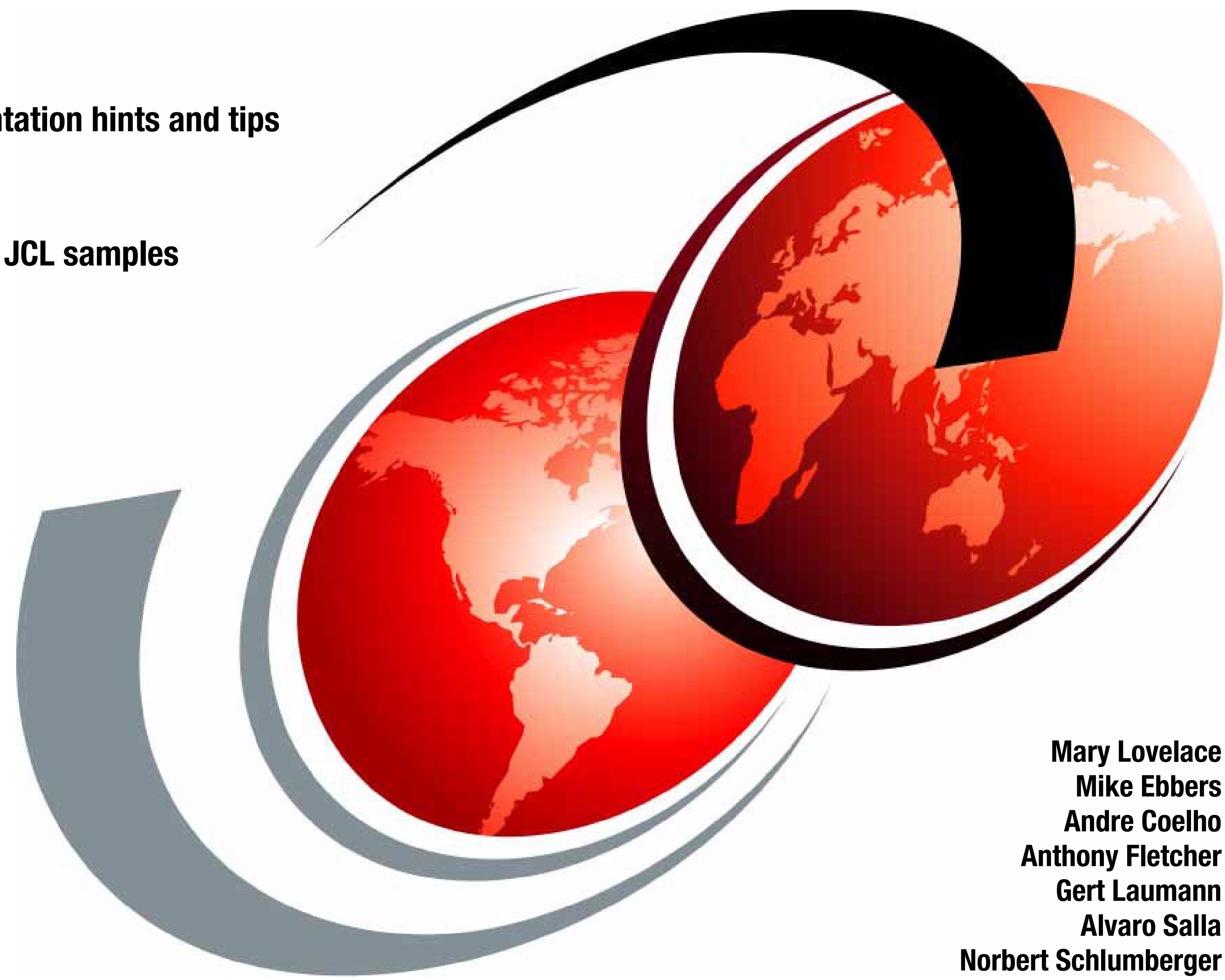
z/OS V1.13 DFSMS Technical Update

- Features and functions of DFSMS V1.13
- Implementation hints and tips
- Code and JCL samples



Mary Lovelace
Mike Ebbers
Andre Coelho
Anthony Fletcher
Gert Laumann
Alvaro Salla
Norbert Schlumberger

Redbooks

ibm.com/redbooks

International Technical Support Organization

z/OS V1.13 DFSMS Technical Update

August 2012

SG24-7961-00

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

First Edition (August 2012)

This edition applies to Version 1, Release 13 of IBM Data Facility Storage Management Subsystem (DFSMS), product number 5694-A01.

Contents

Figures

Tables

Examples

Notices

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

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

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

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

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

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

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

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

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

COPYRIGHT LICENSE:

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

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

AIX®
CICS®
DB2®
DS8000®
FICON®
FlashCopy®
GDPS®
Geographically Dispersed Parallel Sysplex™
HyperSwap®
IBM®
MVS™
OS/390®
Parallel Sysplex®
RACF®
Redbooks®
Redbooks (logo) ®
RMF™
S/390®
System Storage®
System z10®
System z®
TDMF®
Tivoli®
z/Architecture®
z/OS®
z/VM®
z/VSE®
z10™
zEnterprise®
zSeries®

The following terms are trademarks of other companies:

Windows, 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.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

Preface

Each release of IBM® Data Facility Storage Management Subsystem (DFSMS) builds on the previous version. The latest release, IBM z/OS® V1.13 DFSMS, provides enhancements in these areas for the z/OS platform in a system-managed storage environment:

- Storage management
- Data access
- Device support
- Program management
- Distributed data access

This IBM Redbooks® publication provides a summary of the functions and enhancements in z/OS V1.13 DFSMS. It provides information that you need to understand and evaluate the content of this DFSMS release, along with practical implementation hints and tips. This book also includes enhancements that are available by enabling PTFs that have been integrated into z/OS DFSMS V1.13.

This book was written for storage professionals and system programmers who have experience with the components of DFSMS. It provides sufficient information so that you can start prioritizing the implementation of new functions and evaluating their applicability in your DFSMS environment.

The team who wrote this book

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

Mary Lovelace is a Consulting IT Specialist at the International Technical Support Organization. She has more than 20 years of experience with IBM in large systems, storage, and storage networking product education, system engineering and consultancy, and systems support. She has written several Redbooks publications about z/OS storage products, IBM Tivoli® Storage Productivity Center, Tivoli Storage Manager, and Scale Out NAS.

Mike Ebbers is a Project Leader and Consulting IT Specialist at the ITSO, Poughkeepsie Center. He has worked for IBM since 1974 on mainframe projects, including time as an SNA Specialist.

Andre Coelho is an IT Storage Specialist in Rio de Janeiro, Brazil. He works for IBM West Internal Accounts, Boulder, Colorado. He has 27 years of experience in z/OS and prior operating systems. His areas of expertise include DFSMSdfp, DFSMShsm, and DFSMSrmm.

Anthony Fletcher is an IT Specialist working with the Global Services Delivery (GSD) z/OS software platform of IBM Global Services on-demand Infrastructure Services, based in New Zealand. He works for New Zealand and Australia. He has over 35 years of experience in z/OS, IBM OS/390®, and their predecessors and related components, both as a customer of IBM and with IBM Global Services. He is a team leader for the mainframe operations of four diverse clients in the banking, airline, investment, and telecommunications industries. He holds a degree in Electrical Engineering from SALFORD University, Lancashire, UK. His main areas of expertise include DFSMS, DFSMSrmm, and DFSMShsm. In addition, he has a working knowledge of IBM RACF®. He has been involved with writing several IBM Redbooks publications. He also has experience in installing non-IBM products for the GSD platform.

Gert Laumann is an IT Specialist in Integrated Technology Delivery, Server Systems Operations / Storage Management in Denmark. He has 26 years of experience in z/OS, working with storage management since 1989. He is the team leader for the Danish mainframe storage team. His experience is mostly with IBM products (DFSMSdfp, DFSMSdss, DFSMShsm, and DFSMSrmm), but he has also worked with OEM software and hardware products. His focus has mainly been automation and standardization across mainframe customer platforms. He currently helps customers share IBM DASD and tape hardware.

Alvaro Salla is an IBM retiree. He worked in IBM for more than 30 years in large systems. He currently teaches ITSO workshops on z/OS performance around the world. Alvaro has co-authored many IBM Redbooks publications and spent many years teaching about large systems, from the S/360 to the IBM S/390®. He has a chemistry engineering degree from the University of Sao Paulo, Brazil.

Norbert Schlumberger is an IT Architect with IBM Germany. He has 34 years of experience in storage software and storage management for IBM and customer systems, including 22 years of experience in DFSMSrmm. He also has experience in DFSMShsm and possesses a good knowledge of RACF. Norbert's areas of expertise include performing conversions from vendor tape management products to DFSMSrmm, new DFSMSrmm implementations, and marketing support for DFSMSrmm, including IBM 3494 and IBM 3495 ATLs, VTSs, and vendor robotics. He wrote a tool called “Tape Copy Tool: Enhancing DFSMSrmm” to copy data sets residing on tape from one media to another media, including updating the ICF user catalog and DFSMSrmm after all data is successfully copied. He has worked at IBM for 37 years.

Thanks to the following people for their contributions to this project:

Bob Haimowitz
International Technical Support Organization, Raleigh Center

Luiz Carlos Bastos de Amorim
Guillermo Gil Carral
IBM Brazil

Kevin Goldsmith
IBM Development, Tucson

Michael R. Mayne
Huntsville Hospital System, Huntsville, AL

Now you can become a published author, too!

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

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

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

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

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

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

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

- Stay current on recent Redbooks publications with RSS Feeds:

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

1

Introduction to DFSMS V1.13 enhancements

This chapter provides an overview of the IBM Data Facility Storage Management Subsystem (DFSMS) product and its primary functions in the z/OS operating system. It also addresses the enhancements introduced with the IBM announcement of DFSMS Version 1 Release 13 (DFSMS V1.13). Those enhancements are covered in more detail In the rest of this book. The new features are grouped in this chapter by major DFSMS components, such as DFSMSdfp, DFSMSdss, DFSMShsm, DFSMSrmm, and DFSMStvs. Some of the new functions address performance, whereas others promote better reliability, availability and serviceability (RAS) in systems that run z/OS.

This chapter contains the following sections:

- DFSMS highlights
- DFSMSdfp enhancements
- DFSMS components and enhancements

1.1 DFSMS highlights

DFSMS comprises a suite of related data and storage management products for the z/OS system. DFSMS is an operating environment that helps automate and centralize the management of storage (DASD, tape, and optical devices). It does so based on the policies that your installation defines for availability, performance, space, and security. The core of DFSMS is the Storage Management Subsystem (SMS). Using SMS, the storage administrator defines policies that automate the management of storage and hardware devices. These policies describe data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements for the system.

DFSMS is a software suite that automatically manages data from creation to expiration, and is an exclusive element of the z/OS operating system. The following components comprise the DFSMS family:

- DFSMSdfp, a base element and co-requisite of z/OS, has the following functions:
  - Storage management that includes Interactive Storage Management Facility (ISMF), which allows you to define and maintain policies to manage your storage resources. It is used to define data classes, storage classes, management classes, storage groups, aggregate groups, copy pools, and automatic class selection (ACS) routines. You can also use the NaviQuest tool under ISMF to migrate to SMS, maintain your SMS configuration, and perform testing implementation and reporting tasks in batch.
  - Tape mount management improves tape usage and reduces tape costs by emulating tape on direct access storage device (DASD). This configuration is useful if a Virtual Tape Server (VTS) controller is not installed.
  - Data Management helps you store and catalog information about DASD, optical, and tape devices so that it can be quickly identified and retrieved from the system. The access methods, such as Virtual Storage Access Method (VSAM) and I/O support routines (Open/ Close/ End of Volume), are included here.
  - Device management. DFSMSdfp can be used when you define your input and output (I/O) devices to the system and in controlling those devices in the z/OS environment. Included is the I/O driver code, Media Manager, that interfaces directly with the Input/Output Supervisor (IOS) in z/OS.
  - DFSMSdfp utilities, which are simple programs that run commonly needed functions. DFSMS provides utility programs to assist you in organizing and maintaining data.
  - Distributed data access allows all authorized z/OS systems and users in a network to use system-managed storage or automated storage management. DFSMSdfp uses the Distributed FileManager/IBM MVS™ or the z/OS Network File System to enable remote clients in a network. These remote clients access data and storage resources on z/OS systems.
  - Advanced Copy Services includes remote and point-in-time copy functions that provide backup and recovery of data. When used before a disaster occurs, Advanced Copy Services provides rapid backup of critical data with minimal impact to business applications. If a disaster then occurs to your data center, Advanced Copy Services provides rapid recovery of your critical data.
  - Object access method (OAM) provides storage, retrieval, and storage hierarchy management for objects that contain images.

