - Deletes generated by SSM have lower priorities than non-SSM deletes. This allows recalls and user-initiated deletes to be selected before SSM-initiated deletes.
  - There are two priorities for SSM-initiated deletes. If the delete is for a data set on ML1, the priority is set to 1; if the delete is for data sets on ML2 tape, the priority is set to 0. This allows ML1 space to be freed before deleting data sets on tape.
- The default for MAXRECALLTASKS has been increased from 5 to 15.

4.3.3 Migration and coexistence considerations

z/OS V1R6 DFSMShsm can coexist with prior releases of DFSMShsm. There is a plan for toleration PTFs to be made available for lower level systems.

4.4 DFSMSrmm enhancements

DFSMSrmm manages the removable media resources. It provides various functions such as library management, shelf management, volume management, and data set management. DFSMSrmm has been enhanced by adding the following:

- New information about setting DFSMSrmm client/server systems
- Additional considerations for running DFSMSrmm utilities when client/server support is used
- New EDGRMMxx parmlib member OPTION command operands and VLPOOL command operands for client/server support
- New samples to the list of DFSMSrmm-supplied samples

4.4.1 DFSMSrmm client/server support

DFSMSrmm can now run on a system that does not have direct access to the DASD containing the DFSMSrmm CDS. The I/O requests to the CDS are handled over the TCP/IP network and allow multiple sysplexes to have a single tape inventory.

DFSMSrmm subsystems

An RMMplex is one or more MVS systems, each running a DFSMSrmm subsystem sharing a control data set. An RMMplex can optionally include one or more DFSMSrmm subsystems as servers, one or more client subsystems, in addition to standard DFSMSrmm subsystems.

Server subsystem

A server subsystem has the following properties:

- A server subsystem is identified by the OPTION SERVER operand.
- The OPTION SERVER identifies the basic TCP/IP info that allows the server to handle requests from CLIENT subsystems.
- It has direct access to the CDS.
- It processes its own local requests, as well as requests from clients.

Client subsystem

A client subsystem has the following properties:

- A client subsystem is identified by the OPTION CLIENT operand.
- The OPTION CLIENT also identifies basic TCP/IP info that allows the client to send requests to a server.
- The client subsystems have no direct access to the CDS but share the CDS through the server.
- It verifies the CDSID of the client, which matches that of the server.
- Any parmlib options not required for a client are ignored.
- If there is more than one server, the client can only connect to one server at a time.
- If the server is not available, any requests that require server processing are failed with an I/O error.

Standard subsystem

A standard subsystem has the following properties:

- A standard subsystem does not have OPTION CLIENT or OPTION SERVER specified and is the default.
- If the server initialization fails, it can be used as a standard subsystem.
- The standard subsystems have direct access to the CDS.

Processing requests

The processing requests depend on the subsystem, whether it is a server or a client.

- For the client:
  - Most requests are processed by communicating with the server.
  - Requests that can be processed by the client are processed locally.
  - When multiple requests are being processed, a FIFO queue is maintained.
  - The operator command Q A displays the tasks and a summary of the queues.
- For the server:
  - Local requests are processed unchanged.
  - Client requests are accepted and processed synchronously while the client waits.
  - There is no queue of client requests.
  - Request queues are only maintained for local requests.
  - The operator command Q A displays local requests as well as accepted client requests.

TCP/IP considerations

The following TCP/IP considerations are required for the client/server environment.

- DFSMSrmm client/server processing is dependent upon Internet Protocol V4.
- You must set up the firewall, if any, to allow communication between clients and servers for the defined IP addresses and ports. DFSMSrmm does no authentication, encryption, or verification of connection requests received on servers other than to verify that it is a valid request and that the CDSIDs match.
- Consider the use of RACF to protect the use of the IP addresses defined for DFSMSrmm and limit use of the IP address to the DFSMSrmm started task.
- Tracing of the IP communication is enabled by DFSMSrmm support.
  - Use TCP/IP facilities such as component trace to gather info on DFSMSrmm socket processing.
  - Use RMM PDA trace for tracing DFSMSrmm subsystem processing.

EDGRMMxx parmlib member changes

Here we describe the changes to the EDGRMMxx parmlib member.

OPTION CLIENT

The CLIENT option is specified as:

OPTION CLIENT(SERVERNAME(*ServerName*) PORT(*PortNumber*)

where:

ServerName The servername can either be an IP address, a fully qualified domain name, or a server host name. DFSMSrmm uses the domain name system (DNS) to resolve a domain name or a host name into an IP address. SERVERNAME is a required operand when you specify CLIENT. servername can be 63 alphanumeric characters, period (.), and hyphen (-). The host name can be a maximum of 63 characters. The host name must contain one or more tokens separated by a period. Each token must be larger than one character. The first character in each token must start with a letter. The remaining characters in each token can be a letter, number, or hyphen.

PortNumber This operand specifies the port number to be used for IP communication. The PORT operand is required. Specify a value from 1 to 65535. Port numbers 1 to 1023 are reserved.

OPTION SERVER

The SERVER option is specified as:

OPTION SERVER(PORT(*PortNumber*) SERVERTASKS(*number*))

where:

PortNumber This operand specifies the port number to be used for IP communication. The PORT operand is required. Specify a value from 1 to 65535. Port numbers 1 to 1023 are reserved.

number This operand specifies how many DFSMSrmm tasks should be available on the server to handle socket connections from client systems. DFSMSrmm uses this number to determine how many tasks are to be started for processing all client requests on this server. Specify a value from 1 to 999.

OPTION LOCALTASKS

The LOCALTASKS option is specified as:

OPTION LOCALTASKS(*number*)

where:

number Use this operand to set the number of tasks available on each system for processing locally initiated requests. You can optionally specify a value for local tasks on each and every instance of the DFSMSrmm subsystem; client system, server system, or standard system. On a client system, LOCALTASKS is also the maximum number of tasks that can make a socket connection to the server. Specify a value from 1 to 999

Default: LOCALTASKS(10)

VLPOOL

VLPOOL is specified as:

VLPOOL PREFIX(*prefix*)…AUTOSCRATCH(YES | NO)…RELEASEACTION(*NOTIFY*)

where:

prefix Specifies a generic shelf location that is used in operator messages, RMM TSO subcommands, and the DFSMSrmm ISPF dialog. A pool prefix is one to five alphanumeric, national, or special characters

followed by an asterisk. Use a single asterisk to specify the default volume pool that contains all volumes not specifically included in another pool.

AUTOSCRATCH When you specify AUTOSCRATCH(YES), DFSMSrmm automatically performs “return to scratch” processing when expiration processing is running on a system that has access to the catalogs, TCDB, and library for the volume. When you specify AUTOSCRATCH(YES), you do not need to manually confirm the scratch release action. DFSMSrmm performs return to scratch cleanup processing, including: uncataloging data sets based on the parmlib OPTION UNCATALOG command value, deleting RACF profiles based on the parmlib OPTION TPRACF value you set, and updating the TCDB based on the parmlib OPTION SMSTAPE(UPDATE(SCRATCH)) command.

When you specify AUTOSCRATCH(NO) or manually confirm the scratch release action, DFSMSrmm does not perform any release action processing. If you have specified the scratch release action for a volume and the volume is in a pool with AUTOSCRATCH(NO), DFSMSrmm does not return the volume to scratch status. You must perform whatever actions or cleanup activity you require.

RELEASEACTION(*NOTIFY*)
Use this operand with the NOTIFY value to automatically set the NOTIFY release action for all volumes in this pool. If you have an e-mail address for the owner of the volume, DFSMSrmm sends the owner notification that the volume is pending release. By default, DFSMSrmm does not set the NOTIFY release action.

TSO command changes

Changes have been made to TSO commands, as follows:

- LISTCONTROL CNTL ACTIONS MOVES

  This command will return info from the CDS on the server. All other LISTCONTROL subcommands return information from the system on which the command is issued

- CV *volser* CRLSE(SCRATCH)

  This command is available for optional manual confirmation, which enables cleanup activities such as updating control files on other platforms and then performing the confirm. This new capability is not required for z/OS systems in a client/server environment and could be used with EDGUX200 or VLPOOL AUTOSCRATCH(NO) to control return to scratch for volumes managed for other platforms.

Exit EDGUX200

Exit EDGUX200 is called during expiration processing when a volume is about to be returned to scratch. There are new parameters passed to this exit, which are mapped by the mapping macro for the parameter list, EDGPL200. The new parameters are shown in Figure 4-22.

```
PL200_VOLUME_FLAGS     DC XL1'00'    STATUS FLAGS FOR VOLUME
PL200_SMS_VOL          EQU  X'80'    VOLUME IS SMS MANAGED
PL200_HOME_LOCDEF      EQU X'40'     VOLUME IS IN STORAGE LOC DEFINED AS HOME
PL200_MANUAL_SCRATCH   EQU X'20'     VOLUME IS IN VLPOOL WITH AUTOSCRATCH(NO)
PL200_OWNER            CL8' '        VOLUME OWNER
PL200_EDGSVREC_ADDR    DC F'0'       ADDRESS OF VOLUME INFO
PL200_CATSYSID         16CL8' '      CATSYSID LIST FOR THE RUNNING SYSTEM
```

Figure 4-22 New parameters passed in exit EDGUX200

REXX variables

New REXX variables have been created into which client/server data will be returned from the subcommands that return client/server information. This is shown in Table 4-1. The variable names are EDG@SSTY, EDG@CSHN etc.; that is, add the prefix EDG@ to the name shown in the table.

Table 4-1 REXX variables

Name	SFI Number	Type API Rexx	Length API Rexx	Value API Rexx	Commands
Subsystem type SSTY	X'8A2500'	3(Bin(8)) Character	9 8	0 –Standard system 1 – Client system 2 – Server system One of: CLIENT, SERVER, or STANDARD	LC OPT
Client/Server host name CSHN	X'819200'	7 variable character Character	71 63	1 to 63 alphanumeric characters including hyphen and periods or blank	LC CNTL
Client/Server IP address CSIP	X'819250' Character	7 variable character Character	23 15	1 to 15 alphanumeric characters including periods or blank	LC CNTL
Server host name SRHN	X'8A1A00'	7 variable character Character	71 63	1 to 63 alphanumeric characters including hyphen and periods or blank	LC OPT
Server IP address SRIP	X'8A1A30'	7 variable character Character	23 15	1 to 15 alphanumeric characters including periods or blank	LC OPT
Server Port number SRPN	X'8A1A50'	5(Bin(31)) Character	12 5	Binary value 1 to 5 numeric numbers from 1 to 65535. Zero indicates no port number.	LC OPT
Local Tasks LCTK	X'843100'	5(Bin(31)) Character	12 3	Binary value 1 to 3 numeric characters	LC OPT
Server Tasks SRTK	X'8A1AF0'	5(Bin(31)) Character	12 3	Binary value 1 to 3 numeric characters	LCOPT
Scratch Mode SCRM	X'891000'	3(Bin(8)) Character	9 6	Binary value: 0-Auto, 1-Manual AUTO or MANUAL	LC VLPOOL
Actions on release ACT				For LC VLPOOL we only list 'N' or blank	+ LC VLPOOL

SMS tape support

DFSMSrmm processing remains unchanged for system-managed tape processing, initiated by OAM and driven through CBRUXxxx exits. They are:

- Library partitioning
- Cartridge entry
- Cartridge insert

Command-driven volume changes/ejects for volumes in system-managed libraries, which are only accessible from the client system, must be issued from the client or from a system sharing the same TCDB and library manager.

MVS operator commands

There are new commands and some commands were changed:

- The new operator command for the server:
  - F DFRMM,RESTART LISTENER

    This command stops and restarts TCP/IP processing tasks.
- The new operator command for any system:
  - F DFRMM,QUERY ACTIVE

    This command displays active and queued tasks and also lists IP server subtasks and includes them in the following:

    - Count of total subtasks
    - Count of active subtasks

    This also includes a list of active subtasks with a summary of the requests they process:

    - Sysid
    - Jobname
    - Function code
    - Status
    - Subtask token
  - F DFRMM,ABEND(TaskToken)

    This command abends the IP subtasks as well as subsystem requests processing tasks. You may find the task token from the Q A command.

Authorization

There is no change in the way authorization is handled for TSO subcommands and utilities. TSO commands and utilities are authorized by DFSMSrmm on the system on which they are run.

Utility restrictions running on client or server

While running the utilities, some restrictions apply based on whether the utility runs in the server or the client. The restrictions are:

- EDGUTIL on a client

  This can only process a DFSMSrmm CDS that is not already in use. Since you cannot run this utility with the active CDS on the client system, you can recover a backup copy of the CDS to the client system to run VERIFY(VOLCAT), VERIFY(SMSTAPE), and MEND(SMSTAPE).
- EDGUTIL on a server

  When run on a server, it cannot access the TCDB or library or libraries that are only accessible from a client system.
- EDGHSKP on a client

  This utility runs all functions except BACKUP, but elapsed times will be longer since there is no direct connection to the CDS.
- EDGHSKP on a server

  Using this utility, it is recommended to run all inventory functions other than CATSYNCH and EXPROC on either a server or standard system. There are times when EXPROC has to be run on client system, for example:
  - A system with system-managed tape, so that the TCDB can be updated

- RACF profiles to be maintained
- Data sets to be uncataloged

- EDGBKUP on a client

  This utility cannot process an active CDS. Therefore, it can use all functions on the client, if you provide DD statements for the CDS and Journal. This enables the use of the BACKUP and RESTORE options independent of the DFSMSrmm subsystem. EDGBKUP can only be used on a client system to back up, restore, and forward recovery a CDS which is not in use by the DFSMSrmm subsystem.

Note: Any attempt to run a utility using a function not available on the client will cause the utility to end with RC16.

Client/server support for expiration processing

This describes the changes made to expiration processing.

Current processing

Under current expiration processing, when all release actions are completed a volume can be returned to scratch when all the following are true:

- The scratch action is set and there is no other release action.
- The volume is not system-managed or the system-managed library is available.
- All catalogs are shared (CATSYSID(*) or CATSYSID not specified), or catalogs are not shared and the creation systemid for file 1 on the volume matches one of the system's CATSYSID(list).
- The EDGUX200 exit allows return to scratch.

Changed processing

- Expiration processing has been changed. Assuming that the catalogs on client systems are not shared with the DFSMSrmm server system:
  - Catalogs can be successfully shared or unshared as long as CATSYSID is specified to correctly match your environment.
  - If catalogs are shared, CATSYNCH does not have to be run but is recommended for performance reasons.
  - If catalogs are not fully shared, CATSYSID must be specified in the server system with a list of system IDs for which uncatalog processing can be performed.
  - Specify CATSYSID on the client system with a list of system IDs for which uncatalog processing can be performed on the client.
- Running EXPROC on each system is normal for unshared catalog environments. This is to automate the return of volumes to scratch and maintain catalogs, TCDB, and RACF profiles.
- As volumes are set pending release, the notify action is based on VLPOOL settings, as follows:
  - If notify release action is set on for the volume, it is left on.
  - If notify release action is specified at the pool level, the volume notify release action is set.
- If a volume matches a VLPOOL defined with AUTOSCRATCH(NO), return to scratch release action is not performed by EXPROC processing until confirmed by command.

- Uncataloging, RACF TAPEVOL, and system-managed tape processing are only performed during EXPROC processing, if the scratch release action is still set and auto return to scratch is allowed for the volume pool.
- If scratch release is not set, because it was confirmed by command, the volume status is changed to scratch and no catalog, RACF, or TCDB activities are performed.

Recommendation for using the system

- It is recommended for users and administrators who require regular access to DFSMSrmm data to log on to the server system. However, client system users can still use dialog, TSO subcommands, and batch utilities.
- You may log on to the client system for the following reasons:
  - Processing relates to a system-managed library that is known only to the client. For example:
    - Eject a volume from a library
    - Add volumes using STATUS(VOLCAT)
    - Change volume attributes that are also maintained in the TCDB
    - Run expiration processing
    - Confirm move processing for exported stacked volumes
    - Run EDGUTIL VERIFY with the TCDB and optionally Library Manager
  - Processing relates to cataloged data sets, which are cataloged only on a client system, for example UNCATALOG(Y/S) specified and:
    - Confirm the erase or init of a volume
    - Return volumes to scratch status and DFSMSrmm is to perform the return to scratch cleanup actions
    - Delete volumes which contain cataloged data sets

Client/server migration considerations

- All client and server systems must be at least at the z/OS 1.6 level.
- The CDS can be shared with lower and higher level systems.
- Client/server systems can be added, or existing systems can be converted. Existing DFSMSrmm systems can be merged into the RMMplex by merging CDSs, as documented in *DFSMSrmm Primer,* SG24-5983.
- On a server system:
  - Update the parmlib options with the SERVER operand and select appropriate values for the suboperands.
  - If CATSYSID is not set, add the operand to define the list of system IDs that share catalogs with the server, or specify that catalogs are shared.
  - Ensure that TCP/IP definition files are updated to identify the server host name, IP address, and port number.
  - Ensure that the firewall is updated with client and server IP addresses and port numbers.
  - Refresh DFSMSrmm with the new parmlib member.
  - Run EDGHSKP CATSYNCH, unless your catalogs are already synchronized.
- On a standard system:

- If CATSYSID is not already set, add the operand now to define the list of system IDs that share catalogs with the server system.
- Refresh DFSMSrmm with the new parmlib member.
- Run EDGHSKP CATSYNCH, unless your catalogs are already synchronized.

- On a client system:
  - Update the parmlib options with the CLIENT operand and select appropriate values for the suboperands.
  - If CATSYSID is not already set, add the operand now to define the list of system IDs that share catalogs with the client system, or specify catalogs that are shared.
  - Ensure TCP/IP definition files are updated to identify the server host name, IP address, and port number.
  - Ensure that the firewall is updated with the client and server IP addresses and port numbers.
  - Refresh DFSMSrmm with the new parmlib member.
  - Run EDGHSKP CATSYNCH, unless your catalogs are already synchronized.
  - Ensure EDGHSKP with EXPROC is run regularly to return volumes to scratch.

Client/server coexistence

All client/server systems must be at least at z/OS V1R6. We recommend to upgrade the server systems first. As long as toleration is installed on all systems in an RMMplex, they can successfully support client/server processing. All dates and times are local times and any dates and times displayed are exactly as they are stored in the CDS. There is no conversion from server time zone to client time zone.

4.4.2 RMM API command classes

There is a new interface, which allows high-level languages such as C/C++ and Java to issue commands. The Input is an RMM command string and the output can be either structure field introducers (SFIs) or XML.

Output for SFIs is in EBCDIC and the values included are in binary, character, decimal.

Output for XML is converted to character in unicode format, as defined in the XML Schema file for the RMM resources. Each XML object returned from the getBufferXml method of the RmmCommand class contains only the data and tags to define the data. The document rmmxml.xsd is a new file that is shipped and is referenced from each XML object.

The RMM API is called RmmApi. The header file containing the class definitions is contained in part EDGXHCLU. Use the RmmApi class to prepare the environment for using the RmmCommand classes to run DFSMSrmm TSO subcommands via the API.

Table 4-2 shows the RMM API command class descriptions.

Table 4-2 RMM API command class descriptions

Class	Description
RmmInterface	This is the superclass for DFSMSrmm processing. It is used to provide methods that are common to the other DFSMSrmm-provided classes.

Class	Description
RmmApi	This class extends the RmmInterface class. Use this class to create an object to initiate a communication session with RMM. It must be created before any other classes or methods are used. The returned object can be used to create one or more RmmCommand objects to enable you to run DFSMSrmm subcommands. You need one RmmApi object for each z/OS TCB you run under. When you want to end the communication with RMM, and run no more subcommands, delete the corresponding RmmApi object.
RmmCommand	This class extends the RmmInterface class. Use this class to run an RMM TSO subcommand. You must pass a reference to the RmmApi object when you instantiate an instance of this object. You can instantiate multiple instances, enabling you to process multiple commands in parallel. For example, use the output from a SEARCH command to issue LIST subcommands.

Table 4-3 shows the RMM API method descriptions.

Table 4-3 RMM API method descriptions

Method	Description
RmmApi.openAPI	Use this method to check that RMM is active and available for commands to be processed. RMM loads the EDGXAPI callable interface and verifies whether RMM is in use on this system.
RmmApi.closeApi	Use this when you no longer want to communicate with RMM using this command session. RMM deletes the EDGXAPI callable interface and releases any resources that are still in use.
RmmCommand.issueCMD	Use this method to issue a subcommand to RMM. The subcommand return and reason code are returned. To access the output from the subcommand, use the getBufferSfi or getBufferXml methods.
RmmCommand.getBufferSfi	Returns a string containing the SFI output buffer from subcommand processing. Use this after issueCmd and after getNextEntry.
RmmCommand.getBufferXml	Returns a string containing the XML output converted from the SFI output of subcommand processing.
RmmCommand.getNextEntry	Use this method to retrieve information for the next resource when more than one resource is to be returned, such as for SEARCH subcommands and LISTCONTROL.
RmmInterface.getMessageText	Returns a string containing the DFSMSrmm information or error message for the last command issued or the last getNextEntry processing.
RmmInterface.getApiRc RmmInterface.getApiRs	Returns the return code from the last API request. Returns the reason code from the last API request. Refer to DFSMSrmm Application Programming Interface and DFSMSrmm Guide and Reference for the return code values and their meaning. Use the getMessageText method to retrieve the corresponding information or error message.

4.4.3 RMM ISPF usability

The DFSMSrmm ISPF dialog has been enhanced for better usability. You can request DFSMSrmm prime panels with saved variables. You can create a CLIST of DFSMSrmm TSO subcommands using the dialog, use new line operators in dialog lists to better manage DFSMSrmm resources, and also specify a new system option to define the type of output station where you want your system-managed cartridges ejected.

Variable saving and reuse

Variables used for data entry are saved and restored in the shared variable pool so that they can be saved and reused across sessions. You can use the RMM ISPF panel, DFSMSrmm Dialog User Options, or the fastpath command REUSE ON|OFF to set the variable savings option. The initial value is REUSE ON. Variables are always saved independent of the setting, but if REUSE OFF is set, variables are not retrieved and all values must be entered. Figure 4-23 shows the DFSMSrmm Dialog User Options panel.

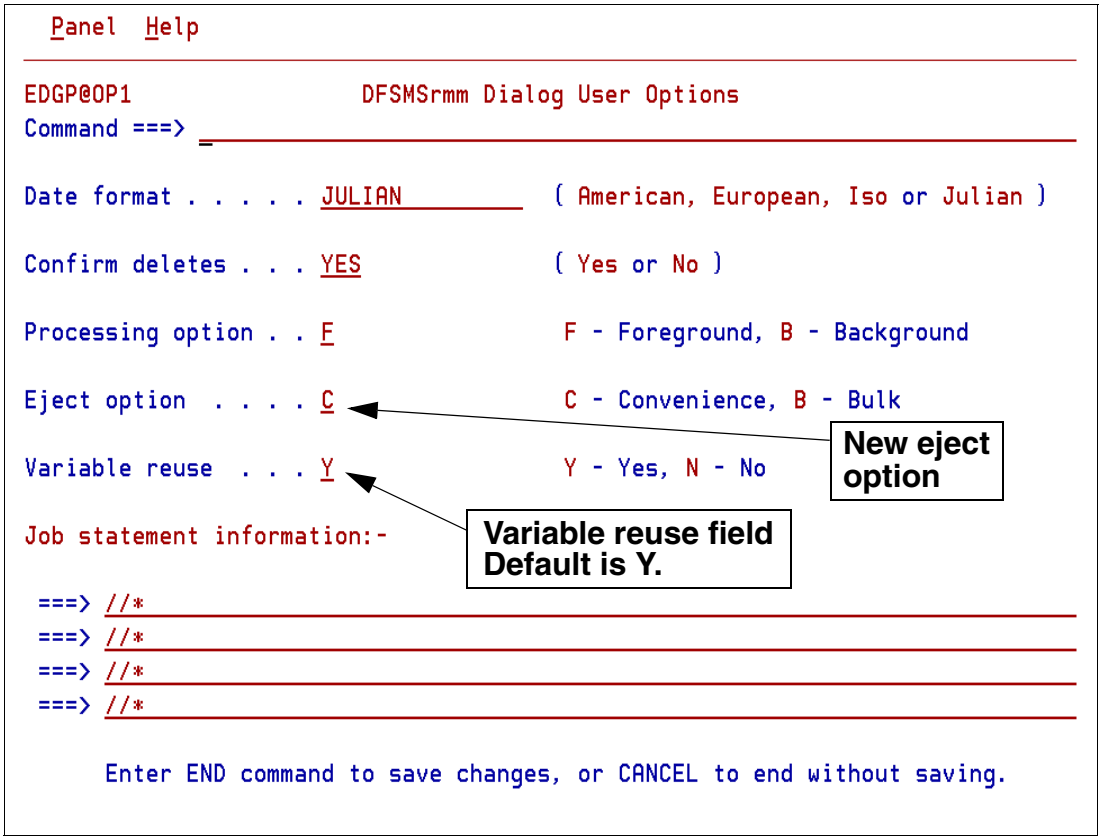

Figure 4-23 DFSMSrmm Dialog User Option panel showing variable reuse field

Listing logical volumes on a stacked volume

The `I` line command has been enhanced. When issued against a stacked volume, it lists the volumes. However, the data sets are not listed along with the volumes.

New line commands supported from search results list

New line commands introduced, which are supported against any search result lists:

VL List the volume chain from the data set search results.

IL List the data set chain from the volume search results.

CA Confirm all actions and moves outside of action summary lists.

CH Change from the action summary list.

CS Confirm scratch supporting new manual scratch release action for volumes.

U This line command calls an exec, EDGRLCL, which is customized for installation-defined commands.

Move the LIMIT input field

The LIMIT field has been moved to the first viewed part of the scrollable area on volume search panels. If the LIMIT field is null or blank, this value is set to "*". Figure 4-24 shows the LIMIT field in the first 24 lines of the scrollable panel.

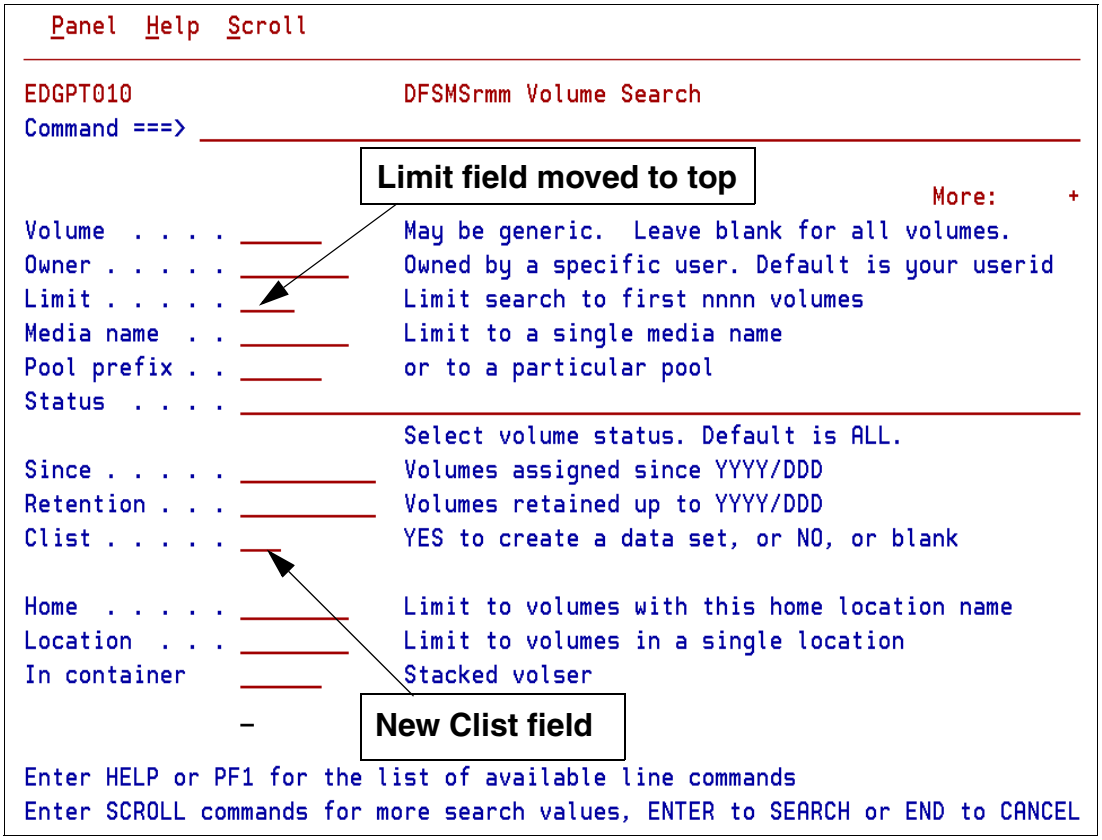

Figure 4-24 DFSMSrmm Volume Search panel showing the Limit field

New EJECT User option

Now you can specify and change the default EJECT station to CONVENIENCE or BULK. This is specified in the Dialog User Options panel or fastpath command EJECT BULK|CONVENIENCE. The initial value is set to EJECT CONVENIENCE. This is shown in Figure 4-23 on page 117.

CLIST option for all SEARCHes

A new field has been added on the first scrollable screen for all search panels. When YES is specified in this field, a new panel is displayed prompting for relevant values, as shown in Figure 4-25.

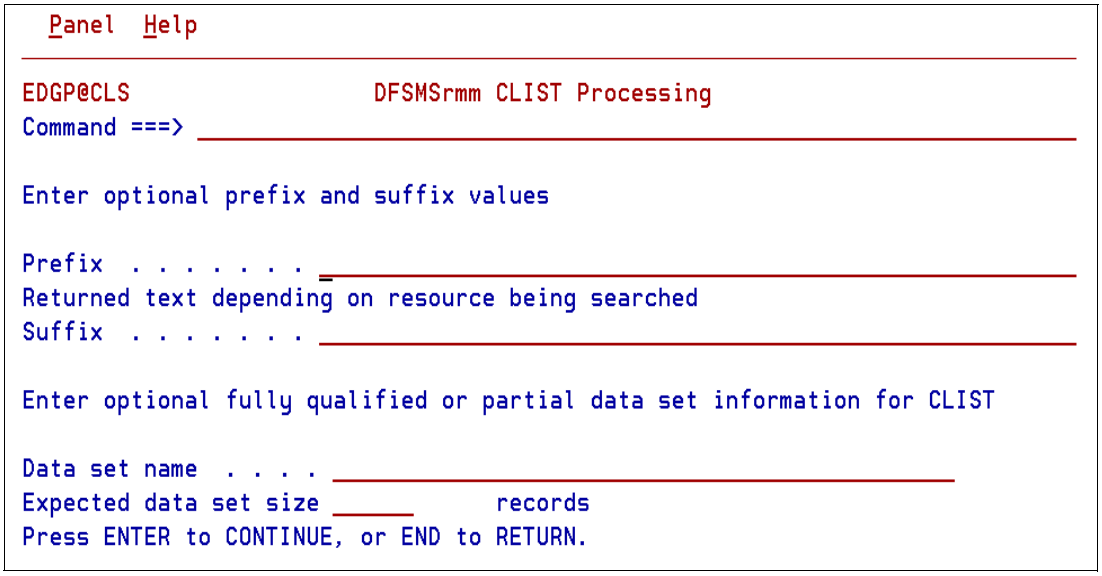

Figure 4-25 DFSMSrmm CLIST Processing panel

Customizing the dialog with the new line command U

The U line command calls an exec, EDGRLCL, and can be customized to provide new line command support. This implementation is a local extension to the search results line commands. A sample has been provided in SYS1.SAMPLIB(EDGXMP3).

Search results list row management

The search results list now has the capability to delete rows for records that are deleted. The following lists are affected:

- Data set
- Product
- Volume (for FORCE and REMOVE only)
- VRS

5

UNIX System Services enhancements in z/OS V1R6

In this chapter we describe the enhancements introduced in z/OS V1R6 UNIX System Services:

- Miscellaneous enhancements
- ISHELL enhancements
- 64-bit support
- RTLS removal

5.1 Miscellaneous enhancements

In z/OS V1R6 there are a number of enhancements that cover various services of the z/OS UNIX environment, as follows:

- Shared condition variables
- RAS improvements
- Spooled output constraint relief
- Automove system list wildcard support
- Increase the 64K per process file descriptor limit
- Automount enhancements
- Fork() accounting
- Superkill function
- Shell and utility enhancements
- BPXWPERM environment variable
- Mount utility enhancements
- USS REXX BPXWDYN enhancements
- Logical file system support of zFS
- Distributed BRLM enhancement

5.2 Shared condition variables

Currently, UNIX-based server applications, like LOTUS DOMINO, make use of mutexes and condition variables in shared memory to serialize resources across multiple processes. On z/OS this was not supported, so these applications had to be rewritten to be ported to the z/OS platform.

Note: A *mutex* is a mutual exclusion lock that can be held by one thread only. Mutexes are used to protect data or other resources from concurrent access. Also, a mutex has attributes, which specify the characteristics of the mutex. On the other hand, a *condition variables* allows threads to wait until some event or condition has occurred. Although it is not easy to program, a condition variables allows the implementation of powerful and efficient synchronization mechanisms.

In z/OS V1R6 support is added to allow both condition variables and mutexes to reside in shared memory. This allows applications like LOTUS DOMINO to have a more common code base across the platforms it can be run upon. With *shared condition variables* it is no longer necessary to rewrite applications in order to serialize resources shared across multiple processes. Now we can use the standard UNIX constructs of mutexes and condition variables in shared memory.

Shared condition variables are intended to be used by an LE C/C++ application for the purpose of exploiting mutexes and condition variables in shared memory. This usage is intended to be the equivalent to that of using these objects in regular private storage. The added benefit is that mutexes and condition variables in shared memory can be used to synchronize operations across multiple processes rather than in just one process. To exploit this feature, an application simply has to do the following:

1. Compile the application with the new _OPEN_SYS_MUTEX_EXT feature switch defined. This will cause the pthread_mutex_t and pthread_cond_t objects to be defined as larger objects that can support the new sharing capability.
2. Initialize a pthread_mutex_attr using the pthread_mutexattr_setpshared() function to set the pshared attribute to pthread_process_shared to indicate that the object can be shared across multiple processes. Do the same for a pthread_condattr using the new pthread_condattr_setpshared() function.
3. Use pthread_cond_init() and pthread_mutex_init() functions to initialize a condition variable and mutex specifying a pthread_cond_attr and pthread_mutex_attr that has pthread_process_shared enabled.
4. Use the standard condition variable functions pthread_cond_wait(), pthread_cond_timedwait(), etc. and mutex functions pthread_mutex_lock() and pthread_mutex_unlock() in the manner they are normally used by UNIX applications to synchronize operations.

Note: To enable cross-address space sharing in z/OS Release 6 a new assembler language callable service was added, the *"shared mutex and condition variable service".* Check the *z/OS UNIX System Services Programming: Assembler Callable Services Reference,* SA22-7803 for more information about the BPX1SMC callable service.

```
BPXO057I 08.51.42 DISPLAY OMVS 284
OMVS     000E ACTIVE          OMVS=(6D)
                              UNIX SERIALIZATION REPORT
RESOURCE     #1:
  NAME=SHARED MUTEX   DATA: SHMID=00000648 OFFS/ADDR=0000000000002428
 JOBNAME ASID TCB        PID    USER DATA        EXC/SHR  OWN/WAIT
 DOMINO1 013A 008EF190 16777220 0000000024780148 EXC    OWN
 DOMINO2 02B2 008FA190 16908357 0000000024825220 EXC    WAIT
```

Figure 5-1 Display OMVS command

The D OMVS,SER command shown in Figure 5-1 gives an example of the contention information.

5.3 RAS improvements

RAS stands for Reliability, Availability, Serviceability and is part of the RAS zSeries strategy for autonomic computing. From time to time customers have experienced hangups in z/OS UNIX for internal processing, which in some cases have led to system outages. An enhancement to z/OS V1R6 attempts to address this by providing additional latch cleanup and identification for z/OS UNIX hang conditions.

With this RAS enhancement, customers are less likely to encounter z/OS UNIX latch hang problems, because of additional cleanup procedures that will be performed. This includes providing cleanup for latches that have been abandoned by abnormally terminated address spaces. Prior to this support, it was possible for an abnormally terminated address space to continue to hold a latch, even though it was no longer active.

Additionally, with the messaging and command support that is being provided, customers will be more able to initiate actions to relieve z/OS UNIX hang conditions. This should lead to fewer system outages for customers.

Starting in z/OS V1R6, if z/OS UNIX Latch Contention is not resolving over several minutes, the action message BPXM056E is displayed:

```
BPXM056E UNIX SYSTEM SERVICES LATCH CONTENTION DETECTED
```

At this point, it has been detected that at least one unit of work in the system is holding onto a z/OS UNIX GRS latch for several minutes. If this message does not eventually get DOMed on its own, then a system programmer should issue the following command to determine which z/OS UNIX latches are the cause of the contention problem:

```
D GRS,C
```

If the system programmer found that a z/OS UNIX latch is owned by a user address space, there is a new console command that can be used to attempt to relieve the contention:

```
F BPXOINIT,RECOVER=LATCHES
```

This command will attempt to abend (422-1A5) user address space tasks that are holding the latches causing the contention. When the abend occurs, a system dump is requested for the 422-1A5 abend to capture a potential internal problem.

If the latch contention is resolved by the issuance of the command, message BPXM056E is DOMed and the following message is displayed:

```
BPXM067I UNIX SYSTEM SERVICES LATCH CONTENTION RESOLVED
```

If the latch contention cannot be resolved, the following message is displayed:

```
BPXM057E UNIX SYSTEM SERVICES LATCH CONTENTION NOT RESOLVED
```

Another RAS-related message that was added to z/OS V1R6 is the BPXP022E message. When jobs attempt to use z/OS UNIX Services while they are unavailable (during shutdown or prior to complete initialization), action message BPXP022E is displayed:

```
BPXP022E ONE OR MORE JOBS ARE WAITING FOR UNIX SYSTEM SERVICES AVAILABILITY
```

At this point, it has been detected that at least one unit of work in the system is waiting for the availability of z/OS UNIX System Services. To determine which jobs are waiting, issue D OMVS,A=ALL. An example of the output from D OMVS,A=ALL for a waiting job during OMVS shutdown is shown in Figure 5-2.

```
BPXO040I 10.48.16 DISPLAY OMVS 225
   OMVS     000D SHUTDOWN
   USER     JOBNAME  ASID        PID       PPID STATE   START     CT_SECS
            TC       0021 **********          0 1D----               .001
```

Figure 5-2 Display OMVS command

Note: The D in the 2nd character of the STATE field indicates the job is waiting for z/OS UNIX System Services availability.

5.4 Spooled output constraint relief

Prior to z/OS V1R6, a z/OS UNIX-based print and output server like Infoprint® Server could be ended by the operating system, 722 Abend, when it reached the system-defined job output limits for cards, bytes, lines, or pages.

Jobs are subject to the JES ESTBYTE NUM and OPT specifications that control the estimated output (SYSOUT) of that job. Figure 5-3 shows the default setting in JES2.

```
$D ESTBYTE
$HASP845 ESTBYTE  NUM=99999,INT=99999,OPT=0
```

Figure 5-3 ESTBYTE, JES2 initialization statement

If a job exceeds the ESTBYTE NUM specification, the action defined by the ESTBYTE OPT specification occurs. The NUM parameter sets the boundary for the estimated spool space utilization in thousands of bytes. When this limit is reached, the $HASP375 message is issued. What happens after that is controlled by the OPT parameter. Following are the options and subsequent actions:

0 Job is allowed to continue execution.
1 Job is cancelled without a dump.
2 Job is cancelled with a dump (if a dump statement was coded for this job step).

However, UNIX System Services programs are treated differently by JES than other jobs. Started tasks and USS programs that run in their own address spaces are not subject to the ESTBYTE NUM specification. Instead, they have a fixed limit of approximately 2 GB. But they are subject to the ESTBYTE OPT specification. This means that if a USS program places more than 2 GB of data on the JES spool, the following occurs:

- If ESTBYTE OPT=1 or OPT=2, the program abends with S722.
- If ESTBYTE OPT=0, a HASP375 message is issued.

A USS program that runs in the same address space as the JCL that starts it, is subject to the configured ESTBYTE NUM value, though it can be overridden using the BYTE parameter on the job statement.

Note: See also APAR II13685 for more information regarding the S722 abend.

A new enhancement of z/OS V1R6 attempts to address this problem by providing a way for a UNIX-based print and output server to be able to continue to spool output even after reaching these job output limits.

Due to the spooled output constraint relief, software developers/vendors will be able to develop a long-running UNIX print and output server for the z/OS UNIX environment. This will provide customers with the advantage of being able to run a z/OS UNIX print and output server on z/OS, like Infoprint Server, with a much smaller chance of outages.

To fully benefit from the constraint relief, the following software is required:

- z/OS V1R6
- z/OS Infoprint Server APAR OA05165

There is also a new environment variable, _BPX_UNLIMITED_OUTPUT=YES, that can be specified at the startup of a UNIX program via one of the spawn family of functions. This will cause the address space started for the spawned program to be set up in a way that if the job output limits are being reached, warning messages are displayed.

The use of this environment variable requires the user to be privileged. The user must either be a superuser or be given read access to the BPX.UNLIMITED.OUTPUT security profile.

5.5 Automove system list wildcard support

In z/OS V2R4, a choice was added to the automove facility. It is now possible to specify a prioritized automove system list to indicate where the file system should, or should not, be moved to when the owning system leaves the sysplex. All system names had to be defined in the AUTOMOVE INCLUDE system list in order to participate in the sysplex. This way, another listed system could take over the file system if the original server system failed.

New to z/OS V1R6 is the wildcard option. The wildcard is permitted to be the last item of the syslist in place of a system name on the AUTOMOVE INCLUDE list on all methods of MOUNT. This includes parmlib, TSO, shell, ishell, C program, assembler programs, and REXX.

Before, customers had to list all the system names in the include system list. Now they just have to list the priority list system names that they want to consider and the rest they can include by adding the wildcard character at the end. This means less work (knowing and typing all the system names in the sysplex) and reduces type errors of system names, etc. This support is more beneficial if you have a large number of systems participating in a sysplex.

For example, wildcard support allows the following on a MOUNT:

```
AUTOMOVE(INCLUDE,SY2,*)
```

Restriction: The wildcard is only allowed with an INCLUDE list, not an EXCLUDE list. It also must be the last item, or the only item, in the list.

AUTOMOVE(INCLUDE,*) is permitted, and is similar to AUTOMOVE(YES), except that if no system can take it over, the file system will be unmounted rather than becoming unowned.

Here are some examples of automove scenarios:

Moving file system Moving a file system to any other system also requires recognizing a wildcard in an Automove Include syslist, and choosing the target system accordingly.

OMVS shutdown (soft) Causes file systems to move and utilizes the Automove Include systems list.

Dead system Recovery If a file system had an Automove Include syslist with a wildcard, dead system recovery and takeover processing must recognize the wildcard and perform takeover accordingly. When a system dies, all remaining active systems will run dead system recovery and may attempt to take over ownership of file systems that were owned by a dead system. If a file system had a Automove Include syslist with a wildcard, then all systems explicitly named in the syslist will try takeover first, in the order in which they were specified. If none of these systems can take it over, then all remaining systems will also try to take it over. If no system can take it over, the file system will be unmounted.

Note: In order to use this support, all systems in a sysplex should be at the z/OS V1R6 level. Mixed-release systems in a sysplex will give unpredictable results during dead system takeover processing and file system move.

5.6 Increase the 64K per process file descriptor limit

Currently a single UNIX process is limited to 64K file descriptors, which are used for all open file system objects, including files, sockets, pipes, terminals, and directories.

This is a known system constraint for very large servers since it limits them to 64K connected clients at any one time. In z/OS V1R6, the limit is increased to 128K (131072) descriptors to provide immediate relief for some customers.

To exploit this new function, you can use the following techniques:

- BPXPRMxx MAXFILEPROC() IPL and RESTART configuration option
- SET OMVS=(xx) console command with BPXPRMxx MAXFILEPROC()
- SETOMVS RESET=(xx) console command with BPXPRMxx MAXFILEPROC()
- SETOMVS MAXFILEPROC() - Operator console command
- SETOMVS PID=ppp,MAXFILEPROC() - Operator console command

Note: You now have support for more clients on TN3270.

5.7 Automount enhancements

The automount function is enhanced in z/OS V1R6. The automount facility provides the following advantages for mounting z/OS UNIX file systems:

- You do not need to mount the user's file systems at initialization time and you do not need to request that they be mounted by an administrator or authorized operator. This makes it easier to add new users, because you can keep your parmlib specifications unchanged. This simplifies management of the user file systems.
- You can establish a simple automount policy to manage user home directories.
- A file system that is managed by the automount facility remains unmounted until its mount point is accessed.
- It enables you to reclaim system resources used by a mount if that file system has not been used for a period of time. You can specify how long the file system should remain mounted after its last use.

Tip: Usually you have more than one user working on your system. Therefore, it is strongly recommended that you use the z/OS UNIX automount facility. It manages the creation of the mount point and the mount of the user file systems for you. Whenever someone accesses a directory managed by the z/OS UNIX automount facility, the mount is issued automatically.