- DFSMSdss, an optional element feature of z/OS, has the following functions:
  - Very fast data movement and replication, allowing you to move or copy data between volumes of like and unlike device types.
  - Space management can reduce or eliminate DASD free-space fragmentation.
  - Data backup and recovery provides you with host system backup and recovery functions at both the data set and volume levels. It also includes a stand-alone restore program that you can run without a host operating system.
  - Data set and volume conversion can convert your data sets and volumes to system-managed storage (SMS).
- DFSMShsm, an optional element feature of z/OS. You must have DFSMSdss to use the DFSMShsm functions:
  - Storage management provides automatic DASD storage management, relieving users from manual storage management tasks.
  - Space management improves DASD space usage by keeping only active data on fast-access storage devices. It automatically frees space on user volumes by deleting eligible data sets, releasing over-allocated space, and moving low-activity data to lower cost-per-byte devices such as tapes. This process occurs even if the job does not request this device type.
  - Tape mount management can write multiple output data sets to a single tape, making it a useful tool for implementing tape mount management under SMS. When you redirect tape data set allocations to DASD, DFSMShsm can move those data sets to tape as a group during interval migration. This method greatly reduces the number of tape mounts on the system. DFSMShsm uses a single-file format, which improves your tape usage and search capabilities.
  - Availability management backs up your data automatically or by command to ensure availability if accidental loss of the data sets or physical loss of volumes should occur. DFSMShsm also provides aggregate backup and recovery support (ABARS) for user-defined groups of data sets (aggregates). This function allows you to restore critical applications at the same location or at an off site location.
- DFSMSrmm, an optional element feature of z/OS with the following functions related to your removable media resources, including tape cartridges and reels.
  - Library Management, where you can create tape libraries to balance the work of your tape drives and assist the operators. Tape libraries are collections of tape media associated with tape drives. DFSMSrmm can manage a removable media library that incorporates all other libraries, such as:
    - System-managed manual tape libraries
    - System-managed automated tape libraries. Examples of automated tape libraries include IBM TotalStorage Enterprise Automated Tape Library (3494) and IBM TotalStorage Virtual Tape Servers (VTS).
    - Non-system-managed or traditional tape libraries, including automated libraries such as a library under Basic Tape Library Support (BTLS) control.
  - Shelf Management, where information about removable media by shelves is grouped into a central online inventory to track the volumes on those shelves.
  - Volume management, to control and track the movement and retention of tape volumes throughout their lifecycle.
  - Data set management that records information about the data sets on tape volumes. DFSMSrmm uses the data set information to validate volumes and to control the retention and movement of those data sets.

- DFSMStvs is an optional feature element of the Transactional VSAM Services. It allows you to share VSAM clusters across IBM CICS®, batch, and object-oriented applications on z/OS or distributed systems. DFSMStvs enables concurrent shared updates of recoverable VSAM data sets by CICS transactions and multiple batch applications.

  DFSMStvs enables 24-hour availability of CICS and batch applications. DFSMStvs is built on top of VSAM record-level sharing (RLS), which allows you to share recoverable VSAM data sets at the record level.

The remainder of this chapter summarizes the enhancements in DFSMS V1.13. Each subsequent chapter details these enhancements.

1.2 DFSMSdfp enhancements

This section explains the DFSMSdfp modifications in DFSMS V1.13, clustered by DFSMSdfp components.

1.2.1 VSAM RLS modifications

The following enhancements were introduced for VSAM RLS:

- Buffer management facility (BMF) has the following enhancements:
  - Using time stamps rather than unreferenced interval count (UIC) to track CI buffer aging in the SMSVSAM data space buffer pools (31-bit).
  - Placing CI buffers on the top of the BMF read buffer chain when referenced.
  - Modifying BMF LRU buffer clean up to process only CI buffers at the end of CI buffer chains.
- New storage class option to disconnect the RLS cluster from buffering. This disconnect is done as soon as the last RLS is closed. Before DFSMS V1.13, the RLS cluster remained connected for a short period after a close by using buffer pool resources.
- VSAM OPEN first time failure memory dump data capture. DFSMS V1.13 release includes a mechanism for taking a supervisor call (SVC) memory dump during a VSAM open if logic errors are encountered. The running task continues and does not abend.

1.2.2 Open/Close/EOV enhancements

The following enhancements were introduced for Open/Close/EOV:

- A new installation option (OCE_ABEND_DESCRIP = YES | NO) in the PARMLIB member DEVSUPxx. This function adds descriptive text to O/C/EOV abend console messages. This text is associated with the most commonly experienced abend and return codes that come from the I/O support routines.
- Change to calculations of the BUFNO value by QSAM for concatenated data sets by using the keyword MULTSDN at DCBE. This function reduces the probability of out-of-storage abend conditions.
- O/C/EOV now accepts subsystem DCBs with XTIOTs, in addition to non-VSAM DCBs. Non-VSAM access methods are also called Basic Access Methods (BAM). These methods include the QSAM, BSAM, BDAM, BPAM, and EXCP methods. Support for recovery of a missing or out-of-order tape volume situations in a tape multivolume data set. The recovery is implemented by the use of the Label Anomaly (LABAN) tape exit routine.

- ~~A new FREEVOL=EOV JCL parameter allows different tasks in the same MVS systems in a sysplex to read a multivolume tape data set concurrently. While one task is reading a volume, another can read another volume (of the same data set) at the same time.~~ A new FREEVOL=EOV JCL parameter allows different tasks in the same or different systems of a z/OS GRS Plex (global resource serialization complex) to read different volumes of a multivolume tape data set concurrently.This process is possible because each VOLSER of the last read volume is dequeued at end-of-volume (EOV).
- New O/C/EOV diagnostic data added to SMF 14/15 Type records Extended Information Segment Type 8. This data provides troubleshooting information when a DCBE is invalidated or a partial release is not run for a DASD cluster.

1.2.3 z/OS catalog modifications

The following enhancements were introduced for z/OS catalog.

Catalog enhancements

The following enhancements were introduced for the catalog:

- New CATALOG PARMLIB member named IGGCATxx can be added. It contains parameters exclusively for the catalog environment.
- Alias number constraint relief by enabling a new catalog record type V, Catalog Connector Extension record, to hold extensions to the connector record. The maximum number of extension records is now 255, which raises the limit from the current 3,500 to 500K aliases.
- VSAM volume data set (VVDS) expanded to store 1048575 CIs, a 16 times increase in capacity. The VVDS allocation size in cylinders can now reach 5825 cylinders (87375 tracks).
- Replace catalog pseudo close with standard VSAM close.

IDCAMS enhancements

The following enhancements were introduced for IDCAMS:

- Removed AMS LISTCAT NOIMBED and NOREPLICATE displays. Starting with DFSMS 1.5, these options are no longer valid in an IDCAMS DEFINE. However, data sets already defined with such options are still supported. At DFSMS V1.13, the AMS LISTCAT command no longer displays either a NOIMBED or a NOREPLICATE, when displaying attributes for a particular data set.
- AMS LISTCAT of Catalog CSI. This function of the Catalog Search Interface (CSI) implements improved filtering of catalog entries to be listed by LISTCAT.
- AMS `Delete UCAT` WTOR. With this DFSMS V1.13 support, IDCAMS issues a confirmatory WTOR message (IDC1999I) before deleting objects cataloged in the UCAT to be deleted. Then the VVDS and volume table of contents (VTOC) entries for objects defined in the User catalog are deleted (and scratched). Whether they are deleted depends on the reply to the message.
- AMS RAS 31-bit Storage Pool Manager (Phase 2). Some AMS commands (but not all) can now issue GETMAINs above the 24-bit addressing line.

1.2.4 BAM enhancements

Basic access methods (BAM) are non-VSAM access methods. The main enhancement is that DFSMS V1.13 release provides a trace facility. It improves the serviceability of the

BSAM, QSAM, and BPAM access methods. This trace facility can be used by IBM service personnel to create a trace table that records a chronology of internal events.

1.2.5 PDSE enhancements

The following enhancements were introduced for PDSE:

- The **DISPLAY SMS** command provides additional diagnostic information about PDSE data set connections. When a PDSE is opened, it is connected preferentially to the PDSE1 address space task. However, if the PDSE data set was opened before the start of this restartable address space, this data set is connected to a PDSE address space task. The command shows the status of connections, mainly for problem determination.
- The **VARY SMS** command, which can be used to run a refresh in the status of a particular PDSE data set.
- The IEBPDSE utility to validate and verify the status of one or more PDSE data sets. If a PDSE is damaged, it might affect the PDSE or PDSE1 address space tasks and multiple users. The IEBPDSE utility can help determine what is causing PDSE problems.

1.2.6 EAV enhancements

The major EAV enhancement is support for data sets of up to 1 terabyte (TB) in size.

1.2.7 OAM enhancements

The following enhancements were introduced for OAM:

- New file system sublevel in the OAM storage hierarchy
- Displaying OAM Status based on **MODIFY OAM**
- Adding wildcard support for the **MODIFY OAM,START,STORGRP**

1.2.8 zFS enhancements

The following enhancements were introduced for zFS:

- zFS automatic takeover of disabled aggregates. In z/OS V1R13, zFS can automatically recover disabled aggregates n both single-system and sysplex environments when multiple systems are running in zFS sysplex-aware mode. This process is intended to eliminate the need to recover the file system manually before applications close and reopen the files to regain access to them.
- zFS refresh. In DFSMS V1.13, zFS maintains existing connections to zFS file systems while recovering from internal errors when possible. This process is intended to provide less-disruptive recovery from most internal zFS problems. It is designed to allow applications with open files to try file system operations again successfully after zFS recovery is complete.
- zFS client direct I/O. zFS in DFSMS V1.13 introduces the new Direct I/O function. This function allows all MVS members of a sysplex to run zFS file system read and write I/O operations directly. This feature is expected to yield substantial performance gains for systems that would not have been zFS owning systems in the prior design. These gains are without performance impact to systems that would have been zFS owning systems.

1.2.9 DADSM enhancements

The following enhancements were introduced for DADSM:

- Volume contention
- Device availability
- Device summary
- IPCS

1.2.10 SDM enhancements

The following enhancements were introduced for SDM:

- XRC time stamp suppression
- CC PARMLIB support
- MaxTotalReader task
- XRCSTART error handling
- XRC query filter option

1.2.11 SMS/ISMF enhancements

The following enhancements were introduced for SMS/ISMF:

- New SMS PARMLIB parameter. Error messages generated during SMS processing are passed back to the caller, who is responsible for externalizing these messages. In DFSMS V1.13, this approach has been modified so that SMS externalizes its own error messages to the hardcopy and job logs.
- Increased retention period to larger than 9999 days. With this support, the maximum retention period has been increased to 93000 days, or about 254 years.
- Provides updated volume space statistics. In certain situations, ISMF displays inaccurate space information about a storage group.
- DFSMSdss cross system sysplex z/OS member notification (XSYS). This function allows DFSMSdss to trigger a mechanism that notifies other eligible z/OS members of a sysplex that:
  - A change to a DASD volume serial number (VOLSER) has occurred.
  - The volume table of contents (VTOC) location or size has changed.
  - The VVDS on the output volume has been overwritten.
  - The free space information in the VTOC index (VTOCIX) has been rebuilt.

1.3 DFSMS components and enhancements

This section outlines the DFSMS components introduced earlier, along with the V1.13 enhancements.

1.3.1 DFSMSdss enhancements

The DFSMSdss cross system sysplex z/OS member notification (XSYS) allows DFSMSdss to trigger a mechanism that notifies other eligible z/OS members of a sysplex whenever:

- A change occurs to a DASD VOLSER.
- The VTOC address is changed.
- An incompatibility arises in the VTOCIX.

1.3.2 DFSMShsm enhancements

The following enhancements were introduced for DFSMShsm:

- A new “on demand” Migration feature
- Performance and availability improvements to CDS backup
- RAS and usability improvements

1.3.3 DFSMSrmm enhancements

The following enhancements were introduced for DFSMSrmm:

- More “last change” details
- Last reference date for vital record specification (VRS)
- Interactive System Productivity Facility (ISPF) navigation enhancements
- Show effective retention/expiration date
- Search data set extensions
- TVEXTPURGE extra days
- More information about expiration date source
- Ability to exclude data sets from VRSEL
- New RETENTIONMETHOD (EXPDT)
- Enhanced tape copy support

2

Open, Close, and end-of-volume

This chapter introduces the functions of the DFSMSdfp routines Open, Close, and end-of-volume (EOV). These are I/O support routines that some manuals and documents call O/C/EOV. The first section is a detailed description of the new I/O support routines features delivered with DFSMS 1.13. If you are an experienced O/C/EOV user, you can continue to 2.3, “I/O support enhancements at DFSMS V1.13” on page 22.

This chapter contains the following sections:

- Summary of O/C/EOV enhancements in DFSMS V1.13
- I/O support routines introduction
- I/O support enhancements at DFSMS V1.13

2.1 Summary of O/C/EOV enhancements in DFSMS V1.13

DFSMS V1.13 includes the following enhancements:

- A new installation option, OCE_ABEND_DESCRIP = YES | NO, in the PARMLIB member DEVSUPxx. This function appends descriptive text for O/C/EOV abend console messages associated with the most commonly experienced abends and the return codes coming from the I/O support routines. See “O/C/EOV text with abend console messages” on page 23 for more details.
- A change in the BUFNO value calculation done by QSAM for concatenated data sets by using the keyword MULTSDN at DCB Extension (DCBE). This function reduces the probability of out-of-storage abend conditions. For more information, see 2.3.2, “MULTSDN changes for QSAM concatenation” on page 23.
- O/C/EOV now accepts subsystem data control blocks (DCBs) with extended task input/output tables (XTIOTs), in addition to non-VSAM DCBs. For more information, see 2.3.3, “O/C/EOV subsystem DCBs with XTIOTs” on page 24.
- Support for recovering missing or out-of-order tape volumes when processing a tape multivolume data set. The recovery uses the Label Anomaly (LABAN) tape exit routine. For more information, see 2.3.4, “Recovering multivolume tape data sets” on page 24.
- A new FREEVOL=EOV JCL parameter allows different tasks in the same z/OS or in other z/OS systems in a sysplex to read multivolume tape data sets concurrently. While one task is reading a volume, another task can read another volume of the same data set. This process is possible because each volume serial number (VOLSER) of the last read volume is dequeued at EOV. For more information, see 2.3.5, “FREEVOL=EOV JCL parameter” on page 25.
- New O/C/EOV diagnostic data added to SMF 14/15 Type records Extended Information Segment Type 8. This data provides troubleshooting information when a DCBE is invalidated or a partial release is not performed for a DASD cluster. For more information, see 2.3.6, “New O/C/EOV diagnostic data added to SMF 14/15” on page 26.
- Improved ISO/ANSI V4 tape label processing. For more information, see 2.3.7, “Improve ISO/ANSI V4 tape label processing” on page 27.
- Saved O/C/EOV RACF return / reason codes and parameter list in a memory dump. For more information, see 2.3.8, “Saved RACF return and reason codes and parameter list” on page 27.
- SAM internal trace facility provides a trace facility to improve the serviceability of the BSAM, QSAM, and BPAM access methods. IBM service personnel can use this trace facility to create a trace table that records a chronology of internal events.

2.2 I/O support routines introduction

Before exploring the new features of DFSMdfp V1.13 O/C/EOV routines, this section covers the basic concepts.

2.2.1 Steps to process a data set

This section addresses the steps necessary to process a Virtual Storage Access Method (VSAM) cluster. This process that does not differ greatly from a non-VSAM data set.

- The installation defines a VSAM cluster by using the IDCAMS utility program or a JCL DD statement with DISP=NEW. This process implies generating a DSCB in a volume table of contents (VTOC) and cataloging this cluster, usually in a z/OS user catalog (UCAT). Optionally, the initial cluster can be initially populated with data by using the Repro function of IDCAMS.
- The system allocates the cluster to establish the logical link between a task (task control block or TCB) application program and the cluster. The MVS Allocation routine (SVC 99) component runs this function. This routine for allocating a cluster can be started in three ways:
  - Through a JCL DD card in a batch job. In this case, the initiator started the allocation routine.
  - A transaction manager (such as CICS or IMS/DC) issues the Dynalloc (SVC 99) service macro. The information for the cluster is provided by the installation in a special parameter list such as file control table (FCT) for CICS.
  - An application task program itself issues the Dynalloc macro to start the allocation routine directly. Certain options of the Dynalloc service macro demand a task with authorized program facility (APF) authorization.
- The application program task opens (with SVC 19) the cluster, identifying it with a DDNAME parameter in the access method control block (ACB). A program accesses data through GET, or READ and PUT, or WRITE macros. These application programming interfaces (APIs) branch to an access method, in this case VSAM. An access method such as VSAM runs the following functions:
  - Produces an I/O channel program
  - Blocking
  - Provides buffering
  - If needed, it synchronizes the task with the I/O event (Wait/Post mechanism)
  - Implements I/O recovery, if needed
  - Passes the control to an I/O driver (a z/OS component) through an SVC instruction.

  When using VSAM, it issues the SVC 121. Other access methods use SVC 00, which is also known as EXCP. The interrupt caused by this instruction passes the control, in supervisor state, to the z/OS component Media Manager, an I/O driver.

  Media Manager, after translating and page fixing the channel program, starts the Input/Output Supervisor (IOS) through a branch instruction. The IOS is another z/OS component.

  IOS issues the SSCH instruction that starts the I/O operation through the system assist processor (SAP) and the channels. Then the processor is sent back to VSAM code that decides whether there is a need of a task wait situation.

  - If there is a need to process a new DASD extent, the EOV or SVC 31 routine is started.
  - At cluster-end processing, the application program issues a Close (SVC 20) command for the cluster. The Close cleans up after tasks run by the Open routine.
  - The same entity that started the Allocation routine does it again but for unallocation purposes.

2.2.2 Control block

The control block is the major information anchor for the Open, Close, and EOV functions. The control block has different contents and names depending on the access method that deals with the data set opened. For VSAM, the control block is the ACB. For all the other access methods, such as QSAM, BSAM, BDAM, BPAM, and EXCP, it is the DCB.

DCB/ACB is constructed by the application source program through the macros DCB, ACB, or GENCB. DCB/ACB contains information that describes the logical aspects of the data set. These include the following logical aspects:

- Logical record size and format
- Access type (sequential or direct)
- Address of the routine processing the end of file condition
- Data set organization

The DCB is located below the 16 MB line for compatibility reasons. It has an optional extension called a DCBE that can be located above or below that line. This DCBE enables new I/O features while keeping compatibility at the application source code level. The DCBE provides functions that augment those provided by the DCB. The DCBE is pointed to by a DCB field. This pointer is created through a keyword in DCB and GENCB macros. DCBE options cannot be declared at DD statements for late binding.

Table 2-1 shows the parameters of the DCBE macro.

Table 2-1 DCBE macro parameters

[label]	DCBE	[,BLKSIZE=*n*] [,BLOCKTOKENSIZE={LARGE | SMALL}] [,CAPACITYMODE=*XCAP*] [EADSCB=OK | NOTOK] [EODAD=*relexp*] [,FIXED=USER] [GETSIZE={YES | NO}] [LOC=[ANY | BELOW}] [,MULTACC=*n*] [,MULTSDN=*n*] [,NOVER={YES | NO}] [,PASTEOD={YES | NO}] [RMODE31=[BUFF | NONE}] [,SYNAD=*relexp*] [,SYNC=[SYSTEM | NONE}]

The physical properties of the data set are usually provided in a job control language (JCL) DD statement. These include the following properties:

- Data set name
- DASD space needed (primary and secondary)
- Device type

Using a DD statement to define those properties at execution time is convenient because the source code is not bound, for example, to a data set name. Therefore, you do not need to recompile the program if, for example, a data set name is changed. In addition, some logical properties are allowed to be defined at DD statement, providing a late binding approach. The connection between the DCB/ACB logical properties and the DD statement physical properties is done through the field DDNAME in the ACB/DCB.

2.2.3 Open routine (SVC 19)

The Open routine (SVC19) has the following functions for a VSAM cluster:

- Enter the remaining options ACB using data from the Job File Control Block (JFCB) that is originated from DD card and the z/OS Catalog. The order of precedence is as follows:
  - DD statement AMP parameters
  - ACB, EXLST, or GENCB parameters
  - Catalog entry for the cluster.

  For example, if both an ACB or GENCB macro and the DD statement have values for buffer space, the values in the DD statement override those in the macro.
- Create, in one specific ACB field, the address of the VSAM access method routine in charge of processing the options described in ACB. This pointer is used later when the task application code asks for reading and writing to or from the cluster.
- Construct the internal control blocks that VSAM needs to process your requests for access to the cluster.
- Construct the Data Extent Block (DEB), indicating to MVS that the cluster is opened.
- Check for consistency of updates to the prime index and data components when you open a key-sequenced data set (KSDS), an alternative index (IBM AIX®) component, or an AIX path. Open issues a warning message to indicate a time stamp discrepancy. The verify function can be started.
- Issue a RACROUTE for RACF authorization for the task program access.
- If the access is for sequential input, as in QSAM, Open does anticipatory buffering, that is, filling the buffer pool with records.

2.2.4 Close routine (SVC 20)

Because there are no modifications in DFSMS 1.13 affecting the Close component of DFSMSdfp, the Close functions are not detailed. The major function of the Close is to reverse the actions run by the Open routine.

2.2.5 EOV (SVC 31) routine

VSAM EOV routine is called when a VSAM cluster requires additional space. VSAM EOV does the following functions:

- Acquires new direct access storage device (DASD) extents for the cluster by inspecting the VTOC of the candidate volumes (the ones in the storage group of the data set).
- Updates the VSAM control block structure for the cluster with the new extent information.
- Updates the critical control block data in common storage so that this new DASD space is accessible by all program tasks that use the cluster. The tasks can be in the same or different address spaces.

If the occurrence of an abend or unexpected error prevents this space allocation from being completed, all address spaces program tasks are prevented from further extending the cluster. To obtain additional space, you must close the VSAM cluster in all address spaces, then reopen it.

2.2.6 Completion codes and reason codes

A code is a number that indicates how a data processing entity has finished its execution. z/OS has the following types of codes:

- Condition code (two-bit), which indicates how a processor instruction finished. It is in current program status word (PSW) and tested by the branch and condition instruction (BC).
- Return code (one-byte), which indicates how a load module, or a PDSE program object, finished. It is in general purpose register (GPR) 15 and is tested by the calling load module. A zero value means that there are no problems. This code is used by the COND parameter in JCL to decide the conditional execution of the next step. This decision depends on the return code of the last load module of the previous step.
- Completion code (two-byte), which indicates how a task finished, set by the ABEND macro. The code is at the TCB of the completed task, and is tested by the mother task.
- Reason code (one-byte), which indicates more detailed information about how a task finished. It completes the completion code in GPR 15. Because there are more errors than the possible number of completion codes (64K possibilities in a 2 byte field), different errors share the completion code. Different reason codes for the same completion code individualize the error. Reason codes are also called descriptor codes.

2.2.7 QSAM blocking

A logical record is a unit of information used by an application task program to store and retrieve data in a data set. An application task program sees only logical records that are its units of data being transferred. The application task program, through a GET or a READ, requests that a specific logical record be moved from the I/O device to a virtual storage application buffer. Through a PUT or a WRITE, the specific logical record is moved from the virtual storage application buffer to an I/O device. However, this application buffer is not the I/O buffer from where the I/O operation is run.

A physical record (or a block) is a set of logical records put together. Non-zHPF I/O architecture demands one channel command word (CCW) for each transferred physical record. The process of packaging logical records into one physical record is called blocking. The size of a physical record can be declared in the BLKSIZE parameter of a DD statement and stored by Open routine DCB/ACB. BLKSIZE can also be automatically determined by a storage management subsystem (SMS) routine.