5.7.1 Add to an existing policy

Now we have the capability with the automount facility to add new automount-managed directories to the existing automount policy. To do this a new flag was added to the command line. The `a` option indicates that the policy being loaded is to be appended to the existing policy rather than replacing it, with the following command:

```
ROGERS @ SC65:/u/rogers>/usr/sbin/automount -a
FOMF0107I Processing file /etc/auto.map
FOMF0108I Managing directory /u
ROGERS @ SC65:/u/rogers>
```

Note: -a is mutually exclusive with -q; both these options are new with z/OS V1R6.

Automount policy update

In Figure 5-4, the first command displays the current automount policy that is in-storage and is currently being used, as follows:

```
/usr/sbin/automount -q
```

Then the user issues an **oedit** command to change the auto.map file to change the duration from 1440 to 1400.

Next, the command issued appends or adds the change in the policy into the current active sysplex in-storage, as follows:

```
/usr/sbin/automount -a
```

When the following command is issued to query the in-storage policy, you can see that it has changed the duration to 1400:

```
/usr/sbin/automount -q
```

```
ROGERS @ SC65:/u/rogers>/usr/sbin/automount -q
/u

name                 *
filesystem           OMVS.<uc_name>.ZFS
type                 ZFS
allocuser            space(1,1) storclas(OPENMVS)
mode                 rdwr
duration             1440
delay                360

ROGERS @ SC65:/u/rogers>oedit /etc/auto.map
ROGERS @ SC65:/u/rogers>/usr/sbin/automount -a
FOMF0107I Processing file /etc/auto.map
FOMF0108I Managing directory /u
ROGERS @ SC65:/u/rogers>/usr/sbin/automount -q
/u

name                 *
filesystem           OMVS.<uc_name>.ZFS
type                 ZFS
allocuser            space(1,1) storclas(OPENMVS)
mode                 rdwr
duration             1400
delay                360
```

Figure 5-4 Example of new automount policy command options

Although automount ensures that loading a new policy is an atomic operation, it does not serialize multiple simultaneous instances of running the automount utility. This remains the case when using the -a option. It should not be used in an automated script such as /etc/rc that can be run at the same time from multiple systems. This may result in changes to the automount policy being done without any indication of this. When automount is run this way—without the -a option—and the same policy is loaded from all systems, it is irrelevant that the policy load from one or more systems is overlaid.

Note: This new function can be used in a shared HFS environment at z/OS V1R6 and up.

Allow for MVS data sets

The automount facility in z/OS V1R6 allows the master and map files to reside in MVS data sets.

The default master file remains /etc/auto.master, and the file name can be specified on the command line. The syntax is enhanced to indicate the name is a data set name. The usual convention of // preceding the name is used. The data set may be a sequential data set or a member of a PDS. The data set name must be specified as a fully qualified name and may be upper or lower case. Single quotes are not needed. Example:

```
/usr/sbin/automount "//sys1.parmlib(amtmst01)"
```

Attention: Notice the double quotes around the name to avoid unwanted shell processing. If the data set is a PDS with a member name and the member does not exist, automount will act similar to the way it acts when the path name does not exist. In addition, an open abend message may be generated by the REXX processor. Automount does not support input files containing sequence numbers, whether they are from HFS files or data sets.

HFS to zFS automount

In previous releases of z/OS a generic automount policy could not automount both HFS and zFS file systems. All file systems had to be of the same type.

In HFS-to-zFS migration scenarios, customers are likely to migrate file systems over time rather than all file systems at once. The capability of allowing automount to be able to manage both HFS and zFS file systems in one automount policy is vital for enabling this migration. Automount was changed in z/OS V1R6 in such a way, that when HFS or ZFS is specified as the file system type, the data set is checked to determine what type of data set it is and then the mount is directed to the appropriate file system type.

No new explicit externals are added. However, some behaviors are changed. Specifically, it is now possible to have a generic automount policy manage both HFS and zFS file systems based simply on the type of the data set. Automount dynamically determines the correct file system type for the data set and directs the mount to the appropriate PFS.

Note: It is not necessary to specify zFS file system names in upper case.

5.8 Fork() accounting

There is a new fork() accounting functionality added in z/OS UNIX V1R6. The reason for that arose from the use of secure FTP. Secure FTP establishes the Secure Socket Layer (SSL) during processing of an FTP client. Part of this processing also includes a setuid() to obtain the client's identity. Normally an exec() would be required to force the termination of the Job Step Task to process the new userid's accounting information. The problem is that the exec() would destroy the SSL environment.

The reason SSL cannot survive an exec() or a spawn() is that it is not relocatable. SSL creates several pointers into LE, and once done the LE heap must remain intact. SSL must be initialized under the daemon's identity and thus before the typical setuid()/exec().

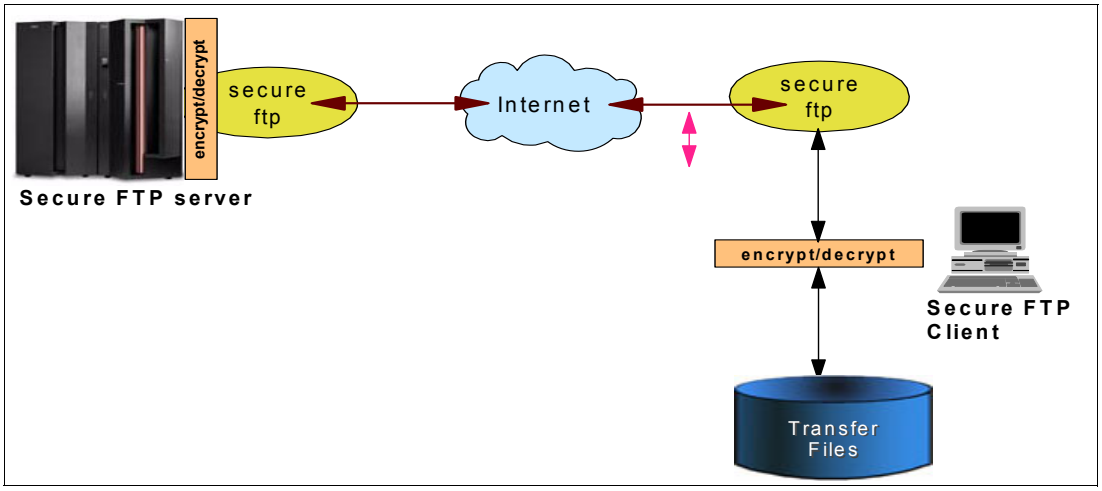

Figure 5-5 Process the account information for the new job while preserving the SSL object

To provide for a mechanism to process the account information while preserving the SSL object, the fork() accounting logic was added to the USS kernel. This way accounting data is available for secure FTP clients. The account data from the client is extracted from the user's work attributes in the RACF WORKATTR segment.

5.9 Superkill function

Using the new superkill OMVS function in z/OS V1R6 enables you to do the following:

- Cancel hung USS processes using UNIX semantics
- Cancel your own hung processes from the shell
- Use the enhanced console support to give operators and automated console applications additional flexibility

The need for a different kind of "cancel"-like solution emerged because:

- USS processes that use MVS services can defer USS signal processing. Even though these restrictions are documented, a hung process can cause other problems if it cannot be terminated.
- USS processes that become hung and cannot be terminated via the kill() service require MVS operator intervention to CANCEL the address space containing the USS process.

The superkill command bypasses the Language Environment, which is in contrast to the "normal" kill. There are some essentials for using superkill. A "normal" kill will do some quiescing towards the Language Environment.

This way higher applications like CICS or DB2 will be more willing to cooperate in the termination of UNIX processes. It is therefore necessary to do a normal kill first, before you try to use the superkill.

Superkill employs a 422 non-retryable abend, directed to the initial thread of the target process. Only one abend per process is allowed. If for some reason the 422 abend cannot terminate the process, it is doubtful another will succeed. Much like when a cancel fails, it would become necessary to terminate the address space.

There are four ways to invoke the superkill:

BPX1KIL / BPX4KIL	These are the USS-callable assembler services. A superkill can be sent by setting the PPSDSUPERKILL bit.
__superkill()	This is the C/C++ service.
kill -K [pid...][job-identifier ...]	The shell command.
F BPXOINIT,SUPERKILL=pid	The operator console command.

Restrictions to the use of superkill are:

- Cannot target Group IDs or –1.
- Must be authorized to send the target process a signal.
- Must be preceded by a normal sigkill signal.

Note: The restrictions have been put in place to ensure that the asynchronous nature of the abend is limited to processes that a truly hung. Limiting the abend to a single process at a time also avoids abusive use.

Shell kill command

The **kill** command is used to send a signal to a process or shell job. It is often used to terminate a process. As described in the previous section, processes occasionally get stuck in a state where the normal KILL signal cannot be delivered. In these situations the superkill can then be used to force termination of these so-called stubborn processes. You must have the usual authority to kill the process, for example the user associated with the process, or a superuser.

The **kill** command is a built-in function in both the sh and tcsh shells, with slight syntax variations. Both shells were updated with the same syntax for the new option.

The procedural flow of a superkill would be as follows:

1. Send a regular KILL signal by issuing, **kill -s KILL** *pid*.
2. Wait 3 seconds.
3. Then send a superkill to force termination, - **kill -K** *pid*.

Note: The superkill cannot be sent to a process group (by using a pid of 0 or a negative number) or to all processes (by using a pid of -1). Therefore, the alternate format of the kill command, **kill –s KILL %2** to kill shell job 2, is not useful.

5.10 Shell and utility enhancements

There are some additional commands in this new release of z/OS V1R6, which found their origin on other UNIX platforms, as shown in Table 5-1. But they where not specified as part of the UNIX standards.

Table 5-1 Commands from other UNIX platforms

Need addressed	Solution
Historical UNIX commands on z/OS	clear, uptime
Terminate stubborn processes	superkill option on the kill command

OMVS clear command

The **clear** command exists on most UNIX platforms, but was not specified as part of the UNIX standards. Customers porting UNIX shell scripts to z/OS UNIX have encountered the **clear** command and the resulting “not found” error message. For ease of porting, IBM supplied a downloadable clear script on the Tools & Toys Web site.

On historical UNIX platforms, **clear** is equivalent to the **tput clear** command. **tput** uses the terminal definition (terminfo) database, which sends a terminal-specific byte stream to the terminal to clear the screen, based on the value of the TERM environment variable.

On z/OS, if the user is logged in on a 3270 session (using the TSO/E command OMVS), TERM is set to “dumb”. The terminfo definition of “dumb” does not implement “clear”, so the z/OS **clear** command is designed to recognize and clear the screen in this case, without modification of the terminfo database.

The **clear** command has no options. It just clears the screen of all output and places the cursor at the top of the screen.

OMVS uptime command

To report how long the system has been running, you can now use the **uptime** command.

This **uptime** command is another command common on historical UNIX platforms, and requested by customers. It outputs how long the system has been “up” on a one-line display:

```
PATRICK @ :/u/patrick>uptime 04:06PM  up 8 day(s), 19:33,  1 users,  load average:
0.00, 0.00, 0.00
```

Note: Load averages are not supported on z/OS UNIX, and are displayed as 0.00.

5.10.1 BPXWPERM environment variable

In z/OS V1R6 it is now possible to allow for the specification of default permissions for **oedit** when run under the shell.

An environment variable, BPXWPERM, is supported by the **oedit** shell command. It specifies the default open permissions used by **oedit**. Permissions are specified in octal format. No validation is done on the supplied permissions and the number will be used as the file mode on an open() call. If the file already exists, the permissions are not changed. If the environment variable is not set, **oedit** will work as before using 0700 as the default permissions. This support is only effective for the **oedit** shell command, not the OEDIT TSO command.

5.11 Mount utility enhancements

The mount utility is enhanced with the -v verbose option. If -v is specified on the mount command and the mount fails, the file system name that had the mount failure will be included in the failure information.

5.12 USS REXX BPXWDYN enhancements

The z/OS REXX functions extend the REXX language on z/OS for z/OS UNIX tasks. One of these functions, BPXWDYN, has been enhanced.

BPXWDYN is a text interface to dynamic allocation (SVC 99) and dynamic output (SVC 109). It supports data set allocation, unallocation, concatenation, and the addition and deletion of output descriptors. It is designed to be called from REXX, but it may be called from several other programming languages, including Assembler, C, and PL/I.

Two enhancements were made to BPXWDYN:

- The capability was added to set the S99GDGNT flag. The new GDGNT keyword can be specified on an allocation request, meaning the S99GDGNT flag will be set in the S99FLAG1 field. This field enables you to specify wether or not the most recent GDG catalog information should be used. For more information on this subject, see *z/OS MVS Programming: Authorized Assembler Services Guide,* SA22-7608.
- Support for the userdata keyword in BPXWDYN, which specifies installation-specific user data for a dynamic output statement. Dynamic output allows up to 16 1-60 character strings.

5.13 Logical file system support of zFS

In z/OS V1R6 a change was made to LFS termination of a PFS, such as zFS, in order to improve the availability of file systems on the system where a PFS is terminating.

File requests are routed by the logical file system (LFS) to the appropriate physical file system (PFS) through the PFS interface, as shown in Figure 5-6, when users request access to file system data.

.

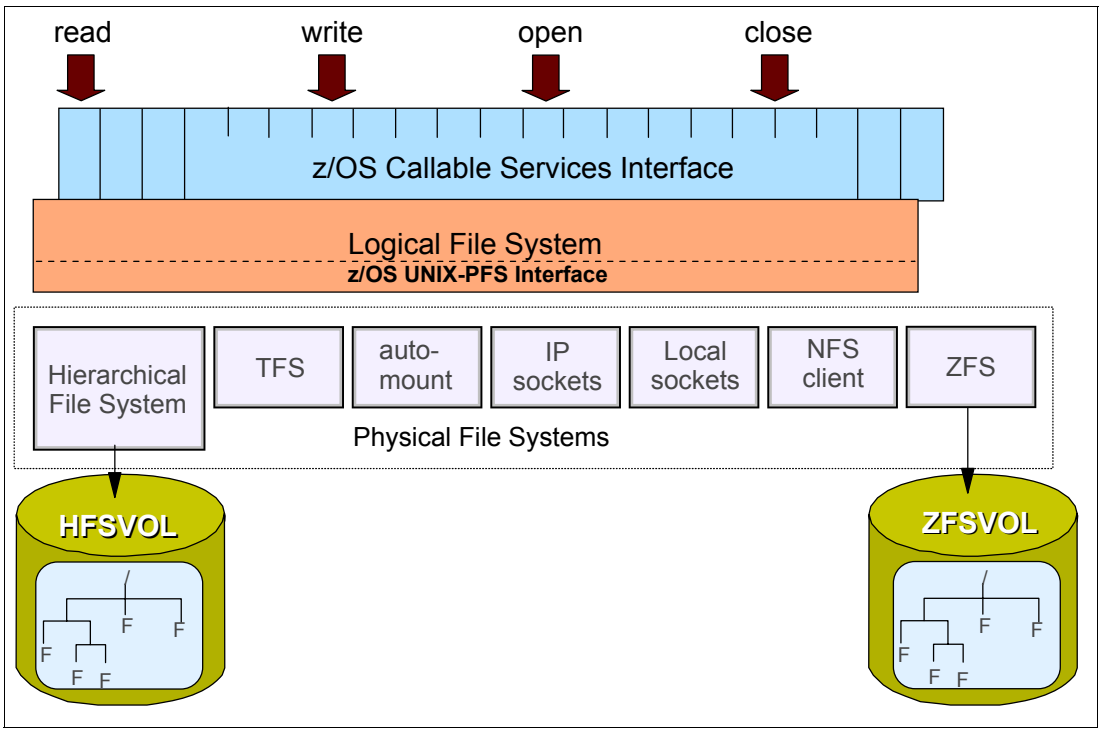

Figure 5-6 User access to file system data

5.13.1 Changes to LFS for zFS

Currently, the design of PFS termination is that file systems for the terminating PFS, and subtrees of those file systems, get moved to another system (if locally owned), and then get locally unmounted and become unavailable on the system where the PFS is terminating. If they could not be moved, then they become globally unmounted.

The new design is that if the ownership of these file systems can be moved to another system in the sysplex, then they should be moved there, and function-ship the requests and avoid the local unmounts. This allows improved availability of file systems on the system where a PFS is terminating.

If the file system is sysplex-aware (locally mounted), but not owned by the system where the PFS is terminating, then it is converted to function-shipping to the owner (no move occurs).

- If the reply to restart the PFS is `I` (do not restart the PFS), then the file systems are locally unmounted as before.
- If the reply to restart the PFS is `R`, then any sysplex-aware file systems convert back from function-shipping to local mount. Sysplex-unaware file systems remain function-shipping to the current owner.

Sysplex aware Capable of mounting locally in the systems. For example, R/O zFS file systems in V1R6.

Sysplex-unaware Not capable of mounting locally in the systems. Function ships the request to owner. For example, R/W zFS file systems in V1R6.

With the new LFS support of sysplex zFS, you can continue to access the file systems owned by the system where PFS was terminated.

Note: Each file system is moved and/or converted to function-shipping one at a time. So there is a window during which the PFS is dead and all file systems are not yet function-shipping. If the operator issues a request during this time, it will fail. This design accepts that window.

The advantages of this support are:

- Improved availability of file systems.
- Applications can continue to access the file systems via function-shipping on the system where a PFS is terminating.

Example of the new support

In Figure 5-7 on page 135 you see a USS sysplex sharing an environment with three systems that share two zFS file systems. In this example, the following is taking place:

- System SY1 owns the zFS file systems OMVS.TEST1.ZFS (R/W) and OMVS.TEST2.ZFS (R/O).
- The other two systems, SY2 and SY3, have the R/O zFS file system locally mounted as read (R/O).
- Any R/W requests from SY2 and SY3 to the R/W file system owned by SY1 must be passed through the XCF messaging function, which is referred to as function-shipping requests.

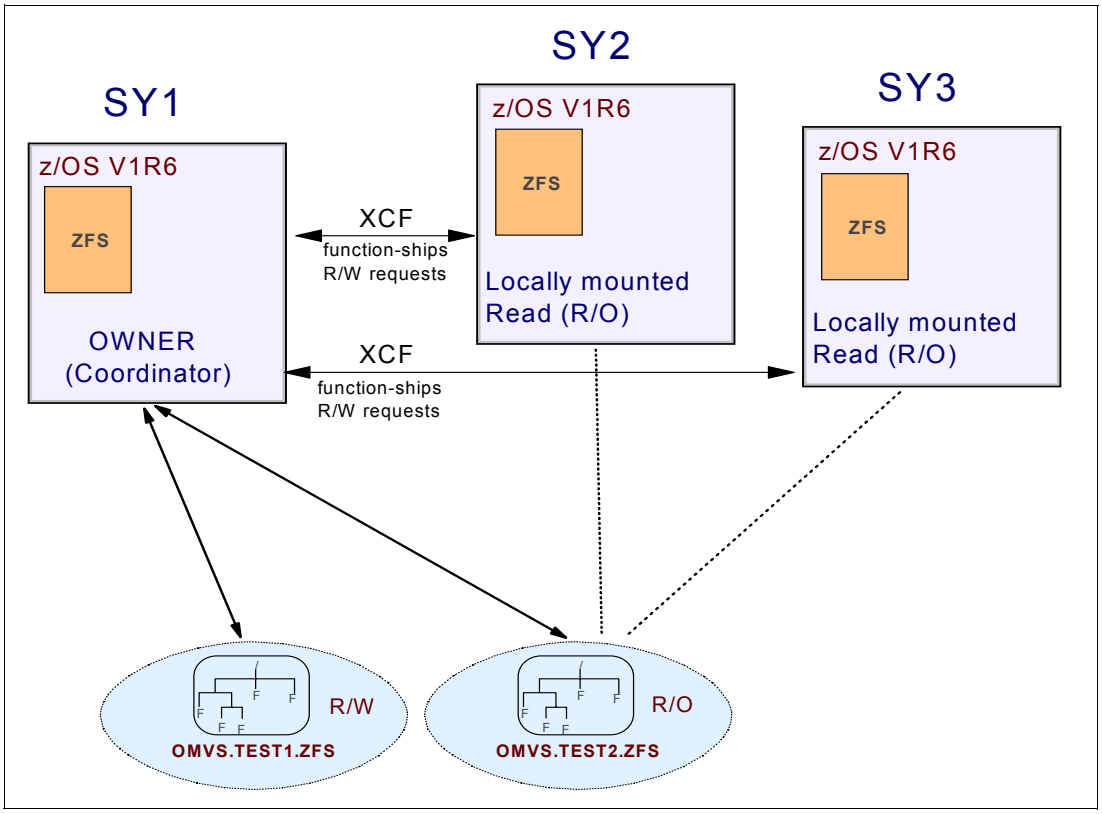

Figure 5-7 Three systems in a USS sharing an environment accessing two file systems

zFS PFS terminates

The zFS PFS can be terminated by an operator console command or by a zFS PFS abend. When the zFS PFS terminates, an attempt is made to move all the file systems owned by the terminating PFS to another system in the sysplex since they are mounted as AUTOMOVE.

When the PFS terminates, there is a system prompt message waiting for a reply in system SY1, for example:

```
*015 BPXF032D FILESYSTYPE ZFS TERMINATED.  REPLY 'R' WHEN READY TO RESTART.
REPLY 'I' TO IGNORE.
```

- If the reply to restart the PFS is `I` (do not restart the PFS), then the file systems will be locally unmounted as before.
- If the reply to restart the PFS is `R`, then any sysplex-aware file systems convert back from function-shipping to local mount. Sysplex-unaware file systems remain function-shipping to the current owner.

Example 1: Before z/OS V1R6

This example shows the processing when a zFS PFS terminates before z/OS V1R6. File systems from the terminating PFS are moved to another system in the sysplex as follows (see Figure 5-8 on page 136):

- The file systems owned by SY1, OMVS.TEST1.ZFS (R/W) and OMVS.TEST2.ZFS (R/O) are automoved to SY2. SY2 becomes the new owner.
- The other two systems, SY2 and SY3, still maintain the R/O zFS file system locally mounted as read (R/O). SY2 is the new owner of the R/O file system.

- Any R/W requests from SY3 to the R/W file system now owned by SY2 is passed through the XCF messaging function which is referred to as function-shipping requests
- All users on the SY1 system that were using the (R/O) and (R/W) file systems on SY1 no longer have any access to them.

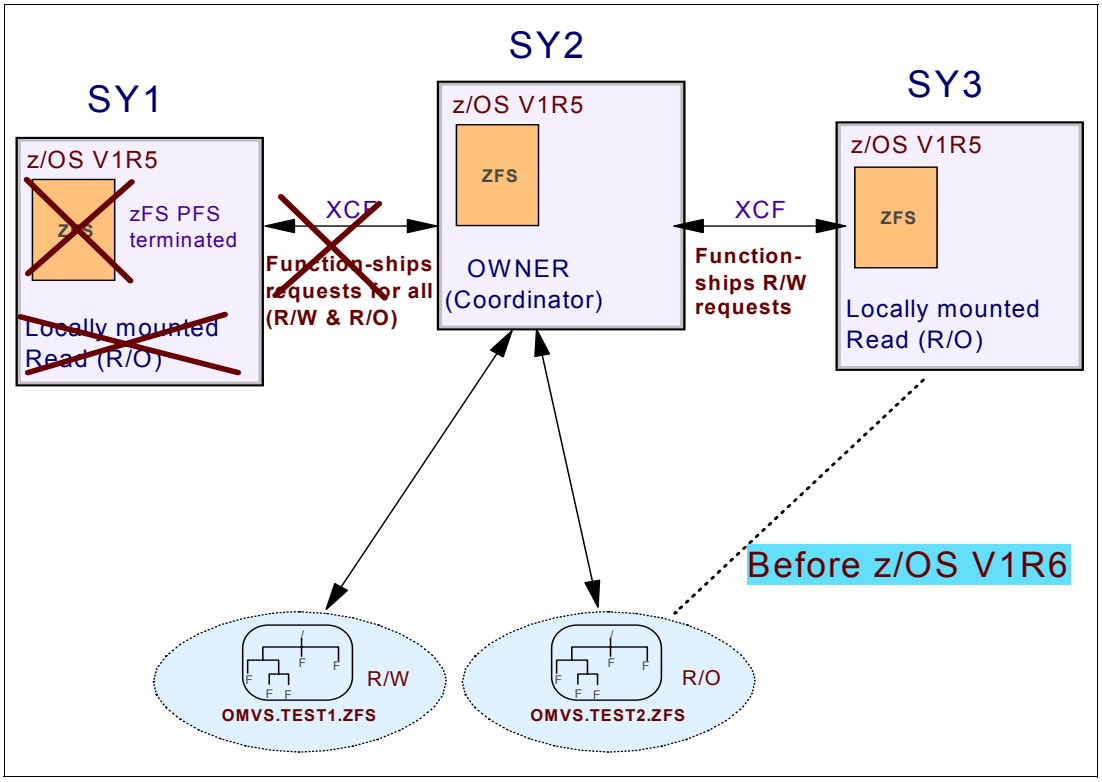

Figure 5-8 For all releases prior to z/OS V1R6 for move of file systems

Example 2: Changes with z/OS V1R6

This example shows the processing with z/OS V1R6 when a zFS PFS terminates. File systems from the terminating PFS are moved to another system in the sysplex as follows and as shown in Figure 5-8:

Figure 5-9 on page 137 shows that SY2 is now the new owner (coordinator) of the two file systems.

- The file systems owned by SY1, OMVS.TEST1.ZFS (R/W) and OMVS.TEST2.ZFS (R/O) are automoved to SY2. SY2 becomes the new owner
- The other two systems, SY2 and SY3, still maintain the R/O zFS file system locally mounted as read (R/O). SY2 is the new owner of the R/O file system.
- Any R/W requests from SY3 to the R/W file system now owned by SY2 are passed through the XCF messaging function, which is referred to as *function-shipping requests*.
- All users on the SY1 system that were using the R/O and R/W file systems on SY1 now have access to them through the XCF messaging function, which is referred to as function-shipping requests.

Note: Notice that the XCF function-shipping is also for a R/O file system.

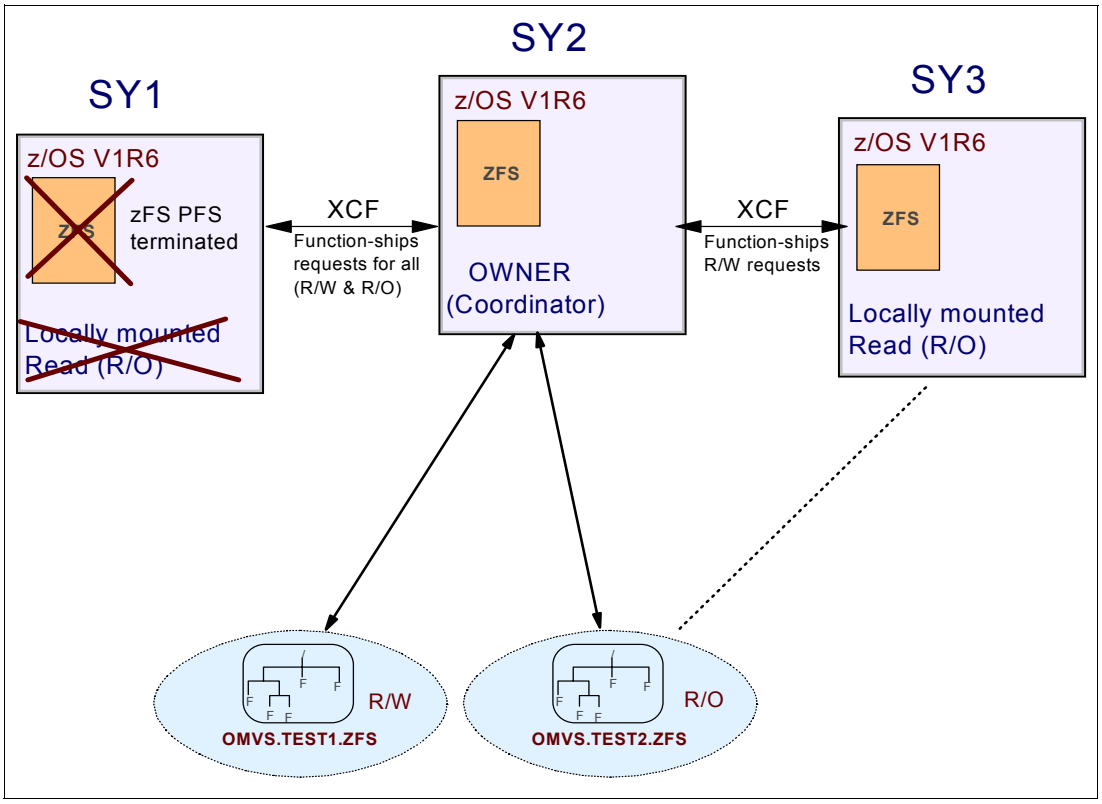

Figure 5-9 Same USS sharing environment after changing the ownership to another system

5.13.2 Automove behavior changes

For sysplex-aware file systems (R/O file systems) the behavior in automove situations is changed. It now has the following characteristics:

- MOUNT will allow AUTOMOVE(YES) or AUTOMOVE(UNMOUNT).
- If AUTOMOVE(NO) or if an automove syslist is specified, it changes to AUTOMOVE(YES) and a new message, BPXF234I, is issued:

  `BPXF234I “FILE SYSTEM OMVS.TEST1.ZFS WAS MOUNTED WITH AUTOMOVE(YES)”`

  Automove is also described in 5.5, “Automove system list wildcard support” on page 126.
- Remount will not change the AUTOMOVE setting. So a remount from R/W to R/O when the AUTOMOVE is NO will not change it to AUTOMOVE(YES), even if it is now sysplex-aware.
- PFS termination ignores AUTOMOVE(NO) or AUTOMOVE(UNMOUNT) if sysplex-aware, and tries to move ownership and then perform a local to function-ship conversion.
- Move to any systems in the sysplex (SYSNAME=*) ignores an automove syslist if sysplex-aware and considers all systems as move candidates. It has always ignored AUTOMOVE(NO) and AUTOMOVE(UNMOUNT) if sysplex-aware.
- Dead system recovery and takeover has always ignored AUTOMOVE(NO) and AUTOMOVE(UNMOUNT) for sysplex-aware, and has still attempted to have all systems try takeover. But it was honoring the automove syslist regardless of sysplex-awareness. It now ignores the automove syslist as well if sysplex-aware, and allows all systems to try takeover.

- For sysplex-aware, if no system could take it over, AUTOMOVE(UNMOUNT) unmounts the file system and its subtree, but for AUTOMOVE(NO) or syslist it becomes unowned.

Note: A move of a file system that is either AUTOMOVE(NO) or has the automove syslist to a new z/OS V1R6 owner will change to AUTOMOVE(YES), and issue BPXF234I. This will be true for manual move and file system Dead System Recovery and unowned file system takeover processing.

And although MOUNTs will now only allow AUTOMOVE(YES) or AUTOMOVE(UNMOUNT) for sysplex-aware, you can still wind up with a sysplex-aware file system with other AUTOMOVE settings by either having it mounted on a down level system, or mounted R/W and remounted R/O.

The combination of sysplex=aware and syslist will be treated as AUTOMOVE(YES): Will try to move it anywhere, and if it cannot, it will turn it into an unowned, rather than unmount it.

Prior to z/OS V1R6, automove syslist for sysplex-aware behaved like sysplex-unaware (honored the syslist and unmounted if it could not be taken over).

5.14 Distributed BRLM enhancement

You can lock all or part of a file that you are accessing for read-write purposes by using the byte range lock manager (BRLM). As a default, the lock manager is initialized on only one system in the sysplex. The first system that enters the sysplex initializes the BRLM and becomes the system that owns the manager. This is called a “centralized BRLM”.

In a sysplex environment, a single BRLM handles all byte-range locking in the shared HFS group. If the BRLM server crashes, or if the system that owns the BRLM is partitioned out of the sysplex, the BRLM is reactivated on another system in the group. All locks that were held under the old BRLM server are lost. An application that accesses a file that was previously locked receives an I/O error, and has to close and reopen the file before continuing.

You can choose to have *distributed* BRLM initialized on every system in the sysplex. Each BRLM is responsible for handling locking requests for files whose file systems are mounted locally in that system. Use distributed BRLM if you have applications that lock files that are mounted and owned locally. With distributed BRLM, each system in the sysplex runs a separate BRLM, which is responsible for locking files in the file systems that are owned and mounted on that system. Because most applications (including cron, inetd, and Lotus® Domino®) lock local files, the dependency on having a remote BRLM up and running is removed.

Running with distributed BRLM is optional. Many applications that lock files that are locally mounted will be unaffected when a remote sysplex member dies. Movement away from a centralized to a distributed BRLM will provide greater flexibility and reliability.

Distributed BRLM is enhanced in z/OS V1R6 to support moving byte range locks. Before this release of z/OS, an ENOMOVE error was presented when it was attempted to externally move a file system while an application had requested a byte range lock for a file in that file system. So previously a file system couldn’t be removed in a sysplex when an application held a byte range lock in that file system. Moving of locks was not supported with distributed BRLM. This restriction was removed with this new item. The file systems mounted in a sysplex are movable from one member of the sysplex to another member, even when locks are held in that file system.

The byte range locks are moved in the following way:

1. The target system purges any residual locks for that file system.
2. The source system unloads the file system locks and ships them to the target.
3. The target system issues a locking command for each unloaded lock.
4. The source system purges its obsolete file system locks.

Other actions are:

- The file system is quiesced during these steps.
- All owners are notified that the file system has a new system owner.
- New lock requests are now routed to the new owner.

Note: All systems in the sysplex must support moving byte range locks in order for the move function to succeed. Any system which is down level will prevent the function from occurring. An ENOMOVE error will result.

With V1R6, Distributed BRLM is the only supported byte range locking method when all systems are at the V1R6 level. The idea is to have all systems automatically move toward Distributed BRLM, which is the superior BRLM solution in a sysplex.

Byte range locks are moved under the following normal conditions:

- SETOMVS FILESYS,FILESYSTEM=,SYSNAME=,…
- /usr/sbin/chmount –d targetsys mountpath
- F BPXOINIT,SHUTDOWN=FILESYS
- F OMVS,SHUTDOWN
- Sysplex member normal termination

Note: Byte range locks are not moved when a sysplex member abnormal termination occurs or any sysplex member is down level.

5.14.1 Migration and coexistence considerations

If you are already running with an OMVS couple data set indicating distributed BRLM enabled, then there is no change required to activate the enhanced distributed BRLM for V1R6.

Also if the sysplex only has systems at the V1R6 level, then there is no change required. A z/OS V1R6 sysplex will automatically use distributed BRLM. However, if V1R6 is joining a mixed level sysplex, then the distributed BRLM needs to be enabled.

To enable distributed BRLM we recommend that you run the IXCL1DSU utility for couple data set versioning. IXCL1DSU no longer support NUMBER(0) for centralized BRLM, when running at the V1R6 level. It will only accepts NUMBER(1) as a keyword when ITEM NAME(DISTBRLM) is specified.

Note: If the customer does not set DISTBRLM NUMBER(1), a pre-release V1R6 system in the sysplex can produce an EC6-BadOmvsCds Abend any time a system enters or leaves the sysplex. An EC6-BadOmvsCds Abend is a notification Abend, indicating that DISTBRLM(0) is set when distributed BRLM is actually active; No loss of function actually occurs. USS still operates normally.

5.15 ISHELL enhancements

With the new ishell enhancements you now can:

- Use wildcard(*) filter on the directory list
- Display permissions in rwxrwxrwx format in the directory list
- Preserve the extended attributes on a copy
- Turn on and off autoskip on action list panels, such as directory list
- Stop processing multiple actions on a directory list after an action failure
- Allow executed shell commands to output in line mode as they are running
- Select to use autouid and/or autogid from RACF support
- Allow null Enter on directory list to refresh the list
- Select to not remember the last path on main panel

5.15.1 Wildcard support with filter command

If you are displaying an directory list using the ishell panels, like the one in Figure 5-10 below, you can now easily filter the displayed items.

```
  File  Directory  Special_file  Commands  Help
_ ┌──────────────────────────────────────────────────────────┐ ──────────
  │                      Enter the filter                    │
  │                                                          │
S │    *.pdf_____________________________________________    │ lect an
a │                                                          │   Select
w │                                                          │  quick
n │                                                          │
E └──────────────────────────────────────────────────────────┘
  Type  Permission  Changed-GMT       ------Size  Filename        Row 1 of 48
_ Dir    rwxr-xr-x  2004-03-04 18:28        8192  .
_ Dir    rwxr-xr-x  2003-09-04 20:18        8192  ..
_ File   rw-r--r--  2003-09-04 22:22      159744  biftags.boo
_ File   rw-r--r--  2003-09-04 22:22      655360  ephzal00.dan
_ File   rw-r--r--  2003-09-04 22:22     2392064  ephzal00.pdf
_ File   rw-r--r--  2004-03-04 18:28     1179648  ephzal01.dan
_ File   rw-r--r--  2003-09-04 22:22     1073152  ephzbl00.deu
_ File   rw-r--r--  2003-09-04 22:22     2359296  ephzbl00.pdf
_ File   rw-r--r--  2004-03-04 18:28     2150400  ephzbl01.deu
_ File   rw-r--r--  2003-09-04 22:22      229376  ephzcl00.esp
_ File   rw-r--r--  2003-09-04 22:22     4096000  ephzcl00.pdf
_ File   rw-r--r--  2004-03-04 18:28     1581056  ephzcl01.esp
_ File   rw-r--r--  2003-09-04 22:22      229376  ephzdl00.fra
_ File   rw-r--r--  2003-09-04 22:22     2981888  ephzdl00.pdf
_ File   rw-r--r--  2004-03-04 18:28      843776  ephzdl01.fra
_ File   rw-r--r--  2003-09-04 22:22      774144  ephzel00.ita
_ File   rw-r--r--  2003-09-04 22:22     1286144  ephzel00.pdf
_ File   rw-r--r--  2004-03-04 18:28      839680  ephzel01.ita
_ File   rw-r--r--  2003-09-04 22:22      774144  ephzfl00.nld
_ File   rw-r--r--  2003-09-04 22:22      839680  ephzfl00.pdf
_ File   rw-r--r--  2004-03-04 18:28      888832  ephzfl01.nld
Command ===> filter
```

Figure 5-10 Directory list

If the **filter** command is entered without any argument, a panel will be displayed to enter the new filter. A filter can specify any characters with * as the wild card character. The * can match any number of characters including no characters. The command filter *.c for example will show only files that end with .c. The filter is case sensitive. The * itself cannot be used as

a specific matching character, but is used only as a wild card character. Multiple * in a row are equivalent to a single *. When a filter is in effect, the directory list panel will indicate this following the EUID. If the filter is short enough (around 10 characters) the actual filter will be displayed, otherwise, it will just show FILTER=ON.

The filter command can also be selected via a pull-down choice on the directory list panel. This will act as though filter was entered as a command without an argument.

5.15.2 Display permissions

The directory list options panel will have a permissions choice for rwx format (See Figure 5-11). This format is also extended to the file attributes panel. On that panel the permissions will always be displayed in both formats. Change mode will not support entering new permissions in this format.

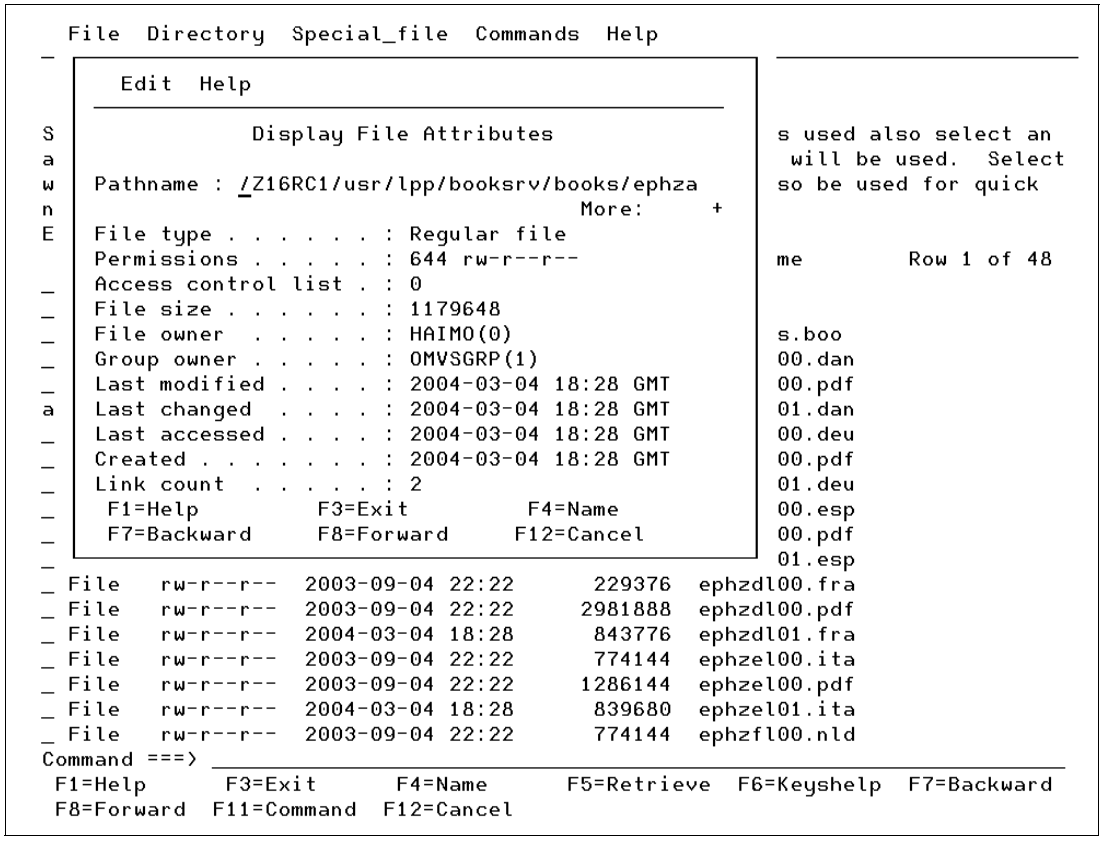

Figure 5-11 Display permissions

5.15.3 Preserve extended attributes

Individual file copy to another file (not data set) will allow selection of an option to preserve extended attributes (previously, panel just asked for permissions).

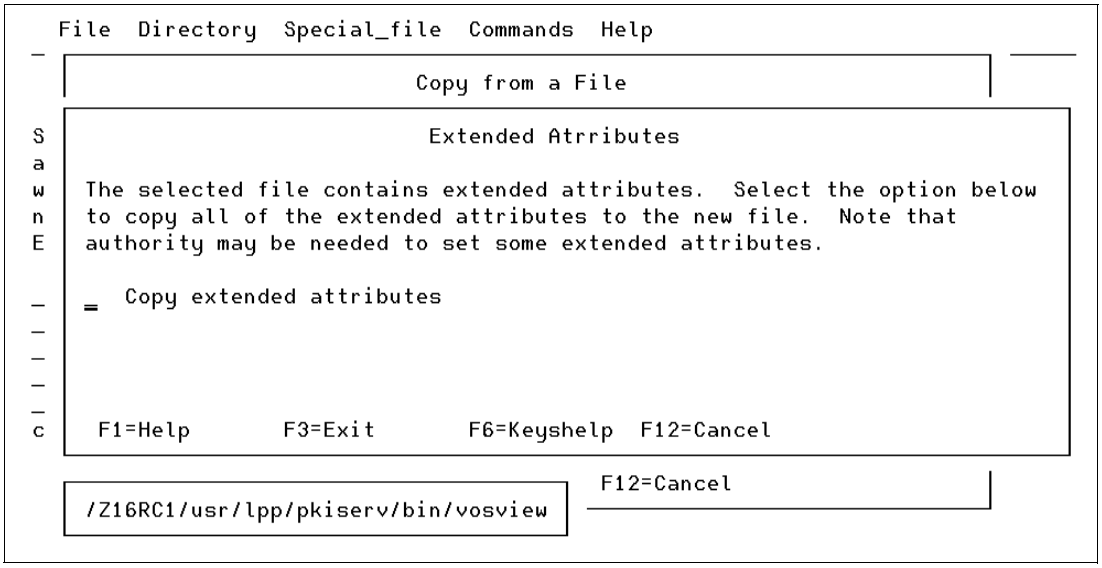

Figure 5-12 Extended attributes panel BPXWP75

5.15.4 Autoskip options

A new enhancement was added to the ishell to turn off autoskip on action list panels. This will be a new option included on the “Advanced” pull-down choice. To get to this panel, as shown in Figure 5-13, you can use:

- The **bpxishop** command and select the desired option.
- Or you can follow the pull-down menu’s from the main ishell (BPXWP99) panel. Select “Options”, number 5 “Advanced” and the desired option.

Autoskip is the characteristic when a character is entered in the one character selection column the cursor moves to the next unprotected area on the screen, usually the next selection field on the next line. With this option enabled, the cursor will move to a protected field and must be moved with a key such as tab. This only affects action list panels that have autoskip. Currently, that is just the directory list panel.

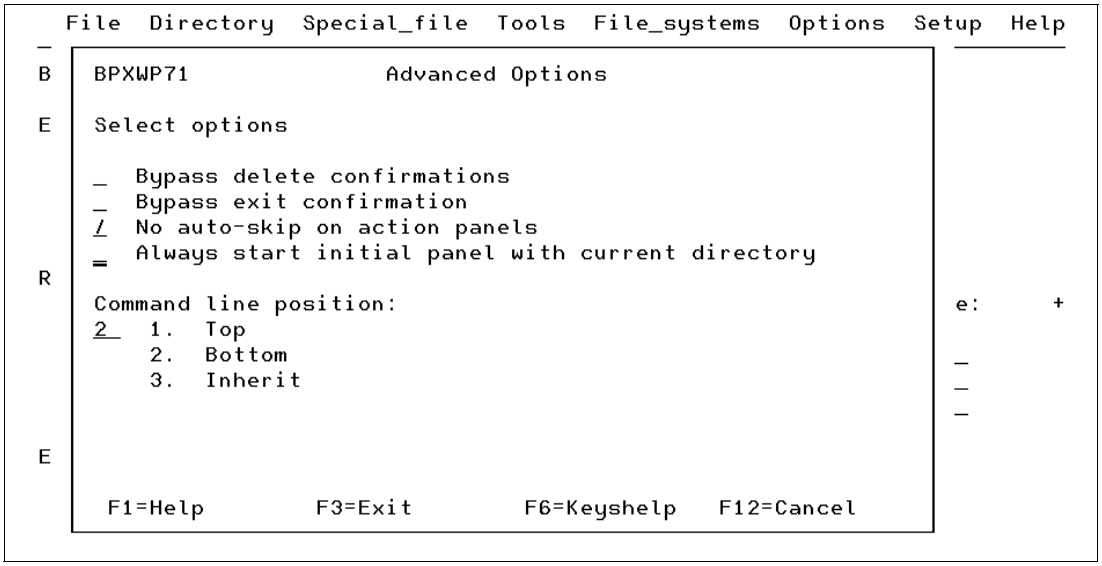

Figure 5-13 Advanced options panel BPXWP71

5.15.5 Stop multiple actions

There is a new option to stop processing multiple actions on a directory list after an action fails. This will be a new directory list option and can be enabled using either:

- The **flfield** command and select the desired option at the bottom of the panel.
- Or you can follow the pull-down menu's from the main ishell (BPXWP99) panel. Select "Options", number 1 "Directory list" and the desired option.

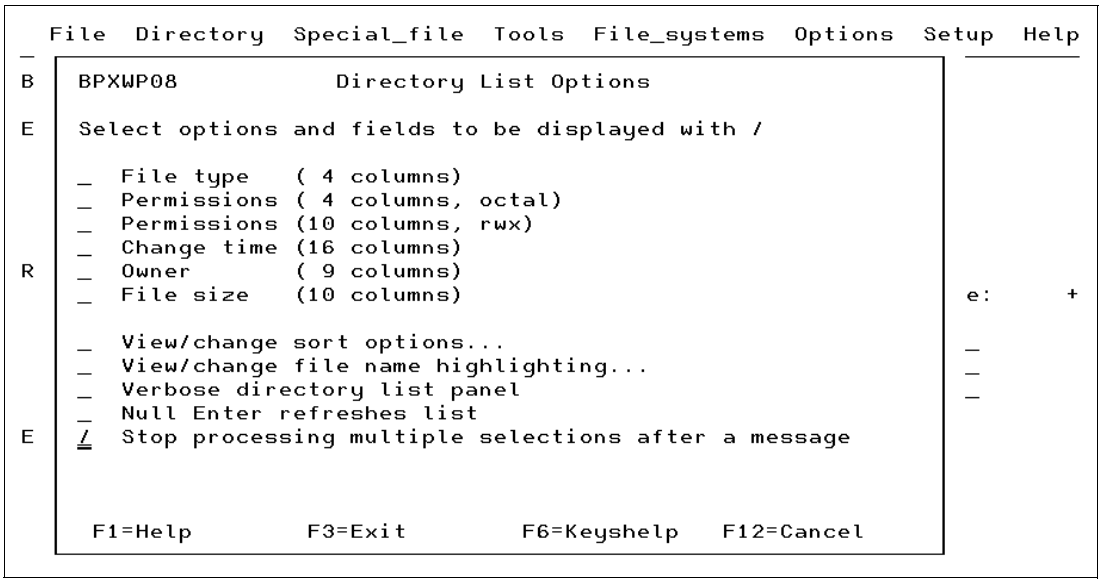

Figure 5-14 Directory list options panel BPXWP08

If enabled, when multiple actions are selected on the directory list, processing will stop if an action results in any message so that the message can be viewed. All subsequent selections will be cleared.

5.15.6 New option for executing shell commands

A new options was added, which can be used for executing shell commands. This option is added to the "Execute a Command" (BPXWP55) panel to indicate that the shell command output should be directed to the terminal in line mode. This option is ignored if the run method TSO is selected (it already works that way).

To get to this panel, as shown in Figure 5-15 on page 144, you can use:

- The **ex** command.
- Or you can follow the "Tools" pull down menu and press number 3 "Run program".

```
 File  Directory  Special_file  Tools  File_systems  Options  Setup  Help
 BPXWP55                     Execute a command

 Enter the command and select how to run it.

 Command . . . ls -la
               ____________________________________________________________
               ____________________________________________________________
               ____________________________________________________________
 Time limit    ____

 Select run method:
 3_  1.  Direct
     2.  Login shell
     3.  Login shell (line mode, no time limit)
     4.  TSO


  F1=Help        F3=Exit        F4=Name        F6=Keyshelp   F12=Cancel
```

Figure 5-15 BPXWK55 panel

5.15.7 Support for autouid/autogid

Setup for user, all users, and all groups will not assign a uid or gid if the RACF autouid or autogid support is enabled and the user requests that autouid or autogid be used. Prior to ishell assigning a uid or gid, it will show a panel that asks the user to select options to use autouid or autogid. This panel will be displayed at most one time in any ishell session. The setting is not remembered between sessions. No determination will be made by ishell as to whether all of the setup has been done to allow autouid or autogid to work or is even supported in the security product. If setup has not been done, RACF will put out an appropriate message to the terminal.

To enable this support, select the “setup” pull down menu from within the main ishell panel and press enter. Choose number 3 “*All users” to enter the BPXWP76 panel as shown in Figure 5-16 below.

Attention: The BPXWP76 panel will be shown only ones per ishell session.

```
  File  Directory  Special_file  Tools  File_systems  Options  Setup  Help
_
B  BPXWP76                Select AUTOUID and AUTOGID

E  If your system is configured to use the AUTOUID or AUTOGID capability to
   assign a new UID or GID and you want to make use of this capability,
   select the appropriate options below.  You must exit ISHELL and run it
   again to change this selection.

   Make selections and press Enter to continue
R  _  Use the AUTOUID capability
   _  Use the AUTOGID capability

    F1=Help       F3=Exit       F6=Keyshelp  F12=Cancel
```

Figure 5-16 BPXWP76 panel

5.15.8 Last path name

There is a new option that changes the behavior of the ishell panels. This option gives you the choice whether you want the ishell to remember the last path on the main panel or not. If you select the option as shown in Figure 5-17, the ishell will always prime the initial panel with the home directory. Neither the last path used, nor a path specified on the command line, is used to prime the initial panel.

Note: This behavior is similar to what it was in z/OS R1V3.

To set this option you can:

- Use the **bpxishop** command to enter the advanced option panel and select **Always start initial panel with current directory**.
- Follow the pull-down menus from the main ishell panel and select **Options**. From the Options menu, select number 5 **Advanced** and select the desired option.

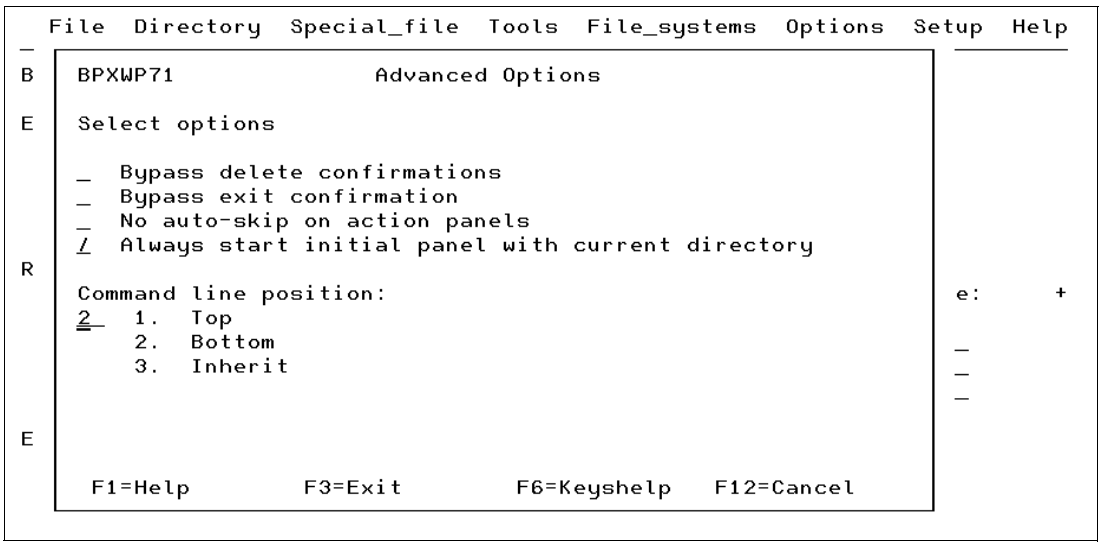

Figure 5-17 BPXWP71 advanced options panel

5.15.9 Allow null Enter

A new option is added to allow null Enter on the directory list to refresh the list. In older releases of OS/390 just pressing Enter on the directory list panel was a good way to refresh your panel. Unfortunately, this was not possible in later z/OS releases, such as z/OS V1R4.

In z/OS V1R6 you are given the choice to bring back the null Enter on the directory list panel to refresh the list. In Figure 5-18 on page 146 you see the directory list Options panel. You have to select **Null Enter refreshes list** to be able to use this feature.

To enter the directory list Options panel:

- Use the **flfield** command and select the desired option.
- Or you can follow the pull-down menus from the main ishell (BPXWP99) panel. Select **Options**, number 1 **Directory list** and the desired option.

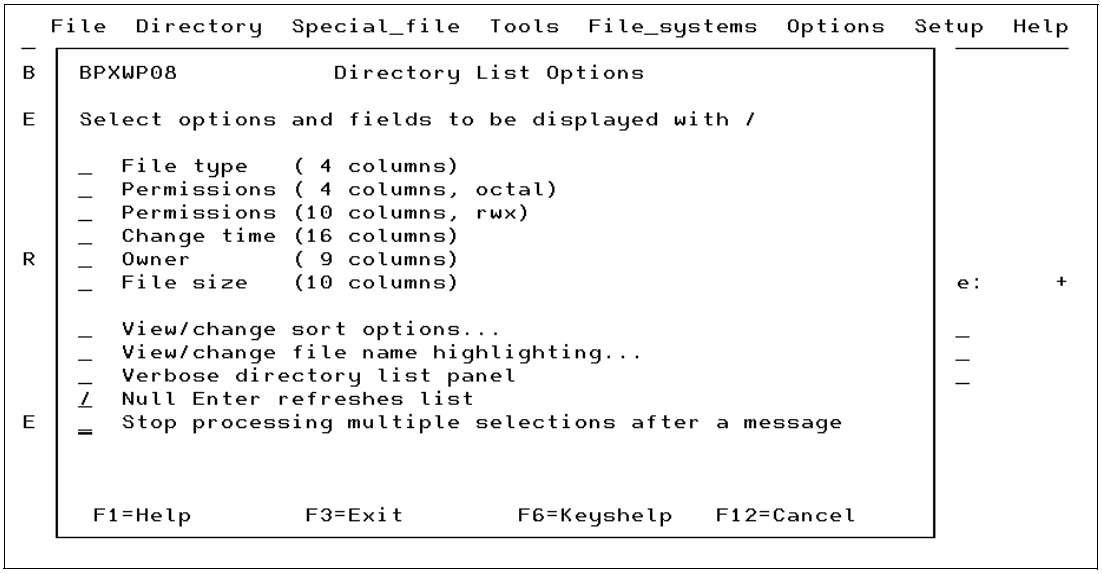

Figure 5-18 BPXWP08 directory list panel

5.16 RTLS removal

The run-time library support (RTLS) is a set of z/OS services. Language Environment uses these services. The z/OS services related to RTLS are not affected by this Language Environment change and are not being removed from z/OS V1R6.

RTLS is used to aide migration to Language Environment from previous run-times. It allows applications to specify a particular level of Language Environment run-time to be used.

5.16.1 Specifying run-time options

Each time your application runs, a set of run-time options must be established. These options determine many of the properties of how the application runs, including its performance, error handling characteristics, storage management, and production of debugging information. Under batch, you can specify run-time options in any of the following places where the installation default options are:

- The CEEDOPT CSECT
- The CEEROPT CSECT
- The CEEUOPT CSECT
- The CEEUOPT CSECT where user-supplied default options
- #pragma runopts in C/C++ source code
- A PLIXOPT string in PL/I source code
- The PARM parameter of the EXEC statement in your JCL
- In z/OS, on the GPARM parameter of the IBM-supplied cataloged procedure
- The assembler user exit
- The _CEE_RUNOPTS environment variable, when your application is running under z/OS UNIX and is invoked by one of the exec family of functions

5.16.2 Migrating without RTLS

Language Environment controlled its use of RTLS services with three run-time options that are being removed:

- LIBRARY
- RTLS
- VERSION

These run-time options are removed from the options reports generated by RPTOPTS(ON), CEEDUMP, and the IPCS verb exit. The CEEXOPT macro has been updated to prevent the use of these run-time options when building new CEEDOPT, CEECOPT, CEEROPT, or CEEUOPT CSECTs. Existing CEECOPT and CEEDOPT members that contain these run-time options must be modified to remove them if these run-time options are encountered in existing CEEROPT or CEEUOPT CSECTs.

For z/OS UNIX applications, this requires changes to CEEDOPT and CEECOPT usermods at z/OS 1.6 installation. This means that existing CEEUOPTs or CEEROPTs that specified any of the above run-time options now produce the following informational message:

```
CEE3611I The run-time option option was an invalid run-time option or is not
supported in this release of Language Environment.
```

To avoid this message, the CEEUOPTs need to be reworked and relinked. The SCEERTLS data set will no longer be shipped by Language Environment. Any JCL that references this data set needs to be updated.

Some user applications might continue to require a lower level of Language Environment than that shipped with z/OS 1.6. Applications can continue to use STEPLIB to access a lower level of Language Environment. However, z/OS elements require the use of the level of Language Environment delivered with the operating system.

5.17 64-bit support

This section discusses the enhancements that were made to z/OS V1R6 for 64-bit support. We talk about the kernel enhancements and the shells and utilities support for 64-bit.

5.17.1 UNIX System Services 64-bit kernel support

For the USS kernel to support 64-bit, about 10% of the syscall interface has been changed. In fact, the total number of syscalls remains basically the same. Changes have to do with:

- Parms with addresses and lengths that were expanded
- New parm added in cases where RV returned an address

Because of better maintenance capabilities, we now have a bimodal kernel (31/64 bit). 99% of the USS kernel is still running in 31-bit mode, indicating that 64-bit program addressing is mapped under the covers.

Note: Syscalls that are not being supported in 64-bit mode are those that have been replaced in functionality by other syscalls. BPX1GPS has been replaced by BPX1GTH, BPX1TYN has been replaced by BPX2TYN, and so on. The kernel will continue to support the BPX1... versions of these syscalls.

Converting applications to 64-bit

The syscall stub and a portion of the syscall layer are entered in the AMODE of the caller so that the callable service table can run in either AMODE 31-bit or 64-bit.

LE will always just run in one mode per process image. LE will provide two libraries. The 64-bit version will run amode64 and invoke the 64-bit kernel services. The 31-bit version will run as today. Converting a 31-bit C program to 64-bit calls for a recompile and a relink.

.

Attention: For applications written in the assembler programming language to run in 64-bit mode requires that all invocations of BPX1... kernel services need a code change.

There are some differences between the syscall layers of 31-bit and 64-bit mode. Check the z/OS V1R6 version of *z/OS UNIX System Services Programming: Assembler Callable Services Reference,* SA22-7803 for more information about this subject.

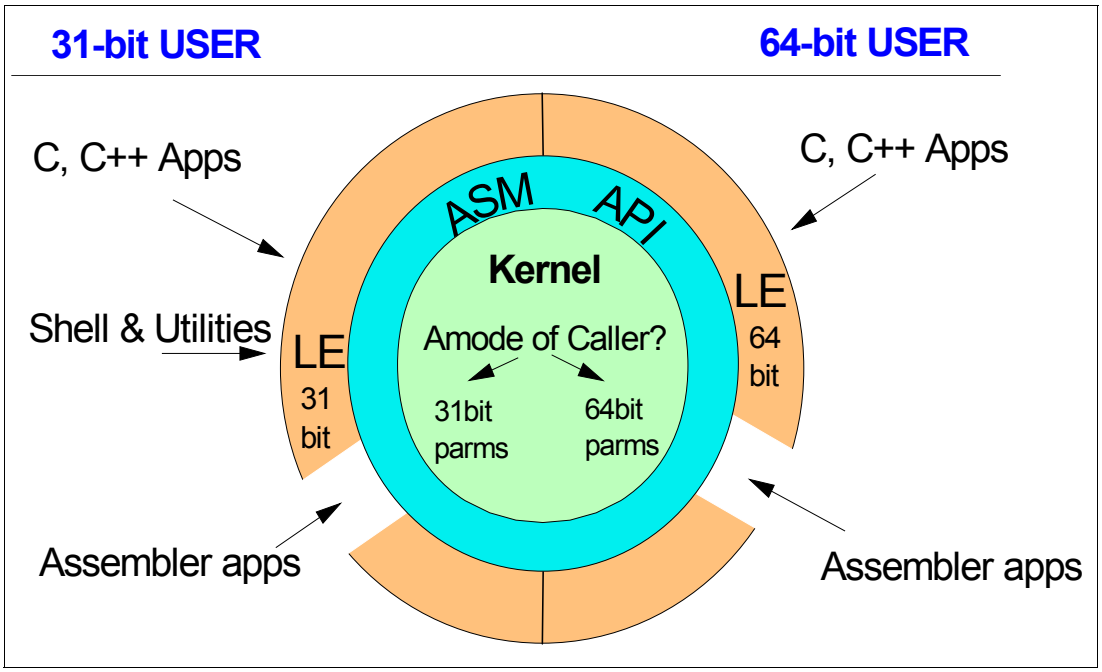

Figure 5-19 Applications can use both AMODE 31 and AMODE 64

5.17.2 Shells and utilities support for 64-bit

For 64-bit virtual addressing support, no shells or commands need to be run with 64-bit AMODE/RMODE. However, changes were made to the utilities to support both 31-bit and 64-bit application development. Table 5-2 provides an overview of the changed utilities.

Table 5-2 Changed UNIX utilities to support 64-bit

Need addressed	Solution
UNIX commands to build 64-bit applications	`ar, nm, file, lex, yacc`
UNIX commands to display 64-bit resources	`ulinit / limit / unlimit, ps, ipcs`

Archive libraries

The **ar** utility maintains archive libraries, which are a collection of files, usually object files (output from compile). These archive libraries are then input to the linkage editor to statically link functions in the archive libraries with object files that reference the library functions. For the linkage editor (including the binder, the pre-linker, and IPALink) to perform this autocall resolution, the archive libraries contain a symbol table member, named "__.SYMDEF". While duplicate entries are currently allowed in the symbol table, it is sorted such that the linkage editor doing a sequential search through the table will find the most recent entry first.

For example, to add or replace two members to the libfun archive, use the following command:

```
ar -ruv libfun.a math.o symbol.o
```

New in z/OS V1R6 is that you can store multiple versions of the same object file in one archive; see Figure 5-20. This will overcome the problem where you have to make separate archives to supply 31-bit and 64-bit versions of the same library. The **ar** utility stores the attributes in the archive and the archive can contain, for example, three versions of an object:

myfunc.o AMODE(31), non-XPLINK

myfuncX.o AMODE(31), XPLINK

myfunc64.o AMODE(64) (AMODE(64) always forces XPLINK)

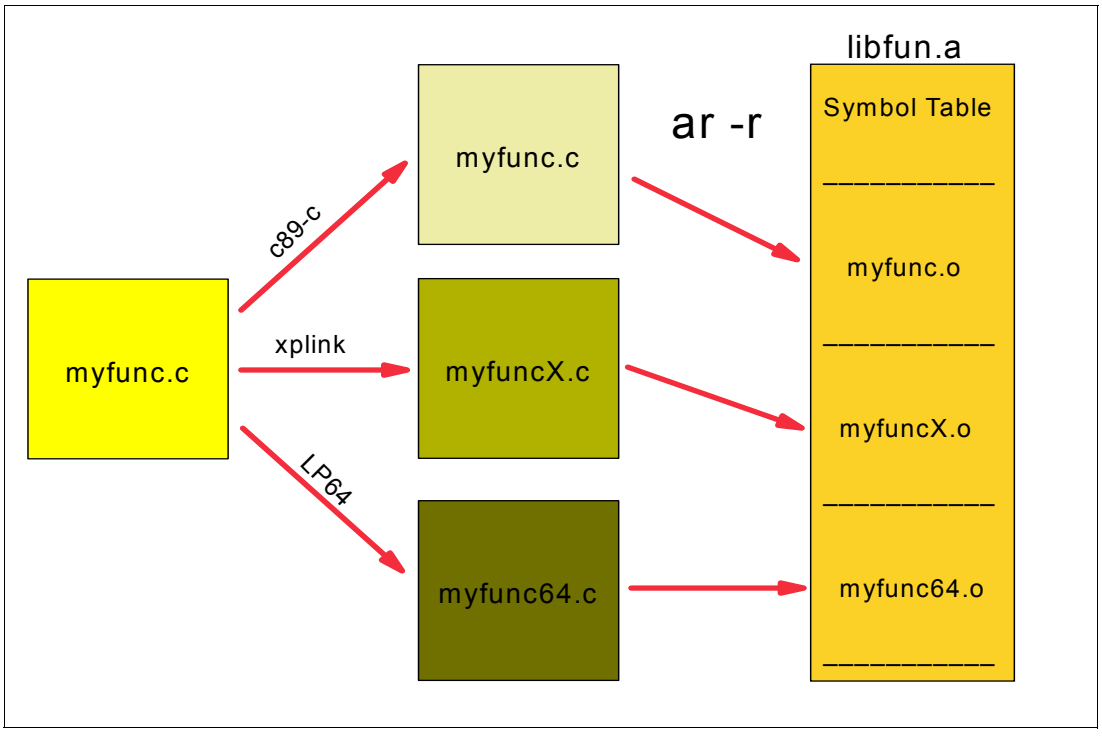

Figure 5-20 Archive example

Display symbols

The **nm** command is used for checking object files and archive libraries for external symbols when the link-edit reports unresolved references. You can display symbols in an:

- Object (.o)
- Archive library (.a)
- Executable

New to z/OS V1R6 is the **-M** option to display RMODE, AMODE and compiler options such as XPLINK and IPA. This is useful for checking whether the symbols have the correct attributes; see Figure 5-21.

```
>nm -M /usr/lib/libl.a
yyerror.o:
          0 D --- --- -    @@DOPLNK
          0 D ANY ANY -    @@INIT@
          0 U  24  24 -    @@XINIT@
          0 U ANY ANY -    CEESG003
          0 U ANY ANY -    CEESTART
          0 D ANY ANY -    FSUMSYYR
          0 U ANY ANY -    FSUSSYYR
          0 D ANY ANY -    FSUSSYYR
          0 U ANY ANY -    fprintf
          0 U ANY ANY -    yyerror
         48 T ANY ANY -    yyerror
```

Figure 5-21 Partial output example of the display symbol command

File type command

The **file** command makes a guess at the file type by examining the named files. It determines the file type by its attributes and by reading the beginning of the file.

In z/OS V1r6 the **file** command also determines the addressing mode (31 or 64) of executable files; see Figure 5-22.

```
>file /bin/pax
/bin/pax:       z/OS Unix executable (amode=31)
```

Figure 5-22 The file command

Note: In case the AMODE check is not desired, the **-E** option bypasses it and uses the magic file (/etc/magic) templates. See Figure 5-23.

```
>file -E /bin/pax
/bin/pax:       OS/390 UNIX executable
```

Figure 5-23 The file command with the -E option

Utilities for writing parsers

The shell commands **lex** and **yacc** can be used to build application parsers. They generate C source code, which is subsequently compiled and link-edited to build the application.

Note: In z/OS V1R6 there are no changes to the **lex** and **yacc** utilities themselves.

z/OS includes archive libraries libl.a and liby.a, which contain collections of compiled object files with functions used by **lex** and **yacc**. These libraries are searched when the parsing application is link-edited. The new archives in z/OS V1R6 contain both 31-bit and 64-bit compiled object files. See also the new **ar** support in “Archive libraries” on page 149.

Set or display resource limits on processes

The command to set or display resource limits on processes created under the current shell differs between the shell types.

ulimit **sh** built-in command

limit / unlimit **tcsh** built-in command

Both control the current shell's process. The limits are inherited by child processes (for example, regular commands invoked from the shell).

There are two new options for the **ulimit** command:

-M Set or display the amount of storage above the 2 GB bar that a process is allowed to have allocated. Memlimit is specified as the number of megabyte increments.

-A Set or display the maximum address space size for the process, in units of 1024 bytes. If the limit is exceeded, storage allocation requests and automatic stack growth will fail. An attempt to set the address space size limit lower than what is already used will fail.

```
PATRICK @ :/u/patrick>ulimit -a
core file         8192b
cpu time          unlimited
data size         unlimited
file size         unlimited
stack size        unlimited
file descriptors  65535
address space     136168k
memory above bar  17592186040320m
```

Figure 5-24 ulimit command

The **ulimit -a** command shows all the current limits, as shown in Figure 5-24.

Notes:

- Superusers can set "hard" limits with **-H** as an upper bound on all users.
- Users can set "soft" limits (default or **-S**) only up to the hard limit.
- System parms (BPXPRMxx) control some of the resources and will be reported as the limit values, if not overridden.

For the tcsh shell commands **limit** and **unlimit** there are two new resource names:

memlimit Maximum storage allocation above the 2Gig bar (in megabytes)

addressspace Maximum address space size (in kilobytes)

The following command limits the above-the-bar use to 25 GB:

```
limit memlimit 25g
```

Status of z/OS UNIX processes

The **ps** command shows the status of z/OS UNIX processes; see Figure 5-25. The format specifications of **ps** allow many optional fields to be displayed. Two new fields related to storage above the bar were added in z/OS V1R6, one for the limit and one for the current allocation:

vszlmt64 Maximum virtual storage above the 2Gig bar

vsz64 Virtual storage used above the 2Gig bar

```
PATRICK @ :/u/patrick>ps -o comm,vsz64,vszlmt64
COMMAND        VSZ64   VSZLMT64
/bin/sh            0          0
/loop_64         100       100G
```

Figure 5-25 ps command

Both new output fields are displayed with possible multiplier abbreviations:

space	multiplier
K	Kilo
M	Mega
G	Giga
T	Tera
P	Peta

UNIX interprocess communication status

The **ipcs** command shows the status of UNIX interprocess communication resources.

Before z/OS V1R6, **ipcs** output SEGSZPG showed the segment size in pages. The size of these pages was always 4K. In addition to SEGSZPG, the following were added in z/OS V1R6:

SEGSZ Segment size in bytes of the shared memory

PGSZ Page size in 4K or 1M

There is also a new z/OS extension, the **-y** option on the **ipcs** command, which gives a *summary system limit status*, including the following fields:

TPAGES The system limit for the number of system-wide shared memory pages (only for 31-bit (below the bar) requests)

SPAGES The system limit for the number of pages per shared memory segment

SEGPR The system limit for the number of segments per process

CPAGES The current number of system-wide shared memory pages

MAXSEG The largest number of shared memory pages allocated to a single shared memory segment

6

z/OS V1R6 RMF

In the recent past, RMF has provided enhancements on an alternating release schedule. Because significant enhancements were made available with z/OS V1R5, there are no enhancements to z/OS V1R6. However, two Special Programming Enhancements (SPE) are now available for z/OS V1R2 and above that do provide enhanced functionality to support IFA processors and ESS customers.

In this chapter, we describe the RMF changes in support of:

- IFA processing units
- New ESS support

6.1 IFA processing units

Integrated Facility for Applications (IFA) is a new type of special-purpose processing unit available for zSeries machines. It is similar to other types of special-purpose processing units, such as the Integrated Coupling Facility (ICF) and the Integrated Facility for Linux (IFL). The primary difference between the IFA and others is that the z/OS operating system recognizes the IFA as a new type. IFAs cannot be IPLed and they do not process interrupts. Additionally, in some circles the IFA may also be referred to as a Java Assist Facility (JAF).

The IFA, also known as the IBM zSeries Application Assist Processor (zAAP), is available on the IBM zSeries 990 (z990) and zSeries 890 (z890) servers. It is an attractively priced specialized processing unit that provides a strategic z/OS Java execution environment for customers who desire the powerful integration advantages and traditional Qualities of Service of the zSeries platform.

6.1.1 RMF IFA support

RMF support has been enhanced via SPE to support IFA processors by extending the statistical reporting available in the following:

- RMF Postprocessor CPU Activity report
- RMF Postprocessor Workload Activity report
- RMF Monitor III Enclave report

In general, RMF distinguishes between regular CP and IFA processing units where appropriate to do so. It will also collect statistics and report IFA service times. For WLM service class and report class periods, IFA using and delay states will be collected and reported.

RMF CPU Activity report

Total and average lines are created for each pool of processing units. The I/O Total Interrupt Rate and % I/O Interrupts Handled VIA TPI are only applicable to standard CPs.

On the CPU Activity report, the following changes support zAAP processing:

- The CPU section is grouped per processor type.
- A new TYPE column indicates whether the processor belongs to the pool of standard CPs or IFA (zAAP) processors. In the testing we did, there were one standard CP and two zAAPs, as shown in Figure 6-1 on page 155.
- The last two columns are only available for standard CPs—not for zAAPs—because zAAPs are disabled for I/O interruption.
- A TOTAL/AVERAGE line is printed per pool.

Figure 6-1 on page 155 shows an example of an updated CPU Activity report that presents IFA processing unit statistics.

```
                                              C P U  A C T I V I T Y
                                                                                            PAGE 1
            z/OS V1R5          SYSTEM ID SC64           DATE 03/23/2004         INTERVAL 09.59.998
                               RPT VERSION V1R5 RMF     TIME 15.50.00          CYCLE 1.000 SECONDS

CPU  2084  MODEL  316

---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       69.41      69.41      011511      58.67              0.00
  1    CP    100.00       70.75      70.75      111511      233.6              0.00
  2    CP    100.00       68.40      68.40      211511      254.2              0.00
  3    CP    100.00       63.64      63.64      311511      63.49              0.00
  4    CP    100.00       67.74      67.74      411511      1380               0.01
CP TOTAL/AVERAGE          67.99      67.99                  1990               0.01

  8    IFA   100.00       39.41      39.41      811511  <-----------
  9    IFA   100.00       40.75      40.75      911511
IFA AVERAGE               40.08      40.08
```

Figure 6-1 CPU Activity report

On the Partition Data section of the CPU Activity report, logical IFA processors are grouped and reported together with the ICF processor pool. This is due to the fact that the hardware recognizes ICFs, IFLs, and IFAs as a single pool of resources.

RMF Workload Activity report

To produce this report, specify:

```
WLMGL(RCPER(option))
```

The Resource Consumption section of the WLMGL report is extended by the following zAAP-related fields:

- The IFA is the zAAP processor service time in seconds. It does not account for the zAAP-eligible work executed on regular CPs.
- APPL% IFA is the percentage of CPU time executed on zAAP processors.
- APPL% IFACP is the percentage of CPU time used by zAAP-eligible work on regular CPs. It is also reported by the IWMRCOLL service. It allows the customer to assess whether additional zAAP processors might be necessary to run all zAAP-eligible work.
  - If crossover is turned on (IFACrossOver= YES) on the IEAOPTxx parmlib member, and honor priorities are turned on or defaulted in the IEAOPTxx=, the CP time was accumulated at priority.
  - If crossover is on and honor priorities are turned off, the CP time for zAAP-eligible work was accumulated after discretionary service class periods were dispatched.
  - If crossover is turned off (IFACrossOver=NO), no zAAP-eligible work runs on CPs. This percentage should be zero.

```
REPORT BY:_POLICY=WLMPOL                         REPORT CLASS=RWASTCZL
                                                 DESCRIPTION =Report class for zaapload light

TRANSACTIONS    TRANS.-TIME  HHH.MM.SS.TTT   --DASD I/O--   ---SERVICE----   --SERVICE TIMES--   PAGE-IN RATES    ----STORAGE---
AVG       9.77  ACTUAL                 175   SSCHRT    0.0  IOC          0   TCB        1599.7   SINGLE     0.0   AVG        0.0
MPL       9.77  EXECUTION              175   RESP      0.0  CPU     34967K   SRB           0.0   BLOCK      0.0   TOTAL      0.0
ENDED    33489  QUEUED                   0   CONN      0.0  MSO          0   RCT           0.0   SHARED     0.0   CENTRAL    0.0
END/S    55.81  R/S AFFINITY             0   DISC      0.0  SRB          0   IIT           0.0   HSP        0.0   EXPAND     0.0
#SWAPS       0  INELIGIBLE               0   Q+PEND    0.0  TOT     34967K   HST           0.0   HSP MISS   0.0
EXCTD        0  CONVERSION               0   IOSQ      0.0  /SEC     58275   IFA        1085.3   EXP SNGL   0.0   SHARED     0.0
AVG ENC   9.77  STD DEV                 17                                   APPL% CP     85.7   EXP BLK    0.0
REM ENC   0.00                                              ABSRPTN   5963   APPL% IFACP  65.7   EXP SHR    0.0
MS ENC    0.00                                              TRX SERV  5963   APPL% IFA   180.9
                                            W O R K L O A D   A C T I V I T Y
                                                                                                                  PAGE    3
     z/OS V1R5             SYSPLEX SANDBOX          DATE 07/30/2004          INTERVAL 10.00.049    MODE = GOAL
                       CONVERTED TO z/OS V1R5 RMF   TIME 15.50.00
```

Figure 6-2 Service time calculations

As illustrated in Figure 6-2, service times are calculated as follows:

- APPL% CP is the percentage of CPU time used by non-zAAP-eligible work plus zAAP-eligible work (APPL%IFACP) running on regular CPs:

  ```
  APPL% CP = (TCB + SRB + RCT + IIT + HST – IFA_normalized) x 100 / interval
  length
  ```

  - The TCB service time is the time spent on regular CPs as well as zAAP processors. It is calculated from the total CPU service units (R723CCPU) together with the CPU service coefficient (R723MCPU) and CPU adjustment factor (R723MADJ).
  - The zAAP time may be normalized for z890 models, where regular CPs (capped) and zAAP processors (not capped) run at different speeds. Thus, a “normalized” zAAP time portion has to be removed.

  Note: If no zAAPs are configured, N/A is shown for the new fields.

- To calculate the percentage of non-zAAP-eligible work, subtract the value of IFA from a CP, as follows:

  ```
  non-zAAP-eligible work % = APPL%CP - APPL%IFACP = 85.7 - 65.7 = 20%
  ```

Goals and Actuals section

The Goals and Actuals section of the Workload Activity report has been changed as well. The changes are:

- The Goals section has been formatted on a separate line for readability.
- The USING% block includes IFA Using.
- IFA delays may appear in the EXECTUTION DELAY% block if it is among the highest contributors to the TOT delay samples.

An example of the new Actuals section of the Workload Activity report follows:

```
RESPONSE TIME GOAL:  00.00.01.000 AVG

ACTUALS: RESPONSE  EX    PERF AVG   ----USING%---  --------- EXECUTION DELAYS % -------  ---DLY%--  -CRYPTO%-  ---CNT%--  %
           TIME    VEL%  INDX ADRSP CPU IFA I/O    TOT CPU IFA I/O AUX AUX SWIN UNKN IDLE USG DLY  USG DLY QUI
         HH.MM.SS.TTT                                          VIO  PRIV
*ALL     00.00.01.854 31.8  1.1  5.7   3.6 2.2 2.6   13.4 8.5  4.3 0.3 0.2 0.1  0.1  58.1 22.7 1.1  3.1  0.0  0.0  0.0
LP1      00.00.01.999 30.5  1.3  2.1   3.5 2.1 2.3   13.5 9.0  4.2 0.2 0.1 0.0  0.1  60.3 20.5 0.2  1.1  0.0  0.0  0.0
LP2      00.00.01.001 49.4  1.0  1.9   3.3 2.0 3.1    6.8 1.8  4.3 0.3 0.1 0.0  0.0  63.1 24.0 1.4  4.1  0.0  0.0  0.0
LP4      00.00.01.003 24.1  1.0  1.8   3.9 2.3 2.3   10.1 14.8 4.3 0.3 0.1 0.4  0.1  50.3 23.7 0.3  0.3  0.0  0.0  0.0
```

RMF Monitor III support

The collection and reporting capabilities of RMF III have also been enhanced to provide support for IFA processors. These enhancements include the following details:

- The definition of CPU UTIL% in the SYSINFO and Workflow Exception report is changed so that only standard CP details are included.
- Processor Using and Delay samples include data from both standard CPs and IFA processors. This impacts the USING%, DELAY%, and WORKFLOW% metrics.
- The ExecVelocity% in SYSSUM reports includes IFA using and delay information.
- The APPL% and EAPPL% metrics in the Sysinfo and Enclave report contain both standard CP and IFA processors.

RMF Monitor III Enclave report

The Enclave Details panel has been updated to include IFA Using and Delay states. Additionally, the Total and Delta enclave CPU time reported includes CPU time for IFA processors. The following is an example of the Enclave Report screen:

```
RMF Enclave Classification Data

The following details are availabe for enclave ENC00003:
Press Enter to return to the Report panel.

 Detailed Performance Statistics:

 - CPU Time -    ------------------ Execution States ------------
Total     26.78   #STS   -- Using --     ----------- Delay --------              IDL UNK
Delta     22.50          CPU IFA  I/O   CPU  IFA  I/O  STO  CAP  QUE
                   592   12  1.0  0.0    88  0.0  0.0  0.0  0.0  0.0    0.0  0.3
```

New overview conditions

The following new overview conditions have been added to provide support for IFA processing units:

Overview Condition	Description
NUMIFA	Number of IFA processors
IFABSY	IFA processor busy
IFASEC	IFA service time
IFACPSEC	IFA service time spent on CPs
APPLIFA	IFA application execution time percentage
APPLIFCP	IFA on CP application execution time percentage
IFAUSGP	IFA using percentage
IFCUSP	IFA on CP using percentage
IFADLYP	IFA delay percentage

6.1.2 SMF record changes

SMF type 30 and type 72 records have been enhanced to provide zAAP usage information:

- SMF 30 reports the amount of standard CP time consumed by the job step and the amount of zAAP-eligible time consumed by the job step executing Java on standard CPs, if any.
- For SMF 72 records, the amount of time spent executing on zAAP processors is reported, as well as Using and Delay sample counts for zAAP-eligible work.

When running the same Java workload with zAAPs as you were running before without zAAPs, you should expect to see less capacity shown in your Sub-Capacity Reporting Tool

(SCRT) reports, as well as less capacity used for standard CPs in your RMF reports. If new Java workload has been added, this increases CP usage.

Refer to *z/OS MVS System Management Facilities (SMF),* SA22-7630 for more information on SMF type 30 and type 72 records. Refer to *z/OS Resource Measurement Facility User's Guide,* SC33-7990 for more information on RMF monitoring.

Note: The diagnosis tools and service aids that you use today, for example SLIP traps and traces, can be used unchanged with zAAPs.

The following SMF record types have been extended to provide support for IFAs:

- SMF record type 70 subtype 1 - CPU Activity
- SMF record type 72 subtype 3 - Workload activity
- SMF record type 79 subtype 1 and subtype 2 - Address space state and Resource data

6.1.3 Special programming enhancement details

The SPE providing the IFA support for RMF is shipped as RMF APAR OA05731.

6.2 ESS support

RMF support for the Enterprise Storage Server (ESS) has been enhanced with a new report that makes available ESS link Performance Statistics. Additionally, a new SMF record has been created for ESS statistics. This support is provided by way of a small product enhancement (SPE) APAR, which is available for z/OS V1R2 and later.

6.2.1 Enhanced reporting

The new ESS Activity report provides link performance statistics for each ESS adapter for:

- SCS/ I/O
- PPRC I/O

The statistics include data from each link in the entire ESS and give a "box-wide" view of the performance. While similar statistics are available in the Cache report, they give a view of the ESS itself.

In order to generate the data necessary to produce the ESS report, a new Monitor I data gatherer option has been supplied. The option is:

```
ESS | NOESS
```

Specifying ESS enables the capture of ESS link measurement data. This data is written to the new SMF type 74 subtype 8 records supplied with this SPE.

Specifying NOESS suppresses the gathering of the link measurement data.

It should be noted that when several systems have access to the same storage subsystem, only one system needs to be started with ESS data gathering enabled. Enabling ESS data gathering in more than one system will simply produce duplicate SMF records.

The RMF Postprocessor REPORTS option has also been updated with this SPE. To Generate the Postprocessor ESS report, specify the following Reports option:

REPORTS(ESS)

This results in the creation of a report similar to the following example, but hopefully with more meaningful link statistics:

```
SERIAL NUMBER 0000022010  TYPE-MODEL 2105-E20          CDATE  07/11/2003    CTIME 17.45.00     CINT  15.0

------ ADAPTER ------   -- LINK TYPE --  MBYTES    MBYTES      OPERATIONS   RESP TIME    I/O
SAID  TYPE                                /SEC      /OPERATION  /SEC         /OPERATION   INTENSITY

000C  FIBRE 2GB          PPRC READ        nnn       nnn         nnn          nnn           nnn
                         PPRC WRITE       nnn       nnn         nnn          nnn           nnn
                                                                                           ------
                                                                                           mmm

0028  FIBRE 2Gb          SCSI READ        nnn       nnn         nnn          nnn           nnn
                         SCSI WRITE       nnn       nnn         nnn          nnn           nnn
                                                                                           ------
                                                                                           mmm

002C  FIBRE 2Gb          PPRC SEND        nnn       nnn         nnn          nnn           nnn
                         PPRC RECEIVE     nnn       nnn         nnn          nnn           nnn
                                                                                           ------
                                                                                           mmm

008C  FIBRE 2GB          PPRC SEND         nnn       nnn         nnn          nnn           nnn
                         PPRC RECEIVE      nnn       nnn         nnn          nnn           nnn
                                                                                           ------
                                                                                           mmm
```

6.2.2 SMF extension

SMF Record type 74 subtype 8 has been created to store ESS link measurement data.

6.2.3 SPE details

The SPE APAR is OA04877; it is available for RMF at z/OS V1R2 and later.

7

SMPE for z/OS and OS/390 Version 3 Release 3

This chapter describes the enhancements and changes that have been incorporated into SMPE V3R3. The following topics are discussed:

- What is new in SMP/E V3R3
- Communications server FTP client exploitation
- GIMZIP and GIMUNZIP Extensions
- RECEIVE FROMNETWORK Service Routine
- IEBCOPY COPYMOD Support
- Extended RECEIVE SOURCEID processing
- REJECT CHECK operand
- Wildcard on the CSI QUERY dialog
- New data sets
- Installation, migration, and coexistence considerations

7.1 What is new in SMP/E V3R3

For a quick reference of what is new in SMP/E V3R3, go the SMP/E primary option menu and select option “w”. The following panel (Figure 7-1) is displayed.

```
TUTORIAL           What is New in SMP/E Version 3 Release 3          TUTORIAL
OPTION  ===> _

The following is a summary of the enhancements and changes that have been
incorporated in SMP/E Version 3 Release 3.  Detailed information can be found
in the SMP/E manuals.  Select a topic number or press ENTER to view the topics
in sequence.
                                                                  More:     +
Significant Changes:

  1 - RECEIVE FROMNETWORK command uses z/OS Communications Server FTP client
  2 - GIMZIP packages may contain Unix directories and files and VSAM data sets
  3 - GIMUNZIP can extract archives into existing data sets
  4 - SMP/E assigns RECEIVE command SOURCEID value to existing SYSMODs
  5 - SMP/E uses COPYMOD to copy load modules

SMP/E Dialog Changes:

  6 - CSI Query allows entry name wildcard

Miscellaneous Changes:

  7 - REJECT command accepts CHECK operand
  8 - SMP/E provides a new service routine to transfer GIMZIP packages
```

Figure 7-1 SMP/E V3R3 online tutorial

7.2 Communications server FTP client exploitation

Currently SMP/E RECEIVE FROMNETWORK operations use an SMP/E-unique FTP client. SMP/E writes to and reads from the control and data sockets directly. SMP/E will now use the FTP client provided by z/OS Communications Server.