QSAM is started by a PUT request that comes from the application task program. It moves the logical record from the application buffer, with a logical record size, to an I/O buffer, with physical record size, to construct a block. At GET time the unblocking is done by moving the logical record from the I/O buffer to the application buffer. Figure 2-1 on page 16 shows that five logical records (from R21 to R25) were moved to two I/O buffers in the buffer pool (BUFNO=4). No I/O operation was done to these logical records so far. The figure captures the moment where the sixth logical record (R26) is moved, completing the second I/O buffer.

For sequential access, it is convenient and efficient to do blocking, that is, to keep several logical records in just one physical record. Using blocking has the following advantages:

- Better utilization of the 3390 track because the larger the size of the block, the fewer gaps there are. To minimize gaps, you can have just one block per track (56 KB). However, the access methods support a maximum of 32 KB for DASD. The best BLKSIZEs for using the 3390 track are:
  - 27648 bytes (27 KB) with two blocks per track
  - 18432 bytes (18 KB) with three blocks per track and 97% utilization of the track.
- For sequential access, the larger the size of the block, the more efficient the I/O is because there is less total I/O connect time. This efficiency is also because you need fewer CCWs to move data.

The drawbacks of larger blocks are:

- Larger blocks demand larger buffers in virtual and real memory to keep the data. However, this greater demand is not usually a problem because of the available virtual storage above the 2 GB bar and adequate main storage in most customer installations.
- For random read access that requires just one specific logical record, many not-needed logical records are moved to and from memory during an I/O operation.

When using BSAM as an access method, the application code is in charge of blocking and unblocking.

2.2.8 QSAM buffering

Buffering is one of the key aspects of I/O performance. An I/O buffer is a virtual storage area where the physical record (or a CI for VSAM clusters) is transferred during an I/O operation. A buffer pool is a set of contiguous I/O buffers of the same size. QSAM takes advantage of having several buffers in the buffer pool by chaining them together into a single I/O operation with several Read or Write CCWs. Defining many sequential buffers decreases the number of I/O operations because QSAM is able to cluster many physical blocks into a single I/O operation. This configuration results in less total I/O connect time and less processor time for sequentially transferring your data set.

The number of buffers in the buffer pool is determined by the BUFNO parameter. This parameter can be determined statically by the installation or dynamically by QSAM. The installation participates in QSAM buffering only by indicating the number of buffers in the buffer pool. The decision about how many buffers are chained in one I/O operation is a QSAM internal decision that changes dynamically.

Figure 2-1 shows that QSAM decided to chain two I/O buffers that contain a physical record and three logical records each, in a single channel program. This is a single I/O operation. Buffering also allows QSAM to implement anticipatory buffering, also called a look-ahead algorithm. Anticipatory buffering provides logical records before that are requested by a reading application task program.

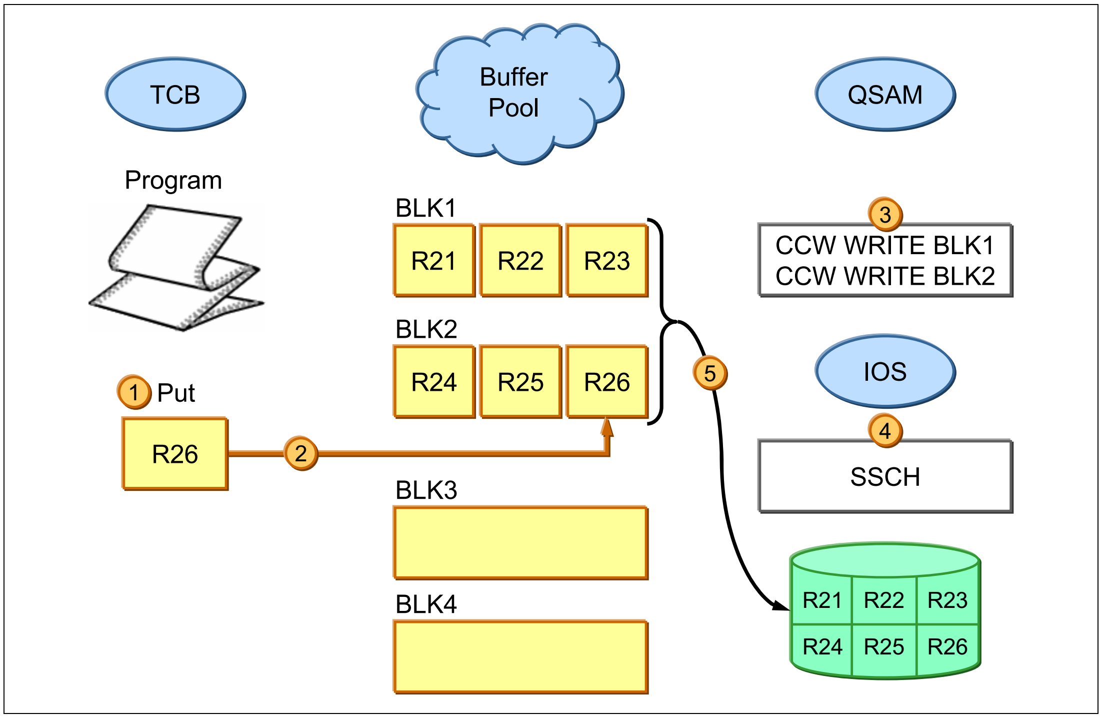


Figure 2-1 QSAM blocking and buffering

2.2.9 QSAM support for MULTSDN keyword

The following three BSAM parameters are set by the installation to allow BSAM dynamically derive the best value from BUFNO:

- MULTSDN in DCBE
- NCP in DCB
- MULTACC in DCB

How BSAM uses these parameters is not covered because this algorithm is not modified in DFSMS V1.13.

Ever since DFSMS V1.9, the MULTSDN keyword DCBE can be used in queued sequential access method (QSAM). It allows the access method to calculate a more efficient value for BUFNO. This function reduces the situations where the installation needs to specify a BUFNO value. Remember that when enough buffers are available for reading ahead or writing behind, QSAM attempts to read or write those buffers in the same I/O operation.

QSAM accepts a MULTSDN value for the following data sets, where it calculates an initial BUFNO:

- DASD data sets that are not extended format data sets. For these data sets, the initial BUFNO value is the number of BLKSIZE-length blocks that can fit on a 3390 track.
- Striped extended format data sets (not in the compressed format). For these data sets, the initial BUFNO value is the number of stripes multiplied by the number of BLKSIZE-length blocks, plus the 32-byte suffix, that can fit on a track.
- Tape data sets with a blocksize less than 32 KB. For these data sets, the initial BUFNO value is the number of BLKSIZE-length blocks that can fit within 64 KB.
- Tape data sets with a blocksize equal to or greater than 32 KB (larger block format). For these data sets, the initial BUFNO value is 2.
- PDS data sets.

For the supported types of data sets, QSAM uses MULTSDN as a multiplier value to calculate a more efficient value for BUFNO. The value is based on the initial BUFNO value when the following conditions are true:

- MULTSDN value is not zero.
- DCBBUFNO has a value of zero after completion of the DCB OPEN exit routine, meaning that this option was not defined by the programmer.
- Data set block size is available.

Table 2-2 shows how the option MULTSDN affects the value of BUFNO for a data set accessed by QSAM. If your installation has enough central and virtual storage to define a QSAM DCBE of sufficient size, the MULTSDN value should be around 10.

Table 2-2 MULTSDN option effects

Data Set Type	DCBBUFNO default without MULTSDN	DCBBUFNO default with MULTSDN
PDSE Member	1	1
Extended format data set in the compressed format	1	1
UNIX file	1	1
Extended format data set (not in the compressed format)	2 * number of stripes * number of blocks per track	MULTSDN * number of stripes * number of blocks per track
Block size equal to or greater than 32 KB (tape)	2	MULTSDN value
Block size less than 32 KB (tape)	5	MULTSDN * number of blocks in 64 KB
IBM 2540 card reader or card punch	3	3
PS, PDS	5	MULTSDN * number of blocks per track
Others, including dummy data sets	5	5
TSO terminal	5	1

The MULTSDN works like a multiplier when enlarging the BUFNO figure. A typical BUFNO value is 5. If a DCB is changed, the program needs to be recompiled.

2.2.10 DEVSUP PARMLIB member

DEVSUPxx is a member of the PARMLIB system data set. Most of its installation options apply to the IOS z/OS component. DEVSUPxx specifies the installation default for I/O device support options, and is processed during the nucleus initialization program (NIP) phase of initial program load (IPL). After IPL, you can use system command `SET DEVSUP=XX` to activate the DEVSUP changes. One of the DEVSUSPxx parameters addressed is NON_VSAM_XTIOT: YES/NO.

2.2.11 Task Input/Output Table (TIOT)

Allocating a data set to a task allows the programs of the task to run I/O operations on that data set. Allocation is not the same as creating a data set, although for DISP=NEW data sets, allocation includes a data set creation (a DSCB F1 is built at VTOC). The allocation routine can be started by an Initiator task for each DD statement present in the Job JCL. It can also be started explicitly by a task program through the Dynalloc macro.

Originally, the allocation routine keeps information about data sets already allocated to a task in a control block named Task Input/Output Table (TIOT). This control block is kept in the system queue area (SQA) below the 16 MB line. Each task TCB points to its own TIOT.

Each TIOT entry describes one data set allocated to the TIOT owner task. The information in the TIOT allows IOS, a z/OS component, to locate the device by the UCB address. The UCB address contains the data set target of the I/O operation. The link between the logical properties described in the DCB/ACB and the physical information in each TIOT entry is done through the data set DDname. TIOT has the following major entry fields:

- TIOEDDNM has the data set DDname.
- TIOEJFCB has the pointer for JFCB (which has the DCB/ACB options on a DD statement in JCL) in the scheduler work area (SWA).
- TIOEFSRT has the address of the UCB representing the device that contains the data set.

A PARMLIB member ALLOCxx has the SIZE(nn) parameter, where the installation can define the TIOT size. This size can be from 16 KB to 64 KB (default is 32 KB). The only reason for not defining 64 KB is the possibility of running out of SQA virtual space below the 16 MB line. The number of TIOT entries limits the number of data sets a task can allocate by Dynalloc plus the ones allocated by DD statements at JCL. The number of TIOT entries depends on these factors:

- Size of a TIOT entry, which varies with the number of volumes that contain the data set. Every SMS candidate volume assigned increases the size by 4 bytes. Also, the TIOT prefix, header, and trailer use 60 bytes of the total TIOT space available.
- Size of the TIOT.

TIOT entries have these limitations:

- Mono-volume data sets and a TIOT with 64 KB have a limit of 3273 data sets per task.
- For data sets with the maximum number of volumes (59) and a TIOT with 64 KB, the limit is259 data sets per task.

Figure 2-2 shows a TIOT formatted in a storage memory dump. This TIOT has five data sets, each of which is identified by a specific DDNAME.

```
TIOT:   009B5000
           JOB......  J906SMF    STEP..... G
                  LN-STA      DDNAME      TTR-STC       STB-UCB
           +0018  14010100    SYSLIN     00014F00   809D6160
           +002C  14010100               00016F00   80000000
           +0040  14010102    SYSLOUT    00018F00   80000000
           +0054  14010102    SYSUDUMP   0001AF00   80000000
           +0068  14020100    SMF        0001CF00   949C56C0
           +007C  14010102    SAIDA      0001EF00   80000000
```

Figure 2-2 TIOT in a memory dump

2.2.12 Extended Input/Output Table (XTIOT)

In prior DFSMS releases, jobs and STCs such as SORT, SMF (IFASMFDP), and IBM DB2® often failed because they have more DD statements than can fit a job step task TIOT.

The solution is the XTIOT, which is an extension TIOT in ESQA above the 16 MB line. There is no space limit for XTIOT, and originally it could be used only by data sets allocated through the Dynalloc macro function. Example 2-1 shows the bit S99TIOEX on the Dynalloc macro parameter list. When this parameter is on, TIOT entries associated with the data set allocated through this Dynalloc macro are stored in an XTIOT entry.

Example 2-1 A bit in the Dynalloc macro parameter list

```
S99TIOEX
Build XTIOT entry. Note: The XTIOT is a non-contiguous TIOT entry that is not
accessible through the contiguous TIOT.
```

Examples of using XTIOT entries and therefore allocating data sets using Dynalloc for data sets are the subsystems DB2, CICS, and IMS/DB. Originally, all the data sets introduced through DD statements in a job step must use a TIOT entry. Beginning with VSAM and EXCP access methods in early releases of DFSMS, this limitation of Dynalloc macro for XTIOT was removed.