RECEIVE FROMNETWORK has been enhanced to:

- Allow user credentials and file data transferred between an FTP client and server to be secured with respect to encryption, authentication, and data integrity using the Transport Layer Security (TLS) enablement for FTP.
- Allow the z/OS FTP client to connect to FTP servers that reside beyond a firewall that runs a SOCKS server.
- Make IPv6 connectivity possible for both the FTP client and server.
- Use the FTP.DATA configuration data set to allow the client to specify local site parameters.
- The FTP.DATA configuration data set is optional, but must be used by the client to specify the parameters for TLS security and SOCKS firewall support.SOCKS firewall navigation.
- Reduces the opportunities for errors associated with an SMP/E unique FTP client.

To exploit SOCKS firewall navigation, Secure FTP, and other aspects of the FTP client, you must configure the FTP client by modifying FTP.DATA in one of these locations:

- $HOME/ftp.data

- userid.FTP.DATA
- /etc/ftp.data
- The SYS1.TCPPARMS(FTPDATA) data set
- The tcpip_hlq.FTP.DATA file

7.2.1 Migration tasks

SMP/E will now use the FTP.DATA configuration data set to allow the client to specify local site parameters. Two of the values specified in the FTP.DATA data set are FWFriendly and FTPKEEPALIVE. These values correspond to the “pasv” and “keepalive” attributes in the CLIENT data set. Therefore, these attributes should no longer be specified in the CLIENT data set.

Note: If the “pasv” and “keepalive” attributes are specified in the CLIENT data set, they will be ignored.

You must now specify FWFriendly or FTPKeepAlive in the FTP.DATA file as shown in Figure 7-2.

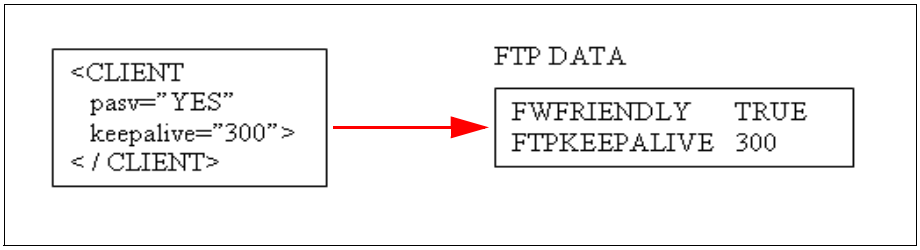

Figure 7-2 FTP.DATA file

To enable TLS security, SOCKS firewall support, and IPv6 addressing, ensure that z/OS Communications Server V1R2 (or higher) is installed.

7.2.2 SMP/E V3R3 and Internet delivery

For Internet delivery of z/OS V1R6 and ServerPac, you need:

- IBM SMP/E for z/OS and OS/390 V3.3 (5655-G44) or higher.

  You can order the latest version of SMP/E as a free product, entitled to z/OS and z/OS.e customers. You can even download SMP/E from the Internet, but if you do, you should also order it separately to ensure that your software profile is updated and that you are registered to receive service. This package includes the function required for Internet delivery, and is also intended to provide the SMP/E network capabilities in situations where the required level of SMP/E is not installed.

To use the SMP/E RECEIVE FROMNETWORK function, or to receive Internet-delivered ServerPacs using the “Receive the order from Server” option, you also need:

- Integrated Cryptographic Service Facility (ICSF)

  ICSF must be configured and active. ICSF is a base element of z/OS. It provides data integrity by performing hash checking on the Internet package. SMP/E RECEIVE FROMNETWORK uses ICSF utilities to verify the hash values defined for the package by IBM when you download directly to your z/OS or z/OS.e system, or you download to a

workstation and RECEIVE FROMNETWORK to access the files on the workstation by configuring it as an FTP server.

CustomPac dialog

To install an Internet delivered ServerPac, you also need *CustomPac dialogs*.

If you are using a dialog whose Package Version is less than 17.00.00, you must migrate the dialog to this level, or higher. You can easily determine if you have the correct dialog level if the text “This dialog supports electronic delivery.” appears at the bottom of panel CPPPOLI. If your dialog is not at the minimum level, the migration scenarios and steps are described in *ServerPac: Using the Installation Dialog,* SA22-7815.

7.3 Enhancements to GIMZIP and GIMUNZIP service routines

GIMZIP and GIMUNZIP are being provided as part of SMP/E to assist in the packaging of a product for shipment via the Internet. The GIMZIP service routine creates portable packages of software and related materials. Typically, the packages contain SYSMODs, RELFILE data sets, HOLDDATA, and associated material such as documentation, samples, and text files. These GIMZIP packages may be transported through a network, processed by the GIMUNZIP service routine, and then processed by the SMP/E RECEIVE command. The GIMUNZIP service routine is used to extract data sets from archive files in GIMZIP packages created by the GIMZIP service routine.

We now describe the enhancements made to the GIMZIP and GIMUNZIP service routines and their processing requirements.

Calling GIMZIP and GIMUNZIP

When executing these programs, as shown in Figure 7-3 on page 165 and Figure 7-4 on page 167, the following DD statements indicate, for UNIX System Services, the following:

SMPDIR Specifies a directory in a UNIX file system that contains a GIMZIP package. This directory is referred to as the package directory.

SMPWKDIR Specifies a directory in a UNIX file system that is used by GIMUNZIP for temporary work files. This is an optional DD statement. If the SMPWKDIR DD statement is not provided, GIMUNZIP will use the package directory specified on the SMPDIR DD statement for temporary work files.

Note: In the input, the existing data set can now be a file or directory in the following parameter NEWNAME, ARCHID can be the HFS root, and PRESERVEIDS identifies the original UID and GID of files.

7.3.1 GIMZIP extensions

Formerly, GIMZIP could create and GIMUNZIP could process packages that contained only sequential and partitioned data sets. Also, GIMUNZIP would only extract data from an archive file into a new data set allocated directly by GIMUNZIP.

The GIMZIP and GIMUNZIP service routines have been enhanced to allow packages to also contain VSAM ESDS, KSDS, LDS, and RRDS data sets, and UNIX files and directories.

Additionally, when building a package using GIMZIP, a unique ID value (as shown in Figure 7-3) may be assigned to each archive. The ID value may then be used to identify an archive that is to be processed by GIMUNZIP, as opposed to using the archive's file name.

More specifically, a GIMZIP package consists of a single package definition file, a set of archive files, and text files. The package definition file describes the total package and identifies the archive files and text files contained in the package. An archive file consists of a portable image of any of the following:

- A sequential data set
- A partitioned data set
- A VSAM data set
- A file in the UNIX file system
- A directory in the UNIX file system

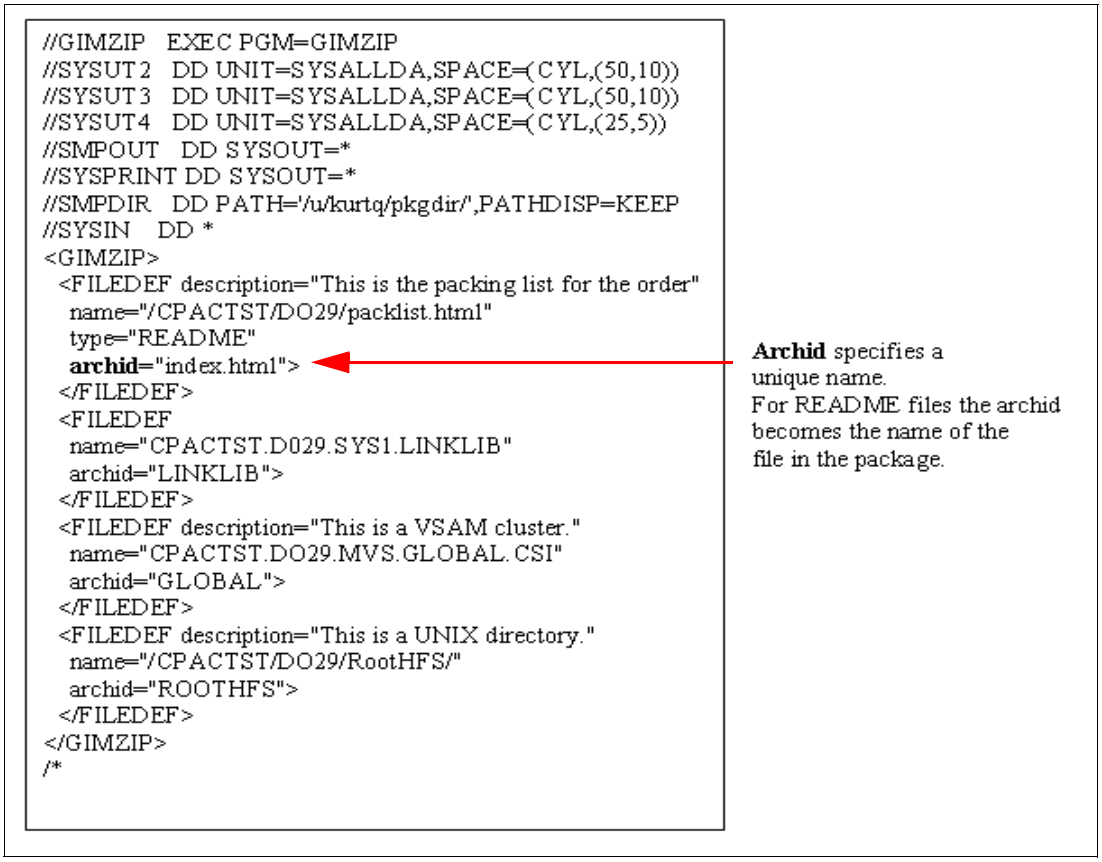

Figure 7-3 Example of using the new archid attribute

7.3.2 GIMZIP processing

You may assign a unique ID to an archive during GIMZIP processing and then use that ID to identify the archive that is to be unzipped during GIMUNZIP processing.

GIMUNZIP now allows GIMUNZIP operations into existing data sets. GIMUNZIP determines whether the data set specified on the <ARCHDEF> tag already exists. If it does, GIMUNZIP copies the archive file into the existing data set. If the data set does not already exist, GIMUNZIP allocates a new data set and then copies the archive file into that new data set.

VSAM data sets

When a VSAM data set is being specified, the true cluster name must be used. Do not reference a VSAM data set by a path name. Although an alternate index may be defined to the cluster, the alternate index does not become part of the archive. If an alternate index is desired at the destination site after the archive is unzipped, then the alternate index must be defined and built at the destination site.

For VSAM data sets, GIMZIP does the following:

- Captures the attributes of the existing data set from the catalog.
- Uses REPRO to produce a portable sequential form of the data.
- Copies the sequential form to a file in the UNIX file system.
- Uses the **pax** command to create an archive file

For UNIX directories and files, GIMZIP uses the **pax** command to create the archive file directly.

7.3.3 GIMUNZIP extensions

The GIMUNZIP service routine is used to extract data sets, files, and directories from archive files in GIMZIP packages created by the GIMZIP service routine. More specifically, the GIMUNZIP service routine extracts data sets, files, and directories from the archive files that compose the GIMZIP package. An archive file consists of a portable image of a sequential, partitioned, or VSAM data set, or a file or directory in a UNIX file system, and the information needed to create that data set, file, or directory from the portable image. The data set, file, or directory into which the archive file is to be extracted can already exist or GIMUNZIP can create a new one of the appropriate type. New sequential and partitioned data sets created by GIMUNZIP are always catalogued.

As stated in the GIMZIP extensions above, the GIMZIP and GIMUNZIP service routines have been enhanced to allow packages to also contain VSAM ESDS, KSDS, LDS, and RRDS data sets, and UNIX files and directories.

Archive files

Additional enhancements include:

- Selection of archives by file name or by new archid name
- Extracting archives into:
  - New data sets and files
    - Allocates PDS and Sequential data sets.
    - Uses IDCAMS DEFINE CLUSTER for VSAM data sets.
    - Creates directories and files in the UNIX file system.
  - Existing data sets and files

    This is useful if the output data set must span volumes.
- When extracting into existing data sets, the archive and destination formats must be the same.
  - Extract PDS archive only to a PDS. Cannot extract PDS members into a UNIX directory, or vice versa.
  - Extract sequential data set archive only to a sequential data set. Cannot extract data set into a UNIX file, or vice versa.
  - Extract directory archive only to a UNIX directory.
- Optionally replace members of an existing PDS or files in an existing directory.

- Optionally preserve the uid and gid of UNIX files versus inheriting the uid and gid from the user executing GIMUNZIP.
  - For general use, inheriting the uid and gid is fine.
  - For system installation (that is, ServerPac) preserving the defined uid and gid is preferred.

VSAM data sets

For a VSAM data set (cluster), replace="YES" indicates that an existing VSAM cluster should be populated with the data from the archive file.

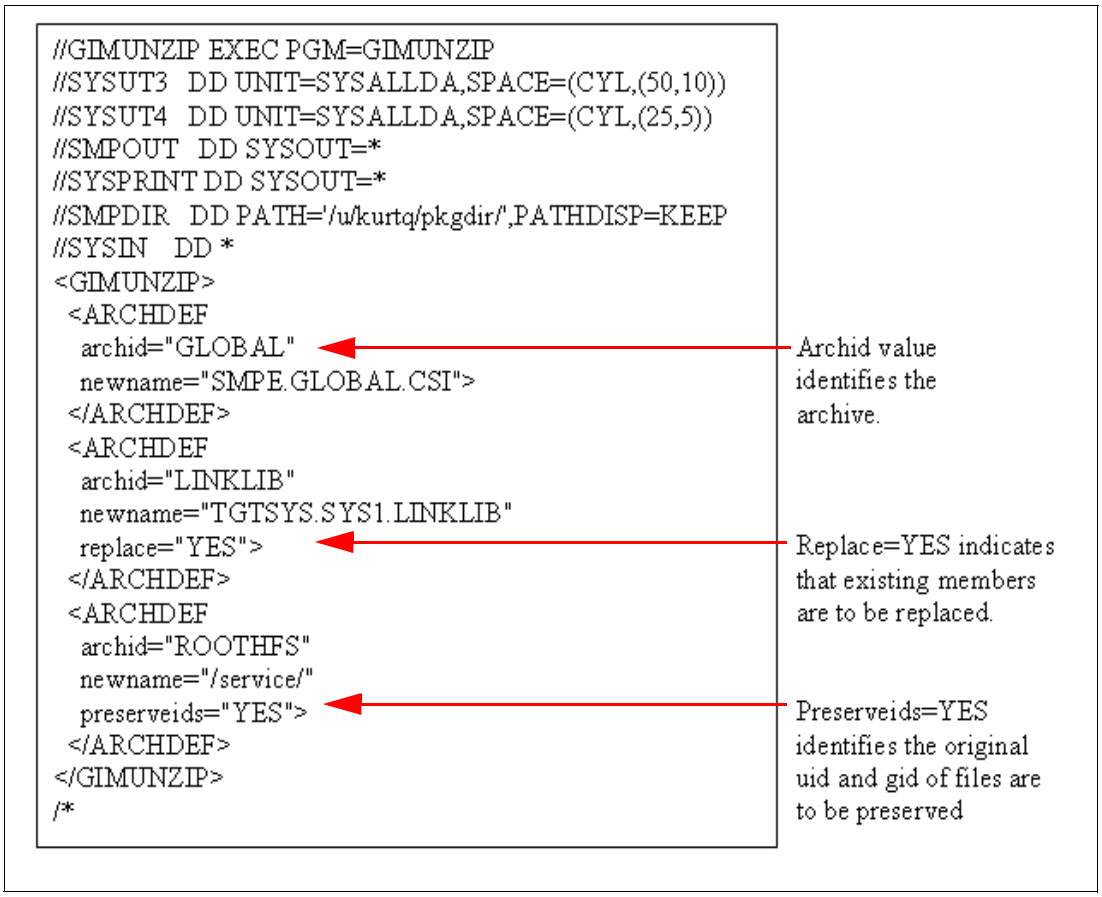

Figure 7-4 GIMUNZIP extensions example

7.3.4 GIMUNZIP processing

The GIMUNZIP service routine extracts data sets and files from the archive files that compose the GIMZIP package. An archive file consists of a portable image of a data set or file and the information needed to reload the data from the portable image. The GIMUNZIP program does the following:

- Uses IDCAMS DEFINE CLUSTER to create new data sets based on the attributes of the original data set (optional).
- Uses the **pax** command to extract the portable unloaded form of the data set.
- Copies the unloaded form into a temporary data set.
- Uses IDCAMS REPRO to load the data set with the unloaded data.

For UNIX directories and files GIMUNZIP:

- Uses the **pax** command to extract from the archive directly into the destination.

7.4 RECEIVE FROMNETWORK service routine

The new GIMGTPKG service routine can be used to get GIMZIP packages from a remote FTP server in a TCP/IP network and store the package on a local z/OS host. GIMGTPKG performs the functions of the SMP/E RECEIVE FROMNETWORK TRANSFERONLY command, but does so independently of SMP/E.

Using GIMGTPKG, you can:

- Transport a GIMZIP package from a remote FTP server to a local z/OS system.
- Use industry standard FTP for transport.
- Support secure transmission (authenticated and encrypted).
- Ensure the integrity of the package files.

Note: The files in the GIMZIP package are stored in the SMPNTS directory for use later by the RECEIVE FROMNTS command, or other applications and offerings.

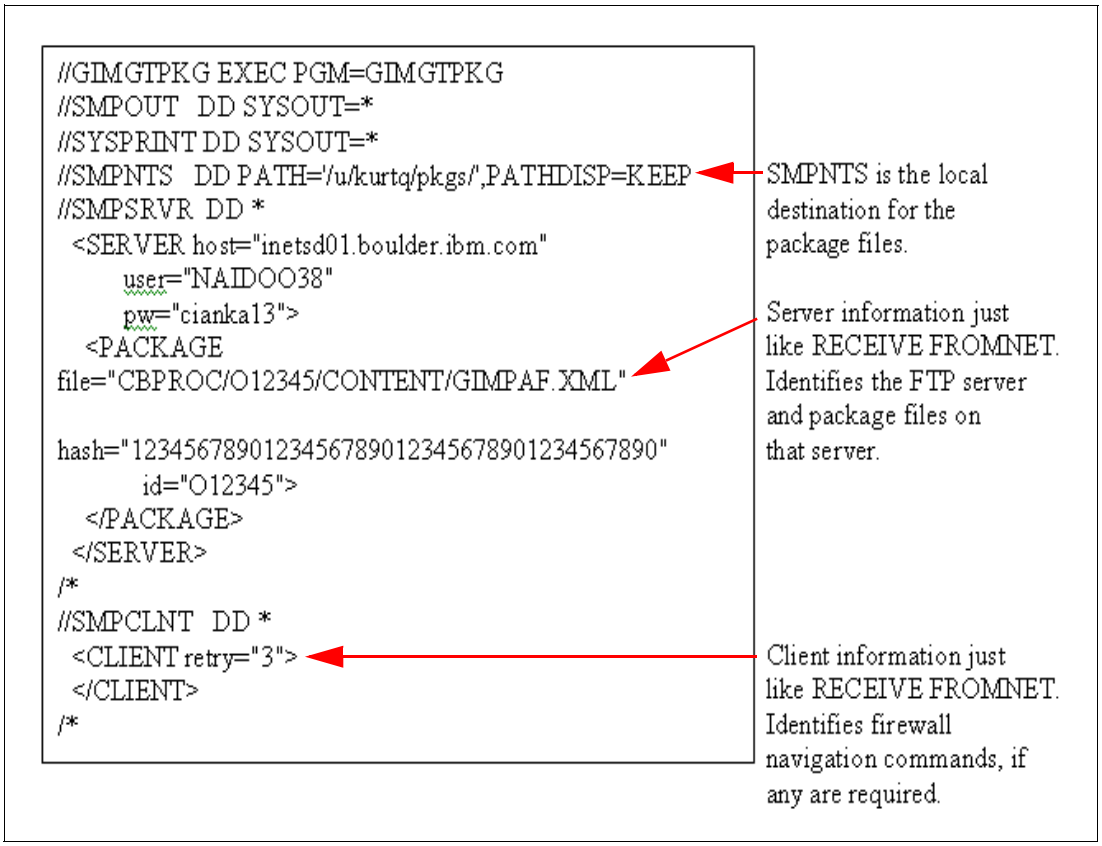

Figure 7-5 GIMGTPKG service routine example

7.5 IEBCOPY COPYMOD support

SMP/E now uses the COPYMOD control statement in conjunction with the SPCLCMOD and CMWA copy execution parameters when the copy utility is invoked to copy load modules.

This results in the following:

- Any load module that is reblockable will be reblocked.
- Any load module that cannot be reblocked, but can be copied without causing fat blocks, will be copied without complaint (RC=0).
- Any load module that cannot be reblocked and cannot be copied without causing fat blocks will not be copied and the copy operation will fail with RC=8.
- JCLIN will recognize the COPYMOD control statement for copy steps.

The result is that more SYSMODs should be applied and accepted without problems and with better space utilization of load module data sets. Additionally, the installation of load modules that are likely to cause a problem in the future is stopped until the user takes a corrective action (such as increasing the blocksize of the destination library).

Note: SMP/E passes new default parameters (SPCLCMOD and CMWA=256K) to the copy utility when copying modules, load modules, or programs.

7.6 Extended RECEIVE SOURCEID processing

The RECEIVE command has been enhanced to assign the source ID specified on the SOURCEID operand of the command to SYSMODs found in the SMPPTFIN input stream, even if they are already received.

Note: Formerly, the source ID was not assigned to SYSMODs that are already received.

7.7 REJECT CHECK operand

The CHECK function has been added to the REJECT command.

CHECK indicates whether SMP/E should do a trial run of a command without actually updating any libraries.

This provides a way to test for errors that might occur during actual processing and to receive reports on the changes that would be made.

7.8 Wildcard on the CSI QUERY dialog

The CSI QUERY dialog allows a wildcard (pattern) for the entry name specification as shown in Figure 7-6 on page 170. A selection list of all entry names that match the specified pattern will be displayed when using a wildcard.

- Patterns of the form ABC* or *ABC may be specified, where ABC is a string from 0-7 characters long.
- An entry type specification is required when using an entry name wildcard.
- Only one wildcard character (*) is allowed in the entry name specification.

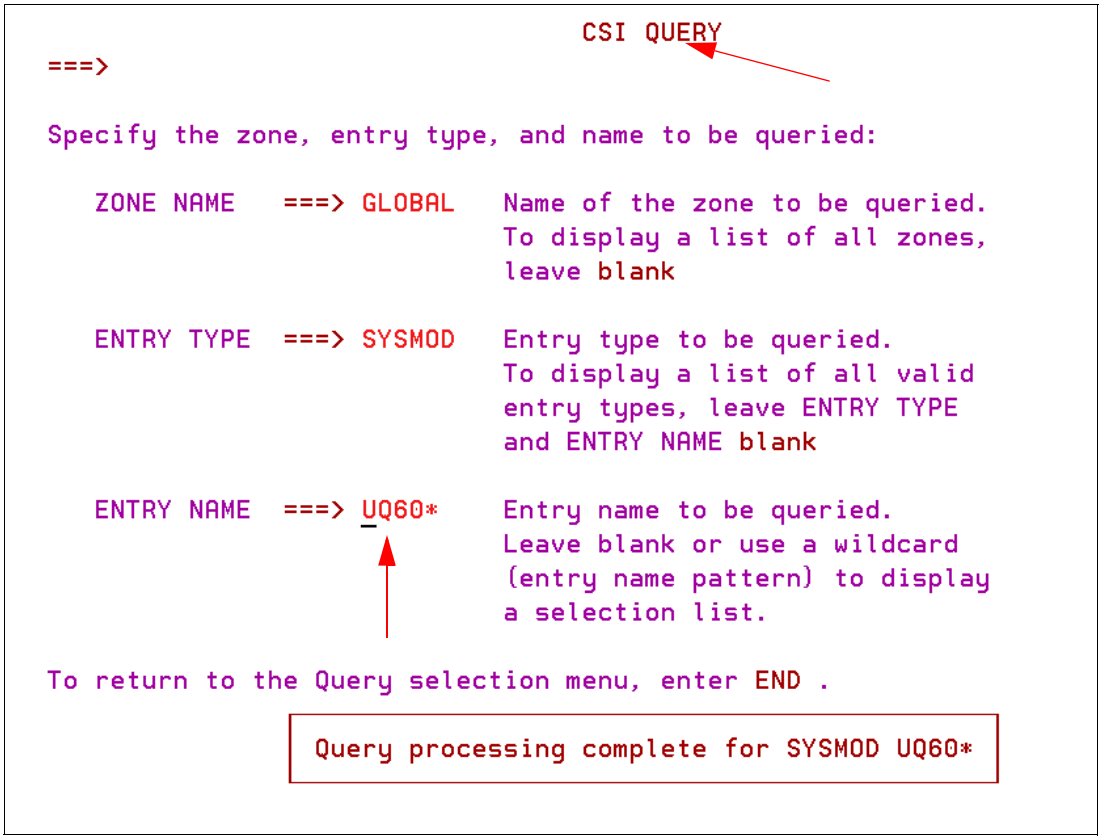

*Figure 7-6 Example of the * wildcard on the CSI QUERY panel*

7.9 New data sets

New data sets SMPCLNT and SMPSRVR have been added. These data sets can reside in a UNIX file system.

The SMPCLNT data set contains information about the TCP/IP client environment of the local machine. It is used by the GIMGTPKG service routine.

The SMPSRVR data set contains information about a TCP/IP-connected host running an FTP server. It is used by the GIMGTPKG service routine.

Note: Specify PATHOPTS(ORDONLY) and FILEDATA=TEXT on the DD statement for the data sets, if they reside in a UNIX file system.

7.10 Installation, migration, and coexistence considerations

There are no unique installation considerations.

For migration do the following:

- Modify the CLIENT file for RECEIVE FROMNETWORK.
- The UPGRADE command is *not* required for migration to SMP/E V3.3.

Coexistence considerations include:

- Coexistence PTFs for prior release levels will be available.
- No incompatible changes are made to SMP/E data sets, but coexistence provides complete support or explicit messages to call out new function operands.

8

z/OS V1R6 Workload Manager (WLM)

WLM facilities continue to be refined, upgraded and enhanced in an effort to improve resource and performance management and reporting capabilities, as well as providing support for new technologies. The WLM enhancements provided in z/OS V1R6 are described in the remainder of this chapter and include the following:

- WLM virtual 64-bit support for UNIX System Services
- WLM virtual 64-bit support for WebSphere
- WLM support for greater than 16 CPUs in one z/OS image
- WLM LAN free backup enhancements
- WLM stateful session placement
- DB2 stored procedures enhancements
- WLM support for integrated facility for applications

8.1 WLM virtual 64-bit support for UNIX System Services

WLM provides callable services that are available to C and C++ server applications to invoke WLM services. To support their operation in 64-bit virtual mode, these interfaces have been upgraded to provide support for both 31-bit and 64-bit AMODE callers.

For z/OS V1R6, fifteen WLM services are enhanced to support 64-bit environments. These services run in both 31-bit and 64-bit address mode. To use 64-bit services, change the names of the services in your application, for example, change IWMCONN to IWM4CON. The prefix of all 64-bit services names is IWM4, as shown in Table 8-2 on page 175.

The services that run in 64-bit address mode in general support the same parameters as their equivalents for 31-bit address mode. Note that the only exception is the PLISTVER parameter, which has slightly changed. The 64-bit services only support PLISTVER=0, in case a PLIST Version is explicitly used. The following example shows how to use the PLISTVER keyword for 31-bit services:

```
12345678 SPACE 1 DS 0H
         IWMxxxxx ETOKEN=ETOKEN
                  RSNCODE=RSNCODE,
                  PLISTVER=2
```

The functions shown in Table 8-1, in WLM.H in PDS CEE.SCEEH.SYS.H, have all been updated to include 64-bit support in the C-Compiler, Language Environment, and UNIX System Services.

Table 8-1 WLM callable service routines

Callable service	WLM Service Invoked
ExportWorkUnit	IWMEXPT
UnDoExportWorkUnit	IWMUEXPT
ImportWorkUnit	IWMIMPT
UnDoImportWorkUnit	IWMUIMPT
CreateWorkUnit()	IWMECREA
ContinueWorkUnit()	IWMECREA
ConnectWorkMgr()	IWMCONN
ConnectServer()	IWMCONN
DisconnectServer()	IWMDISC
JoinWorkUnit()	IWMEJOIN
LeaveWorkUnit()	IWMELEAV
DeleteWorkUnit()	IWMEDELE
ExtractWorkUnit()	IWMESQRY
QueryMetrics()	IWMWSYSQ
QuerySchEnv()	IWMSEQRY
CheckSchEnv()	IWMSEDES
QueryWorkUnitClassification()	IWMECQRY

Callable service	WLM Service Invoked
ConnectExportImport	IWMCONN

The following four functions in IWMWDNSH.H in PDS SYS1.SIEAHDR.H have been updated to include 64-bit operation, but no support has been provided for them in the C-Compiler, the Language Environment, or UNIX System Services:

- IWMDNGRP
- IWMDNSRV
- IWMDNREG
- IWMDNDRG

Refer to *z/OS MVS Programming Workload Management Services,* SA22-7619 and *z/OS C/C++ Run-Time Library Reference,* SA22-7821 for more detailed information on these interfaces.

8.2 WLM virtual 64-bit support for WebSphere

Support has been provided with z/OS V1R6 so that 64-bit applications can now call WLM services in 64-bit mode. Fifteen new 64-bit WLM services are available to be called by applications running in either 31-bit or 64-bit mode. Additionally, applications calling these new WLM services in 64-bit mode can pass parameters located above the 2 GB bar.

WebSphere is the first exploiter of these services.

Table 8-2 shows the old WLM services and the new services that have been enabled for 64-bit operation.

Table 8-2 WLM services

Old WLM Service	New WLM Service	Function
IWMCONN	IWM4CON	Connect to Workload Manager.
IWMDISC	IWM4DIS	Disconnect from Workload Manager.
IWMECREA	IWM4ECRE	Create an Enclave.
IWMEDELE	IWM4EDEL	Delete an Enclave.
IWMMCHST	IWM4MCHS	Change state of work request service.
IWMMCREA	IWM4MCRE	Create monitor environment service.
IWMMINIT	IWM4MINI	Monitor environment initialization.
IWMQDEL	IWM4QDE	Delete a request from the queue for an execution address.
IWMQINS	IWM4QIN	Insert a request in the queue for an execution address.
IWMSLIM	IWM4SLI	Application Environment Limit Service.
IWMSSEL	IWM4SSL	Select a request from a caller's work manager queue.
IWMSSEM	IWM4SSM	WLM server select secondary service.

Old WLM Service	New WLM Service	Function
IWMSTBGN	IWM4STBG	Begin a request from a caller's work manager queue.
IWMSTEND	IWM4STEN	End a request from a caller's work manager queue.
IWMTAFF	IWM4TAF	WLM Temporal Affinity service.

Important: The new WLM services will only run on z/OS V1R6 or higher and can be used in either 31-bit or 64-bit environments.

8.3 WLM support for greater than 16 CPUs

WLM has been enhanced in z/OS V1R6 to support up to 64 CPUs per z/OS image. This removes the long-standing limitation of 16 processors per z/OS image and permits an installation to take advantage of new zSeries hardware technology.

8.4 WLM LAN free backup enhancements

With z/OS V1R6, IOS provides a new general use programming interface (GUPI) that authorized applications can use to mark a device offline and in use by a system component. IOS also maintains configuration integrity and initializes device-dependent features such as dynamic pathing. WLM has been enhanced to support the gathering of measurement data for devices that are offline and in use by system components.

8.5 WLM stateful session placement

New work requests inserted by a queue manager causes WLM to wake up a suspended server task to process it. The wake-up process attempts to keep the first bound server address space highly utilized while also trying to decrease the work utilization in order to promote an environment with idle servers in server address spaces. Successfully achieving this result can be beneficial to SRM when it decides to stop server address spaces.

However, it is counterproductive when an application inserts too many work requests that create an affinity to a specific server region. These affinities are created when a data object is built in the server address space that will be required by subsequent client requests. In this situation, it would be better to spread the work requests across all of the available server regions.

WLM stateful session placement now allows for a round-robin method of scheduling new transactions across all active WebSphere Access Services (WAS) server regions in one WAS control region. This provides a simplified environment for horizontal scaling. Multiple control regions do not have to be created and the external round-robin facility does not need to be maintained as application volumes grow. Rather, they can be controlled using the WAS control region parameters min_srs and max_srs. The number of server instances is managed to handle predicted transaction volumes.

8.6 DB2 Stored Procedures enhancements

WLM makes application interfaces available to applications that can be used to pass queue/server management work requests from an application daemon to work regions that can process these requests. WLM will queue these requests based on service class queue classification, and will manage the server regions based on the service class goal definition, the level of goal achievement, and demand.

DB2 is one exploiter of these application interfaces in order to get WLM management of their stored procedures address spaces. The customer is required to define an application environment and to provide a startup procedure that WLM can use to start the server regions.

The initial design for this support assumed that all of the work requests would be independent of each other. In the case of DB2 Stored Procedures, this is not always true, and has led to problems in WLM's management of the server regions.

Each stored procedure is executed by a server task within the WLM-established DB2 Stored Procedure address space. In the case of recursive calls, each server task waits for the completion of the stored procedure it called. For quasi-parallel calls, DB2 wakes up the server tasks but only one task is active at any given time while the other server tasks are suspended.

In order for the initial work request to complete its processing, all subsequently called work requests must be selected by server tasks. The suspended tasks do not consume resources and the dependencies between the work requests are unknown. WLM may not detect that a problem exists because service class goals are being met and will therefore project that adding more server address spaces would not provide any additional benefit.

The WLM queue/server management application interfaces have been enhanced to allow an application, such as DB2 Stored Procedures, to inform WLM that a work request with a dependency to other work already in progress is inserted. WLM recognizes these situations and attempts to assist these applications by starting new server address spaces as long as the available system resources and WLM service class goals will support this.

8.7 WLM support for Integrated Facility for Applications

Integrated Facility for Applications (IFA) is a new type of special purpose processor available for zSeries machines. They are similar to the other types of special purpose processing units, such as the Integrated Coupling Facility (ICF) and the Integrated Facility for Linux (IFL). The primary difference between the IFA and the other special purpose processors is that the z/OS operating system recognizes the IFA as a new type of processing unit. IFAs cannot be IPLed and they do not process interrupts. Additionally, in some circles the IFA may also be referred to as a JAF or Java Assist Facility.

The objective for the IFA was to create a lower cost environment for eligible workloads executing zSeries software. Currently, the only kind of work that is eligible to be dispatched on an IFA is work that is executing in the Java Virtual Machine (JVM). The JVM has been enhanced to provide a switch interface which contacts the dispatcher and signals that IFA-eligible work is beginning. This workload is then removed from a regular CP and queued for execution on an available IFA. When an IFL becomes available for work, it selects the next highest priority work request from the IFA dispatcher queue and begins execution.

It is important to note that IFA-eligible work can still run on a normal CP as well as an IFA. This is known as crossover. The standard CP can select work from either the system work unit queue or the IFA work unit queue. If a CP and IFL are both available when an IFA-eligible work unit is dispatched, it is dispatched to and processed on the CP. Restrictions on

crossover may be set by the installation by means of parameters in the IEAOPTxx member of SYS1.PARMLIB.

Integration of IFAs into the zSeries environment has required changes to System Resource Manager (SRM). WLM and SRM work in tandem to provide support for IFAs and the IFA-eligible workloads access to system resources. The following areas have been enhanced to provide this support:

- New IEAOPTxx parameters
- Calculation of CPU and IFA “Using” and “Delay”
- Calculation of CPU times and service
- Modifications for starting WLM server address spaces
- Exclusion of IFAs from LPAR management
- Extensions to the SMF 99 record

8.7.1 New IEAOPTxx parameters

There are two methods available to the dispatcher to select work to be executed on regular CPs from the IFA work unit queue (WUQ).

The dispatcher on the normal CP reviews both dispatch queues and selects the work with the highest dispatching priority.

When no work is queued in the system work unit queue (SWUQ), the IFA-eligible work executes on the regular CP as if its dispatch priority were below the level of discretionary work. This, again, is known as crossover work.

Under normal operation, the dispatcher executing on a regular CP selects the highest priority work from both of the work unit queues. Two new IEAOPTxx parameters are made available with z/OS V1R6 that can be used by the installation to change the normal mode of CP work selection.

```
IFACROSSOVER=Yes|No
IFAHONORPRIORITY=Yes/No
```

IFACROSSOVER parameter

FA crossover and priority specifications are passed to the supervisor so that it knows if these functions are enabled or not.

Specifying YES to IFACROSSOVER indicates that WLM will *manage* crossover, not that WLM will always cause a dispatch before a wait state. Specifying NO disables the crossover capability.

IFAHONORPRIORITY parameter

Specifying YES to IFAHONORPRIORITY indicates the WLM will *manage* priority selection, not that WLM will always dispatch in priority sequence. Specifying NO disables priority-based dispatching.

LPAR capping

Additionally, there is one other situation that can impact whether or not IFA honor priority is active, and it potentially impacts customers using LPAR capping.

WLM communicates with the LPAR Hipervisor to turn LPAR capping on and off. The basis for the on or off decision is a partition’s rolling four hour average. When the rolling four hour

average exceeds the capacity limit defined by the installation, WLM performs several functions. First, it deactivates honor priority and communicates that to the supervisor. Then it contacts the LPAR Hipervisor and instruct it to begin capping the partition.

During the time that the partition is capped, regular CPs can continue to process IFA-eligible work, but it will only be at below discretionary dispatch priority because crossover mode remains unaffected by LPAR capping.

WLM enables honor priority dispatching when the rolling four hour average falls below the defined capacity limit and LPAR capping is turned off.

There is also an additional undocumented IEAOPTxx parameter that can be used to control supervisor processing when a switch to or from an IFA is requested. The parameter is:

```
IFASWITCHIMMEDIATE=Yes|No
```

If YES is specified, the supervisor forces the dispatch when the switch is requested. If NO is specified, the dispatch is deferred and the work running on the CP is allowed to continue until the time slice ends and a redispatch occurs.

8.7.2 Calculation of CPU and IFA Using and Delay

Three new CPU Using and Delay states are introduced to reflect the fact that IFA-eligible work can also run on a regular CP. They are:

IFA using	Work is detected executing on an IFL.
IFA on CP using	IFA-eligible work is detected executing on a regular CP.
IFA delay	Work is delayed for an IFA.

The following related Using and Delay states already exist:

CPU using	Work is detected executing on a regular CP.
CPU delay	Work is delayed for a regular CP.

IFA-eligible work that executes on a regular CP is recorded as “IFA on CP using”.

If the dispatcher is honoring priorities for selecting work from the IFA work unit queue, these “using” samples are also recorded as “CPU using” and the amount of “IFA delay” is reduced proportionally to the amount of “IFA and CPU using”. To demonstrate:

```
CPU using = CPU using + IFA on CP using
IFA delay = IFA on CP using * IFA delay / all IFA using
All IFA using = MAX(1, IFA using + IFA on CP using)
CPU delay = CPU delay + IFA delay
```

If the dispatcher is not honoring priorities either because WLM or the customer has turned it off, IFA on CP using is added to the IFA using. Further, IFA on CP using is always contained in either the CP using or IFA using metric.

IFA using and IFA delay are also included in the calculation of execution velocity. Even though IFAs are not managed as a resource, reported goal achievement does reflect their impact.

8.7.3 Calculation of CPU times and service

All service units that are externalized by SRM and made available for reporting include the consumption of Regular CP and IFA service time. Service units that are used internally by SRM algorithms do not include IFA service time since IFAs are not managed as a resource.

8.7.4 Modifications for starting WLM server address spaces

WLM factors in CPU demand when it considers starting a new server address space. It compares the CPU demand with CPU utilization to determine whether it is appropriate to start another server address space. Because it is possible that DB2 Stored Procedures and WebSphere could run Java programs, IFA demand and IFA utilization are also now considered in the decision to start a new server address space.

8.7.5 Exclusion of IFAs from LPAR management

LPAR weight and CPU management are performed only for regular CPs. They do not include any consideration of IFA processor impact. Additionally, Defined Capacity management also ignores IFAs. This means that the rolling four hour average is calculated using only work executed on the regular CPs. However, an adjustment is made for IFA-eligible work that executes on a regular CP. As mentioned previously, WLM will turn off honor priority if the define capacity limit is reached so that IFA-eligible work does not compete with regular work when the system is capped.

8.7.6 Extensions to the SMF 99 record

The Basic System Data for the SMF 99 record has been extended to include data similar to existing CPUs, for the IFA. The added report fields are:

IFAs-C Number of IFAs available on the CEC

IFAs-S Number of shared IFAs

IFAs-P Number of IFAs available to the logical partition

IFAS-O Number of online IFAs

IFA IFA Utilization

The System Processor Priority Data and License Manager Data section required no update.

The Service Class Data section was updated to reflect the new IFA states:

IFAU IFA Using

IUCP IFA Using on CP

IFAD IFA Delays

The Processor Priority Data section was updated to add columns for:

- IFA Using
- IFA Using on CP
- IFA Delay

8.8 WLM support for Enterprise Workload Manager (eWLM)

Enterprise Workload Manager (eWLM) is IBM's business priority-based, response time-driven, platform-independent resource management and reporting solution of the future. eWLM will work in cooperation with middleware applications such as database, message processing, or application servers to dynamically manage the resources of an enterprise in an effort to achieve end-to-end response time goals.

z/OS WLM provides support and instrumentation for middleware applications whose tasks can be monitored and managed through WLM goal-based policies. This support exists exclusively for the z/OS environment and is not available to other platforms.

Application Response Measurement (ARM) is a developing standard owned by the Open Group to develop cross-platform response time measurement instrumentation interfaces. Middleware applications intending to take advantage of the management and reporting capabilities made possible by ARM will provide workload definitions and operational data for the transactions they run.

eWLM will then use combined facilities of WLM and ARM to effectively manage enterprise resources to increase efficiency, performance, and availability, and potentially lower the total cost of ownership.

For more information on eWLM as the emerging multi-tied workload strategy, refer to *The Great Balancing Act*, available on the IBM thinkresearch Web site at:

http://www.research.ibm.com/thinkresearch/pages/2002/20020529_ewlm.shtml

9

OSA 3270 Support for z/OS V1R6

This chapter describes the new OSA 3270 feature of the z990 processors.

The following topics are discussed:

- Introducing the OSA-Express console controller
- Installation requirements
- Migration considerations
- HCD configuration process
- HMC configuration process
- PCOM session configuration
- Integrated console controller specifications

9.1 Introducing the OSA-Express console controller

IBM has announced a new console controller for the zSeries 990. One (or two) ports on an OSA-Express 1000BASE-T ethernet card is used to connect to ethernet LAN-attached TN3270E consoles. Each port can connect up to 120 client console sessions across multiple LPARS. OSA-3270 is supported by a new channel type called OSC and a new control unit type also called OSC. A device type of 3270-X has to be used. The introduction of the OSA-3270 feature has simplified the configuration by consolidating prior console controller capacity onto an integrated footprint. This provides significant space savings and eliminates the need for standalone or rack-mounted controllers. Coexistence with the IBM 2074 and older 3174 controllers is also possible.

The following sections describe the OSA-3270 configuration process.

9.2 Installation requirements

The following requirements must be met to use the OSA-Express console controller.

Hardware:

- You must have a z990 with May 2004 licensed internal code (LIC).
- OSA-Express 1000BASE-T Ethernet (FC 1366).

Software:

- z/OS 1.3 or later releases require APAR OA05738 for OSA-3270.
- z/VM® 4.4 or later releases.

9.3 Migration considerations

A new support level has been introduced for a z990 processor. To get the new functions, after the appropriate PTFs have been applied, the processor definition has to be changed to the new support level (“XMP, 3xx models, OSC”).

Figure 9-1 on page 185 shows the new support level in HCD.