In DFSMS V1.12, non-VSAM access methods such as BSAM, BPAM, QSAM, and the Open/Close/EOV routines were enhanced to support the following functions:

- XTIOT
- Nocapture UCB

  Capture UCBs, as defined in an HCD, are located above the 16 MB line. However, through the implementation of the IBM z/Architecture® concept of shared pages, they can be accessed in z/OS by AMODE 24 programs. The nocapture UCBs are above the 16 MB line, but are not mapped or accessible below the line.
- DSAB-above-the-line options of dynamic allocation

Previously, VSAM and EXCP were the only access methods that supported these dynamic allocation options. In DFSMS V1.12 the EXCP support is enhanced, and BSAM, BPAM, and QSAM support is provided. This new support applies to dynamic allocation of DASD, tape, and dummy data sets, and cases where PATH= is coded. EXCP support includes the EXCPVR and XDAP, a type of EXCP that does not require writing channel programs. Macros are also affected.

These enhancements provide virtual storage constraint relief, such as UCB nocapture and DSAB above-the-line, especially in the areas of DASD and tape support. They also allow you to have more than 3273 mono-volume dynamically allocated data sets. To use these enhancements, the programs named in the following section must be changed. In addition, installations using them must have the DEVSUPxx PARMLIB option NON_VSAM_XTIOT: YES/NO set to YES. The default is NO.

To enable this DFSMS V1.12 enhancement, specify the LOC={ANY|BELOW} option before issuing the OPEN or RDJFCB macro. If you specify LOC=BELOW or do not code LOC=, the program does not support the XTIOT, UCB nocapture, or DSAB-above-the-line options of dynamic allocation.

By setting LOC=ANY, you indicate that the program is either not affected by or that it allows for any of the following possibilities:

- The DCBTIOT field, offset in TIOT to an entry, can contain zeros or contain a TIOT offset.
- The DEBXDSAB field, address of DSAB, can point above the line.
- The DSABTIOT field can point to an XTIOT or to a TIOT entry.
- The UCB address field in the DEB can be 4 bytes or 3 bytes (test the DEB31UCB bit).
- The TIOEFSRT field can contain zeros instead of a UCB address.

Regardless of dynamic allocation and the NON_VSAM_XTIOT setting, always specify DCBE LOC=ANY when your program does not reference XTIOT, UCB, and DSAB. Also specify this setting when your program is correctly modified to support the XTIOT, UCB nocapture, or DSAB-above-the-line options.

2.2.13 Multivolume tape data set serialization

A multivolume tape data set is a data set in several tape volumes. At end of each volume sequential processing, the EOV routine gains control. It then provides the mounting of the next volume (for read or write access) in the same tape drive or in another tape drive. This EOV processing is not apparent to the application.

A tape data set and a DASD data set are serialized by the Initiator/Allocation routine through the issue of ENQ SYSDSN.*datsetname*. However, a key difference between these two data sets types is that the DASD volumes are shareable by nature, but a tape volume has an exclusive access type. Therefore, just one task at a time can access a data set in a tape volume. Shareable, conversely, means that several tasks can concurrently access the same or different data sets in the same volume.

To avoid mounting the same tape volume by different tasks in the same z/OS, or in other z/OS systems but in the same sysplex, the Initiator/Allocation routine serializes the volume. This serialization is done by issuing an exclusive ENQ SYSZVOLS.VOLSER command for each volume. For a multivolume tape data set, this ENQ is issued at data set allocation time to all the volumes that contain the data set.

In DFSMS releases before V1.13, the DEQ SYSZVOLS.VOLSER command is run at the end of processing of the last tape volume by the DEQ at Demount Facility for all tape volumes. Because of this, different tasks cannot process, with some limitations, different volumes of the same multivolume data set in parallel. For more information, see 2.2.14, “DEQ at Demount facility” on page 21 for details.

Figure 2-3 shows an IBM RMF™ Monitor III Enqueue Delay Report that shows contention between tasks for tape volumes. During the 60 second RMF range, the tape volume P35641 was being used by a task in the address space BDTBP01 for 70% of that range. EO under STAT means that it was an exclusive owner. As a consequence, another task in the address space D3P5PRD1 was delayed (EW under STAT means exclusive wait) for the same 70% of the range. This resource is identified by the major name SYSZVOLS and the minor name P35641.

```
                RMF V1R2 ENQ Resource Delays
Command →                                      Scroll → CSR

Samples: 60   System: PROD Date: 02/09/11 Time: 16.40.00 Range: 60  Sec

----------Resource Name---------   ---Delayed---            ---Holding---
Major/Minor   (Scope)          %  Name      STAT     % Name/SYS STAT

SYSZVOLS   (SYSS)           70 D3P5PRD1 BW      70 BTDBP01   EO
P35641
```

Figure 2-3 RMF Monitor III enqueue delay report

2.2.14 DEQ at Demount facility

The DEQ freeing the tape volumes in a multivolume tape data set is done at the end of the last tape volume processing. The different tasks in the same or other z/OS systems cannot concurrently read different tape volumes in the same multivolume tape data set. This limitation is caused by these factors:

- ENQ is still up
- Volume is not unloaded after processing end, or sometimes just rewound.

Releases before DFSMS V1.13 included a function called DEQ at Demount facility that circumvented this problem. The DEQ of the volume was run at data set close time. However some aspects of this function were not useful, namely:

- It requires APF authorization for the running task
- Changes at application code are required
- Depending on the disposition, the volume is rewound and not unloaded.

A better solution in DFSMS V1.13 is addressed in 2.3.5, “FREEVOL=EOV JCL parameter” on page 25.

2.2.15 ISO/ANSI V4 tape labels

ISO/ANSI V4 tape labels are a standard for tape labels based on ANSI X3.27-1987 level 4 and ISO 1001-1986(E) level 4. ISO/ANSI labels are similar to IBM labels. ISO/ANSI labels and IBM standard labels have the following principal differences:

- ISO/ANSI labels are written in ASCII characters. IBM standard labels are written in EBCDIC.
- IBM standard labels are 80 bytes long. ISO/ANSI labels have a length of 80 bytes or more.
- IBM labels have a maximum of nine user volume labels can be in the beginning-of-volume group. Version 4 labels can also have VOL2–VOL9 after VOL1 label in the beginning-of-volume group. An unlimited number of ISO/ANSI user labels can be placed

at the beginning and end of a file. These labels do not have to be sequentially numbered or lettered.

- The formats of the ISO/ANSI labels VOL1, HDR2, EOF2, and EOV2 are slightly different from the formats of the corresponding IBM labels.
- A maximum of nine user EOF or EOV labels can be in the file section label group.

The different types of ANSI/ISO tape labels are shown in Table 2-3.

Table 2-3 Types of tape labels

Label Identifier	Label Definition
VOL1 - VOL9	Volume label set (optional: VOL2 - VOL9 not produced by z/OS)
UVL1 - UVL9	User volume labels (optional: *Not* produced by z/OS)
HDR1	Data set header label 1
HDR2	Data set header label 2 (produced by z/OS, but optional for input)
HDR3 - HDR9	Optional (not produced by z/OS)
UHLa	User header labels (optional: Unlimited number allowed)
EOV1	End-of-volume trailer label 1 (produced by z/OS, but optional for input)
EOV2	End-of-volume trailer label 2 (produced by z/OS, but optional for input)
EOV3 - EOV9	Optional (not produced by z/OS)
EOF1	End of data set trailer label 1 (produced by z/OS, but optional for input)
EOF2	End of data set trailer label 2 (produced by z/OS, but optional for input)
EOF3 - EOF9	Optional (not produced by z/OS)
UTLa	User trailer labels (optional: Unlimited number allowed)

2.3 I/O support enhancements at DFSMS V1.13

This section provides a detailed explanation of the DFSMS V1.13 enhancements related to the I/O support routines Open/Close/EOV.

2.3.1 O/C/EOV text with abend console messages

This new DSMS V1.13 function address reliability, availability, and serviceability (RAS) disciplines. Normally, O/C/EOV routines detect hundreds of error conditions. The routines then issue the ABEND macro to pass control to the recovery termination manager (RTM). The RTM, a z/OS component, abnormally terminates the O/C/EOV calling task, and an abend message is issued containing a numeric abend code and reason code.

Description

A new installation option is available by using DEVSUPxx through the keyword OCE_ABEND_DESCRIP. It appends a descriptive text for messages associated with the more commonly experienced abend and return codes of the I/O support routines. This new referred keyword is:

```
OCE_ABEND_DESCRIP = YES | NO
```

This enhancement improves RAS because it minimizes the time needed to do problem determination. You no longer need to refer to the message manuals or the LOOKAT website to interpret the O/C/EOV abend completion and reason codes for those messages.

Example 2-2 shows a sample descriptive text that is appended to the determinant abend messages associated with a subset of O/C/EOV abends. In the example, the device type does not support the recording mode requested by the calling program.

Example 2-2 Sample descriptive text during abend

```
IEC145I 413-40,IFG0194F,RDASL1,RDSL1,SYSUT1,0920,,DATASETX
ERROR DESCRIPTION:
THE DEVICE DOES NOT SUPPORT THE RECORDING MODE REQUESTED BY THE USER OR DETERMINED
BY THE SYSTEM.
END ERROR DESCRIPTION: IEC145I
```

Compatibility and coexistence

The default for OCE_ABEND_DESCRIP is NO, to maintain compatibility of automation products that manage the console. You need to investigate your product rules to make sure that they are not affected by this change in the message text. Usually there is no compatibility issue because the automation rules are connected to messages IDs, not to the text of the messages.

2.3.2 MULTSDN changes for QSAM concatenation

This DFSMS V1.13 enhancement is applied to the MULTSDN parameter in the QSAM access method. This new function increases availability by improving the way that BUFNO is calculated by QSAM for concatenated data sets.

Description

QSAM uses the MULTSDN value in the DCBE macro to calculate a better performance BUFNO value for tape and specific types of DASD data sets. However, in releases before DFSMS V1.13, if data sets are concatenated, the BUFNO value is calculated based on the first data set in the concatenation. When EOV provides the next concatenated data set, there might be a lack of virtual storage abend because EOV might getmain a large amount of virtual storage. The area that must be getmained might be large because its size is the size of the BUFNO multiplied by the BLKSIZE. The size of the block of the next data set is usually much larger than the current data set from where the BUFNO was derived.

The solution in DFSMS V1.13 is that QSAM can dynamically recalculate the BUFNO value when switching to the next concatenated data set. This recalculation applies only when the installation is processing a data set with QSAM, where the MULTSDN option is specified at DCBE. For concatenated data sets, if MULTSDN is specified, QSAM dynamically recalculates the BUFNO value when switching from one data set to another in the concatenation.

Compatibility and Coexistence

There are no compatibility and coexistence issues because the installation is already using the MULTSDN for QSAM.

2.3.3 O/C/EOV subsystem DCBs with XTIOTs

This enhancement improves z/OS serviceability. Using XTIOT entries instead of TIOT entries for each task allocated data sets allows more data sets to be allocated per task.

Description

All the O/C/EOV access methods support the use of XTIOT entries as addressed in 2.2.12, "Extended Input/Output Table (XTIOT)" on page 19. This configuration increases the limit of 3273 mono volume allocated data sets to a task. To use these enhancements, set the DEVSUPxx PARMLIB option NON_VSAM_XTIOT to YES during installation.

XTIOT support for BAM DCBs was included in z/OS V1.12. However, although VSAM clusters were supported before release, that support did not include subsystem DCBs or non-VSAM access methods. These are also known as basic access methods (BAM), and include including QSAM, BSAM, BDAM, BPAM, and EXCP.

In the enhancement in DFSMS V1.13, O/C/EOV accepts Subsystem DCB/ACBs associated XTIOTs. Subsystem DCB/ACBs are a SYSIN/SYSOUT DCB/ACB combination. The application passes a subsystem DCB/ACB to a common Open routine, and the OPEN executors build a subsystem DCB/ACB and issue OPEN TYPE=J. To associate DCB/ACB with XTIOTs, the following settings are required:

- DEVSUP parameter NON_VSAM_XTIOT=YES must be set.
- The subsystem must support XTIOT, DSAB above the 16 MB line, and nocapture UCB.
- DCBE with LOC=ANY option must be set.
  - If DCBE with LOC=ANY is set but NON_VSAM_XTIOT=NO, and ABEND is issued with completion code 113 and descriptor code 4C.
  - If there is no DCBE or if there a DCBE but with LOC=BELOW, then Open fails with return code 8.

Subsystem ACBs associated XTIOTs require these settings:

- The subsystem supports XTIOT, DSAB above the 16 MB line, and nocapture UCB.
- DEVSUP parameter NON_VSAM_XTIOT=YES must be set. If it is not set, ABEND113-4C is issued.

Compatibility and Coexistence

There are no compatibility and coexistence issues associated with this enhancement.

2.3.4 Recovering multivolume tape data sets

This enhancement improves availability of z/OS systems. This line item covers the situation when the following events are detected at tape volume mounting time in a tape multivolume O/C/EOV data set processing:

- An out of sequence tape volume
- A missing volume.

In DFSMS V1.11, the EOV routine calls the LABAN exit routine. LABAN provides an ignore (default) or an abend option for the message conditions associated with multivolume tape data set as shown in Example 2-3.

Example 2-3 Messages showing error conditions

```
Tape volumes processed out of order:
IEC709I…. EXPECTED VOLSEQ: nnnn, FOUND: nnnn
 Missing last volume:
IEC710I… ANOTHER VOLUME EXPECTED
 Missing last volume when reading backward:
IEC711I…..RDBACK-NOT LAST VOLUME OF DATA SET
 Missing first volume:
IEC712I…READ-NOT FIRST VOLUME OF DATA SET CORRECTED is issued.
```

The problem is that there is no recovery option for the LABAN exit routine. The solution in DFSMS V1.13 is that when O/C/EOV detects the anomaly, it passes control to the LABAN exit. The LABAN exit then returns the recovery option to be exercised by reliable multicast messaging (RMM). It can provide the missing or correct next volume to process.

The O/C/EOV processes an RMM volume list and runs the following tasks:

- Ensures that all volumes are enqueued.
- Adds all volumes to the JFCB and JFCB extensions that ensure no duplicate volume serial numbers are introduced.
- Dynamically obtains any required JFCB extensions.
- Issues a demount for the incorrect volume followed by a mount for the correct next volume.
- Issues new message: IEC716I ddnamexx: TAPE MULTIVOLUME LIST CORRECTED.
- Processes the rest of the volumes in the updated JFCB volume list.

2.3.5 FREEVOL=EOV JCL parameter

This enhancement improves performance for tape processing and increases the possibility of parallelism.

Description

FREEVOL=EOV is DD statement parameter introduced in DFSMS V1.13. It allows different tasks that run in the same z/OS, or in other z/OS systems but in the same sysplex, to concurrently read multivolume tape data sets. In other words, this facility issues the DEQ SYSZVOLS at each invocation of the EOV routine for each tape volume already read. It also forces a demount of the volume instead of a rewind.

This function is similar to the DEQ at Demount Facility (see 2.2.14, “DEQ at Demount facility” on page 21), but has the following advantages:

- It does not require APF authorization. Also, because it is implemented in the JCL, no changes to the application are required. It is accepted only for input processing
- EOV and CLOSE volume disposition processing unload the volume when the disposition would otherwise be rewound.
- For an input-specific mount request and when FREEVOL=EOV is requested, OPEN and EOV I issue an abend. this process occurs if a task using the same JCL DD statement attempts to reprocess a previously DEQueued volume serial number.

Compatibility and Coexistence

The implementation of parallelism when reading a multivolume tape data set might change the flow of your batch window execution. You need to evaluate this effect to ensure that there are no new bottlenecks that can impair the time gain.

2.3.6 New O/C/EOV diagnostic data added to SMF 14/15

This enhancement improves the DFSMS serviceability. Before DFSMS V1.13, the OPEN routine verified and validated the DCBE parameters by checking them against the DCB. If an error was found, the DCBE was invalidated.

One error could be a declared data set organization of PS, DA, or PO, which cannot be an ACB. However, the OPEN processing continued as though no DCBE was provided.

Description

DSMS V1.13 support lists the DCBE discrepancy in the SMF14/15 record. New O/C/EOV diagnostic data is added to SMF 14/15 Type records Extended Information Segment Type 8. This data provides reasons when a DCBE is invalidated.

The SMF 14/15 is also used to log partial release errors. This support lists the reason why Close did not call partial release. Table 2-4 shows DCBE and partial release errors described at SMF 14/15.

Table 2-4 SMF 14/15 list of errors

SMF14RAS	ECU *		EXTENDED INFO SEGMENT TYPE 8
SMF14DCBEEXCP	EQU	X'80'	DCBE INVALIDATED BECAUSE EXCP AND NO FOUNDATION EXTENTION PRESENT
SMF14DCBEDSORG	EQU	X'40'	DCBE INVALIDATED BECAUSE DSORG IS NOT PS, PO, OR DA
SMF14DCBEFREE	EQU	X'20'	DCBE INVALIDATED BECAUSE STORAGE IS NOT ADDRESSABLE
SMF14DCBEKEY	EQU	X'10'	DCBE INVALIDATED BECAUSE DCBE STORAGE IS NOT IN KEY OF CALLER
SMF14DCBEID	EQU	X'08'	DCBE INVALIDATED BECAUSE THE DCBEID IS NOT 'DCBE'
SMF14DCBEMIN	EQU	X'04'	DCBE INVALIDATED BECAUSE IT IS NOT AT LEAST THE MINIMUM LENGTH REQUIRED (56 BYTES)
SMF14NODCBE	EQU	X'02'	DCBEHIARC FLAGS SET BUT DCBDCBE IS ZEROS

Compatibility and Coexistence

To get the benefit of this enhancement, have the SMF reduction routine take the new error information provided at SMF 14/15 into consideration.

2.3.7 Improve ISO/ANSI V4 tape label processing

DFSMS V1.13 improves z/OS systems performance when processing tape label types.

Description

O/C/EOV pre-reads the load point, data set header (HDRx), or data set trailer label (EOFx) data into UCB Tape Class extension. See Table 2-3 on page 22 for the location of such labels in a tape volume. After that O/C/EOV intercepts by using an EXCP SIO appendage exit positioning and reads channel programs that are targeted within the pre-read label structure. Performance is improved because the I/O operation is not run and because the tape is physically moved only in a forward direction during OPEN label processing.

This idea can improve stacking data sets performance. Stacking data sets on a tape means creating multiple data sets, and leaving the tape positioned at the end of the data set. This process leaves the tape ready to write the next data set. To improve performance during CLOSE, the trailer labels about the created data set are saved in the UCB Tape Class extension. Then, during OPEN processing for the next data set, the system reads the previous data set's trailer labels. This saved information is used to create pseudo trailer labels, eliminating the need for I/O to reposition the tape and physically read the labels.

Due to restrictions associated with ANSI/ASCII Version 3 standards, this pseudo label build is not done. This function requires no application changes and there are no externals for this function.

Compatibility and Coexistence

There are no compatibility and coexistence issues associated with this enhancement.

2.3.8 Saved RACF return and reason codes and parameter list

This enhancement improves the RAS of your z/OS systems. In this process, O/C/EOV starts RACF for data set and tape volume permission. On return from a call made to RACF using RACROUTE from O/C/EOV, if the return or reason code is nonzero, these values are saved together with the RACROUTE parameter list. This process makes those values available in a memory dump associated with the RACF failure during O/C/EOV processing.

3

DSS enhancements

This chapter describes the DSS features introduced with z/OS V1.13. One enhancement allows DFSMSdss (also called DSS) to trigger a mechanism that notifies other eligible members in a sysplex. This notification can be about a change to a DASD volume serial number (VOLSER) or the volume table of contents (VTOC) address. It can also be about an incompatibility of the VTOC Index (VTOCIX). This is called the DSS cross-system (XSYS) enhancement. If DSS changes volume information while running in a member of a sysplex, you must update the UCB information in the other members of a sysplex.

Before this enhancement, you had to vary an affected volume offline before making a change. In this case, the UCB was refreshed when the volume was varied back online. If the change was made without varying the volume offline, you had to vary the volume offline, then online after the change in any affected member of the sysplex.

DSS achieves this by generating an event to the Event Notification Function (ENF64). This event is recognized by the DEVMAN task if a member of the sysplex (on z/OS V1.13) has enabled the DEVMAN function. If DEVMAN is enabled to respond to ENF64 to refresh the UCB, DEVMAN will issues the following command. This command is issued in all affected members of the sysplex where necessary and applicable.

```
VARY ddd,ONLINE,UNCOND
```

On systems that use a level of z/OS before V1.13, if the DEVMAN REFUCB function is not enabled, ENF64 is ignored. The UCB refresh must then be done manually.

This chapter contains the following sections:

- DFSMSdss cross system sysplex member notification
- DSS user changes required
- Dynamic Volume Expansion and copy services

3.1 DFSMSdss cross system sysplex member notification

DSS has been enhanced to trigger various processes by building an ENF64 event when it changes certain VTOC information about a target volume. The ENF64 event contains information such as a new volume serial or a change in the VTOC location.

3.2 DSS user changes required

DSS users do not have to make any changes in their jobs to use this new function because it is started automatically. If the function has not been enabled in a particular member of the sysplex, the ENF64 event notification is ignored without error. ENF64 is not generated if the VOLSER or the VTOC location does not change. However it might be issued if the VTOCIX becomes invalid (if the target volume is indexed).

3.2.1 DSS operations that trigger the XSYS function

The following DSS operations can trigger this function:

- Full volume copy and RESTORE processing. If COPYVOLID is specified, the target VOLSER is changed and an ENF64 event created.
- COPY FULL with DUMPCONDITIONING does not cause the VOLSER to change. However the VTOC might be in a different position, or the size of the VTOC might be different. If either of these conditions occur, an ENF64 event is generated.
- Implicit DSS change. If the target volume of a DSS operation is larger than the source volume, the free space information in the VTOCIX can become incorrect. In this case, DSS starts ICKDSF, and ICKDSF issues the ENF64. For more information, see 3.2.6, “Use of the ENF64 event by other functions” on page 35.
- DFSMShsm functions that use DSS as the data mover implicitly take advantage of this XSYS enhancement. This process occurs if the DSS functions starts result in one or more of these triggers.

3.2.2 DSS XSYS enhancement system compatibility and coexistence

The new DSS function is part of z/OS DFSMS V1.13, together with the related functions required to support it. No compatibility update is required on prior level systems because they ignore any ENF64 event received.

3.2.3 DSS cross system enablement

This enhancement notifies other members of a Sysplex about a change in the VOLSER or the VTOC address on a volume shared across members of a Sysplex. You must enable the automatic notification function in all members of a Sysplex where automatic UCB refreshes are required.

> **Consideration:** You do not need to implement these changes on the volume that runs DSS for ENF64 event generation if it is at the z/OS V1.13 level. However, at some stage any one of the members of the Sysplex can become a target member. Therefore, set all members of the Sysplex up to recognize ENF64 and enable DEVMAN to issue the REFUCB update.

The default is for the new function not to be enabled so that the implementation of the new function can be managed by the user.

The function is implemented through the DEVMAN task, which when enabled issues the necessary command to refresh the unit control block (UCB) information for the affected volume. DEVMAN can be enabled to carry out the REFUCB function dynamically. You can also use the PARMLIB DEVSUPxx member to cause DEVMAN to start at initial program load (IPL) with the REFUCB option enabled.

DEVMAN for REFUCB support initialization at IPL

To enable the DEVMAN function, update member DEVSUPxx of SYS1.PARMLIB to add the ENABLE(REFUCB) option.

> **Attention:** When updating the DEVSUPxx member, be sure to manage the commas properly. Add ENABLE(REFUCB) to the list of options without also adding a comma to the end of the previous option. If you add a comma, the ENABLE(REFUCB) option is ignored without any indication of error.

Example 3-1 shows the member DEVSUPAF, which has been updated to add ENABLE(REFUCB).