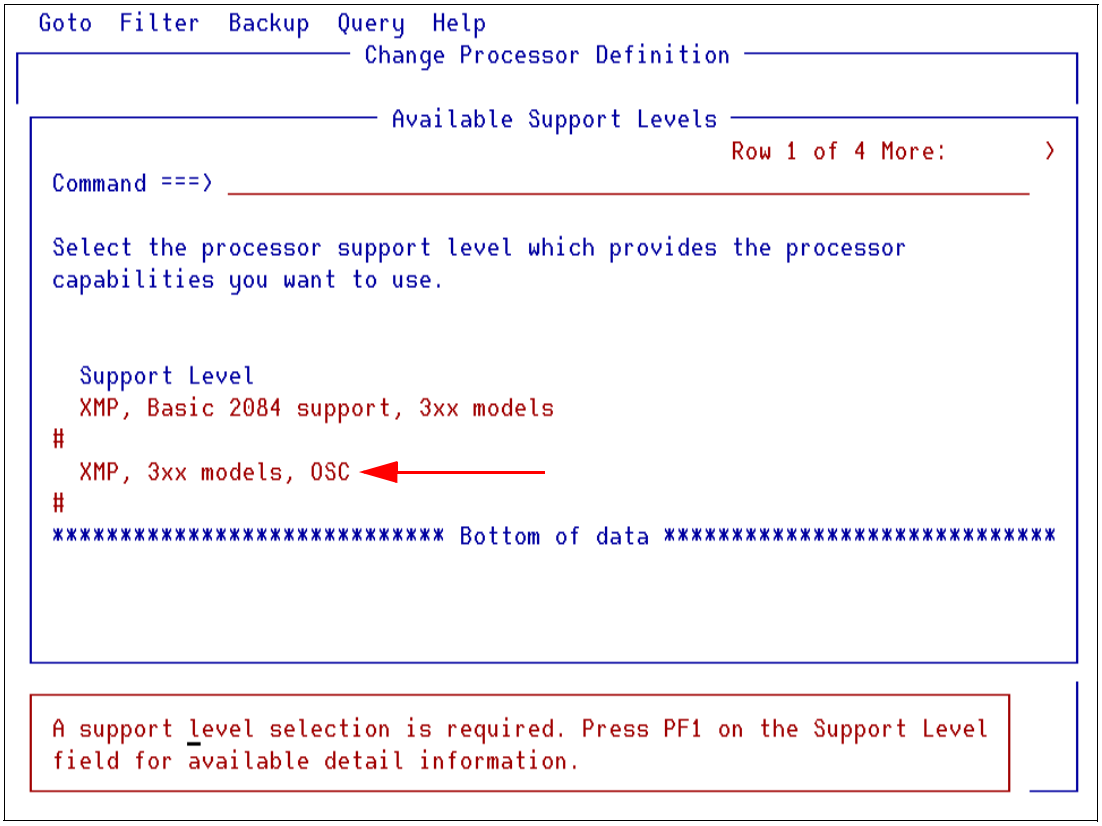

Figure 9-1 HCD processor support level

9.4 HCD configuration

Let's start by doing the HCD configuration process. We will also list the generated IOCP statements. We take an existing OSD CHPID 07 in LCSS 0 assigned to PCHID 380 and convert it to a channel type called OSC, and then define the attached control unit and devices.

Start with the CHPID. Change the CHPID type from OSD to OSC by overtyping the OSD with OSC on the channel path definition frame as shown in Figure 9-2 on page 186.

```
Specify or revise the following values.

Processor ID . . . . : SCZP901
Configuration mode . : LPAR
Channel Subsystem ID : 0

Channel path ID  . . . . 07    +            PCHID . . . 380
Channel path type  . . . OSC   +
Operation mode . . . . . SHR   +
Managed  . . . . . . . . No    (Yes or No)   I/O Cluster ________  +
Description  . . . . . . 1000BaseT

Specify the following values only if connected to a switch:

Dynamic entry switch ID  __   + (00 - FF)
Entry switch ID  . . . . __   +
Entry port . . . . . . . __   +
 F1=Help     F2=Split    F3=Exit     F4=Prompt   F5=Reset    F9=Swap
F12=Cancel
```

Figure 9-2 Change channel path definition frame

Select **CHPID 07** from the channel path list and use PF11 to add a new control unit of type OSC, as shown in Figure 9-3.

```
CBDPCUF1                    Control Unit List
Command ===> ____________________________________________ Scroll ===> CSR

Select one or more control units, then press Enter.  To add, use F11.

Processor ID . . : SCZP901    CSS ID . : 0   Channel path ID  : 07
                          Add Control Unit
/ CU    CBDPCU10
*****
        Specify or revise the following values.

        Control unit number  . . . . E200  +
        Control unit type  . . . . . OSC_________  +

        Serial number  . . . . . . . __________
        Description  . . . . . . . . ________________________________

        Connected to switches  . . . __  __  __  __  __  __  __  __  +
        Ports  . . . . . . . . . . . __  __  __  __  __  __  __  __  +

        If connected to a switch:

        Define more than eight ports . . 2   1.  Yes
                                             2.  No
        Propose CHPID/link addresses and
        unit addresses . . . . . . . . . 2   1.  Yes
                                             2.  No
```

Figure 9-3 Add a new control unit

Press Enter to advance to the next panel and specify the CHPID number to be used, as shown in Figure 9-4 on page 187.

```
CBDPCUP0                    Select Processor / CU     Row 1 of 5 More:       >
Command ===> _____________________________________________ Scroll ===> CSR

Select processors to change CU/processor parameters, then press Enter.

Control unit number . . : E200     Control unit type . . . : OSC

             ---------------Channel Path ID . Link Address + ---------------
/ Proc.CSSID 1------ 2------ 3------ 4------ 5------ 6------ 7------ 8------
_ SCZP701    _______ _______ _______ _______ _______ _______ _______ _______
_ SCZP702    _______ _______ _______ _______ _______ _______ _______ _______
_ SCZP801    _______ _______ _______ _______ _______ _______ _______ _______
_ SCZP901.0  07_____ _______ _______ _______ _______ _______ _______ _______
_ SCZP901.1  _______ _______ _______ _______ _______ _______ _______ _______
******************************* Bottom of data ********************************
```

Figure 9-4 Specify CHPID number for CU

Press Enter and define the starting unit address and range, as shown in Figure 9-5. We will define 254, which is the maximum value allowed for the control unit specification.

```
CBDPCUP1                    Select Processor / CU     Row 4 of 5 More: <     >
Command ===> _____________________________________________ Scroll ===> CSR

Select processors to change CU/processor parameters, then press Enter.

Control unit number . . : E200     Control unit type . . . : OSC

                CU   --------------Unit Address . Unit Range + -------------
/ Proc.CSSID Att ADD+ 1----- 2----- 3----- 4----- 5----- 6----- 7----- 8-----
_ SCZP901.0      __   00.254 _____ _____ _____ _____ _____ _____ _____
_ SCZP901.1      __   _____ _____ _____ _____ _____ _____ _____ _____
******************************* Bottom of data ********************************
```

Figure 9-5 Specify unit address and range for CU

Press Enter twice. You will be returned to the control unit list panel.

Note: The control unit is now attached to the CHPID.

Use option S to select the attached devices, as shown in Figure 9-6 on page 188.

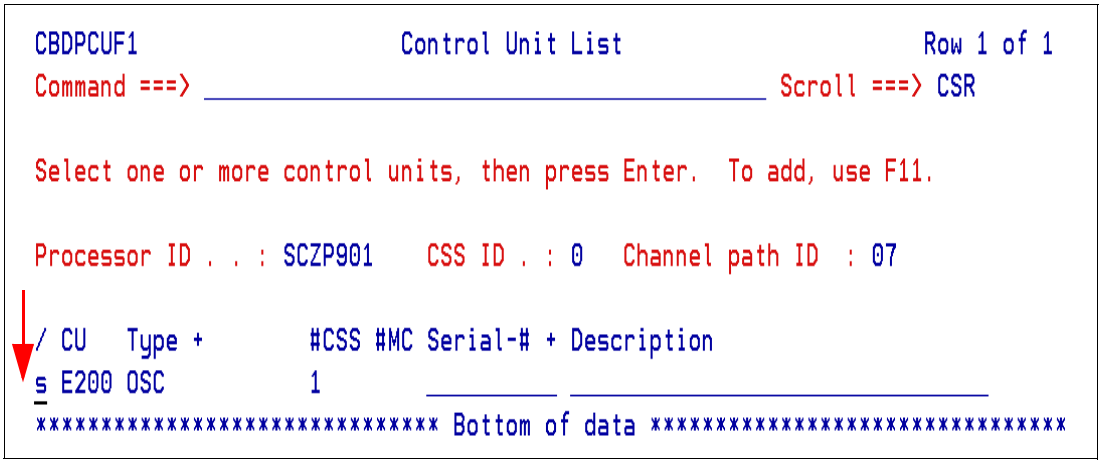

Figure 9-6 Select the attached devices

There are no existing devices. Press PF11 to add devices to this control unit, as shown in Figure 9-7. We will define 120 devices, which is the maximum allowed for the CHPID. The required device type must be 3270-X.

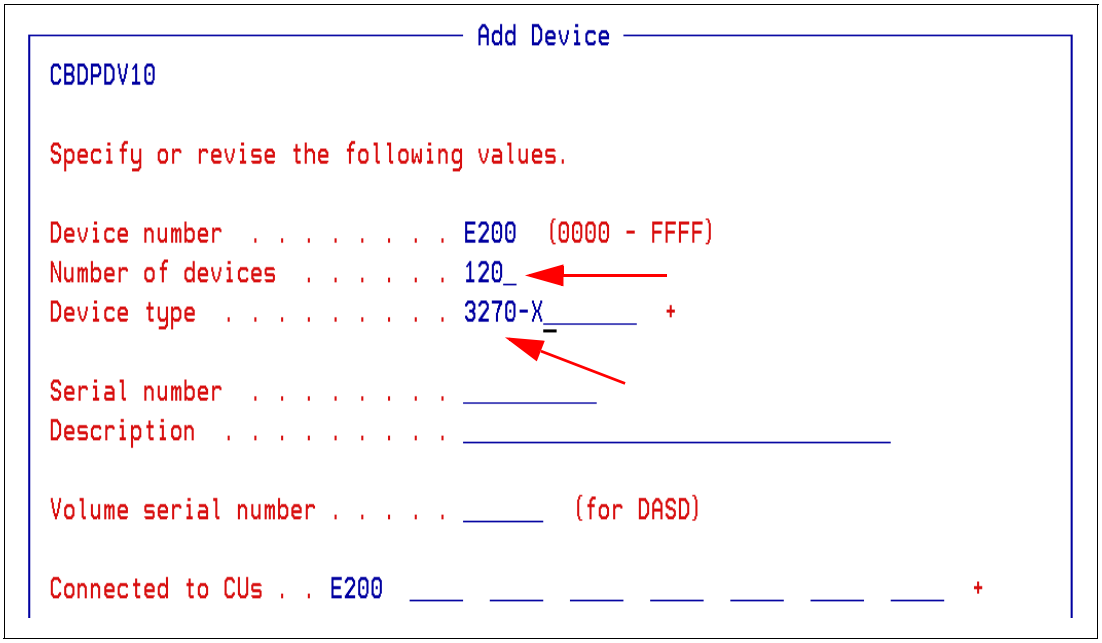

Figure 9-7 Add the device to the control unit

Press Enter. On the next panel (Figure 9-8 on page 189) fill in the starting unit address (UA) as 00.

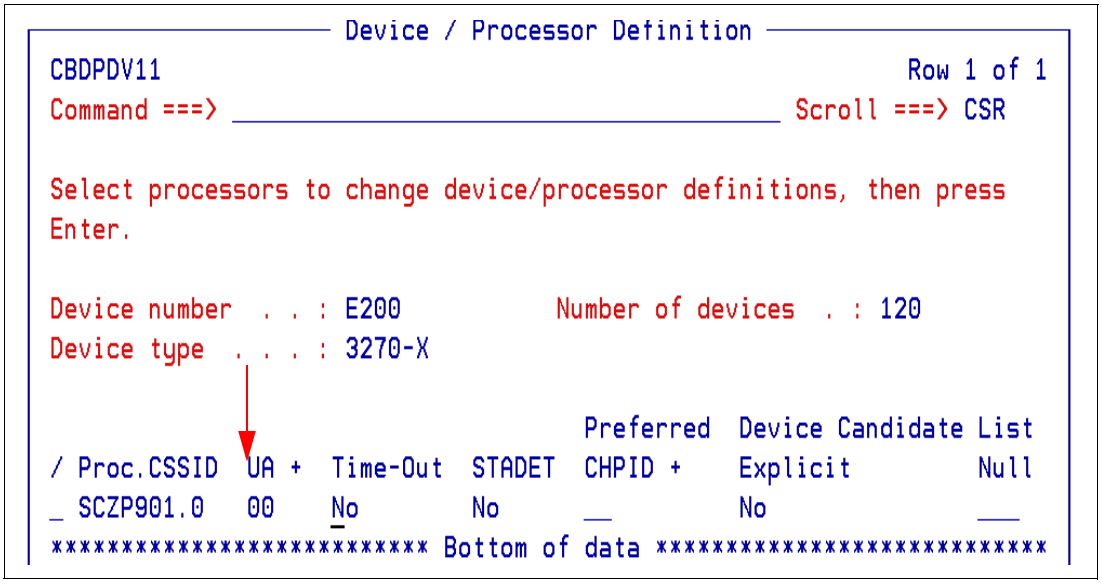

Figure 9-8 Starting unit address

Press Enter. We now connect these devices to one of the Operating System (OS) configurations using the S option, as shown in Figure 9-9.

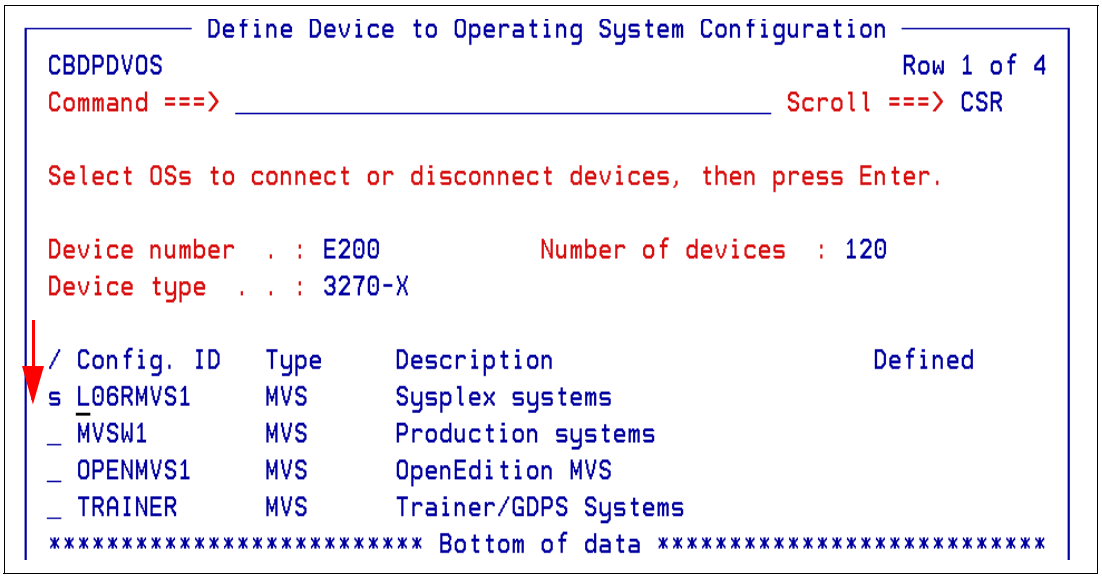

Figure 9-9 Define the device to OS configuration

Press Enter. On the Define Device Parameters panel, change OFFLINE to No and LOCANY to No, as shown in Figure 9-10 on page 190.

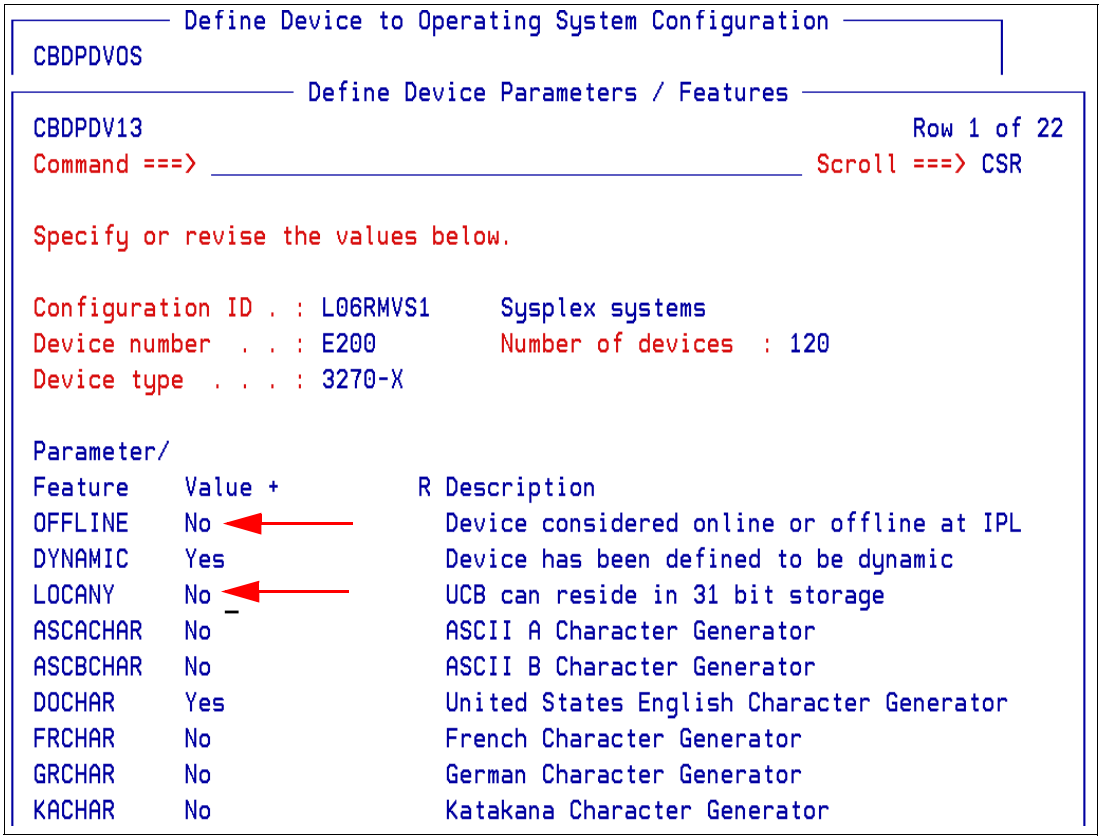

Figure 9-10 Define device parameters

Press Enter. We are taken to the Assign/Unassign Device to Esoteric panel. We make no assignments to any esoterics for these devices, so just press Enter again. The Define Device to Operating System Configuration panel is displayed, as shown in Figure 9-11.

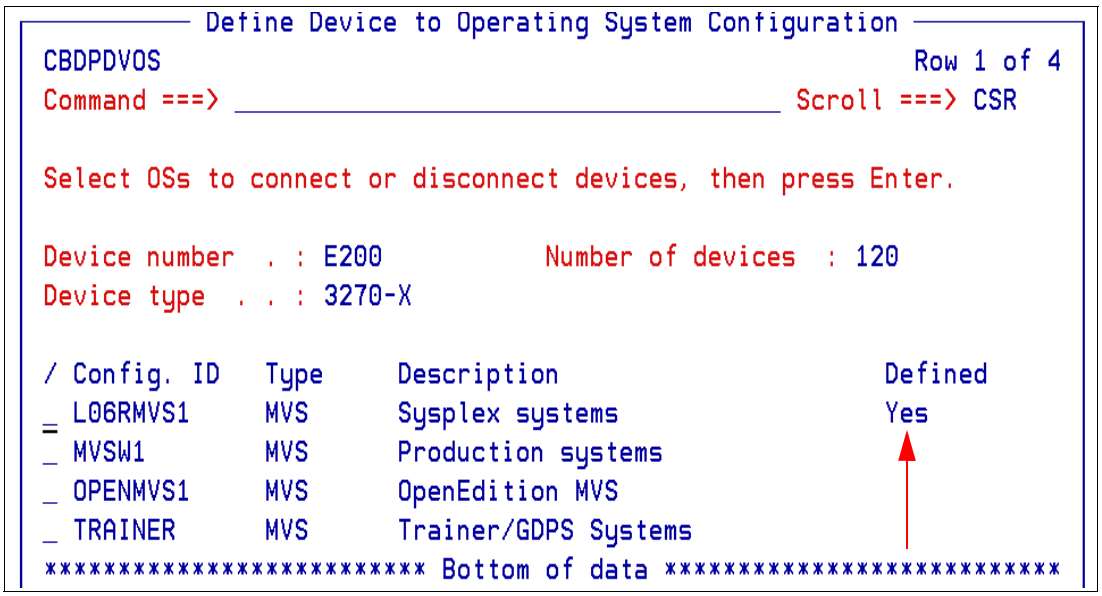

Figure 9-11 Define device to OS configuration

The new devices have been defined to the L0GRMVS1 OS configuration. Press Enter and we now see our newly defined devices (Figure 9-12).

```
CBDPDVF1                        I/O Device List        Row 1 of 120 More:       >
Command ===> _____________________________________________ Scroll ===> CSR

Select one or more devices, then press Enter.  To add, use F11.

Control unit number  : E200     Control unit type  . : OSC

  -------Device------- --#--- --------Control Unit Numbers + --------
/ Number Type +        CSS OS 1--- 2--- 3--- 4--- 5--- 6--- 7--- 8--- Base
_ E200   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E201   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E202   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E203   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E204   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E205   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E206   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E207   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E208   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E209   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E20A   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E20B   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E20C   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E20D   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E20E   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E20F   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E210   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
_ E211   3270-X        1   1  E200 ____ ____ ____ ____ ____ ____ ____
```

Figure 9-12 I/O device list

We have now completed the CHPID, CU and I/O device definitions. Complete the HCD process by creating a new production IODF and write a new IOCDS.

Note: At present a dynamic I/O activation for OSC is restricted, so a POR is required to activate the configuration. Also remember to add the devices that you are going to use for MCS consoles to NIPCON.

9.4.1 IOCP generated statements

The same HCD process was used to convert CHPID 07 in LCSS 1 assigned to PCHPD381 from OSD to OSC, define control unit E300, and devices E300-E377. Listed in Figure 9-13 on page 192 are the IOCP statements generated by HCD for the configuration of the LPARS, both OSC CHPIDs, and the associated control units and device definitions.

```
ID NAME=SCZP901,UNIT=2084,MODEL=A08,SERIAL=026A3A2084,          *
MODE=LPAR,LEVEL=H040331,SNAADDR=(USIBMSC,SCZP901)

RESOURCE PARTITION=((CSS(0),(A0A,A),(A0B,B),(A0C,C),(A0D,D),(A*
      0E,E),(A0F,F),(A01,1),(A02,2),(A03,3),(A04,4),(A05,5),(A*
      06,6),(a07,7),(A08,8),(A09,9))),MAXDEV=((CSS(0),64512))

RESOURCE PARTITION=((CSS(1),(A1A,A),(A1B,B),(A1C,C),(A1D,D),(A*
      1E,E),(A1F,F),(A11,1),(A12,2),(A13,3),(A14,4),(A15,5),(A*
      16,6),(A17,7),(A18,8),(A19,9))),MAXDEV=((CSS(1),64512))

CHPID PATH=(CSS(0),07),SHARED,                                  *
      PARTITION=((A0A,A0B,A0C,A01,A02,A03,A04,A05,A06,A07,A08,*
      A09),(=)),DESC='1000BaseT',TYPE=OSC,PCHID=380

CHPID PATH=(CSS(1),07),SHARED,                                  *
      PARTITION=((A1A,A1B,A11,A12,A13,A14,A15,A16,A17,A18,A19)*
      ,(=)),DESC='1000BaseT',TYPE=OSC,PCHID=381

CNTLUNIT CUNUMBR=E200,PATH=((CSS(0),07)),UNIT=OSC
IODEVICE ADDRESS=(E200,120),UNITADD=00,MODEL=X,CUNUMBR=(E200),*
UNIT=3270

CNTLUNIT CUNUMBR=E300,PATH=((CSS(1),07)),UNIT=OSC
IODEVICE ADDRESS=(E300,120),UNITADD=00,MODEL=X,CUNUMBR=(E300),*
UNIT=3270
```

Figure 9-13 IOCP generated statements

9.5 The HMC configuration process

The work that follows can be done from either the HMC or SE. We chose to use the HMC, and completed all steps using a system programmer ID. From the Defined CPCs Work Area, drag the selected processor to OSA Advanced Facilities, as shown in Figure 9-14 on page 193.

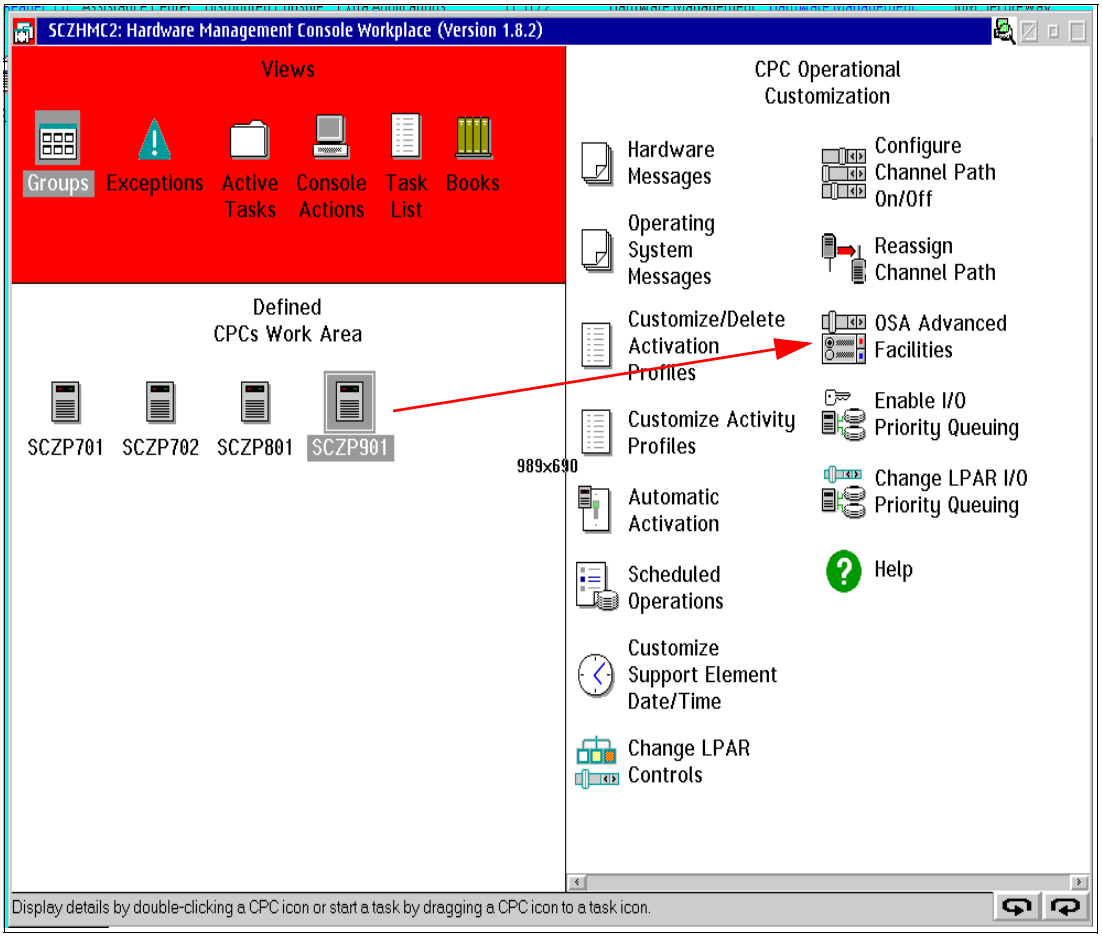

Figure 9-14 HMC OSA Advanced Facilities

Then select the PCHID you wish to define and select **OK**, as shown in Figure 9-15 on page 194. Notice that we used a different PCHID number from the ones defined in the HCD process. You will obviously use the same PCHID numbers defined in your HCD definitions.

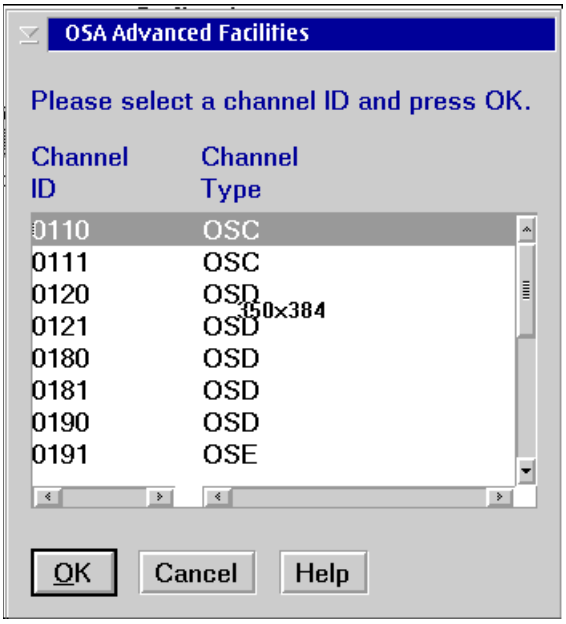

Figure 9-15 Select PCHID

Figure 9-16 is then displayed. Select the **Card specific advanced facilities** option and click **OK**.

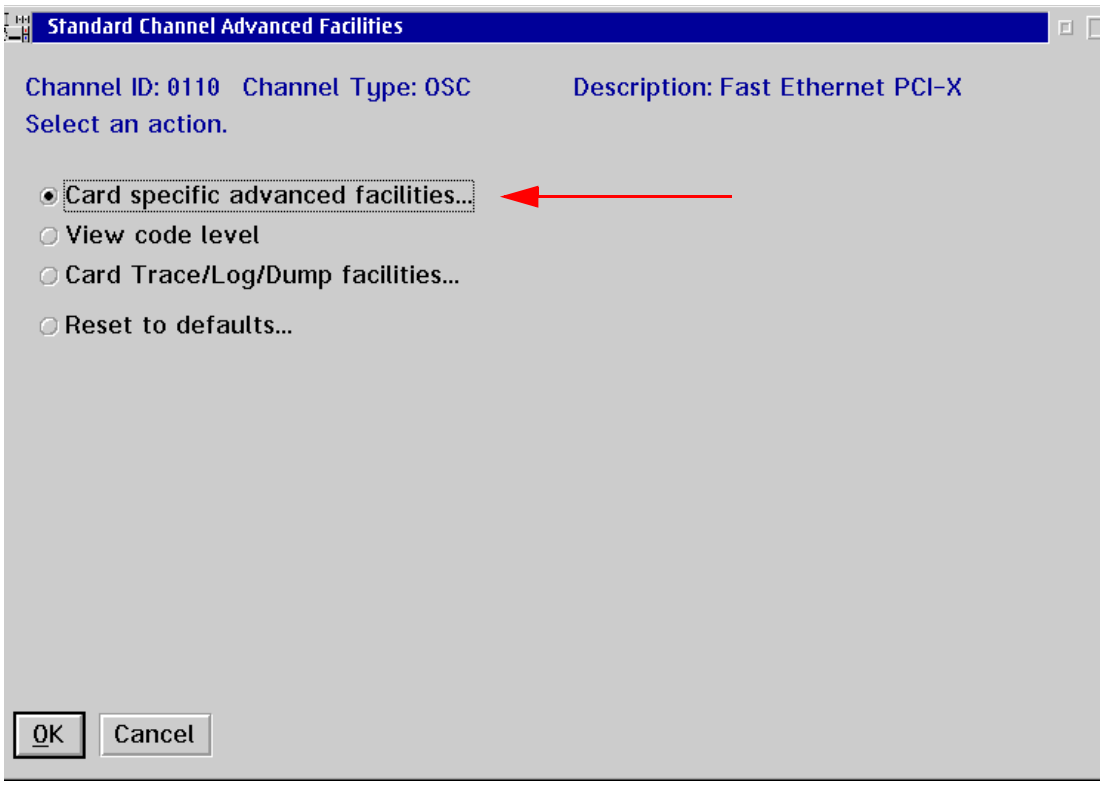

Figure 9-16 Card specific advanced facilities

Figure 9-17 on page 195 is then displayed. Select **Panel configuration options** and click **OK**.

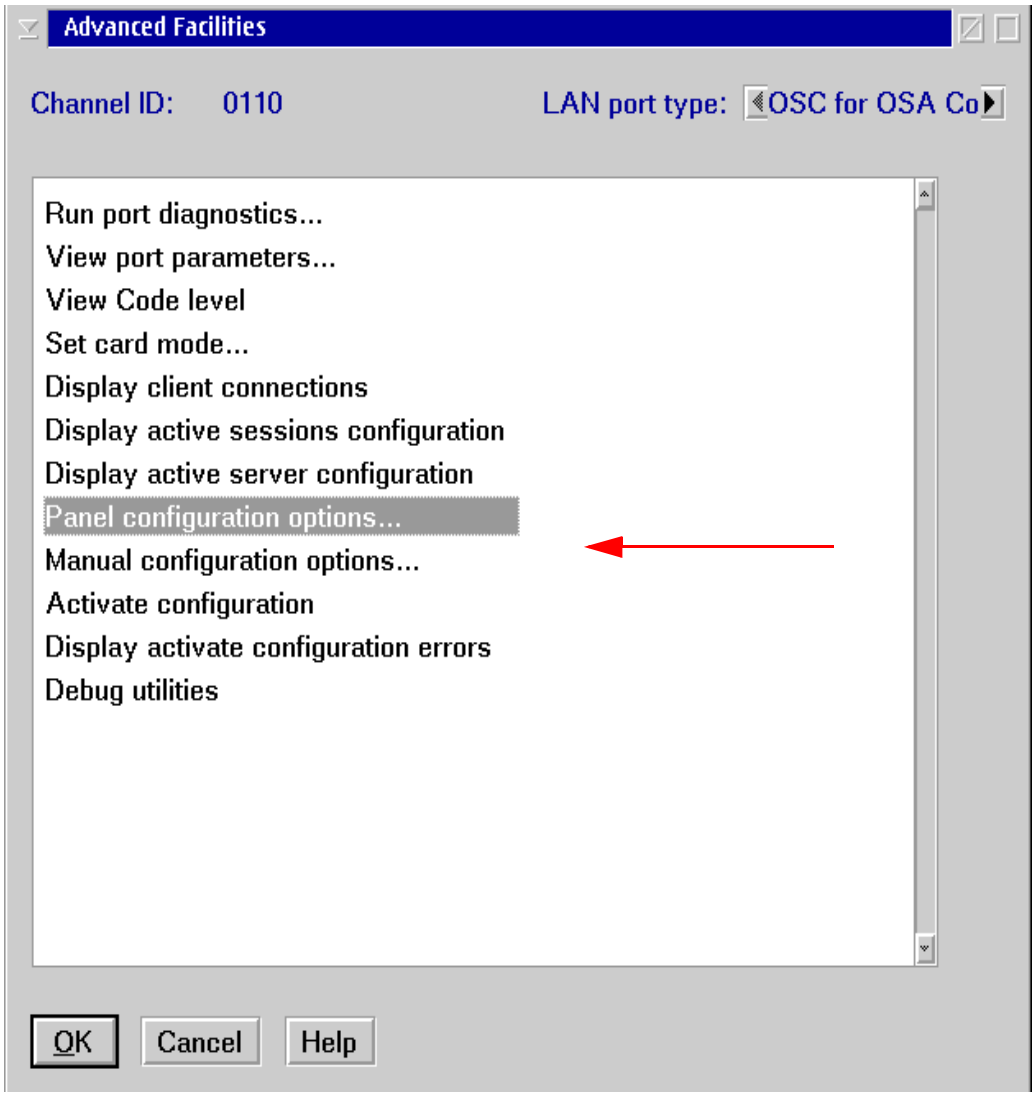

Figure 9-17 Panel configuration options

Figure 9-18 on page 196 is then displayed. Select **Edit server configuration** and click **OK**. This option defines the values used to initialize the TCP/IP stack that is in the OSA-ICC.

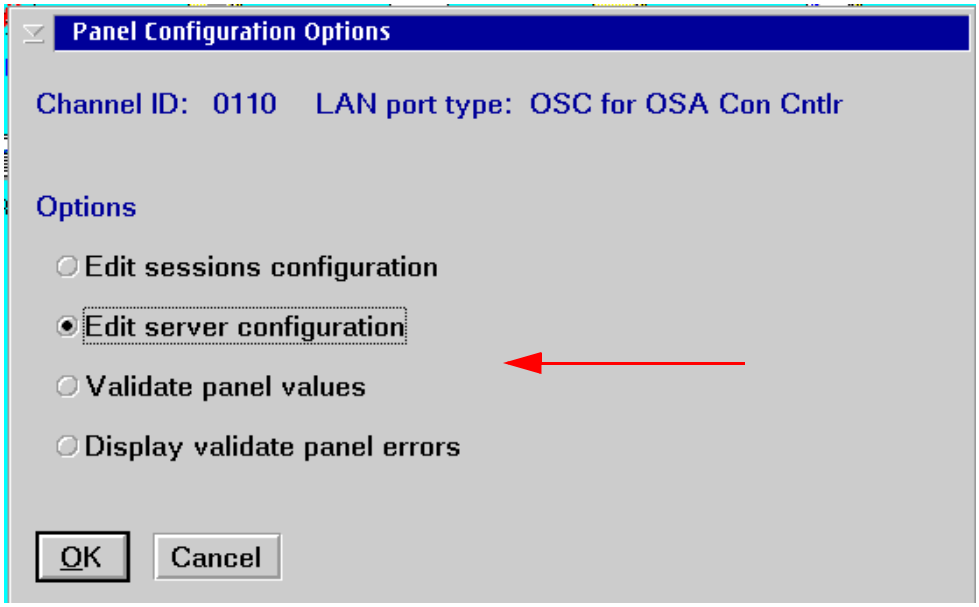

Figure 9-18 Edit server configuration

Figure 9-19 on page 197 is then displayed. The values we inserted were as follows:

Server Name	We are defining the values for CU E000, so we chose the name OSAE000. You will see this name when a PCOM session is connected to this server and the host is not ready for communication.
Host IP address	Use an appropriate address for your installation.
TCP Port	The value you specify here will be used later when defining a PCOM session.
Default Gateway	Use an appropriate address for your installation.
Subnet mask	Use an appropriate address for your installation.
Frame type	The DIX option worked for us.
MTU Size	We used the default value of 1492.

When you press **OK**, you will see a pop-up confirming that `The Command Completed`.

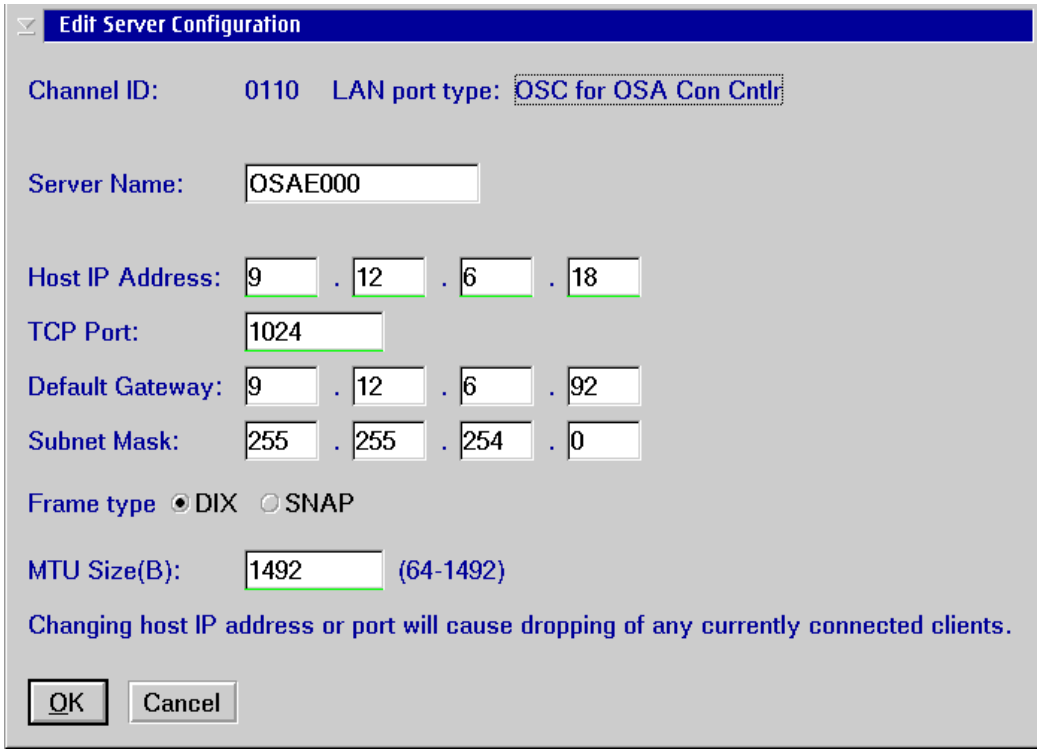

Figure 9-19 LAN server configuration

We now select the **Edit session configuration** option from the Panel Configurations Options menu previously shown in Figure 9-18 on page 196. This takes us to Figure 9-20 on page 198. We are defining the downstream PCOM sessions that connect to this OSA-ICC PCHID.

Notice that the first two sessions have already been defined.

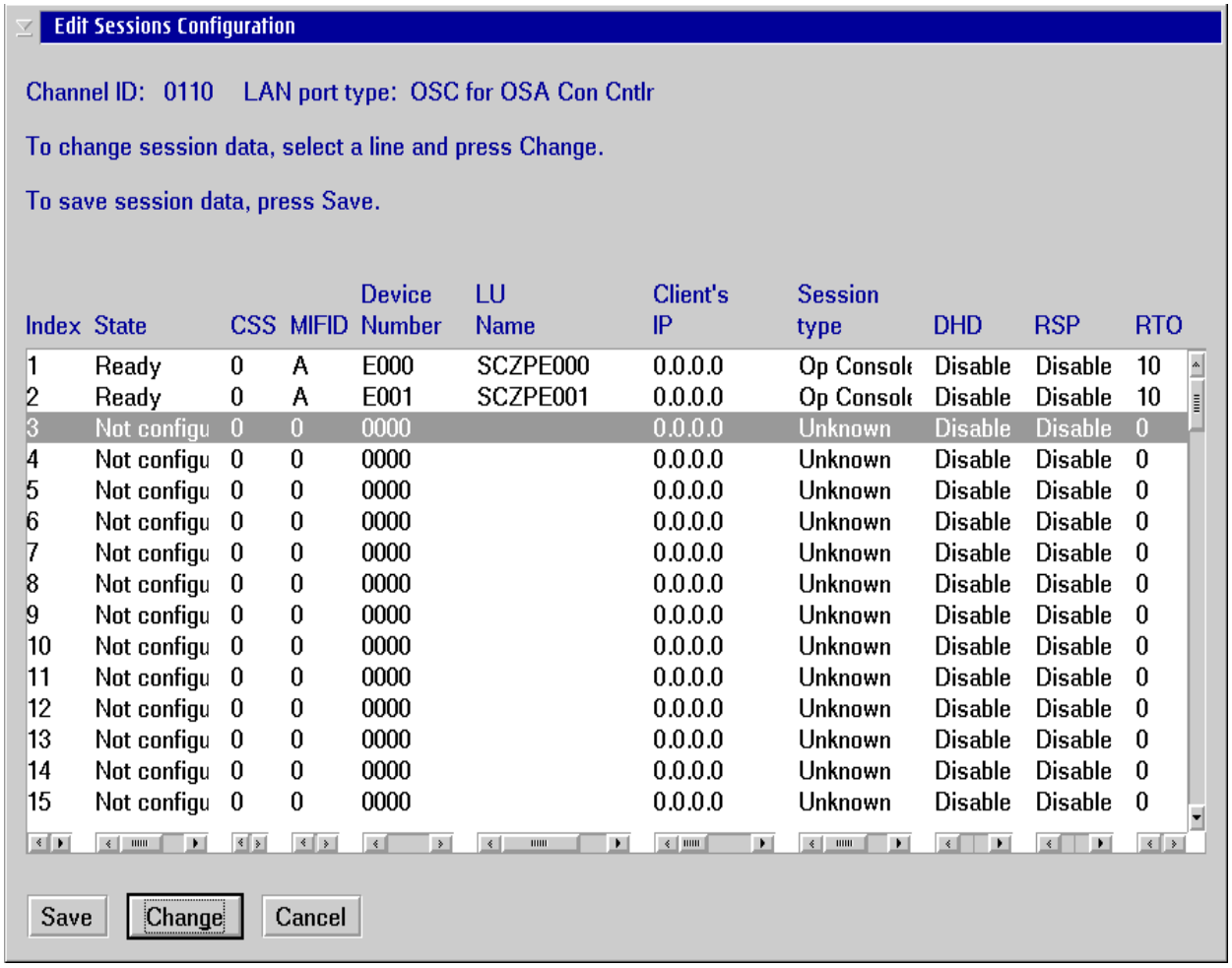

Figure 9-20 Edit sessions configuration

To define another session, highlight the line and click **Change**. This takes you to the Edit Session Configuration panel displayed in Figure 9-21 on page 199.

Here is an explanation of the values we used:

CSS Value The LCSS number (0 in this example).

MIF ID The LPAR identifier between 1 and F within the LCSS that will communicate with this device.

Device Number One of the devices defined previously using HCD.

LU Name A name that will be used when defining the PCOM session to connect to this session.

Client's IP Address It seems that 0.0.0.0 will allow any client to connect. A specific value will restrict the connection to a specific client workstation address.

Session type Use TN3270 for a TSO/E session, and Console for an MCS console session

Defer Host Disconnect Using `disable` worked for us.

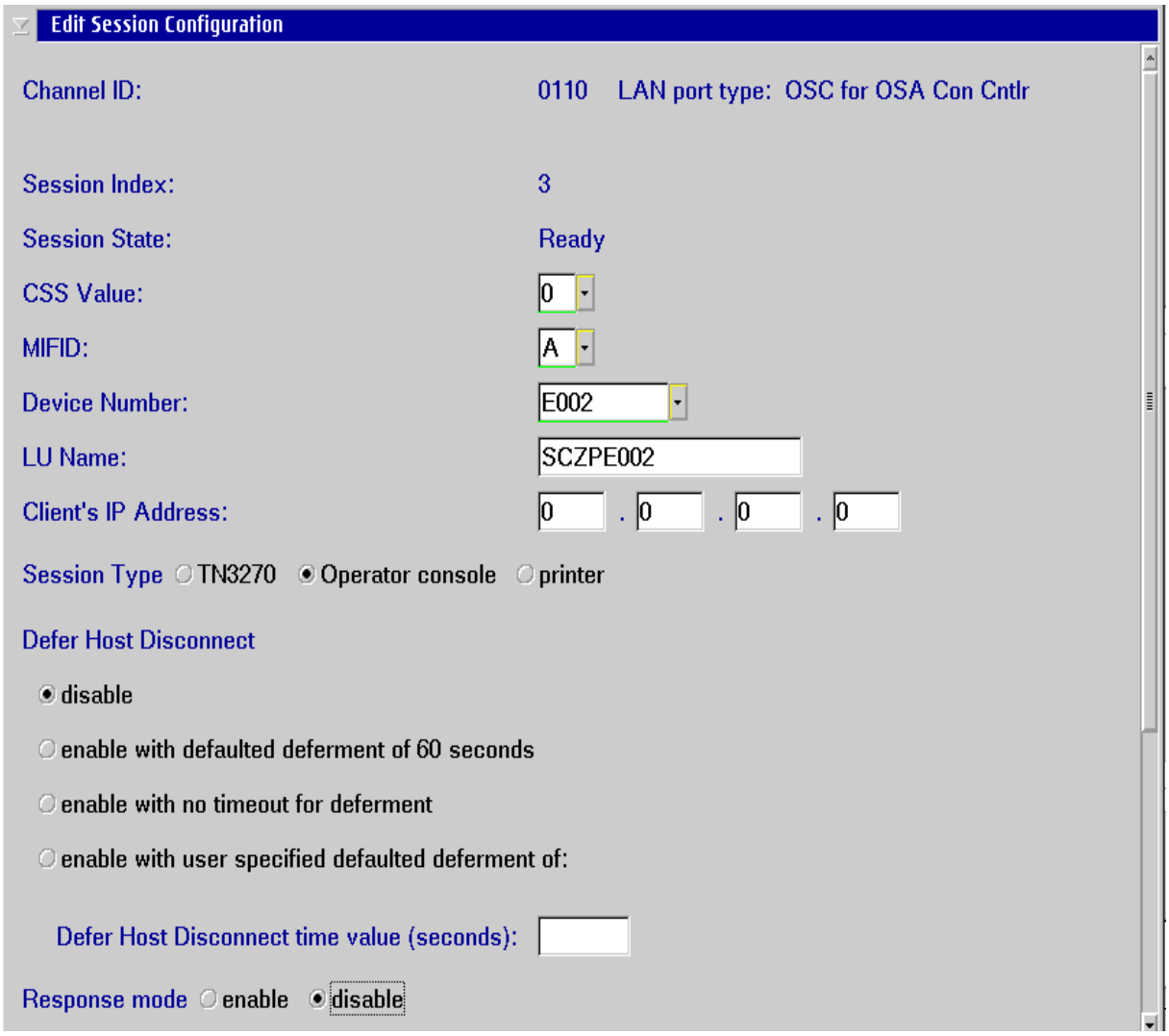

Figure 9-21 Edit session configuration panel

Scroll down the see the remainder of the panel, as shown in Figure 9-22 on page 200.

Select **disable** for the Response mode and select **medium** for the Read Timeout value.

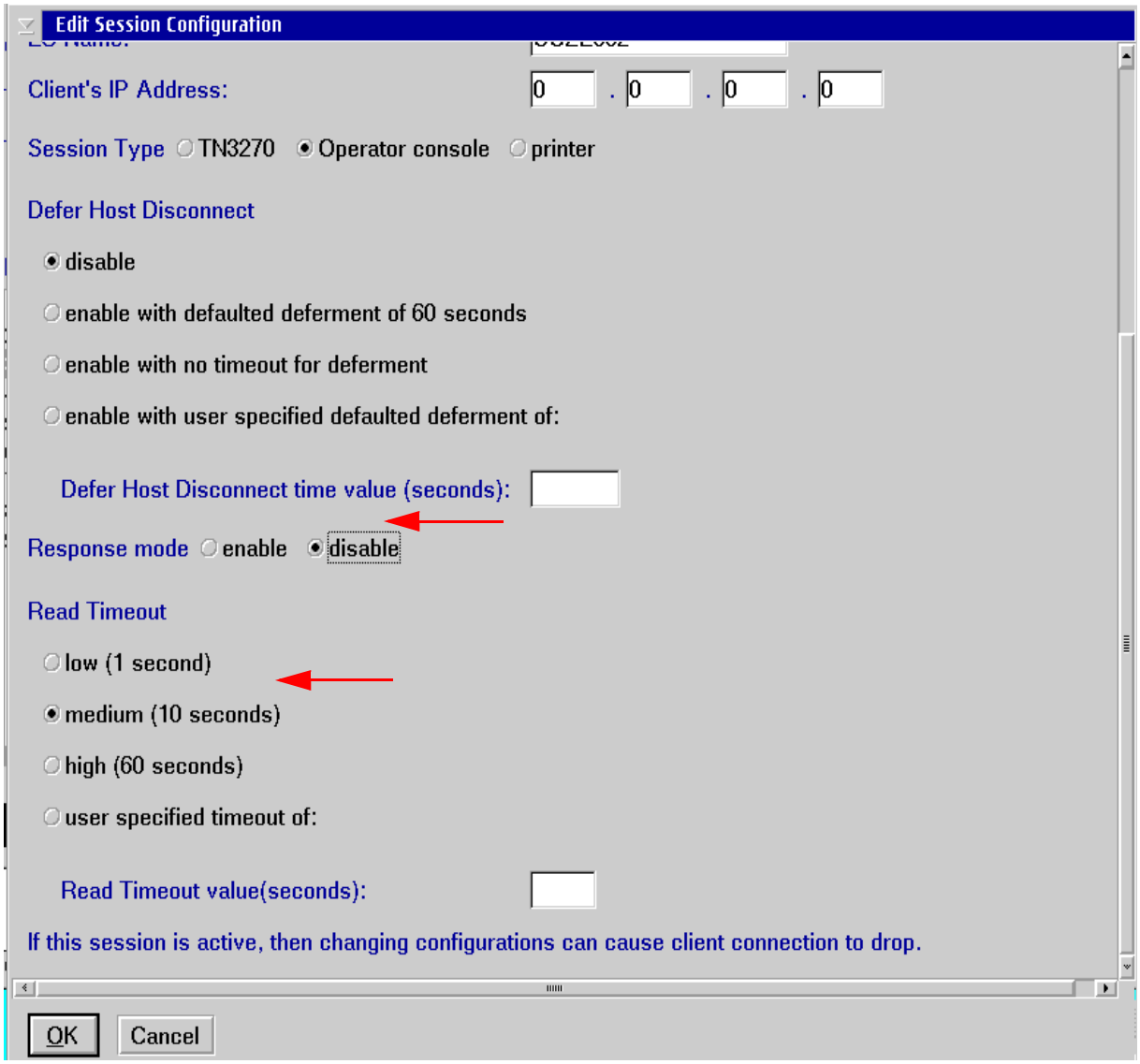

Figure 9-22 Edit session configuration panel

Thereafter, click **OK**. Figure 9-23 on page 201 is displayed, showing the updated session information.

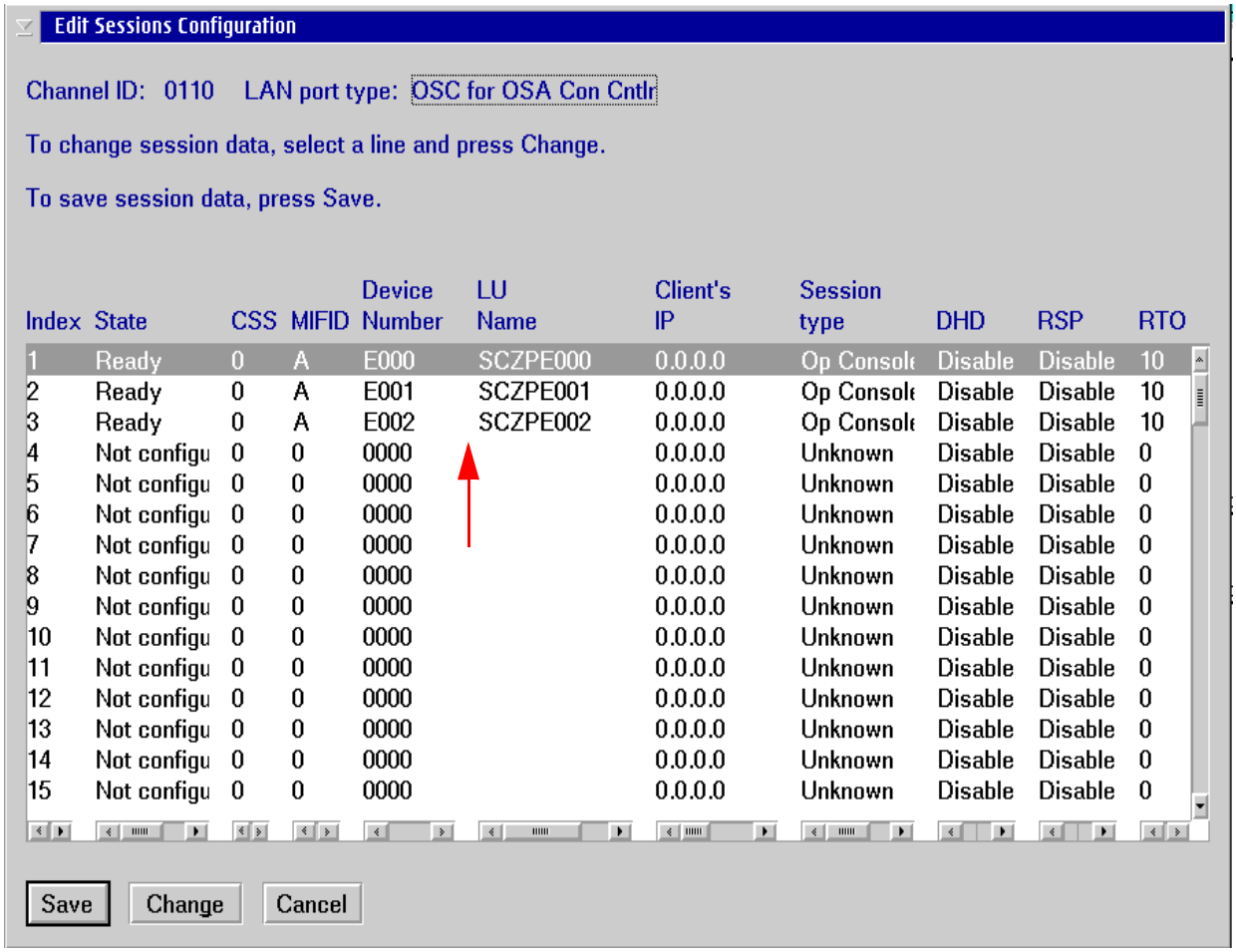

Figure 9-23 Edit sessions configuration panel

Note: Make sure that you save the configuration by clicking **Save**.

We are now back at the Panel Configurations Options panel, as shown in Figure 9-24 on page 202. Select the **Validate panel values** option. If all goes well you will get a pop-up saying that the command completed successfully. If there are errors, use the **Display validate panel errors** option to see the problems. Once they are fixed, validate again.

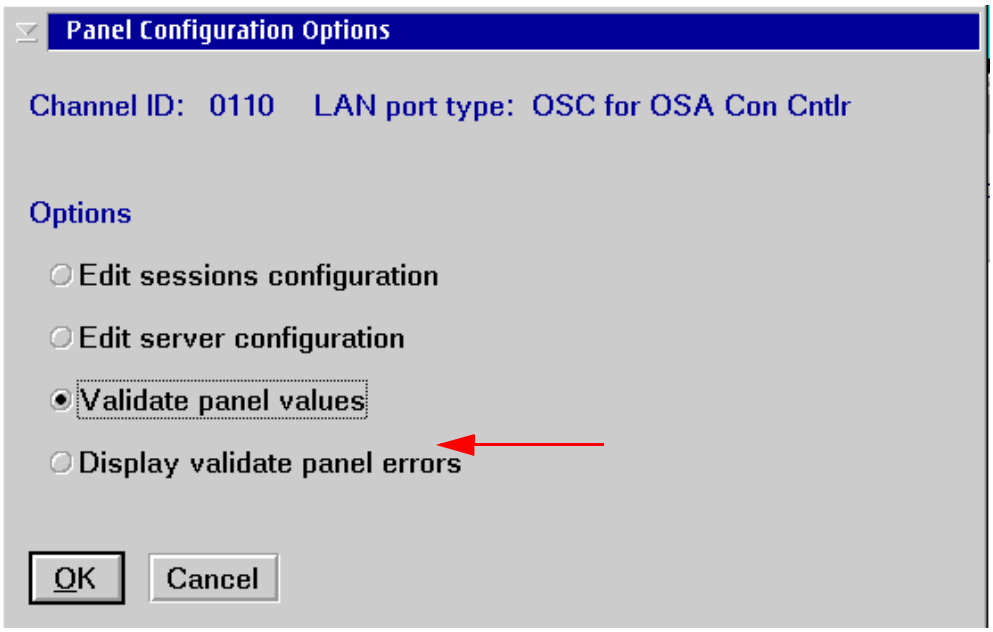

Figure 9-24 Panel Configuration Options

After a successful validation, cancel from the Panel Configuration Options and return to Advanced facilities. You should have a display as in Figure 9-25 on page 203. It is now time to activate the configuration.

Select **Activate configuration** and press **OK**. You should see a pop-up indicating that the command completed successfully.

Note: There is no need to configure the PCHID offline and then online to pick up the new configuration.

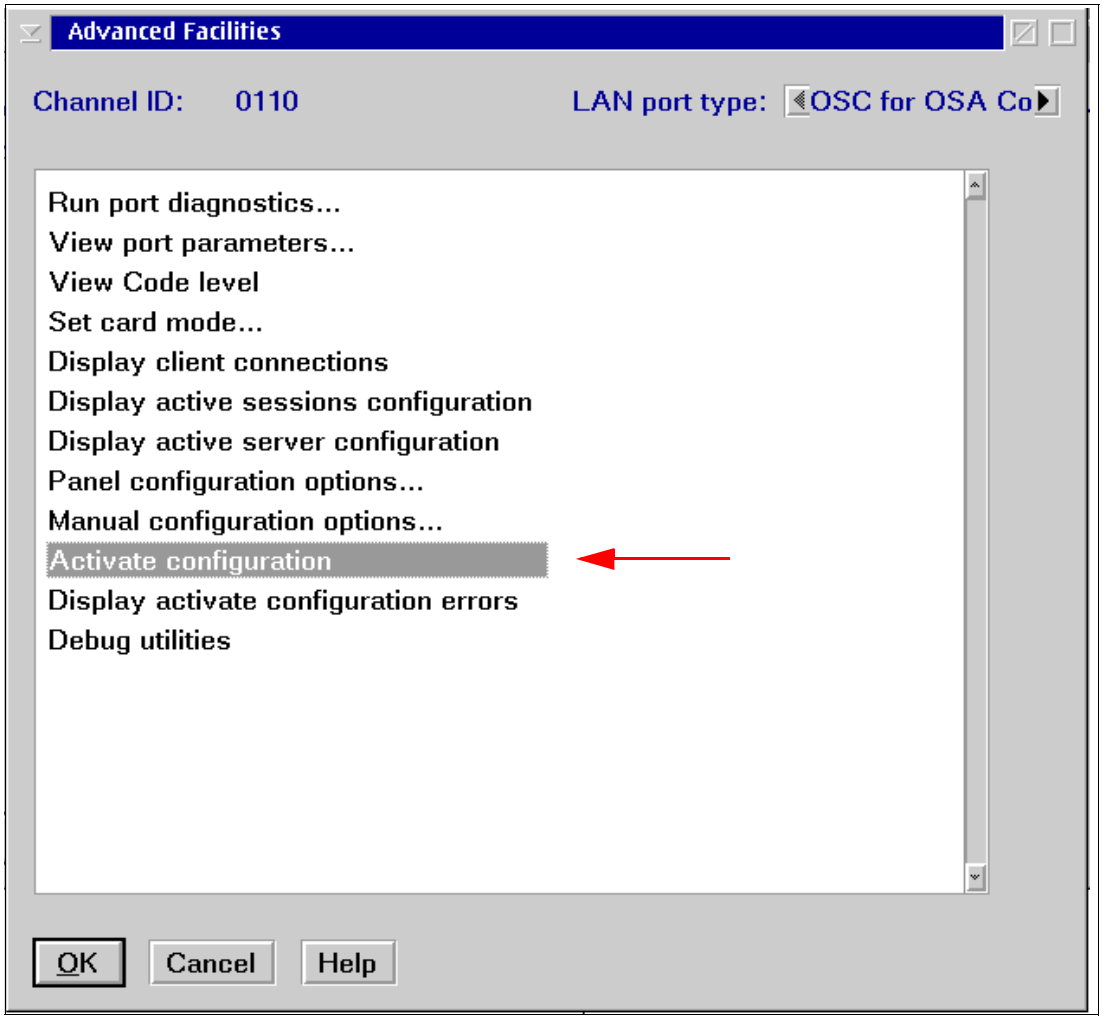

Figure 9-25 Advanced facilities

This completes the HMC configuration. Cancel all the way out of OSA customization.

9.6 PCOM customization

You can use any TN3270 emulation software. We chose PCOM V5.6 for Windows. Defining the PCOM session is very simple. Your panels may look a little different from Figure 9-26 on page 204.

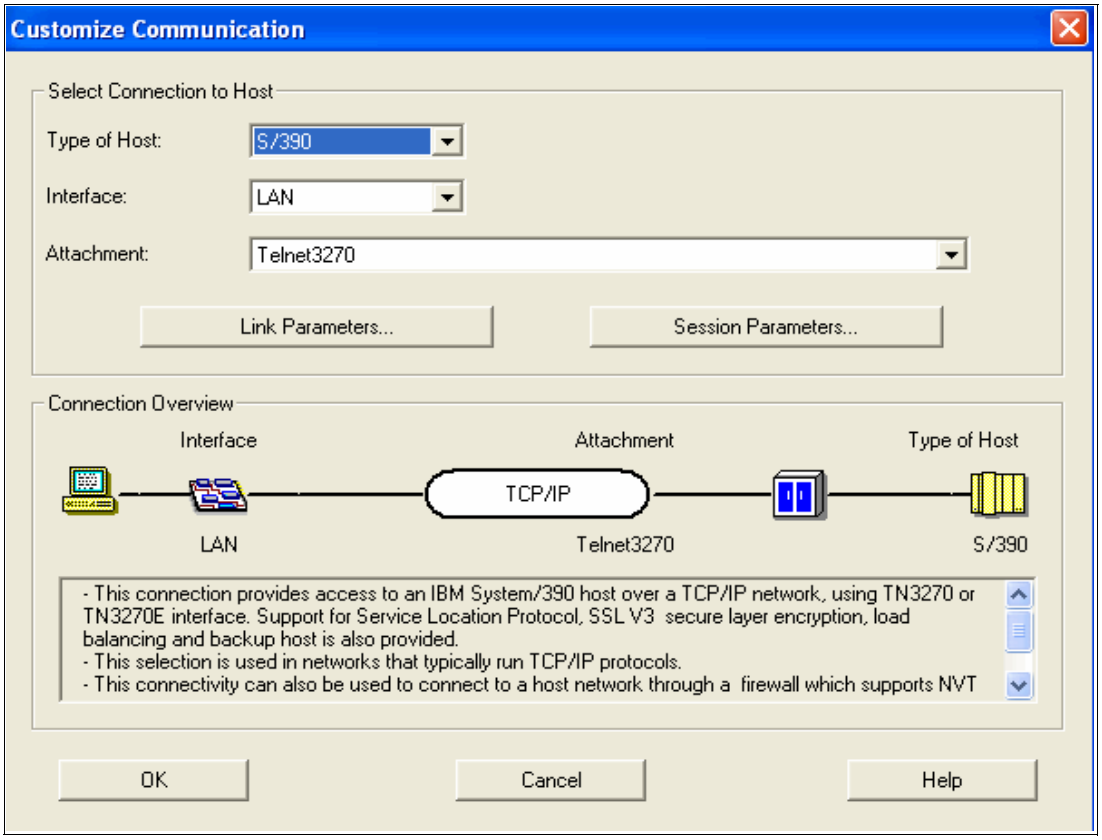

Figure 9-26 PCOM customization

We need to define an Ethernet-attached session to the host. Click **Link Parameters**. This will take us to Figure 9-27 on page 205. Here we define the connection from the workstation to the OSA-ICC server.

The values in Host Name or IP Address and Port Number were specified when defining the OSA-ICC server configuration.

The value in LU or Pool Name was specified when defining the session configuration.

When complete, click **OK** on the Telnet3270 panel. This causes PCOM to initiate the connection to the host.

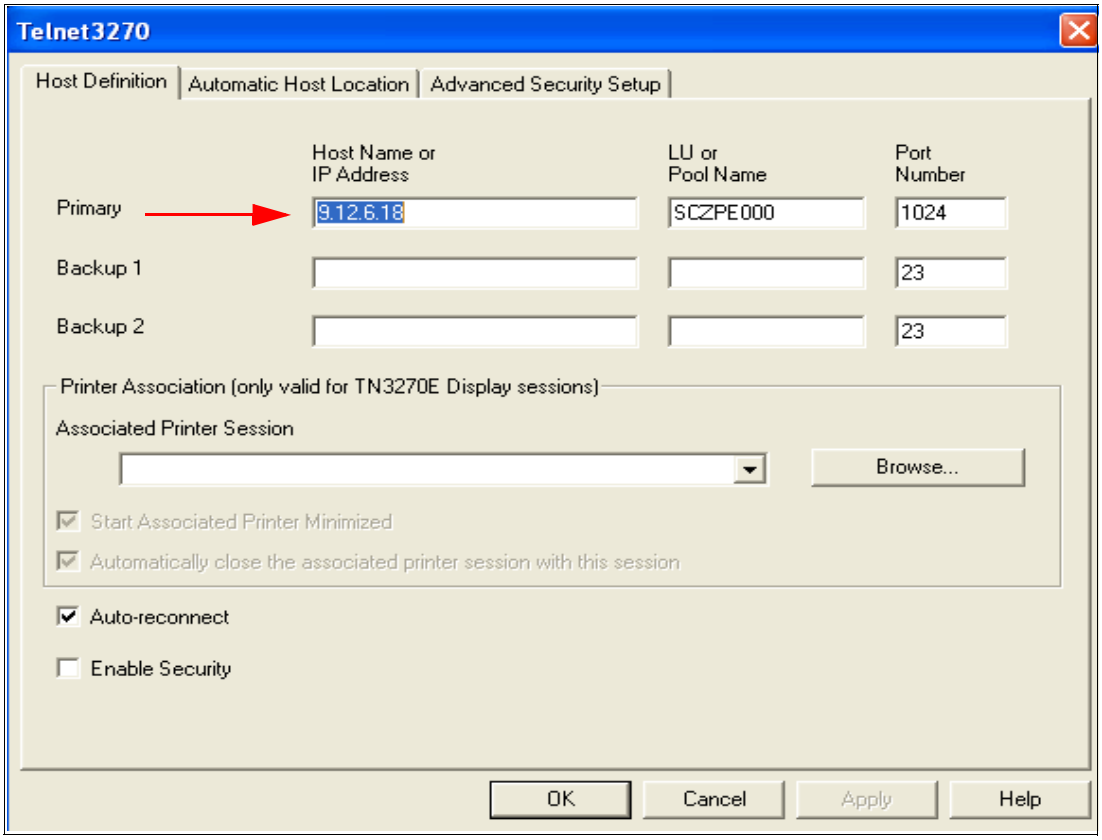

Figure 9-27 Telnet3270 host definition

If the host session is not ready for communication, the screen displayed will show connection information for this session as displayed in Figure 9-28 on page 206. An explanation of each line follows:

Line 1 OSAE000 is the defined server name; 9.12.6.18:1024 shows the defined server address and port number.

Line 2 Session index; LCSS number; LPAR number; logical CU number (always 0); unit address for this device; LU name.

Line 3 Information for the connected processor.

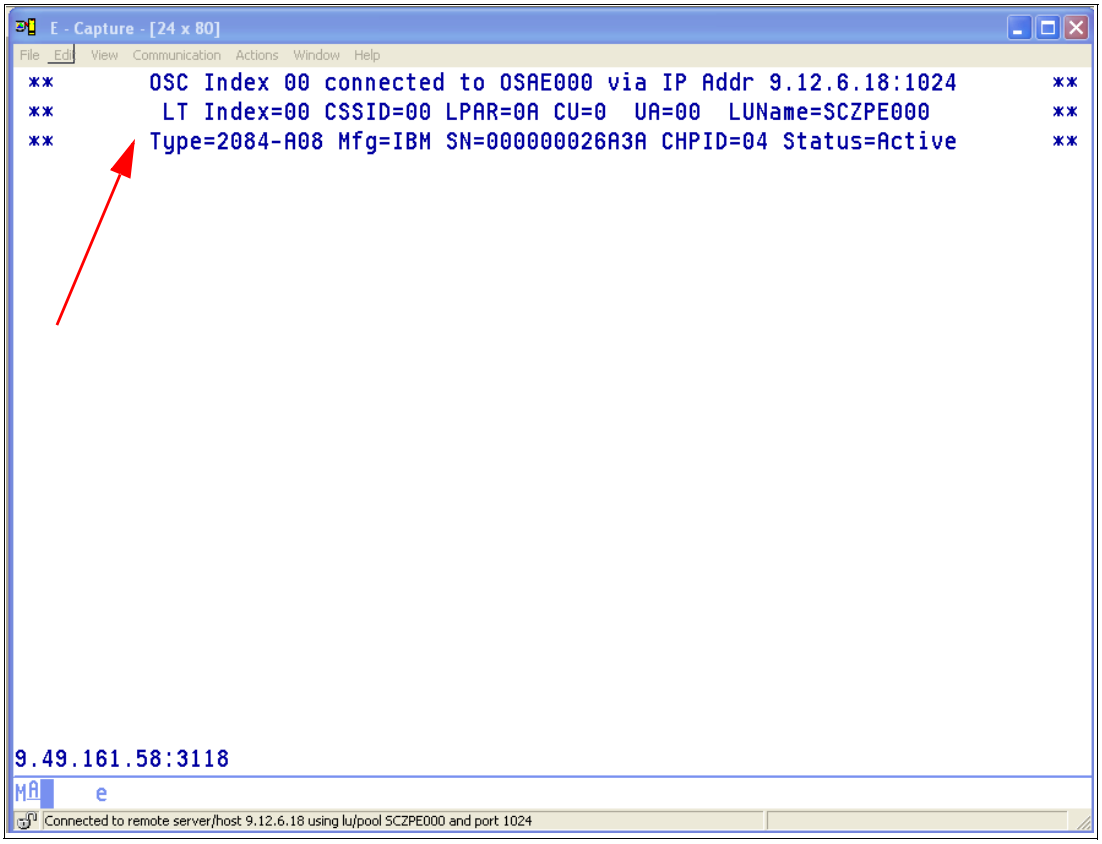

Figure 9-28 TN3270 host session

9.7 Integrated console controller specifications

The following are the OSA-Express 1000BASE-T Ethernet card specifications:

- Up to 120 console sessions per port.
- Port operation is defined with new CHPID type OSC.

  OSC is mutually exclusive with QDIO (OSD) or non-QDIO (OSE) CHPIDs on port.
- One or both ports can be individually configured for Console Controller Support.
- Spanned channels allow ports sharing among all z990 LCSS and LPARS.

The following are the LAN connection specifications:

- Supports only LAN-attached consoles running TN3270E or TN3270 clients.
- Is capable of operating at 10, 100, or 1000 Mbps (1 Gbps).
- Token ring is not supported.

Configuration support is provided via the Support Element or Hardware Management Console.

10

z/OS V1R6 ISPF enhancements

This chapter describes the various enhancements to ISPF for z/OS V1R6. The enhancements are aimed at improving end-user productivity.

This chapter covers the following topics:

- File tailoring enhancements
- REXX support for panel procedures
- ISPF EDIT enhancement
- ISPF services enhancements
- ISPF configuration changes
- SCLM enhancements

10.1 File tailoring enhancements

ISPF skeleton definitions are stored in a skeleton library and accessed through the ISPF file-tailoring services. Skeletons are created or changed by editing directly in the skeleton library. ISPF interprets the skeletons during execution. No compile or preprocessing step is required.

There are two types of records that can appear in the skeleton file:

Data records These are a continuous stream of intermixed text, variables, and control characters that are processed to create an output record.

Control statements Control the file-tailoring process. Control statements start with a right parenthesis in column 1. Records containing a “)” in column 1 and a blank in column 2 are interpreted as data records. Records containing a “)” in column 1 and a non-blank character in column 2 are interpreted as control statements. A)DEFAULT control statement can be used to assign different special characters for syntactical purposes.

Under z/OS V1R6, new control statements have been added for ISPF file tailoring. These are now described.

10.1.1 File tailoring—iterative processing support

The following four new control statements are added to ISPF file tailoring to support iterative (or loop) processing defined within a file tailoring skeleton:

-)DO
-)ENDDO
-)LEAVE
-)ITERATE

The)DO statement must be terminated by an)ENDDO statement, which must appear in the same skeleton (or imbed) member and at the same select level. Use of the)LEAVE and)ITERATE statements is optional.

)DO and)ENDDO statements

The skeleton input records between the)DO and the corresponding)ENDDO statements are repeatedly processed until a condition causes the)DO loop to terminate. Processing then continues with the input record immediately following the)ENDDO statement. The processing of a)DO loop can be prematurely ended using the)LEAVE statement, or the current iteration of the)DO loop can be terminated using the)ITERATE statement. Figure 10-1 shows the syntax of the)DO statement.

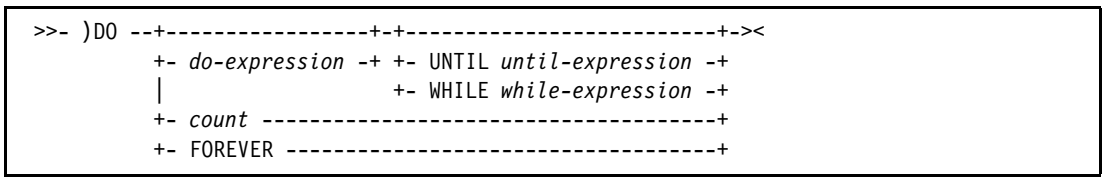

Figure 10-1 Syntax of)DO statement

do-expression This is specified as ***var* = *n* [TO *m*] [BY *incr*] [FOR *cnt*]**

where:

var	Control variable name
n	Starting value
m	Ending value
incr	Increment value
cnt	Maximum number of iterations
until-expression	This is a relational expression that is evaluated for a true or false condition. The)DO loop will continue while the *until-expression* evaluates to a false condition. The test is performed at the end of each loop prior to updating the control variable. The loop is always performed at least one time.
while-expression	This is a relational expression that is evaluated for a true or false condition. The)DO loop will continue while the *while-expression* evaluates to a true condition. The test is performed at the start of each loop, once the control variables are initialized.
count	This is an integer used to control the number of iterations of the)DO loop. The number can be either positive or negative in the range -2147483648 to 2147483647. If the *count* is less than 1, the)DO statement is skipped.
FOREVER	Continues processing the)DO loop until a)LEAVE statement within the loop terminates the)DO loop. All other parameters are ignored when using the FOREVER parameter. File tailoring makes no attempt to determine if a)DO FOREVER loop can be suitably terminated.

There are several variations of syntax supported for the)DO statement:

-)DO *do-expression*
-)DO *do-expression* WHILE *while-expression*
-)DO *do-expression* UNTIL *until-expression*
-)DO UNTIL *until-expression*
-)DO WHILE *while-expression*
-)DO FOREVER
-)DO *count*

Figure 10-2 on page 210 shows some examples.

Example 1: This example performs a loop 10 times with the control variable, I, starting at 1 and increasing by 1 each time. The control variable will have the value 11 at the end of the loop.

```
)DO I = 1 to 10
.....
)ENDDO
```

Example 2: This example shows that the)DO loop continues till the variable EOF is nonzero.

```
)SET EOF = 0
)DO WHILE &EOF = 0
.....
)ENDDO
```

Example 3: This example shows that the)DO loop continues till the variable EOF is zero.

```
)SET EOF = 9
)DO UNTIL &EOF = 0
.....
)ENDDO
```

Example 4: This example shows that)DO loop continues till the variable EOF is zero. The)LEAVE statement terminates the loop.

```
)SET EOF = 9
)DO FOREVER
.....
)IF &EOF = 0 THEN )LEAVE
....
)ENDDO
```

Example 5: This example shows that the)DO loop performs five times.

```
)DO 5
.....
)ENDDO
```

Figure 10-2)DO statement examples

)ITERATE statement

The)ITERATE statement terminates the current iteration of the)DO structure and repeats the loop, providing that any conditions that would cause the loop to terminate have not yet been reached. A severe dialog error occurs if the)ITERATE statement is used outside a)DO structure.

)LEAVE statement

The)LEAVE statement immediately terminates the innermost)DO statement. A severe dialog error occurs if the)LEAVE statement is used outside a)DO structure.

10.1.2 File tailoring—IF-THEN-ELSE processing support

The following three new control statements have been added to support the IF-THEN-ELSE processing:

-)IF
-)ELSE
-)NOP

The syntax of IF-THEN-ELSE statement is shown in Figure 10-3.

)IF *relational-expression* THEN [*control-statement*]

)ELSE [*control-statement*]

Where:

relational-expression is evaluated for a true or false condition.

control-statement is any ISPF file tailoring control statement.

Figure 10-3 Syntax of IF-THEN-ELSE statement

relational-expression This is evaluated for a true or false condition. If the condition is true, then either the *control-statement* on the)IF control statement is processed, or the next non-comment line is processed. A subsequent)ELSE statement, if present, is skipped. If the condition is false, the *control-statement* or next non-comment line is skipped, and the subsequent)ELSE statement, if present, is processed.

control-statement This can be any ISPF file tailoring control statement. However, the)CM (comment) control statement and the remainder of the input record are ignored. Some control statements, namely)DO,)SEL, and)DOT require more than one input record. Similarly, the)IM control statement imbeds another ISPF skeleton member. The processing of the)IF or)ELSE statement is not completed until the *control-statement* specified on the)IF or)ELSE statement is also completed.

The)NOP control statement does not generate any output, but can be used as a null control-statement for either the)IF or)ELSE statements.

Figure 10-4 on page 212 shows an example of the IF-THEN-ELSE statement.

Example 1: This example combines the)IF,)DO and)NOP statements to process a block of input records between)DO and)ENDDO when the variable RC has a value of zero, or do nothing ()NOP) when its value is nonzero.

```
)IF &RC = 0 THEN )DO
. . .
)ENDDO
)ELSE )NOP
```

Example 2: This example shows that when the variable B is equal to or greater than 1, it imbeds another member, whose name is in variable SKEL.

```
)IF &B >= 1 THEN )IM &SKEL
```

Figure 10-4 Example of the IF-THEN-ELSE statement

10.1.3 File tailoring—TBSARG filter for)DOT

This feature allows file tailoring to selectively process rows within an ISPF table. A new SCAN keyword has been added on the)DOT control statement. The syntax is shown in Figure 10-5.

)DOT *table-name* [SCAN [(*name-cond-pairs*)]]

Where:

name-condition-pairs specifies a list of names and conditions for determining the search argument conditions for scanning the table-name.

Figure 10-5 Syntax of)DOT statement

When the SCAN keyword is not provided, each row of the table is processed by the file tailoring services between the)DOT and)ENDDOT keywords. This is the default behavior.

When the SCAN keyword is provided without the additional *name-cond-pairs*, a valid search argument must have already been established for the ISPF table, *table-name*, using the TBSARG service prior to invoking the file tailoring services. This requires the ISPF table to have already been opened. A severe dialog error occurs if the search arguments have not yet been established. The ISPF file tailoring will also recognize the NEXT/ PREVIOUS parameter established on the TBSARG service.

When the *name-cond-pairs* are specified on the SCAN keyword, ISPF file tailoring services uses the variable names and condition values to process the table. The dialog variables must already be initialized to the required values for the TBSCAN service. This can be done from the invoking dialog or using the file tailoring)SET control word. When the ISPF table is not already opened, the file tailoring services open it prior to scanning the table. They also close it when they have finished processing the table. The syntax of the *name-cond-pairs* is exactly the same as for the TBSARG *name-cond-pairs* parameter.

10.1.4 File tailoring—other enhancements

The following miscellaneous enhancements have been implemented for the ISPF file tailoring services:

- Support has been added to ISPF file tailoring to support continuation of control statements over multiple lines, and to increase the maximum number of parameters that can be specified on a control statement from 31 to 63.
- The previously documented maximum number of nested select statements was 8. With the implementation of IF-THEN-ELSE support, the maximum limit for nested IF and SEL statements has been increased from 8 to 32.
- The previously documented maximum number of nested imbed statements was 3. The maximum limit for nested IMBED levels has been increased from 3 to 15.
- With the implementation of)DO loop support, file tailoring will now read a skeleton member into storage, improving performance by eliminating the reading and re-reading of records from disk. Also, by reading the skeleton into storage, the exclusive SPFEDIT enqueue issued for the skeleton member has been removed.

Note: For details, refer to Chapter 10 of *z/OS V1R6.0 ISPF Dialog Developer's Guide and Reference,* SC34-4821.

10.2 REXX support for panel procedures

New panel definition statements are provided to allow the inclusion of REXX code within a panel's)INIT,)REINIT, and)PROC sections. This enables the programmer to use the powers of the REXX language to perform operations such as arithmetic, verification, transformation, translation, and formatting of dialog variables.

The new *REXX panel procedure statement is used to invoke REXX code within a panel procedure. The names of ISPF dialog variables that need to be passed to REXX can be specified as parameters on the *REXX statement. The syntax of the *REXX statement is shown in Figure 10-6.

REXX[([,]*value*,*value*,...[,(*member*)])]

....

[*ENDREXX]

Where:

*	Specifies all ISPF variables defined in the)BODY section are passed to REXX.
value	Specifies the name of an ISPF dialog variable passed to REXX.
member	Specifies the name of a member in the standard search sequence used to load REXX programs.

*Figure 10-6 Syntax of the *REXX statement*

10.2.1 Using the *REXX statement

Specifying an * as the first parameter causes all the dialog variables defined in the)BODY section of the panel to be passed to REXX.

The user has the option of specifying a member name on the *REXX statement. This causes ISPF to look for the member in the standard search sequence (for example, SYSEXEC and SYSPROC DDs) for REXX programs. If found, the REXX program in the member is loaded and invoked. The member can contain either an interpreted or compiled REXX program.

The alternative to providing a member name is to code the REXX statements directly within the panel procedure. When the REXX code is in line, it must be terminated by the *ENDREXX statement.

The ISPF dialog variables that can be processed by panel REXX code are made available via the parameters specified on the *REXX statement.

The ISPF module ISPPRXVP is used to make the ISPF dialog variables specified via the *REXX statement available to panel REXX, and to update the dialog variables after they have been processed by panel REXX. The parameter “I” or “T” is passed to the module ISPPRXVP.

When the parameter “I” is passed, ISPPRXVP sets up corresponding REXX variables for the ISPF dialog variables.

When the parameter “T” is passed, ISPPRXVP updates the ISPF dialog variables with any changes made by the panel REXX.

Compiled REXX

For compiled REXX, it is necessary for the programmer to include the calls to ISPPRXVP in the REXX code.

When the panel REXX is an interpreted REXX (that is, the REXX statements are coded directly in a panel procedure or the member specified on a *REXX statement contains interpreted REXX), ISPF creates calls to ISPPRXVP to perform the following tasks:

- Set up corresponding REXX variables for the ISPF dialog variables before the panel REXX is invoked.
- Update the ISPF dialog variables with any changes made by the panel REXX after it has finished.

ISPF does this by generating the following REXX statements ahead of and after the supplied panel REXX code, as shown in Figure 10-7 on page 215.

```
Call ISPPRXVP 'I‘

if rc!=0 then do
   say 'ISPPRXVP Init failed rc=' rc
   Return

End

Call P_01A2B3C0

Call ISPPRXVP 'T'

if rc!=0 then
   say 'ISPPRXVP Term failed rc=' rc

Return

P_01A2B3C0:

....

panel REXX code

....

Return
```

(**Bold text** indicates REXX generated by ISPF.)

Figure 10-7 Generated REXX statements

Panel REXX code

Panel REXX code cannot issue requests for ISPF services. The ISPEXEC interface terminates a call to an ISPF service when the call comes from panel REXX.

The example shown in Figure 10-8 shows the use of panel REXX (inline REXX) to display the current user ID, system name, and sysplex name.

```
 )PANEL
)ATTR DEFAULT(%+_) FORMAT(MIX)
 # TYPE(OUTPUT) INTENS(HIGH)
)BODY
+                   %Who Am I?+                              +
+
+User#USR     +logged on system#SYSM    +on SYSPLEX#SPLEX   +
+
)INIT
*REXX(*,ZUSER)
 usr = ZUSER
 sysm = MVSVAR('SYSNAME')
 splex = MVSVAR('SYSPLEX')
*ENDREXX
)PROC
)END
```

Figure 10-8 Example of using inline REXX

*REXX statement

The *REXX statement is used to invoke REXX coded directly in the)INIT procedure section of the panel. The "*" parameter on the *REXX statement causes all the ISPF dialog variables defined in the)BODY section (that is, USR, SYSM, and SPLEX) to be made available for processing by the REXX code. Also, the ISPF system variable ZUSER is passed to REXX. The REXX code obtains and sets values for the variables displayed on the panel.

The example shown in Figure 10-9 is similar to the previous example but uses an external REXX program to process the panel variables.

```
)PANEL
)ATTR DEFAULT(%+_) FORMAT(MIX)
 # TYPE(OUTPUT) INTENS(HIGH)
)BODY
+                    %Who Am I?+                              +
+
+User#USR     +logged on system#SYSM    +on SYSPLEX#SPLEX   +
+
)INIT
*REXX(*,ZUSER,(PWHO))
)PROC
)END
```

Figure 10-9 Example using external REXX

In this example, the REXX program is in the member PWHO found in either the SYSEXEC or SYSPROC DD concatenations. The "interpreted REXX" version of PWHO is the same as the inline REXX code in the example shown in Figure 10-8 on page 215.

The "compiled REXX" version of PWHO contains calls to ISPPRXVP to obtain the dialog variable values from ISPF and to update the values of these variables back into the ISPF variable pool. Figure 10-10 shows an example of interpreted REXX and compiled REXX. The difference is the call made to module ISPPXVP in the compiled REXX program.

PWHO interpreted REXX coding

```
usr = ZUSER
sysm = MVSVAR('SYSNAME')
splex = MVSVAR('SYSPLEX')
```

PWHO Compiled REXX coding

```
call ISPPRXVP 'I'
usr = ZUSER
sysm = MVSVAR('SYSNAME')
splex = MVSVAR('SYSPLEX')
call ISPPRXVP 'T'
```

Figure 10-10 Example of interpreted and compiled REXX code

Note: For details, refer to *z/OS V1R6.0 ISPF Dialog Developer's Guide and Reference,* SC34-4821.

10.3 ISPF EDIT enhancement

The ISPF editor is a full screen editor. It is designed for a display screen instead of being like a typewriter terminal. It displays a full screen of data, and allows you to overtype any data that is being displayed. You can scroll the data in any direction (up, down, left, or right) by a half or full page, or by any number of lines (or columns). Scrolling is performed by means of the scroll commands. You perform line-oriented editing operations by entering a line command directly on the line that is affected. For example, you type `D` on a line to delete it or `R` to repeat it. You can perform commands on several lines at the same time.

The following ISPF EDIT enhancements are implemented in z/OS V1R5.

10.3.1 Remove excluded lines from display

The EXCLUDE command is used to exclude specific lines in the data set or member being edited. It can be entered as `EXCLUDE, EX`, or `X`. When the lines are excluded, the lines are not displayed. Instead, a message line is displayed indicating the number of lines not displayed. Sometimes, these messages create inconvenience for editing and also occupy one line in the display panel.

A new HIDE edit primary command and edit macro command has been provided to hide these messages. This command removes the “excluded lines” message from the display where lines have been excluded by the EXCLUDE command. Instead, the line number field of the preceding line is underscored (where the terminal supports the underscore attribute) to alert the user that part of the data is not being displayed.

The HIDE command syntax is:

```
HIDE X
```

Also, the RESET edit primary command and edit macro command has been enhanced with the HIDE parameter to support redisplaying the excluded lines messages within the Edit display. RESET without any parameters also acts to reset the HIDE function.

The reset command syntax for HIDE is:

```
RESET HIDE
```

Figure 10-11 on page 218 shows an example where some lines are excluded from the display by issuing the EXCLUDE command.