Example 3-1 DEVSUPAF member showing ENABLE(REFUCB)

```
COMPACT = YES,                  /* INSTALLATION DEFAULT FOR IDRC */
MEDIA1 = 0021,
MEDIA2 = 0022,
MEDIA3 = 0023,
MEDIA4 = 0024,
MEDIA5 = 0025,
MEDIA6 = 0026,
MEDIA7 = 0027,
MEDIA8 = 0028,
MEDIA9 = 0029,
MEDIA10 = 002A,
ERROR = 002E,
PRIVATE = 002F,
TAPEAUTHDSN = YES,              /* NEW 1.8 TAPE SECURITY NORBERT */
TAPEAUTHF1 = YES,               /* NEW 1.8 TAPE SECURITY NORBERT */
TAPEAUTHRC4 = ALLOW,            /* NEW 1.8 TAPE SECURITY NORBERT */
TAPEAUTHRC8 = WARN,             /* NEW 1.8 TAPE SECURITY NORBERT */
ENABLE(REFUCB)
```

The updates can also be implemented by issuing the SET (or T) command as shown in Example 3-2.

Example 3-2 Command to set DEVSUP from DEVSUPAF PARMLIB member

```
T DEVSUP=AF
```

The result of issuing the **T DEVSUP=AF** command is shown in Figure 3-1. If the message IEA253I DEVSUP REFUCB FUNCTION IS ENABLED is not displayed, the DEVSUPAF member has probably not been constructed properly.

```
T DEVSUP=AF
IEE252I MEMBER DEVSUPAF FOUND IN SYS1.PARMLIB
IEA253I DEVSUP  REFUCB  FUNCTION IS ENABLED
IEA253I DEVSUP  3480X RECORDING MODE DEFAULT IS COMPACTION.
IEE536I DEVSUP   VALUE AF NOW IN EFFECT
IEA253I DEVSUP  ISO/ANSI TAPE LABEL VERSION DEFAULT IS V3
IEA253I DEVSUP  TAPE OUTPUT DEFAULT BLOCK SIZE LIMIT IS 32760
IEA253I DEVSUP  COPYSDB DEFAULT IS INPUT
IEA253I DEVSUP  STORAGE LIMIT FOR TAPE DDR SWAP DEFAULTED TO 1000M
IEA253I DEVSUP  TAPEAUTHDSN: YES
IEA253I DEVSUP  TAPEAUTHF1: YES
IEA253I DEVSUP  TAPEAUTHRC4: ALLOW
IEA253I DEVSUP  TAPEAUTHRC8: WARN
IEA253I DEVSUP  PERFORM NORMAL EXPIRATION DATE PROCESSING
```

Figure 3-1 ST DEVSUP=AF result

DEVMAN for REFUCB support initialization after IPL

The DEVMAN task can be enabled for REFUCB support after IPL. If it is enabled, DEVMAN starts during the IPL process. However, if the ENABLE(REFUCB) option is not present at IPL time, REFUCB support is not enabled.

To enable DEVMAN REFUCB support after IPL, issue the MODIFY(F) DEVMAN,ENABLE(REFUCB) command as shown in Example 3-3.

Example 3-3 DEVMAN command to enable REFUCB

```
F DEVMAN,ENABLE(REFUCB)
```

The result of issuing the modify DEVMAN command is shown in Figure 3-2.

```
F DEVMAN,ENABLE(REFUCB)
DMO0012I DEVICE MANAGER REFUCB ENABLED
```

Figure 3-2 DEVMAN MODIFY for REFUCB result

3.2.4 DSS XSYS enhancement example

If the DSS functions listed in 3.2.1, “DSS operations that trigger the XSYS function” on page 30 are used, the XSYS refreshes the UCB on applicable members of the Sysplex. This scenario shows the effect of using DSS volume RESTORE with COPYVOLID. The scenario has the following characteristics:

- A DSS FULL volume backup exists on volume MHL1A1 in data set MHLRES1.DSSDUMP.MHL1A1.
- Volume MHLTA1 is online in one member of the Sysplex (SC64) and at least one member of the Sysplex (SC70).
- One other member of the Sysplex (SC70) has the DEVMAN REFUCB function enabled.
- The objective is to run a job in Sysplex member (SC64) to RESTORE from data set MHLRES1.DSSDUMP.MHL1A1 to volume MHLTA1. The updated volume is then made available in the member of the Sysplex that the RESTORE job is run in (SC64). It also

becomes online with its UCB updated in another member of the Sysplex (SC70) automatically.

Tip: No additional action is required to refresh the UCB in member SC64 because that happens as a result of the DSS job completion

The job control language (JCL) shown in Figure 3-3 was run. The COPYVOLID statement causes the VOLSER of volume MHL**T**A1 to be changed to the volume referenced in data set MHLRES1.DSSDUMP.MHL1A1, which is MHL**1**A1.

```
//MHLRES2D JOB 99990000,UAALFO,CLASS=A,NOTIFY=&SYSUID
/*JOBPARM S=*
//*
//REST EXEC PGM=ADRDSSU,REGION=0M
//SYSPRINT DD SYSOUT=A
//DASD     DD UNIT=3390,DISP=OLD,VOL=SER=MHLTA1
//TAPE     DD DISP=(OLD,KEEP),
// DSN=MHLRES1.DSSDUMP.MHL1A1
//SYSIN    DD *
  RESTORE INDD(TAPE) OUTDD(DASD) COPYVOLID PURGE
/*
```

Figure 3-3   DSS RSTORE job to volume MHLTA1

When the job ran, it restores the data and changes the VOLSER to MHL1A1. Because the VOLSER was changed, it also generates an ENF64 event that is sent to all members in the Sysplex. In this test, only Sysplex member SC70 had the DEVMAN REFUCB function enabled.

The messages as result of the DSS RESTORE job are shown in Figure 3-4.

Remember: There is no indication in the DSS job output that the ENF64 event was created.

```
IEF403I MHLRES2D - STARTED - TIME=03.10.03 - ASID=009B - SC64
034 ADR369D AUTHORIZE FOR WRITE ACCESS A VTOCIX DATA SET ON MHLTA1, MHLRES2D, REST    ,
REPLY U OR T
R 34,U
IEC604I VTOC CONVERT ROUTINE ENTERED ON 8106,MHLTA1,DOS,DEVMAN
ICK502I   BUILDIX FUNCTION STARTED
ICK503I   8106 REQUEST RECEIVED TO CONVERT VTOC TO IXFORMAT
ICK504I   8106 VTOC FORMAT IS CURRENTLY OSFORMAT, REQUEST ACCEPTED
ICK513I   8106  BUILDIX PROCESSING COMPLETED:  VTOC IS NOW IN IXFORMAT
ADR320I (001)-SBRTN(01), VOLUME SERIAL MHLTA1 ON UNIT 8106 IS CHANGED  830
TO MHL1A1
```

Figure 3-4   DSS RESTORE job output

The output shown in Figure 3-4 is from the SYSLOG. It shows activity from Sysplex members SC64 and SC70. Member SC64 is where the DSS RESTORE job ran. SC70 is the member that picked up the event notification ENF64, and shows the activity that resulted.

Note the following activity:

- In SC64, the message from DSS (ADR320I) indicating that the VOLSER MHL**T**A1 has changed to MHL**1**A1:

  ADR320I (001)-SBRTN(01), VOLUME SERIAL **MHLTA1** ON UNIT 8106 IS CHANGED TO **MHL1A1**

- In SC70, the message from DEVMAN DMO0061I indicating that SC70 has picked up and detected the need to refresh the UCB for the device address 8106 that volume MHLTA1 is on:

  DMO0061I 8106,MHLTA1,REFUCB STARTED

- In SC70, message IEE302I indicating that a VARY ONLINE has completed:

  IEE302I 8106   ONLINE BY DMOAT002

- In SC70, the message from DEVMAN DMO0062I indicating the REFUCB function has completed and that the VOLSER is now MHL1A1:

  DMO0062I 8106,MHL1A1,REFUCB SUCCESSFUL

Example 3-4 SYSLOG output of activity for systems SC64 and SC70

```
SC64     2011270 03:10:03.16 JOB10403 80000010  IEF403I MHLRES2D - STARTED - TIME=03.10.03 - ASID=009B -
SC64
SC64     2011270 03:10:03.28 JOB10403 00000090 *034 ADR369D AUTHORIZE FOR WRITE ACCESS A VTOCIX DATA SET ON
MHLTA1,
                                          MHLRES2D, REST    , REPLY U OR T
SC64     2011270 03:10:21.61 MHLRES2  00000290  R 34,U
SC64     2011270 03:10:21.62 JOB10403 00000090  IEE600I REPLY TO 034 IS;U
SC64     2011270 03:10:21.98 JOB10403 00000090  IEC604I VTOC CONVERT ROUTINE ENTERED ON
8106,MHLTA1,DOS,DEVMAN
SC64     2011270 03:10:22.17 JOB10403 00000090  ICK502I   BUILDIX FUNCTION STARTED
SC64     2011270 03:10:29.18 JOB10403 00000090  ICK503I   8106 REQUEST RECEIVED TO CONVERT VTOC TO IXFORMAT
SC64     2011270 03:10:29.18 JOB10403 00000090  ICK504I   8106 VTOC FORMAT IS CURRENTLY OSFORMAT, REQUEST
ACCEPTED
SC64     2011270 03:10:29.83 JOB10403 00000090  ICK513I   8106  BUILDIX PROCESSING COMPLETED:  VTOC IS NOW
IN IXFORMA
SC64     2011270 03:10:29.86 JOB10403 00000090  ADR320I (001)-SBRTN(01), VOLUME SERIAL MHLTA1 ON UNIT 8106
IS CHANGED
                                           830
                              830 00000090  TO MHL1A1
SC70     2011270 03:10:29.87          00000290  DMO0061I 8106,MHLTA1,REFUCB STARTED
SC64     2011270 03:10:29.88 JOB10403 00000290  -                                            --TIMINGS
(MINS.)--
                                             ----PAGING COUNTS---
SC64     2011270 03:10:29.88 JOB10403 00000290  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB
CLOCK   SERV
                                          PG   PAGE   SWAP    VIO SWAPS STEPNO
SC64     2011270 03:10:29.88 JOB10403 00000290  -MHLRES2D          REST        00    971    .00    .00
.44   3014
                                           0      0      0      0     0     1
SC64     2011270 03:10:29.88 JOB10403 80000010  IEF404I MHLRES2D - ENDED - TIME=03.10.29 - ASID=009B - SC64
SC64     2011270 03:10:29.88 JOB10403 00000290  -MHLRES2D ENDED.  NAME-UAALFO             TOTAL CPU TIME=
.00
SC64     2011270 03:10:29.88 JOB10403 00000090  $HASP395 MHLRES2D ENDED
SC70     2011270 03:10:29.89 INTERNAL 00000290  IEE302I 8106    ONLINE BY DMOAT002
SC70     2011270 03:10:29.89          00000290  DMO0062I 8106,MHL1A1,REFUCB SUCCESSFUL
SC64     2011270 03:10:29.90          00000090  $HASP309 INIT 9   INACTIVE ******** C=A
```

```
SC64     2011270 03:10:29.90 INTERNAL 00000290  SE '03.10.29 JOB10403 $HASP165 MHLRES2D ENDED AT WTSCPLX2
MAXCC=0000
                                                ,LOGON,USER=(MHLRES2)
SC70    2011270 03:11:34.79 MHLRES5  00000290  D U,,,8106
SC70    2011270 03:11:34.80 MHLRES5  00000090  IEE457I 03.11.34 UNIT STATUS 739
                             739 00000090  UNIT TYPE STATUS       VOLSER    VOLSTATE
                             739 00000090  8106 3390 O           MHL1A1    PRIV/RSDNT
                             739 00000090  8107 3390 O           PAG311    PRIV/RSDNT
                                 739 00000090  8108 3390 O            MHL1A3     PRIV/RSDNT
```

3.2.5 DSS cross system notification implementation background information

After the DEVMAN ENABLE(REFUCB) option is activated in any member of a Sysplex, the DSS user does not need to do anything during the DSS job. Internal system functions detect the ENF64 event and notify the DEVMAN tasks in the other members of the Sysplex.

If the DEVMAN ENABLE(REFUCB) function is not enabled, or the member of the Sysplex is lower than z/OS V1.13, the ENF64 event notification is ignored.