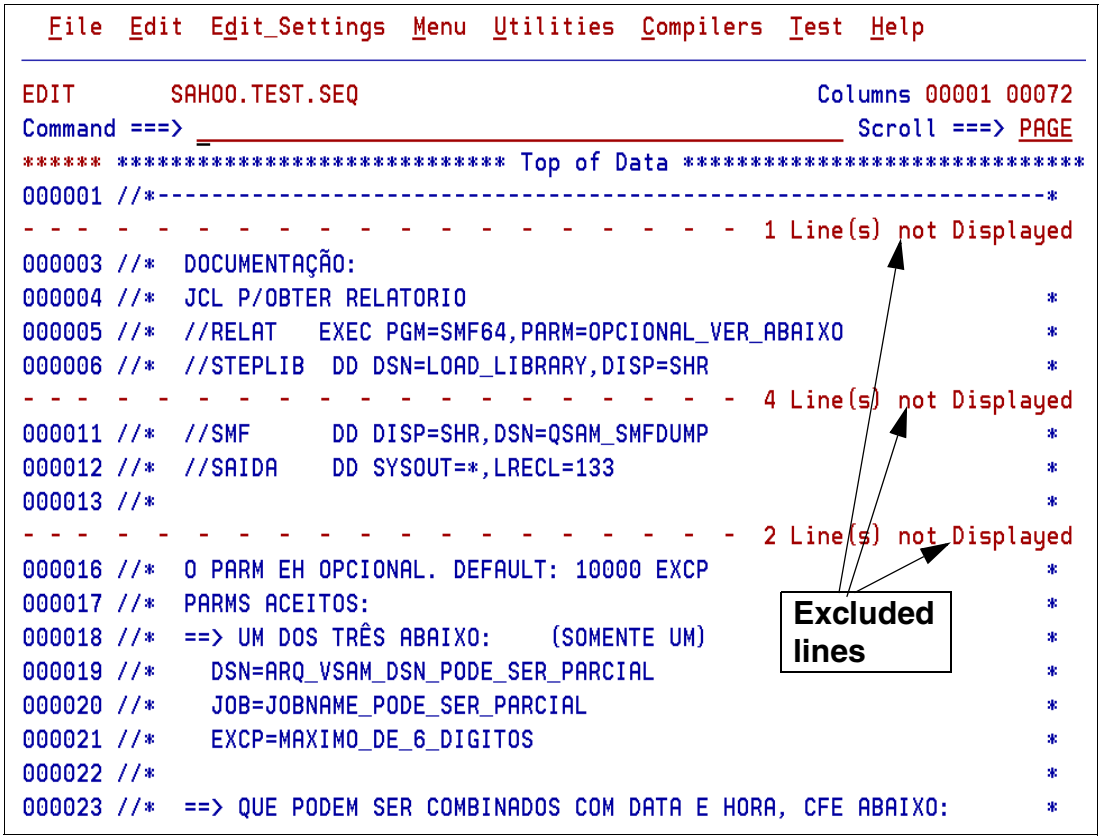

Figure 10-11 Edit panel display with lines excludes

In this panel, the HIDE EXCLUDED (HIDE X) primary command is issued to remove the excluded line messages from the Edit/View display, as shown in Figure 10-12 on page 219.

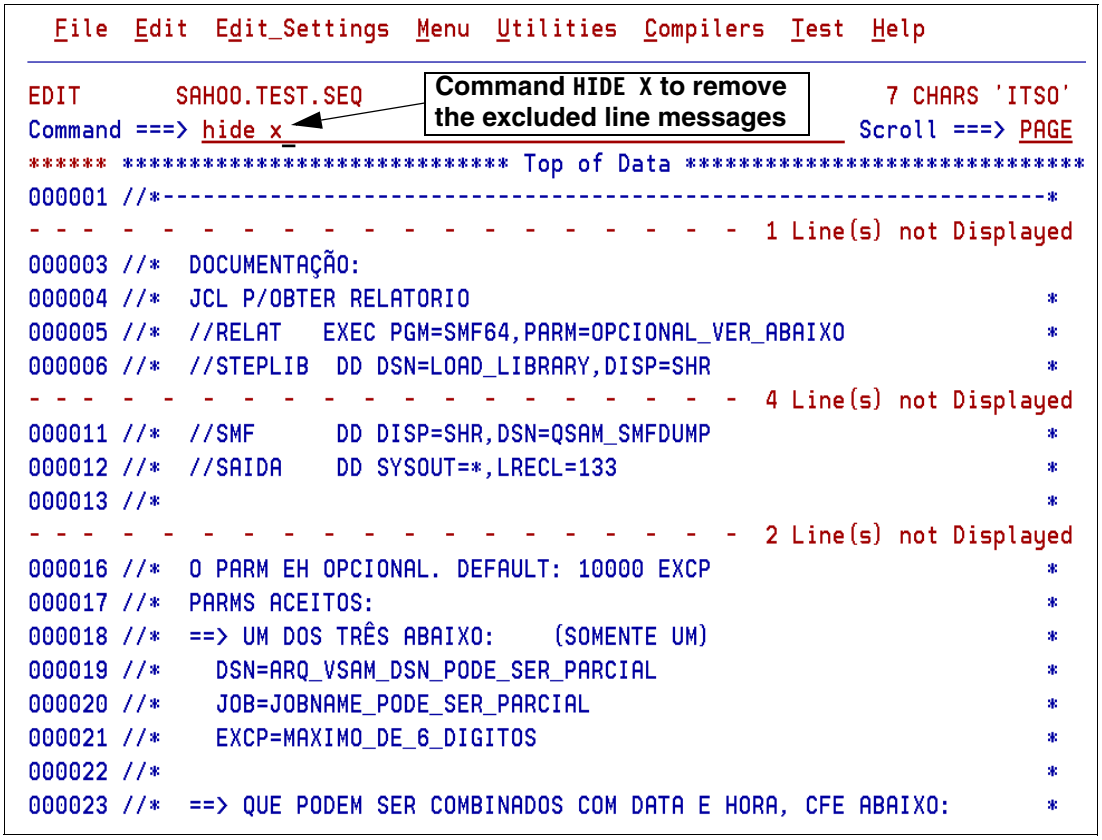

Figure 10-12 Edit panel showing the HIDE X command

Panel display messages

Now the excluded line messages are removed from the panel display. The line number field of the line preceding an excluded line message is underscored to indicate that part of the data is not being displayed. This is shown in Figure 10-13 on page 220.

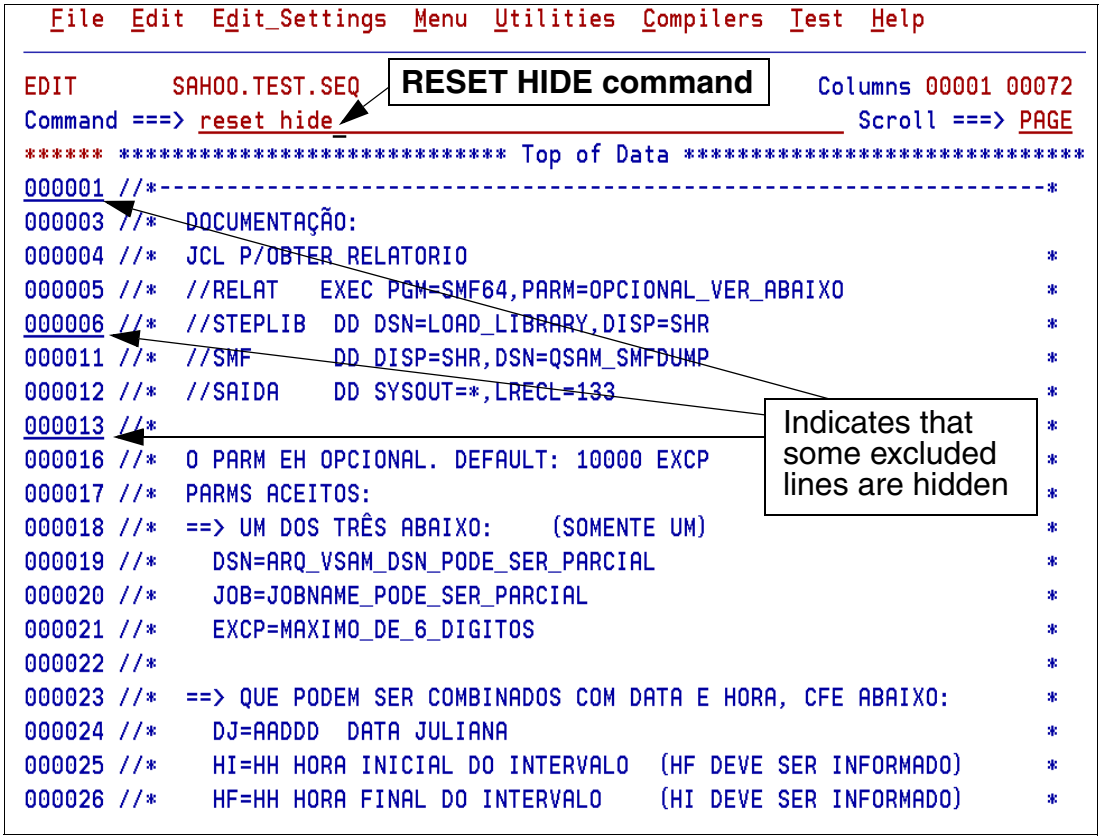

Figure 10-13 Edit panel showing hidden lines and the command RESET HIDE

The RESET HIDE command

The RESET HIDE primary command can be used to redisplay the excluded line messages. When the RESET HIDE command is issued in the panel in Figure 10-13, the panel shown in Figure 10-11 on page 218 is displayed.

10.3.2 Non-scrolling columns line

ISPF has been enhanced to display a non-scrolling columns line in Edit or View panel. This works in the same manner as the columns line under Browse.

A new COL primary command has been added to display the columns line at the top of each Edit/View data screen. This line looks identical to that presented by the **cols** line command. But the line command field, displayed using the COL primary command, is protected and this line cannot be copied, moved or deleted by overtyping with line commands.

The COLS command syntax is shown in Figure 10-14.

COLS [ON|OFF]

Where:

ON	Causes the non-scrolling columns line to display.
OFF	Removes the non-scrolling columns line from the display.

Figure 10-14 Syntax of COL primary command for EDIT/VIEW

You may issue the COL command without the parameters ON/OFF. When the non-scrolling column line is not displayed and you issue the COL primary command without the ON/OFF parameter, the line is displayed. And, when the non-scrolling line is displayed and you issue the COL primary command without the ON/OFF parameter, the line is removed from the display. Figure 10-15 shows the non-scrolling columns line, displayed when COLS ON is issued in an Edit panel.

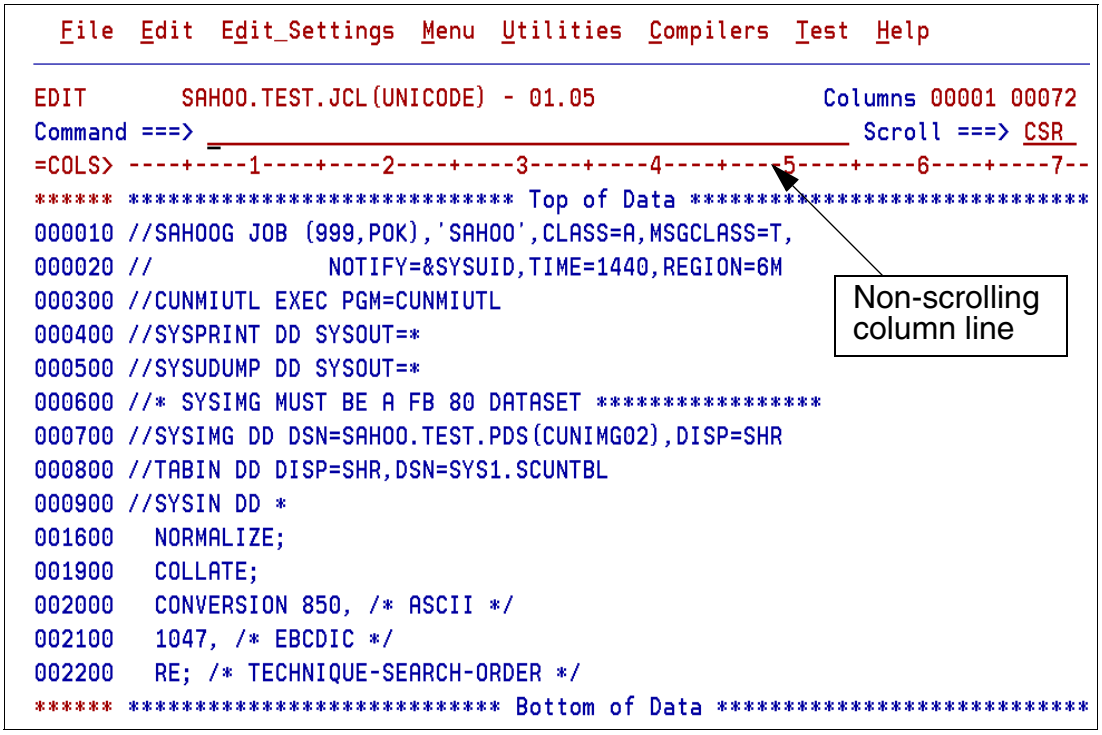

Figure 10-15 Non-scrolling column line display in an Edit panel

Note: For details, refer to *z/OS V1R6.0 ISPF User's Guide Volume II,* SC34-4823.

10.4 ISPF services enhancements

A new service, QTABOPEN, has been added. It allows an ISPF dialog to obtain a list of currently open ISPF tables. The TBSTATS or TBQUERY service can then be used to obtain more detailed information about each table.

The TBQUERY service has been enhanced to allow an ISPF dialog to obtain:

- The sort arguments passed on the last invocation of the TBSORT service
- The name-list that was last presented to the TBSARG service
- The list of name-cond pairs that was last presented to the TBSARG service
- The current direction of the search (NEXT or PREVIOUS) established for TBSCAN

10.4.1 Invocation of QTABOPEN

The service QTABOPEN is invoked either by using a command procedure (REXX or CLIST) or the CALL API. Both formats are now described.

The command procedure format is:

ISPEXEC QTABOPEN LIST(*list-var*)

The call invocation format is:

CALL ISPLINK ('QTABOPEN ',*list-var*);

where *list-var* specifies the prefix to be used to construct the names of ISPF variables that contain the list of open tables. Each variable name is constructed by appending a sequence number to the prefix. The total number of variables created is returned in a variable constructed by appending "0" (zero) to the prefix.

Here is an example of a CLIST that reports the number of open tables using the QTABOPEN service:

```
ISPEXEC QTABOPEN LIST(myvar)
IF &LASTCC = 0 THEN DO
   WRITE THE NUMBER OF TABLES OPEN ARE MYVAR0
```

Note: For details, refer to *z/OS V1R6.0 ISPF Services Guide,* SC34-4819.

10.5 ISPF configuration changes

The ISPF configuration utility has been changed to add the following:

- Support for zero block size for dynamic allocation of the ISPLIST, ISPLOG, ISPCTLx, ISPLSTx, and ISPWRKx data sets.
- Support for specifying primary and secondary space for the ISPCTL0 and ISPLSTx data sets.
- New keywords to control what happens when an explicit member list request is made for an empty PDS or PDSE. The new keywords are:
  - DISPLAY_EMPTY_MEMBER_LIST

    Controls whether an empty member list is displayed. The default is NO.
  - DISPLAY_EMPTY_MEMBER_LIST_PATTERN

    If the DISPLAY_EMPTY_MEMBER_LIST option is set, this field controls whether an empty list that results from a nonmatching pattern will be displayed. The default is NO.
  - DISPLAY_EMPTY_MEMBER_LIST_FUNCTION

    Controls whether empty member list options apply to non-edit functions such as View and Browse. The default is YES.
  - RESET_EMPTY_MEMBER_LIST_OPTIONS

    Resets the values specified in the DISPLAY_EMPTY_MEMBER_LIST fields. The default is NO.

Figure 10-16 on page 223 shows the new options added to the ISPF sitewide defaults panel.

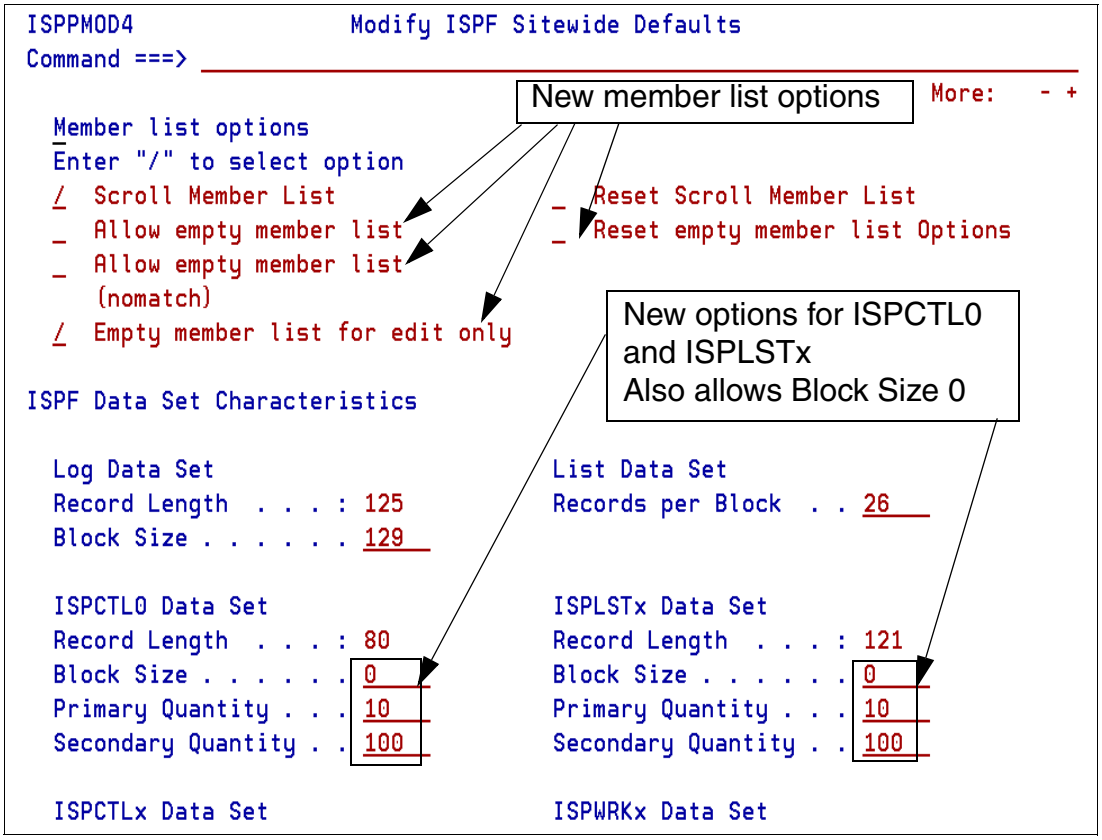

Figure 10-16 ISPF sitewide defaults panel

You can control the empty member list options using the ISPF settings panel shown in Figure 10-17.

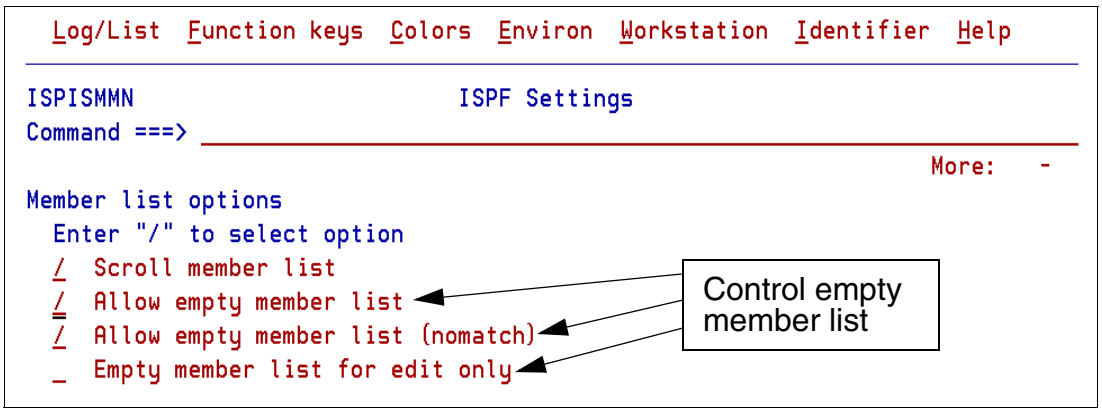

Figure 10-17 ISPF settings panel showing empty member list options

A brief description of the empty member list options shown in Figure 10-16 and Figure 10-17 follows:

Allow empty member list
Enter a “/” to indicate that ISPF can display empty member lists when a PDS or PDSE has no members.

Allow empty member list (nomatch)
Allows the user to customize whether the empty member list option applies

to lists, which are empty as a result of a non-matching pattern. This option has no effect unless “Allow empty member list” is checked.

Empty member list for edit only
Allows the user to customize whether the empty member list option applies to Edit functions or all ISPF-generated member lists. This option has no effect unless “Allow empty member list” is checked. The default is “/”, edit functions only.

Reset empty member list Options Enter “/” to reset each user’s empty member list options to the values specified. The reset is done once each time the Sitewide Defaults Version Level field is incremented.

Note: For details, refer to *z/OS V1R6.0 ISPF User’s Guide Volume II,* SC34-4823 and *z/OS V1R6.0 ISPF Planning and Customizing,* SC34-4814.

10.6 SCLM enhancements

Software Configuration and Library Manager (SCLM) is used to create, control, maintain, and track software components for a project. SCLM runs in a user’s address space and there is no started task. The SCLM project database consists of a series of related ISPF libraries (partitioned data sets). These contain source and non-source software components. SCLM project definition and control information is contained in an assembled and linked PROJECTDEFS data set. SCLM project cross-reference and accounting data sets are VSAM clusters.

The following enhancements have been implemented to SCLM under z/OS V1R6.

10.6.1 SCLM—the Unit of Work utility

The new Unit of Work (UOW) utility allows a user to group together and work with a set of editable elements required for a development line item or maintenance task. The editable elements can be of different types and are defined for the Unit of Work as entries in an ARCHDEF. The ARCHDEF contains an INCLD or PROM statement for each editable element requiring modification for the Unit of Work. SCLM creates a member list based on the entries in the ARCHDEF. Standard SCLM functions are available as line commands from the member list. Also, users have the option to create their own customized line commands.

Unlike the SCLM Library utility, which constrains the user to work with one Type at a time, the Unit of Work utility provides access to all of the members associated with an ARCHDEF, regardless of Type.

Figure 10-18 on page 225 shows the SCLM Unit of Work processing Entry Panel (ISPF option 10.3.11) with the Options action bar exploded.

This panel provides options for processing the Unit of Work defined by an ARCHDEF. These include options to bypass or process options for the Edit, Build, and Promote functions.

The Unit of Work Entry Panel contains Options action bar choices to do the following:

- Set the default prefix for all Unit of Work output data sets
- Specify a job card for batch jobs generated during Unit of Work processing
- Create a customized list of commands that display on the UOW Member List panel
- Create a customized list of commands that display on the UOW Member Contents panel

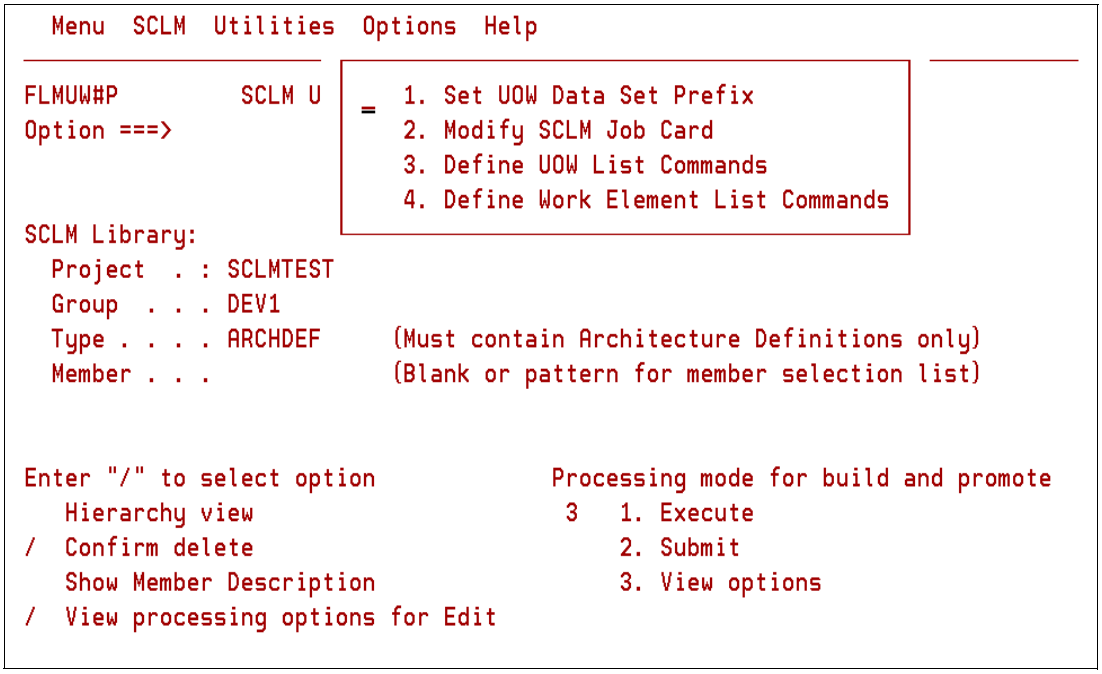

Figure 10-18 SCLM Unit of Work processing - Entry Panel

UOW Member List panel display

The UOW Member List panel displays the list of ARCHDEFs that match the member name pattern specified in the SCLM Unit of Work processing Entry Panel. The user can apply standard SCLM line commands (as displayed under the command line) or user-defined UOW Member List commands to each member in the list. Figure 10-19 shows a list of members displayed where the SCLM line commands can be used for further processing. In this panel, the line command S has been entered against member UOW001.

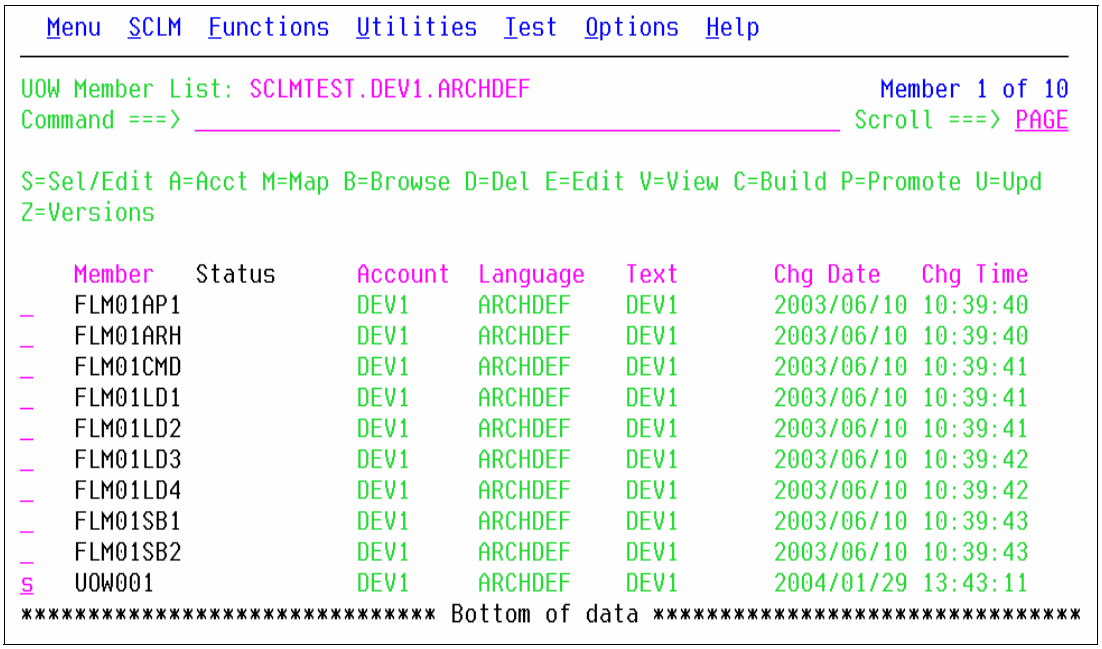

Figure 10-19 UOW Member List

Work Element List panel

The Work Element List panel displays the contents of the selected UOW (ARCHDEF) as a list of members. The user can apply the standard SCLM line commands (as displayed under the command line) or user-defined Work Element List commands to each member in the list. Figure 10-20 shows a list of members displayed for UOW001 selected in Figure 10-19.

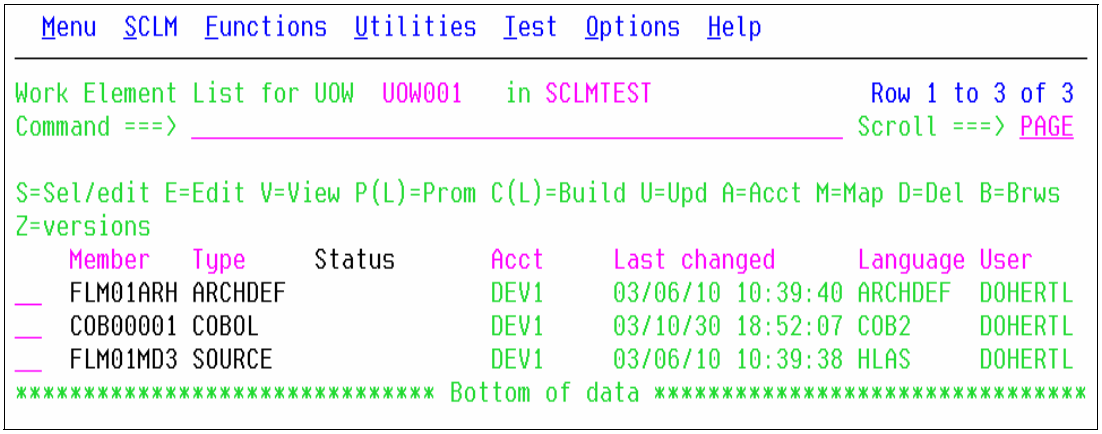

Figure 10-20 Contents of selected UOW

10.6.2 SCLM—the Explorer utility

The SCLM Explorer utility allows a user to identify the relationships between elements in a project. The utility displays a list of elements within a project and the user can then select an element and display all its related elements. The initial list can be a list of parts or architecture definitions (ARCHDEFs).

Element relationships are generally of a hierarchical nature, for example:

- Archdefs can include source parts.
- Source parts include other source parts.
- Archdefs can include other archdefs.

The panel displaying the list of elements for a project supports the following line commands:

U (UP command) to identify the parents of the selected element

D (DOWN command) to show the child elements

L (LMOD command) to identify the related executable components

Related elements

The related elements (if any) are then displayed 8 to a line. Each displayed element is a point-and-shoot field allowing the user to position the cursor on the field, press Enter, and obtain a display showing its parent or child elements.

The information describing element relationships is stored in ISPF tables. The SCLM Explorer batch utility FLMEUXTR creates these ISPF tables from element relationship data extracted from the SCLM project accounting files. A pull-down menu on the Explorer entry panel provides an option to create the JCL to run the batch utility.

SCLM Explorer entry panel

The SCLM Explorer entry panel has options to display a list of parts or architecture definitions (ARCHDEFs). The panel contains Tables Action Bar choices to do the following:

- Specify the name of the ISPF table library containing the information extracted from the SCLM project accounting files
- Build the JCL for a batch job to extract data from the SCLM project accounting files and populate the ISPF tables

This is shown in Figure 10-21.

.

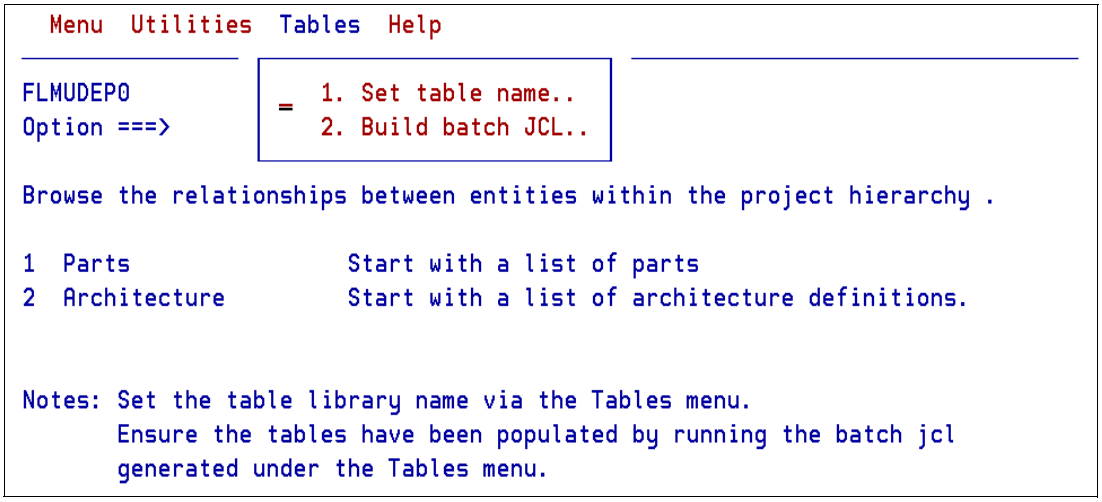

Figure 10-21 SCLM explorer panel

View: Parts panel

The View: Parts panel is displayed when Option 1 is entered on the SCLM Explorer entry panel. This panel allows the user to define the list of displayed elements by entering patterns for the values in the Member, Group, Type, and Language columns. The following line commands are supported on this panel:

U (**UP**) Shows related parent elements.

D (**DOWN**) Shows related child elements.

L (**LMOD**) Shows related executable elements.

A sample panel is shown in Figure 10-22 on page 228.

```
                              View: Parts                    Row 1077 from 54511
Option  ===> ____________________________________________________ Scroll ===> CURS
  DCLS*___ *_______ *_______ *______   <== Show entries matching these pattns
S Member   Group    Type     Language  <== Point and shoot heading to sort col
u DCLS$CMD SVT      SOURCE   PLASMVS
_ DCLS$EXT SVT      SOURCE   PLASMVS
_ DCLS$SYS SVT      SOURCE   PLASMVS
_ DCLSAFPL SVT      SOURCE   PLASMVS
_ DCLSAIDS SVT      SOURCE   PLASMVS
_ DCLSAPI  SVT      SOURCE   PLASMVS
_ DCLSAPM  SVT      SOURCE   PLASMVS
_ DCLSARL  SVT      SOURCE   PLASMVS
_ DCLSASCB SVT      SOURCE   PLASMVS
_ DCLSASVT SVT      SOURCE   PLASMVS
```

Figure 10-22 View: Parts panel

Related elements panel

Figure 10-23 shows an example of a panel displaying related elements, when the U line command is entered against DCLS$CMD to obtain a display of the elements that include DCLS$CMD. As you can see, the elements are displayed 8 to a line. You can place the cursor on an element and press Enter to obtain a panel displaying that element's parents. This process can be continued until the end of the parent chain is reached.

```
                              View: Parts                    Row 1 to 11 of 11
Option  ===> ____________________________________________________ Scroll ===> CURS

  Path= DCLS$CMD
  DCLS$CMD is included by these parts:
------------------------------------------------------------------------------
  ISRBCD   ISRCBAI  ISRCBB   ISRCBF   ISRCBG   ISRCBR   ISRCMLF  ISRECE
  ISRECP   ISRECR   ISRECS   ISREDDI  ISREDI   ISREDO   ISREDP   ISREDSU
  ISREED   ISREFC   ISREFD   ISREFR   ISREICR  ISREIDI  ISREIDO  ISREIMC
  ISREIO   ISRELAF  ISRELCP  ISRELMO  ISRELOV  ISRELX   ISREMC   ISREMD
  ISREME   ISREPAD  ISREPAL  ISREPAN  ISREPAS  ISREPCA  ISREPCM  ISREPCO
  ISREPCP  ISREPCU  ISREPDF  ISREPDL  ISREPDR  ISREPED  ISREPEX  ISREPFC
  ISREPFI  ISREPHI  ISREPHX  ISREPLO  ISREPMD  ISREPMV  ISREPNL  ISREPNM
  ISREPNN  ISREPNO  ISREPPA  ISREPPF  ISREPPK  ISREPRN  ISREPRY  ISREPSB
  ISREPSH  ISREPSL  ISREPSN  ISREPST  ISREPSU  ISREPTB  ISREPTR  ISREPXS
  ISRESC   ISRESE   ISRESRL  ISRETL   ISRNLDES ISRNLDEU ISRNLENP ISRNLENU
  ISRNLJPN ISRPLFND ISRUDFND ISRUDMEM ISRUDS4
------------------------------------------------------------------------------
```

Figure 10-23 View: Parts - List of parent elements

10.6.3 SCLM service command panels

The FLMCMD service interface provides a lot of flexibility to SCLM users. But there was no user interface provided via panel. Now, a panel interface has been provided for each of the services available via the FLMCMD command.

The new Option 6A on the SCLM main menu invokes the SCLM FLMCMD services menu, which has options to display a panel supporting each of the FLMCMD services.

Figure 10-24 on page 229 shows the SCLM main menu panel showing the new Option 6A.

```
  Menu  Utilities  Help
------------------------------------------------------------------------------
FLMDMN                          SCLM Main Menu
Option ===> _____________________________________________________________

Enter one of the following options:

   1  View         ISPF View or Browse data
   2  Edit         Create or change source data in SCLM databases
   3  Utilities    Perform SCLM database utility/reporting functions
   4  Build        Construct SCLM-controlled components
   5  Promote      Move components into SCLM hierarchy
   6  Command      Enter TSO or SCLM commands
   6A Easy Cmds    Easy SCLM commands via prompts <-- New option
   7  Sample       Create or delete sample SCLM project
   X  Exit         Terminate SCLM
```

Figure 10-24 SCLM main menu

Figure 10-25 shows the SCLM FLMCMD Services Menu, when Option 6A is selected from the SCLM main menu.

```
  Menu  Utilities  Options  Help
------------------------------------------------------------------------------
FLMSRV#P                 SCLM FLMCMD Services Menu
Option ===> _____________________________________________________________
                                                              More:      +
1  ACCTINFO  Retrieve accounting information
2  AUTHCODE  Retrieve or set authorization code for selected members
3  BUILD     Build a member
4  DBUTIL    Create reports and tailored data sets against an
               SCLM database
5  DELETE    Delete database components
6  DELGROUP  Delete group database components
7  DSALLOC   Allocate data sets for group or type
8  EDIT      Edit a member of a controlled library
9  EXPORT    Extract SCLM accounting information
10 GETBLDMP  Retrieve build map information
11 IMPORT    Incorporate exported data into the hierarchy
12 LOCK      Lock a member or assign an access key
13 MIGRATE   Register the contents of a library with SCLM
14 NEXTGRP   Find the name of the next group in a hierarchy
15 PROMOTE   Promote a member from one library to another
16 RPTARCH   Create an architecture report
17 SAVE      Lock, parse, and store a member
```

Figure 10-25 SCLM FLMCMD service menu

You can also enter the following command to display the panel supporting the service *srvname*:

TSO FLMCMD *srvname*

where *srvname* is the name of the SCLM service.

Each service panel displays input fields for the service parameters. Once the input fields are validated, the service is invoked through FLMCMD and the results are displayed back to you.

Figure 10-26 shows a panel displayed when the command TSO FMLCMD GETBLDMP is issued. The same panel is displayed by selecting Option 10 in SCLM FLMCMD Services Menu.

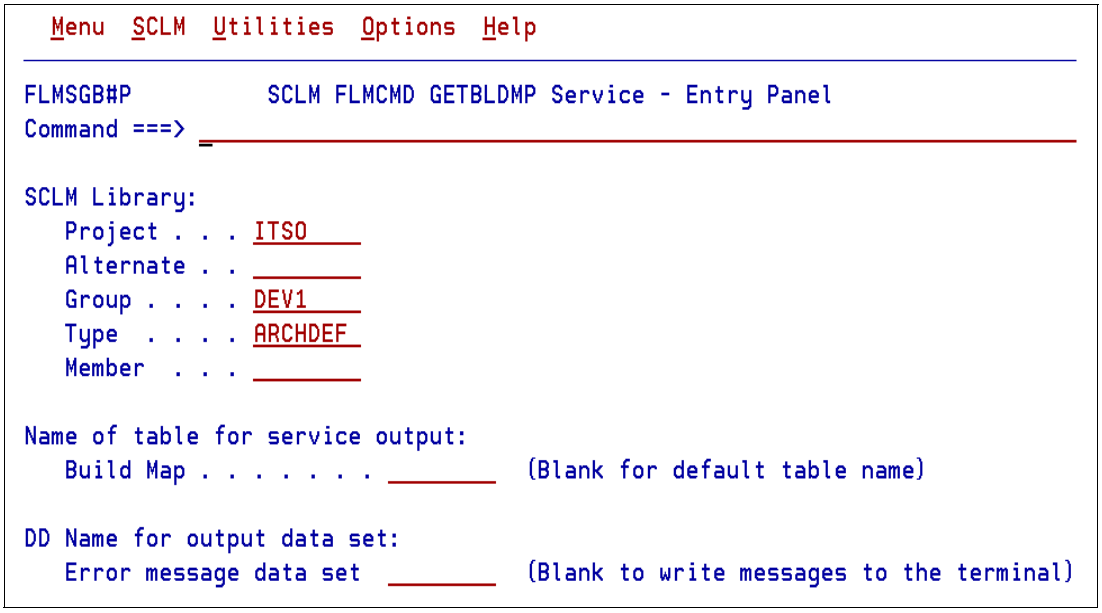

Figure 10-26 SCLM FLMCMD GETBLDMP Service, Entry Panel

Note: For details, refer to *z/OS ISPF Software Configuration and Library Manager (SCLM) Reference,* SC34-4818.

11

Communications Server for z/OS V1R6

z/OS Communications Server is a network communication access method. It provides both Systems Network Architecture (SNA) and Transmission Control Protocol/Internet Protocol (TCP/IP) networking protocols for z/OS.

The TCP/IP protocol suite (also called stack), includes associated applications, transport and network protocol layers, and connectivity and gateway functions. For more information on z/OS Communications Server IP protocols, refer to *z/OS Communications Server: IP Configuration Guide,* SC31-8775.

The SNA protocols are provided by VTAM® and include Subarea, Advanced Peer-to-Peer Networking (APPN), and High Performance Routing protocols. z/OS Communications Server provides the interface between application programs residing in a host processor, and resources residing in an SNA network; it also links peer users in the network. For more information on z/OS Communications Server SNA protocols, refer to *z/OS Communications Server: SNA Network Implementation Guide,* SC31-8777-04.

11.1 Communications Server z/OS V1R6 overview

Communications Server is a z/OS base element that supports secure TCP/IP, SNA, and z/OS UNIX networking on enterprise systems, connecting different types of communication subsystems and applications to each other and supporting usage of various communication devices (such as terminals and printers) listed in a system's hardware configuration, as shown in Figure 11-1. The major components of Communications Server are IP Services and SNA Services.

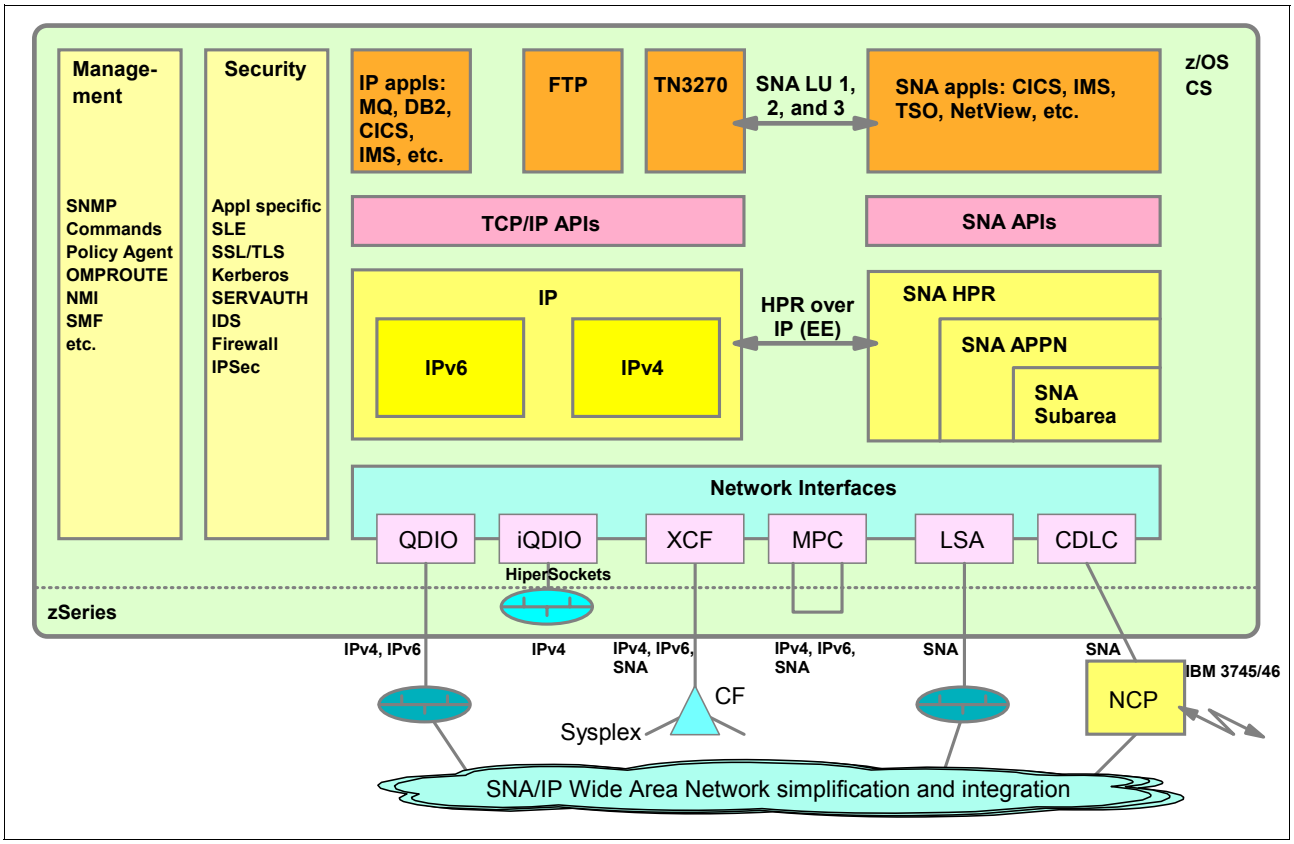

Figure 11-1 Communications Server on z/OS - a technical overview

IBM Communications Server for z/OS V1R4 introduced a new version of the standard Internet Protocol stack, IPv6. IPv6 is the next generation of the Internet Protocol designed to replace the current version, Internet Protocol Version 4 (IPv4). The most significant IPv4 characteristic is its 32-bit addressing space, which theoretically allows over 4 billion nodes. In practice, the interaction between routing and addressing makes it impossible to utilize more than a small portion of available nodes. Continued growth of the Internet could lead to the exhaustion of IPv4 addresses early in the 21st century.

IPv6 uses a 128-bit address space. That, according to RFC 2374, provides practically limitless global addressability. It also adds many improvements to IPv4 in areas such as routing and network autoconfiguration. IPv6 is expected to gradually replace IPv4. During the transitional period, the two Internet protocols will coexist for a number of years.

Not all IPv6 features are supported on z/OS V1R6. z/OS V1R6 CS continues the effort started in z/OS V1R4 to provide IPv6 on z/OS and to also provide key non-IPv6 function.

The IPv6 focus in V1R6 is in the following areas:

- Sysplex exploitation
  - Dynamic VIPA
  - Dynamic VIPA takeover
  - Sysplex Distributor functions
- Dynamic routing protocol with OSPFv3 for OMPROUTE
- Additional SNMP MIBs functions

The non-IPv6 focus in V1R6 is in the following areas:

- Multilevel security
- TN3270 server enhancements
- FTP enhancements
- Sysplex enhancements
- Network management
- SNA enhancements

11.2 IPv6 support in z/OS V1R6 Communications Server

IPv6 is an evolutionary step from IPv4. Functions that work well in IPv4 have been kept in IPv6, and functions that did not work well in IPv4 have been removed. z/OS Communications Server Version 1 Release 4 was the first release to incorporate IPv6 features. Not all IPv6 features are supported. z/OS V1R6 Communications Server enables you to do the following:

- Build an IPv6 network
- Start using IPv6-enabled applications
- Enable existing IPv4 applications to be IPv6 applications
- Access your SNA applications over an IPv6 network

IPv6 uses a 128-bit address space, which has no practical limit on global addressability and provides 3.4×10^{50} unique addresses. This is enough so that every person could have a single IPv6 network with many nodes, and still the address space would be almost completely unused.

The greater availability of IPv6 addresses eliminates the need for private address spaces, which in turn eliminates one of the needs for network address translators (NATs) to be used between the private intranet and the public Internet.

11.2.1 IPv6 sysplex support

Changes to IPv6 support for sysplex were made in z/OS V1R6 Communications Server in the following areas:

- TCP/IP sysplex functions now support IPv6.

  If you are deploying IPv6 applications, you can take advantage of the availability and workload balancing capabilities of z/OS TCP sysplex functions for your mission-critical IPv6 applications, as you have been able to do in past releases for IPv4 applications. Refer to the discussion on IPv6 special considerations in *z/OS Communications Server: IP Configuration Guide,* SC31-8775 for more information.

  In addition, the Policy Agent is changed to support IPv6 addresses for the Policy Agent-to-Policy Agent connections that are established for the Sysplex Distributor Performance Monitoring function.

- The netstat,config display for the IPv4 section now displays either a subnet or the num_mask_bits with the DYNAMICXCF address, depending on how it was configured.
- Sysplex exploitation (Dynamic VIPA, Sysplex Distributor functions)
- Dynamic Routing Protocol with OMPROUTE (OSPFv3)
- Additional Network Management MIBs

11.2.2 z/OS V1R6 IPv6 support

IP addresses are allocated to each TCP/IP services address on a TCP/IP Internet. Each address is a unique 32-bit (an IPv4 Internet address) or a unique 128-bit (an IPv6 Internet address) quantity defining the host's network and the particular host. A host can have more than one IP address if it is connected to more than one network (a so-called multihomed host).

Installation tasks

If you want to use the IPv6 support for sysplex enhancements, perform the following tasks:

- Exploit the So_Clusterconntype option of getsockopt().
- Enable source VIPA for IPv6 Dynamic XCF.
- Define IPv6 Dynamic VIPAs.
- Define backup stacks for the defined IPv6 Dynamic VIPAs.
- Define a range of potential IPv6 Dynamic VIPAs on a stack within the sysplex.
- Cause a defined IPv6 Dynamic VIPA to be distributed to other sysplexes.
- Define an IPv6 TCP stack source VIPA for connections using IPv6 routes.
- Modify the Policy Configuration file, the PolicyAction statement OutboundInterface, to include IPv6 addresses.
- Allow an IPv6 DVIPA to be activated on a backup TCP/IP before it has been activated on the TCP/IP where it is defined with VIPADEFINE.
- When using VIPARANGE, control which applications may bind() to create a DVIPA.

11.2.3 IPv6 support for sysplex

Installations could not deploy IPv6 applications that take advantage of the availability and workload balancing capabilities of the z/OS TCP Sysplex Dynamic VIPA and Sysplex Distributor functions for their mission-critical applications because Sysplex Dynamic VIPA and Sysplex Distributor do not support IPv6.

z/OS V1R6 sysplex functions

In order to support Sysplex Dynamic VIPA and Sysplex Distributor, the following changes are made in z/OS V1R6:

- Dynamic VIPA (DVIPA)

  VIPA, in general, improves availability since connectivity does not have to be tied to a single physical interface. It can be used to identify a network service or application.

  In Figure 11-2 on page 235 the connection can be to 1234::5678 versus connections 2234::2 or 2244::2.

- Dynamic VIPA Takeover

  The changes improve application availability for planned and unplanned outages and help the setup for backup and recovery.

  In Figure 11-2 on page 235, if Stack1 is down, then stack2 takes over. It illustrates that when connection 1234::5678 goes down, then Stack 2 advertises 1234::5678.

- Sysplex Distributor

  The changes improve performance and availability with load balancing between server applications.

  In Figure 11-2, Sysplex Distributor on Stack 2 load-balances connections for CICS on port 5555.

- Underlying connectivity for sysplex uses Dynamic XCF.

  Note that Dynamic XCF supports IPv6 in V1R5.

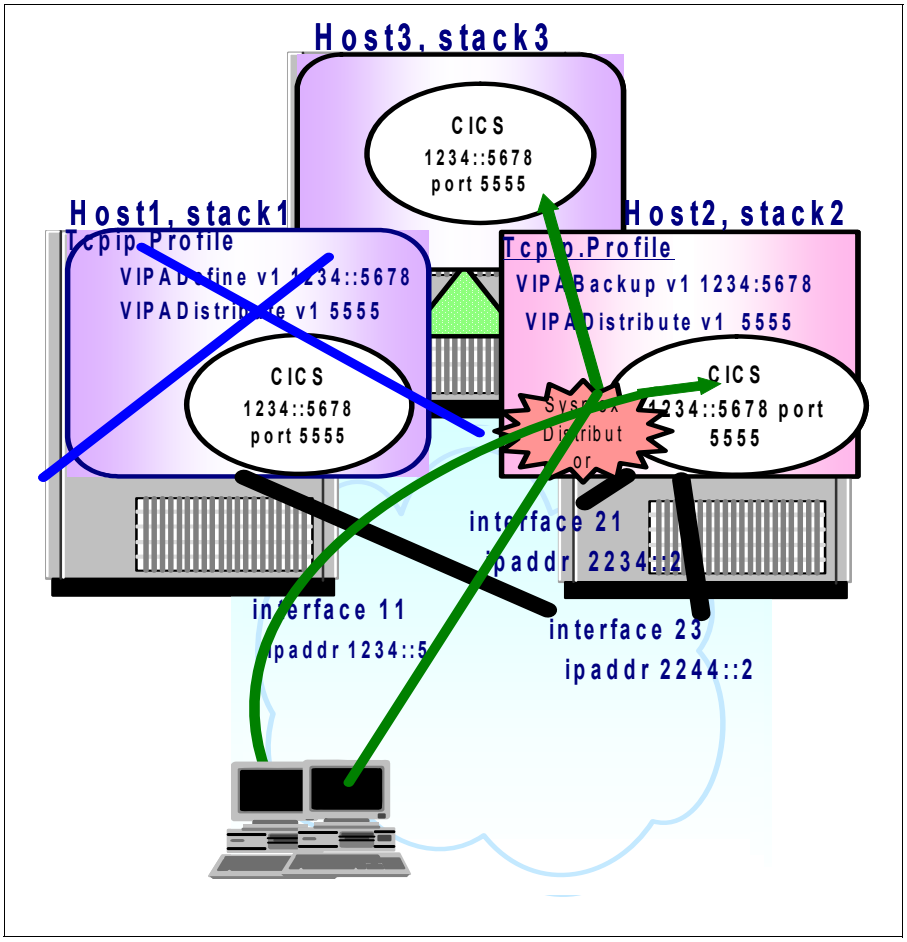

Figure 11-2 Dynamic VIPA takeover

Note: All participating stacks of Dynamic VIPA takeover and sysplex distribution of ports for load balancing must be at least at the V1R6 level.

Other sysplex functions

Other functions associated with sysplex are supported for IPv6, as follows:

Sysplex Sockets This function is via the SO_CLUSTERCONNTYPE option of getsockopt(). Previously, SO_CLUSTERCONNTYPE_NONE was always returned when the SO_CLUSTERCONNTYPE option was used with getsockopt() for an AF_INET6 socket. Now, the proper value will be returned. The types returned for an IPv6 connection are the same as those returned for an IPv4 connection.

TCPSTACKSOURCEVIPA This function allows users to specify a single Dynamic VIPA (DVIPA) or static VIPA to be used as a source IP address for TCP applications that initiate outbound connections on that stack.

Sysplexports Sysplex Distributor is enhanced with a facility to allow assignment of ephemeral ports for outbound connections to be managed across the entire sysplex, such that for a particular Distributed DVIPA, a particular port value is assigned to a socket on only one TCP stack in the sysplex. This ensures that inbound connection data can always be uniquely routed to the correct application instance.

Fast Connection Reset after System Failure

This function allows the client stack to notify the client application of a system failure. This improves availability and allows quicker initiation of connection failure recovery. Without this function, the client was unaware of a system failure until it attempted to send data.

When the target stack fails, the routing stack should attempt to notify the clients of the failure of the connections that terminated in the failed target stack.

When the stack deletes a DVIPA, the deleting stack should notify any clients of the failed connections.

When the routing stack fails, or deletes a Distributed DVIPA, and there is no available designated backup stack for that DVIPA, the target stacks should send RST.

Enhance Workload Distribution (application Server Affinity)

This function allows affinities to be established between a specific client (identified by its IP address) and a particular instance of a server application for which work is being balanced with Sysplex Distributor, using a Distributed Dynamic VIPA. This feature ensures that a client that establishes a relationship with a server will be directed to that particular server for subsequent connections. TIMEDAFFinity is an optional parameter on the VIPADISTRIBUTE statement that enables this function.

Dynamically Assign Sysplex Ports

Distributed DVIPAs configured without a PORT parameter on the VIPADISTRIBUTE statement determine where to distribute work based on where there are applications with listening sockets bound to the distributed DVIPA, regardless of how many different ports are involved. Applications must bind specifically to the designated distributed DVIPA (or have a

BIND parameter configured on the PORT statement to accomplish this) in order to be identified as server applications to Sysplex Distributor when no PORT statement is coded on the VIPADISTRIBUTE statement.

Activation of DVIPAs through VIPABACKUP
VIPABACKUP is enhanced so that an IPv6 Dynamic VIPA may be activated on a backup TCP/IP before it is activated elsewhere in the sysplex with the VIPADEFINE statement. An interface name, a DVIPA IPv6 address, and a new MOVEABLE parameter allow this to occur.

DYNAMICXCF-SOURCEVIPAINT
This function allows the specification of a static VIPA interface to be used as SOURCEVIPA for dynamic XCF link.

Sysplex Distributor Round-Robin Distribution
Add to the IPv6 VIPADISTRIBUTE statement an optional DISTMethod parameter specifying the method of distributing new connection requests without existing affinity for this Distributed DVIPA. The DISTMethod parameter has two values: BASEWLM (the default distribution that consults WLM and Service Policy Agent for distribution), and ROUNDROBIN (for round-robin distribution to the target stacks).

Sysplex Distributor Policy This function is enhanced to support IPv6 addresses for the Policy Agent-to-Policy Agent connections that are established for the Sysplex Distributor Performance Monitoring function. Policy Agent IPv6 support was already implemented in V1R5.

11.2.4 Defining sysplex IPv6

In a general configuration, you cannot mix IPv4 and IPv6 on the same statement, but you can mix within the VIPADYNAMIC and ENDVIPADYNAMIC statements, as shown in the PROFILE.TCPIP data set in Figure 11-3 on page 238. These new definitions are very similar to IPv4.

IPCONFIG parameters

Use the IPCONFIG statement to update the IPv4 IP layer of TCP/IP. Use the IPCONFIG6 statement to update the IP layer of TCP/IP with information that pertains to IPv6.

SOURCEVIPA For the SOURCEVIPA option to work properly, the receiving nodes in the network must be configured to recognize the SOURCEVIPA addresses using the static or dynamic routing protocols. Otherwise, timeouts for the connection or request responses will occur as a result of the VIPA addresses being network-unreachable.

SOURCEVIPA enables interface fault tolerance for z/OS clients that establish outbound connections. When SOURCEVIPA is set, outbound datagrams use the corresponding virtual IP address (VIPA) in the HOME list instead of the physical interfaces IP address. SOURCEVIPA has no effect on RIP servers such as OROUTED, NCPROUTE, or OMPROUTE.

TCPSTACKSOURCEVIPA
If TCPSTACKSOURCEVIPA is specified on the IPCONFIG statement, it overrides SOURCEVIPA for outbound IPv4 TCP connections. If the SRCIP profile statement block is defined to establish one or more job-specific source IPv4 addresses, these IP addresses override

TCPSTACKSOURCEVIPA or the VIPAs in the HOME list for IPCONFIG SOURCEVIPA, or both, for the specified job names.

TCPSTACKSOURCEVIPA allows z/OS clients to specify a sysplex-wide source IP address for TCP connections. When TCPSTACKSOURCEVIPA is set, outbound TCP datagrams use the IP address specified in the TCPSTACKSOURCEVIPA statement instead of static VIPA addresses or physical interface addresses.

SOURCEVIPAINT To use autoconfiguration, the IPADDR cannot be specified. To assign a VIPA address for an interface, use SOURCEVIPAINT. To have IPv4 and IPv6 share a physical device, define IPv4 using DEVICE/LINK/HOME and IPv6 using INTERFACE.

SOURCEVIPAINT is added in V1R6 to allow associated source VIPA.

VIPADYNAMIC Dynamic VIPAs can be either IPv4 or IPv6, and both can be configured within the same VIPADYNAMIC block. A single statement (for example, VIPADEFINE or VIPABACKUP) must contain either IPv4 addresses or IPv6 addresses, but not both. However, statements containing IPv4 addresses can be intermixed with statements containing IPv6 addresses within the same VIPADYNAMIC block in any manner desired.

```
IPCONFIG SYSPLEXROUTING SOURCEVIPA TCPSTACKSOURCEVIPA 201.2.10.11
 DYNAMICXCF 193.9.200.1 255.255.255.240 1
IPCONFIG6 DYNAMICXCF 2001:0DB8::
151:0001 INTFID 6:7:8:9
 SOURCEVIPAINT SVIPA1 SOURCEVIPA TCPSTACKSOURCEVIPA DVIPA1

VIPADYNAMIC
VIPADEFINE   MOVEABLE IMMED v1name  3838::1234
VIPABACKUP  v2name   5555:1122:2244::1234
VIPADEFINE  MOVEABLE  IMMED 255:255: 255:0  9.24.112.3
VIPABACKUP  10.12.105.2
VIPADELETE v1name
VIPADELETE 10.12.105.2
VIPARANGE  r1name 9922:3344:5566:1122/96
ENDVIPADYNAMIC
```

Figure 11-3 TCPIP.PROFILE data set

Note: Including the SOURCEVIPA and TCPSTACKSOURCEVIPA parameters on the IPCONFIG and IPCONFIG6 statements, on each target stack with the same dynamic VIPA specified, enables a single DVIPA address to be used as a sysplex-wide source DVIPA address for outbound TCP connections.

Defining/deleting an IPv6 DYNAMIC VIPA (DVIPA)

When defining or deleting using VIPADEFINE and VIPABACKUP statements in the TCPIP profile, there is one IPv6 address specified per statement versus multiples for IPv4. When defining an IPv6 statement, you must also specify an interface name. This interface name for IPv4 was automatically generated from the IP address.

Note: The MOVEABLE IMMEDIATE parameter is not supported for IPv6. For IPv4, MOVEABLE IMMEDIATE is preferred over MOVEABLE WHENIDLE. WHENIDLE will be removed in a future release.

Consider the following when using IPv6 and DVIPA:

- For IPv6, an address mask is not required. This was needed for IPv4 due to broadcast support.
- A VIPADELETE statement in the TCPIP profile only supports interface name specification for IPv6.
- VIPARANGE, MODDVIPA program utility, and SIOCSVIPA6 ioctl() can be used to define or delete a DVIPA.
- When adding a VIPARANGE statement in the TCPIP profile, the application issues a bind(), if the system administrator runs the MODDVIPA utility for DVIPA in the range to define the DVIPA, or the application issues SIOCSVIPA6 (ioctl for IPv6 DVIPAs).
- You must define an interface name for IPv6 DVIPAs created in the VIPARANGE statement. The interface name is automatically generated for IPv4. Each IPv6 DVIPA created in the VIPARANGE uses the same interface name.
- VIPARANGE MOVEABLE DISRUPTIVE is not supported for IPv6 (only NONDISRUPTIVE). DISRUPTIVE will be removed for IPv4 in a future release.
- A new RACF SERVAUTH class profile is added with the following resource name:

  EZB.BINDDVIPARANGE.sysname.tcpname

  This provides controls over which applications (users) can bind() to create a DVIPA for IPv4 and IPv6.

Define sysplex distribution for the DVIPA /port

The VIPADISTRIBUTE statement in the TCPIP profile supports distribution of an IPv6 interface name/port. With IPv4, an IP address/port is used.

Distribution destinations must match the type specified for distribution; for example, IPv6 interfaces or IPv4 addresses cannot be mixed.

11.2.5 Migrating sysplex applications to IPv6

There are considerations for stacks and applications to support IPv6 clients because server applications must be modified:

- The TCP/IP stack is dual for both IPv4 and IPv6 with open AF_INET6 sockets.
- An application that binds explicitly now needs two binds, one for IPv4 and one fir IPv6.
- An application that binds implicitly via a port reservation in the TCPIP profile (typically applications bind to inaddr_any/unspecified), as follows:
  - It cannot specify an IPv4 and IPv6 for the same jobname.
  - It could start separate instances with separate jobnames if the application can support this.
- Probably the most flexible is a bind() to inaddr_any. If this field is set to the constant INADDR_ANY, as defined in IN.H, the caller is requesting that the socket be bound to all network interfaces on the host. Subsequently, UDP packets and TCP connections from all interfaces matching the bound name are routed to the application. This becomes important when a server offers a service to multiple networks. By leaving the address unspecified, the server can accept all UDP packets and TCP connection requests made of its port, regardless of the network interface on which the requests arrived.
- For IPv6 addresses to be used for backup and recovery, partner stacks must be at the V1R6 level or above.

- To distribute IPv4 and IPv6 workloads, DYNAMICXCF and DYNAMICXCF6 statements must be defined in the TCPIP profile.

Migration from IPv4 to IPv6

A migration from IPv4 to IPv6 will take some time to achieve. A shown in Figure 11-4 on page 241, the migration can take place through various stages.

Stage 1

In stage 1, tunneling provides a way to utilize an existing IPv4 routing infrastructure to carry IPv6 traffic. IPv6 nodes (or networks) that are separated by IPv4 infrastructure can build a virtual link by configuring a tunnel. IPv6-over-IPv4 tunnels are modeled as single-hop. That is, the IPv6 hop limit is decremented by 1 when an IPv6 packet traverses the tunnel. The single-hop model serves to hide the existence of a tunnel. The tunnel is opaque to users of the network, and is not detectable by network diagnostic tools such as traceroute.

z/OS Communications Server does not support being a tunnel endpoint. This means that the z/OS Communications Server stack will have to have an IPv6 interface connected to an IPv6-capable router. The router is relied upon to handle all tunneling issues.

Stage 2

There are two logical networks:

- An IPv4 network, which is a separate IPv4 network. Assign a separate subnet to this IPv4 network.
- An IPv6 network. which is a separate IPv6 network. Assign a separate prefix to this IPv6 network.

The two logical networks may share the same OSA-Express adapters and the same physical network infrastructure, such as cabling, switches, etc.

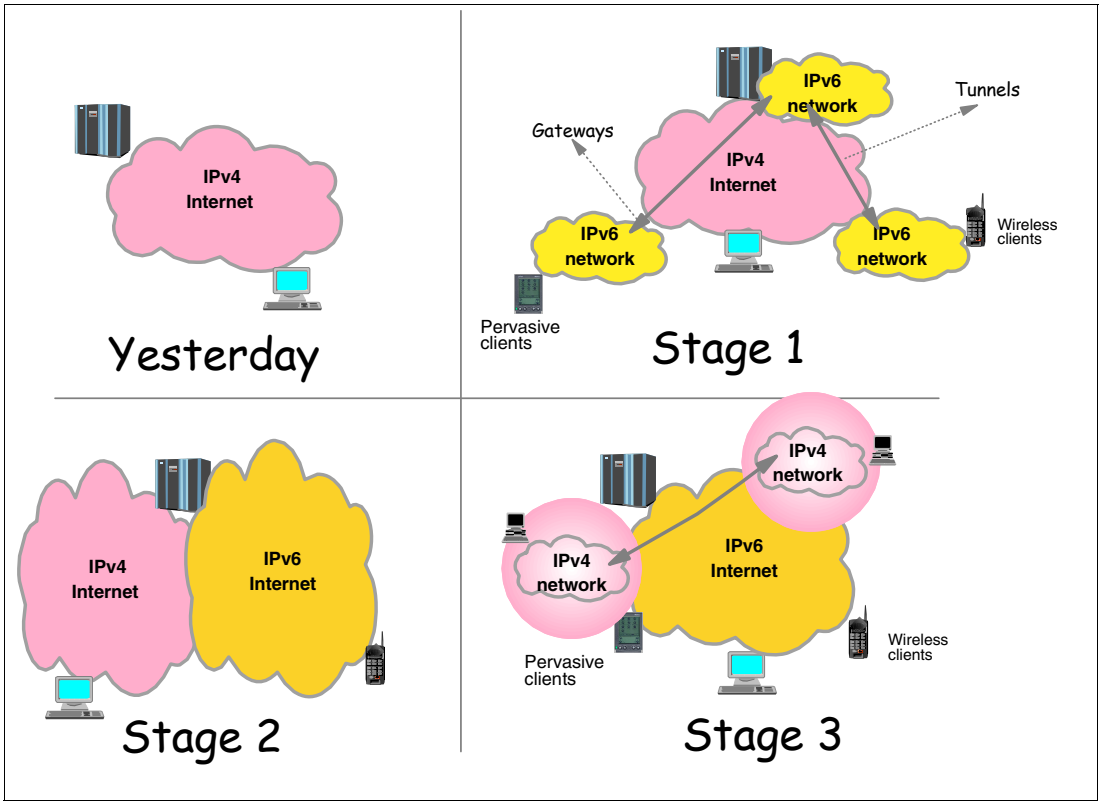

Figure 11-4 Migrations from IPv4 to IPv6

11.2.6 IPv6 OSPF support for dynamic routing (OSPFv3)

IPv6 OSPF is a dynamic routing protocol. It quickly detects topological changes in the AS (such as router interface failures) and calculates new loop-free routes after a period of convergence. Dynamic routing is needed on z/OS to support sysplex functions like VIPA, Dynamic VIPA, and Dynamic VIPA takeover. OMPROUTE has implemented IPv6 OSPF alongside its existing protocols IPv4 OSPF, IPv4 RIP, and IPv6 RIP with the following changes:

- There are new OMPROUTE configuration statements and display commands for IPv6 OSPF, which are similar to those for IPv4 OSPF.
- OMPROUTE implements RFC 2740 except for NSSA and MOSPF support, which is also not in OMPROUTE's IPv4 OSPF.

IPv6 RIP protocol

For IPv6, OMPROUTE implements the IPv6 RIP protocol described in RFC 2080 (RIPng for IPv6) and the IPv6 OSPF protocol described in RFC 2740 (OSPF for IPv6). It provides an alternative to the static TCP/IP gateway definitions. The MVS host running with OMPROUTE becomes an active OSPF or RIP router in a TCP/IP network. Either or both of these routing protocols can be used to dynamically maintain the host IPv6 routing table. For example, OMPROUTE can detect when a route is created, is temporarily unavailable, or if a more efficient route exists. If both IPv6 OSPF and IPv6 RIP protocols are used simultaneously, IPv6 OSPF routes are preferred over IPv6 RIP routes to the same destination.

Considerations when running OMPROUTE for OSPFv3

If at least one IPv6 OSPF interface is configured, IPv6 OSPF is enabled. The IPv6 OSPF interface is defined with an IPV6_OSPF_INTERFACE statement.

MTU is no longer coded on the IPv6 OSPF interface; it is obtained from the TCPIP stack. Maximum Transmission Unit (MTU) considerations are as follows:

- TCP/IP uses the MTU to determine the largest size frame to send. The MTU in effect for a given outbound send depends on several guidelines:
  - Enable path MTU discovery in configurations where traffic originating in the z/OS TCP/IP stack will traverse multiple hops with different MTU sizes.
  - When using OSA-Express Gigabit Ethernet (which supports an interface MTU of 8992), be aware that not all routers and switches support a value this large. Either ensure that all routers and switches in your configuration support 8992 or specify a lower configured_route_MTU.
  - When using OMPROUTE, specify the MTU keyword for each IPv4 interface.
  - When using OMPROUTE, configure all nodes on a LAN to use the same MTU value. Otherwise, you might encounter problems, such as OSPF adjacency errors.

Unlike OSPF for IPv4, IPv6 OSPF uses the IPv6 IPSEC protocol for authentication.

11.2.7 Network management support for IPv6

You must be able to manage IPv6 networks before they are deployed. Therefore, you should be able to monitor network interfaces, connections, and sysplex functions.

IPv6 support for SNMP TCP/IP subagent

In this release, there is added additional support for IPv6 MIB data. The IPv6 MIB data is supported in version-neutral MIB objects. Version-neutral MIB objects can support both IPv4 and IPv6 processing. In V1R6, the TCP/IP subagent was enhanced to support changed or additional version-neutral standard MIB data from the following IETF Internet drafts:

- IP-MIB from draft-ietf-ipv6-rfc2011-update-04.txt
- IP-FORWARD-MIB from draft-ietf-ipv6-rfc2096-update-05.txt
- TCP-MIB from draft-ietf-ipv6-rfc2012-update-04.txt

SMF119 records support

The redesign of SMF records from SMF118 to SMF119 in z/OS V1R2 did factor in IPv6 addresses, so most subtypes are already in z/OS V1R4 supporting IPv6 addresses. Some changes are needed to selected records to capture additional IPv6-related data, such as interface records and statistics records.

11.2.8 LDAP IPv6 support

The LDAP server can be configured to listen for secure and nonsecure connections from clients on one or more IPv4 or IPv6 interfaces on a system. With the listen configuration option on the LDAP server, the hostname or the IPv4 or IPv6 address, along with the port number, can target one or multiple IPv4 or IPv6 interfaces on a system.

IPv6 addressing in the LDAP server can only be used on a network that supports IPv6. You can continue to use IPv4 addressing in the LDAP server. IPv4 applications will not accept IPv6 addresses. However, the enhanced LDAP server works with IPv4 and IPv6 clients.

IPv6 addresses can be used in the LDAP server in the following:

- Listen, referral, and altserver config options
- Listen option on he server **start** command
- Referral and replication entries

Use of IPv6 addresses in LDAP must look like the following:

```
[FEC0::F4F7:0:0:7442:7501]:389
```

Where FEC0::F4F7:0:0:7442:7501 is the IPv6 address and 389 is the port.

11.3 Multilevel security

Multilevel security was introduced for Communications Server in z/OS V1R5. With z/OS V1R6, the following enhancements were made:

- Socket option access control

  This support is designed to give system administrators the ability to assign permission for z/OS users to set selected socket options using a SAF security server.

  Access control is provided for the SOL_SOCKET level, SO_BROADCAST option. This support is added to control the ability of a z/OS application to set the SO_BROADCAST socket option required to send broadcast datagrams. Without this support, the ability to send UDP or RAW datagrams to a broadcast address could be abused to create packet flood attacks or to circumvent network access controls.

  Socket option access control is provided with the SERVAUTH class resource profile EZB.SOCKOPT.sysname.tcpname.SO_BROADCAST. If this profile is defined, users of applications that set the SO_BROADCAST socket option must be permitted at least READ authority to this profile. Certain TCP/IP applications attempt to send datagrams to a broadcast address.

  TCP/IP programs known to set the SO_BROADCAST socket option include:

  - OMPROUTE
  - OROUTED
  - DHCP
  - binlsd
  - sntpd, when invoked with the -b option
- Inspect networking applications, such as sendmail, to determine if they are supported in a multilevel secure environment
- A new option to terminate the TCP/IP stack if inconsistent configuration information is detected in a multilevel secure environment

11.4 Job-specific source IP addressing

Job-specific source IP addressing allows each job to have its own IP address. A requirement for a single sysplex IP address, inbound and outbound, was provided in z/OS V1R4 with TCPSTACKSOURCEVIPA. To have a single IP address for an application is now supported with a job-specific source IP address in z/OS V1R6.

This may be a Distributed DVIPA, with sysplex-wide ephemeral port support. Each job has its own:

- Same source IP address for outbound as applications listen for client requests.

- Stack configuration with the SRCIP statement.

  You use the SRCIP/ENDSRCIP profile statement to designate the IP address or the interface name to be used by that job or jobs.

- Works with explicit bind() to inaddr_any.

  Job-specific source IP addressing establishes an IP address to be used as the source IP address for TCP connections initiated by an application that has not bound the socket or has bound the socket to inaddr_any or inaddr6_any before issuing the connect().

- Overrides all other source IP address selection functions.

In addition, job-specific source IP addressing allows a specific job or set of jobs to have a unique source IP address for outbound-initiated connections, different from other jobs, and independent of the order of addresses in the HOME list. If it is a VIPA source IP address designation, it overrides TCPSTACKSOURCEVIPA and normal SOURCEVIPA processing. This function would be useful when a job communicates with specific partners by ensuring that a single address gets used as the source address, which can simplify the firewall configuration. It may also be useful for a server application supported by Sysplex Distributor, so the Distributed DVIPA can be used as the source IP address for connections initiated by the server application instances.

11.4.1 SRCIP profile statement

This statement is used to designate that a specific job or jobs should use a particular Virtual IP Address or interface name (static VIPA or Dynamic VIPA, or real IP address) as the source address for outbound TCP connections. The statement has the following format:

```
>>SRCIP ----+-------------------------------------------------+-- ENDSRCIP --><
           +----JOBNAME-- jobname --+-- ipv4_address   ----+--+
                                    +-- ipv6_address ------+
                                    +-- ipv6_interface_name+
```

Where:

JOBNAME Indicates to create a job-specific source IP address designation with a designated IP address or interface name.

jobname The name of the job for which the designated interface should be used as the source IP address when the job initiates outbound TCP connections. The job name may end in an asterisk (*), and the job name of any job executing later that begins with the same characters before the asterisk will match this designation. If several different designations exist, then the actual source IP address used will be determined by the most complete match—either an exact match, or the most characters before the asterisk in the job-specific source IP address designation. If the jobname “*” is specified and is a VIPA source IP address, then this provides the same function as the TCPSTACKSOURCEVIPA parameter on the IPCONFIG and IPCONFIG6 statements.

ipv4_address A dotted-decimal notation for an IPv4 address. The IPv4 address is specified as it is in the TCP profile for the static VIPA, Dynamic VIPA (defined by VIPADEFINE DVIPA or previously activated via bind() or IOCTL SIOCSVIPA within a VIPARANGE) or any real IPv4 IP address. The specified IP address does not need to be defined prior to the processing of the SRCIP statement, but it must be defined before the first TCP connect

request is issued by the associated job, otherwise the connect request will fail.

ipv6_address A standard IPv6 notation for an IPv6 address. The IPv6 address is specified as it is in the TCP profile for the static VIPA, Dynamic VIPA (defined by VIPADEFINE DVIPA or previously activated via bind() or IOCTL SIOCSVIPA6 within a VIPARANGE) or any real IPv6 IP address. The specified IP address does not need to be defined prior to the processing of the SRCIP statement, but it must be defined before the first TCP connect request is issued by the associated job, otherwise the connect request will fail.

ipv6_interface_name
An IPv6 interface name. This name is specified as it is in the TCP profile for the static VIPA, Dynamic VIPA (defined by VIPADEFINE or VIPARANGE) or any real IPv6 interface. The specified interface name does not need to be defined prior to the processing of the SRCIP statement, but it must be defined before the first TCP connect request is issued by the associated job, otherwise the connect request will fail.

SRCIP example

The following is an example of defining the SRCIP/ENDSRCIP statements.

```
SRCIP
 JOBNAME USER15  9.43.242.5
 JOBNAME USER*   9.43.242.4
 JOBNAME USER15  2EC0::092B:F203
 JOBNAME JOB*    ETHER1
 JOBNAME *       9.43.242.3
ENDSRCIP
```

Note: To change or remove the JOBNAME entries from a previously defined SRCIP/ENDSRCIP profile statement, issue the VARY TCPIP,,OBEYFILE command for the modified SRCIP/ENDSRCIP profile statement. The new designations will completely replace the existing designations.

11.5 FTP-callable application programming interface

A new CALL interface is provided for applications to invoke the FTP client programmatically. This API supports programs that utilize a standard call interface. Samples are provided for COBOL, PL/I, and Assembler.

Specifically, a new callable API to the z/OS CS FTP client supports a CALL instruction to the module EZAFTPKS. CALL must pass parameters that include a place to put results of a request, the type of request, and other supporting parameters. The CALL interface is well-defined and is fully described in the following documents:

- FTP Callable Application Programming Interface (API) in *z/OS Communications Server: IP Programmer's Reference,* SC31-8787
- *z/OS Communications Server: IP Configuration Guide,* SC31-8775
- *z/OS Communications Server: IP User's Guide and Commands,* SC31-8780

11.5.1 z/OS FTP client programming interface

Provides an interface that allows an application to programmatically invoke the FTP client on z/OS from common environments (UNIX shell, TSO, or MVS batch job). The characteristics of the interface are:

- z/OS V1R6 provides a callable interface to be used from Assembler, Cobol, PL/I (or any z/OS-supported programming language that supports a call interface). There are plans to add C and REXX APIs in a later release.
- It is reentrant and supports multiple parallel FTP client sessions by tasks within an address space.
- For communication between the program and the interface, a simple set of commands and data areas are used (mappings for common programming languages are provided).
- Both blocking (wait for a response) and non-blocking (polling-mode) calls are supported.
- In non-blocking mode, progress replies can be returned to the calling application as the transfer progresses.
- The simple commands tell the interface what to do, for example: initialize, terminate, execute an FTP client command, process output from the FTP client command that was executed, or poll for command completion.
- Results are returned as structured fields in communication area control blocks (return codes from interface and server replies or possibly local commands), along with free-format replies from the FTP client code.
- Debugging options are provided.

11.6 TN3270 Server support

Currently, the TN3270 Server runs as a subtask of the IBM TCPIP stack address space. In z/OS V1R6, the following options are provided:

- Run the TN3270 Server as a separately started address space from TCPIP.
- Continue to run TN3270 Server as a subtask of the TCPIP address space.

The benefits of running TN3270 Server as a separate address space allows for prioritization of the TCP/IP address space versus the TN3270 Server address space. This reduces the chance for a TN3270 Server failure to cause a total TCP/IP address space failure. It also allows for easier problem diagnosis for both TCP/IP and TN3270. It is now also easier to control starting and stopping the TN3270 Server.

11.6.1 TN3270 Server considerations

The following considerations are necessary when running the TN3270 Server in its own address space:

- Profile statements are the same and must be in a file separate from TCP/IP.
- Commands are the same but must be directed to the intended TN3270 procedure name.
- Multiple TCP/IP stacks are supported:
  - One server per stack (affinity)
  - One server associated with all stacks (Generic Server)
- You must run the TN3270 Server with affinity for the following functions:
  - TN3270 SNMP subagent (and must be only Telnet to that TCP/IP)

- Generate an 8-character hostname in the Telnet SMF record
- WLM functions

11.6.2 TN3270 Server support for SNA Character Stream (SCS)

SNA Character Stream (SCS) format is now supported by the TN3270 Server for the Unformatted System Service (USS) processes. Until now, only USS tables and formats have been supported.

VTAM supports SCS, which can provide the same look for migrations. SCS is only supported for TN3270e connections (not TN3270) since this data is sent on the SSCP-LU session. A SCS table can be configured for TN3270e clients in the TN3270 Server profile.

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 251. Note that some of the documents referenced here may be available in softcopy only.

- *z/OS Version 1 Release 2 Implementation*, SG24-6235
- *z/OS Version 1 Release 3 and 4 Implementation*, SG24-6581
- *z/OS Version 1 Release 5 Implementation*, SG24-6326

Other publications

These publications are also relevant as further information sources:

- *z/OS MVS Planning: Operations*, SA22-7601
- *ServerPac: Using The Installation Dialog*, SA22-7815
- *z/OS and z/OS.e Planning for Installation,* GA22-7504
- *z/OS MVS Programming: Workload Management Services*, SA22-7619
- *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- *z/OS JES2 Commands,* SA22-7526
- *z/OS JES2 Diagnosis*, SA22-7531
- *z/OS JES2 Initialization & Tuning Guide,* SA22-7532
- *z/OS JES2 Initialization & Tuning Reference*, SA22-7533
- *z/OS JES2 Installation Exits*, SA22-7534
- *z/OS JES2 Introduction*, SA22-7535
- *z/OS JES2 Macros*, SA22-7536
- *z/OS JES2 Messages*, SA22-7537
- *z/OS JES2 Migration*, GA22-7538
- *z/OS Planning for Multilevel Security*, GA22-7509
- *z/OS MVS System Management Facilities (SMF),* SA22-7630
- *z/OS SDSF Operation and Customization,* SA22-7670
- *z/OS Resource Measurement Facility (RMF) Report Analysis*, SC33-7991
- *z/OS Resource Measurement Facility User’s Guide, SC33-7990*
- *z/OS Resource Measurement Facility (RMF) Programmer’s Guide*, SC33-7994
- *z/OS MVS Assm Services Reference IAR-XCT*, SA22-7607
- *z/OS UNIX System Services Planning*, GA22-7800
- *z/OS Security Server RACF Security Administrator’s Guide,* SA22-7683

- *z/OS UNIX System Services Programming: Assembler Callable Services Reference,* SA22-7803
- *z/OS MVS Programming: Authorized Assembler Services Guide*, SA22-7608
- *z/OS V1R6.0 ISPF Dialog Developer's Guide and Reference,* SC34-4821
- *z/OS ISPF Dialog Tag Language Guide and Reference*, SC34-4824
- *z/OS DFSMS Migration*, GC26-7398
- *z/OS Integrated Security Services Enterprise Identity Mapping (EIM) Guide and Reference*, SA22-7875
- *z/OS Security Server LDAP Client Programming*, SC24-5924
- *z/OS ISPF Dialog Developer's Guide and Reference*, SC34-4821
- *z/OS V1R6.0 ISPF User's Guide Volume II,* SC34-4823
- *z/OS V1R6.0 ISPF Planning and Customizing,* SC34-4814
- *z/OS ISPF Software Configuration and Library Manager (SCLM) Reference,* SC34-4818
- *z/OS MVS Program Management: Advanced Facilities*, SA22-7644
- *Support for Unicode: Using Conversion Services,* SA22-7649
- *z/OS MVS Programming Workload Management Services,* SA22-7619
- *z/OS C/C++ Run-Time Library Reference,* SA22-7821
- *z/OS V1R6.0 ISPF Services Guide,* SC34-4819
- *z/OS Cryptographic Services PKI Services Guide and Reference,* SA22-7693
- *z/OS OCSF Applications Programming*, SC24-5899
- *z/OS Introduction and Release Guide,* GA22-7502
- *z/OS Language Environment Vendor Interfaces*, SA22-7568
- *DB2 Universal Database™ for z/OS: RACF External Security Module Guide and Reference*, SA22-7938
- *Security Server LDAP Server Administration and Use*, SC24-5923
- *z/OS Communications Server IP CICS Sockets Guide*, SC31-8807
- *z/OS Communications Server: IP Programmer's Reference,* SC31-8787
- *z/OS Communications Server: IP Configuration Guide,* SC31-8775
- *z/OS Communications Server: SNA Network Implementation Guide,* SC31-8777-04
- *z/OS Communications Server: IP User's Guide and Commands,* SC31-8780
- *z/OS DFSMS: Using Data Sets,* SC26-7410
- *z/OS C/C++ Compiler and Run-Time Migration Guide for the Application Programmer,* GC09-4913
- *Tivoli Netview OS/390 Installation: Migration Guide Version 1,* SC31-8768
- *DFSMSrmm Primer,* SG24-5983

Online resources

These Web sites and URLs are also relevant as further information sources:

- The RMF Spreadsheet Reporter is available for download via the RMF homepage:

  http://www-1.ibm.com/servers/eserver/zseries/zos/rmf/

- The RMF Spreadsheet Reporter Version 5.1 is available for download from the RMF tools Web site:

  http://www-1.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/rmftools.htm#spr_win

- The most recent version of RMF PM can be downloaded from the RMF PM Web site at:

  http://www-1.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/pmweb/pmweb.htm

- There is a very useful summary of metrics in RMF PM that are available to monitor your system and your sysplex resources at the following Web site:

  http://www-1.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/pmweb/pm_metrics.htm

- There is a document available on the Web that describes what's new in DFSOR. It can be found at:

  http://www.storage.ibm.com/software/sort/mvs/release_14/pdf/sortnew.pdf

- The Unicode collation algorithm is described in detail in the Unicode Consortium's technical report #10. For the detail report, refer to:

  http://www.unicode.org/unicode/reports/tr10/

- Allkeys.txt Unicode file can be found at:

  http://www.unicode.org/unicode/reports/tr10/allkeys.txt

- For a detailed explanation of normalization, including specific information about the normalization forms, refer to The Technical Report #15 provided by the Unicode Consortium at:

  http://www.unicode.org/unicode/reports/tr15/

- Unicode consortium Collation Technical Report:

  http://www.unicode.org/reports/tr10/

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

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

z/OS Version 1 Release 6 Implementation

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

z/OS Version 1 Release 6 Implementation

z/OS, ServerPac, WLM, RMF

DFSMS, SMP/E, z/OS UNIX

OSA, ISPF, Communication Server

This IBM Redbook discusses the many enhancements to z/OS Version 1 Release 6, and provides information to help you install, tailor, and configure this release.

It first offers a broad overview of z/OS Version 1 Release 6, and then goes into detail on how to install and tailor z/OS and the many components that have been enhanced, such as: the z/OS base control program (BCP), ServerPac, DFSMS, Workload Manager (WLM), RMF, SMP/E, z/OS UNIX, ISPF, and Communication Server.

This redbook is intended for systems programmers and administrators responsible for customizing, installing, and migrating to the newest levels of z/OS.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6377-00 ISBN 0738492345