The DEVMAN task in all members in the Sysplex where REFUCB is enabled initiate activity as required and if applicable:

- If the volume that DEVMAN is notified about is OFFLINE, nothing is done.
- If the VOLSER that DEVMAN is notified about is ONLINE, the following command is issued:

  ```
  VARY dddd,ONLINE,UNCOND
  ```

  This command causes the UCB for unit dddd to be refreshed with the new VOLSER or VTOC location.

3.2.6 Use of the ENF64 event by other functions

DSS is not the only function to issue the ENF64 event. DFSMShsm takes advantage of the XSYS enhancement implicitly if DSS is the defined data mover.

ICKDSF, the utility for VTOC management, also issues the ENF64 event for the specific situation where the VTOC Index must be rebuilt. This process occurs even though the VOLSER is not changed and the VTOC location is not moved. In this case, DEVMAN in other logical partitions (LPARs) that have REFUCB enabled refreshes the UCB.

3.3 Dynamic Volume Expansion and copy services

It is not possible to use the Dynamic Volume Expansion (DVE) function while a relationship is established. To extend a volume in a Peer-to-Peer Remote Copy (PPRC) relationship, the PPRC relationship must be deleted. The two volumes in the pair can then be dynamically updated to the same larger size and the PPRC relationship reestablished. You must then restart the mirroring.

4

Changes to DFSMShsm in DFSMS V1.13

This chapter explains the changes to DFSMShsm in V1.13. The focus of this release is on improving performance and reducing processor consumption, especially in the space management area.

This chapter contains the following sections:

- Summary of DFSMShsm changes
- DFSMShsm on-demand migration
- DFSMShsm control data set enhancements
- Small data set packing performance improvement
- DFSMShsm RAS and usability improvements
- Release of recalls: DASD only

4.1 Summary of DFSMShsm changes

In DFSMS V1.13, there are two major changes: A new feature (on-demand migration), and an improvement to CDS backup for performance and availability. Additionally, a number of RAS improvements have been added. V1.13 of DFSMShsm has been enhanced in many areas to lower processor consumption and increase usability and reliability, availability, and serviceability (RAS).

4.2 DFSMShsm on-demand migration

During interval migration, DFSMShsm runs a top-of-the hour space check on every volume it manages. This check determines whether it should move data from the volume to meet the specified threshold occupancy. Because DFSMShsm checks every volume during this process, interval migration typically causes a spike in DFSMShsm processor usage and can consume a lot of time.

Beginning in V1.13, installations have a new option in DFSMShsm event driven space management processing. DFSMShsm on-demand migration queues a volume for space management processing immediately after the volume exceeds its high threshold. It does not wait for the top-of-the hour space management window as in interval migration.

The DFSMShsm team collaborated with the z/OS Core Component and Storage Management Subsystem (SMS) teams to create Event Notification Facility code 72 (ENF72). When a volume exceeds its specified threshold because of a new data set allocation or extension, SMS issues ENF72 on each system within the SMSplex. In on-demand migration mode, a new DFSMShsm listening exit captures each ENF72 volume over-threshold event and relays the information to DFSMShsm. If the volume is in a storage group that is defined with the auto migrate option set to Yes (AM=Y), DFSMShsm immediately queues the volume for space management processing. The volume is queued for processing on each DFSMShsm system that has on-demand migration enabled. However, it is processed by the first host to select the volume. Space management continues on the volume until the volume reaches its low threshold or there are no more data sets to migrate, expire, or delete.

Consideration: Interval migration is still available. It is the default event-driven migration option for DFSMShsm.

The ENF notification happens from DFSMS every time the volume threshold is exceeded. The notification is sent to all the members in the SMSplex sharing the actual COMMDS through which the communication happens. Even though all systems are notified, only one runs the space management. Space management on individual volumes continues until low threshold is reached or there are no more data sets to migrate, delete, or expire.

To avoid too many notifications, DFSMS remembers the previous trigger value. It only issues a notification on the same volume if it grows to 25% of capacity above threshold. This process is done four times: at 25%, 50%, 75%, and 100%. If the growth is more than 60% of the threshold capacity, the first two notifications are skipped, but the next one is issued at 75%. If the capacity reaches a critical value (97% used), an ENF notification is issued on all volume checks. At 100% usage of the volume, another notification is issued. For EAV volumes, both high thresholds (track-managed and cylinder-managed) are monitored. If either is exceeded, an ENF notification is issued to activate on-demand migration for this volume.

4.2.1 Enabling on-demand migration

To activate or deactivate on-demand migration, there is a new SETSYS command: SETSYS ONDEMANDMIGRATION (Y|N). When set to Y, on-demand migration replaces interval migration for all DFSMShsm SMS-managed volumes. DFSMShsm limits its processing to those volumes in storage groups with an AM setting of Y. Interval migration no longer runs on DFSMS managed volumes in these storage groups. However, interval migration continues to process non-SMS managed volumes and volumes in storage groups with an AM setting of I.

Setting the SETSYS ONDEMANDMIGRATION parameter to N makes interval migration act as it did before DFSMS V1.13. By default, the SETSYS ONDEMANDMIGRATION parameter is set to N (not using on-demand migration). The abbreviation for ONDEMANDMIGRATION is ODM.

On-demand migration does not run concurrently when interval migration or primary space management (PSM) runs on a host. When interval migration or PSM starts, on-demand migrations pause and begins again after PSM or interval migration are finished.

> **Requirement:** Storage groups with an AM value of I must be analyzed for conversion to AM=Y to use the new on-demand migration function.

After migrating to DFSMS V1.13, the hourly check of an SMS configuration change will stop. It will be replaced by an ENF notification that sends a signal after every DFSMS configuration change. This process reduces the processor usage of this check significantly. DFSMShsm now only runs the SMS configuration change inventory record updates at the beginning of primary space management, on-demand migration, and interval migration if the ENF15 notification has been received.

Currently there is a patch that can disable the hourly configuration check, MCVT.+C8 BITS(1.......). Because of the new SMS configuration change checking in V1.13, this patch can be removed.

A new notification limit can be set for to alert the volumes selected for on-demand migration. If this limit is exceeded, a message is displayed as shown in Example 4-1.

Example 4-1 Example of notification based on ODMNOTIFICATIONLIMIT

```
ARC1901I NUMBER OF VOLUMES ELIGIBLE FOR ON-DEMAND MIGRATION HAS REACHED nnnn
```

The limit is set by the new setsys command ODMNOTIFICATIONLIMIT(limit). The default value for this number is 100. Issuing QUERY SETSYS also displays this value.

4.2.2 Compatibility and coexistence

After you enable on-demand migration, each DFSMShsm host at V1R13 or later will process primary SMS volumes on demand. You must ensure that there are enough hosts available to run other critical DFSMShsm activities as well. This is also true in a mixed HSMplex environment (pre-V1R13 and V1R13 systems) where the V1R13 hosts might be busy running migrations. Remember that your V1R12 hosts can run only interval migration.

Messages related to the new function have been updated to reflect the status of on-demand migration.

4.3 DFSMShsm control data set enhancements

To assure integrity, all DFSMShsm activities must be quiesced while the DFSMShsm CDS backup is running. This process causes perceptible delay for any other activities during the backup window for customers, such as waiting for recalls requested by users or batch jobs. This pause is because the CDS backup must wait for all active requests to finish before the backup can start.

Using DSS Concurrent Copy point-in-time backup for CDSs can reduce the time spent in backup. However, point-in-time backup is only supported for the CDSs, not for the journal. If the physical copy fails as well, you will lose data.

The implementation of the CDS backup has changed in DFSMShsm V1.13 for those users who use DSS point-in-time copy. Access to the journal is possible during most of the backup window, reducing the time that all other DFSMShsm activity is locked out.

4.3.1 Description

Before DFSMS V1.13, CDS backup enqueued exclusively on ARCCAT and held it for some time. This configuration caused subsequent requests to queue up behind it. In DFSMS V1.13, DFSMShsm releases ARCCAT fairly quickly from all hosts in the HSMplex after completing pending requests. Other tasks can therefore start more quickly. This process provides better performance while reducing elapsed time on the CDS backup.

> **Remember:** This process applies only to a CDS backup using DSS Concurrent Copy.

During a CDS backup in DFSMS V1.13, first the journal is backed up. After completion, CDS backups start and run concurrently, ensuring that the journal and CDSs are in sync.

The journal is backed up in two steps:

1. Back up the records already written to the journal at the time when CDS backup started.
2. Back up the records written to the journal after the backup started.

After this process is completed, the journal will be nullified, and backup of the CDSs can start. Enqueue on the ARCCAT resource is needed only through the second phase of the journal backup. The logical time is spent in backing up the CDSs with point-in-time copy. Even if the backup window is slightly increased, the lockout time of other DFSMShsm activities will be noticeably reduced, improving availability. This way of backing up the journal is known as *non-intrusive*. The alternative way of backing up the journal is known as *quiesce*. If possible, the non-intrusive way is automatically selected.

The non-intrusive backup has these requirements:

- The SETSYS JOURNAL(RECOVERY) setting is in effect (by command or through PARMLIB).
- All control data set (CDS) clusters are SMS-managed. The management class for each cluster indicates a Backup Copy Technique of concurrent copy rather than standard.
- DSS must be the data mover for CDS backup.

Message ARC0750I has been updated to display which of the backup techniques ia used. A sample is shown in Example 4-2.

Example 4-2 Example of ARC0750I showing backup technique

```
ARC0750I BACKUP FOR dsid STARTING AT time ON date, BACKUP TECHNIQUE IS {
NON-INTRUSIVE | QUIESCED(note) }
```

The note value shows the first incompatible condition encountered. The value in the note can be one of those shown in Table 4-1.

Table 4-1 Possible note values in ARC0570I message

Note value	Explanation
a	SETSYS CDSVERSIONBACKUP(DATAMOVER(DSS)) is not specified
b	SETSYS JOURNAL(RECOVERY) is not specified
c	Non-intrusive backup was attempted during a previous journal backup and the journal backup failed. Quiesce journal backup will be attempted. After a quiesced journal backup is successful, the journal will become eligible again for non-intrusive backup.
d	One or more CDS clusters are not SMS-managed, or the Management class for one or more CDS clusters indicates STANDARD for Backup Copy Technique (Concurrent Copy)
other	For special conditions other than a, b, c, or d, the note itself indicates the actual reason

The ARC0744E message during journal backup has been updated to reflect two new return codes (RC88 and RC92). These codes indicate non-intrusive errors while backing up the journal as shown in Example 4-3.

Example 4-3 Example ARC0744E message

```
ARC0744E dsid COULD NOT BE BACKED UP, RC=retcode, REAS=reascode MIGRATION, BACKUP,
FRBACKUP, DUMP, AND RECYCLE HELD
```

Figure 4-2 shows the explanations behind the two new messages.

Table 4-2 ARC0744E message together with RC88 and RC92

Return code	Explanation
RC88	An error was encountered attempting to serialize the CDSs during non-intrusive journal backup
RC92	Intrusive journal backup ended before expected

To handle these two types of problems, collect problem determination aid (PDA) data and a job log and contact the IBM Support Center.

In DFSMS V1.10, XCF support reduced the time the CDS backup waited for an enqueue on ARCCAT. It did so by requesting the resource through XCF from all the DFSMShsm tasks that share the CDS. Some functions, however, were still not released.

In environments using RLS serialization of the CDS or in multiple address space DFSMShsm environments, the support assumes use of XCF. XCF presence is not mandatory, but it gives better performance, resulting in ARCCAT being released faster.

4.3.2 Compatibility and coexistence

The new support applies for environments that use point-in-time copy (DSS Concurrent Copy).

> **Consideration:** In mixed environments (DFSMS V1.13 systems along with prior level systems), this support still provides a benefit if the CDS backup is run from a DFSMS V1.13 LPAR.

When migrating to DFSMS V1.13, the prior level systems must have coexistence support added. This support is needed so that journal updates happen in a consistent way for all the DFSMShsm tasks in the HSMplex.

The optimum performance benefit in this support will be achieved only when all records in the journal are written after migrating to DFSMS V1.13.

4.4 Small data set packing performance improvement

Small data set packing is a DFSMShsm function that reduces the space requirement when migrating smaller data sets to ML1. Instead of being migrated as individual data sets, these data sets are chosen based on a size limit. They are migrated into a single VSAM data set on ML1 called SDSP (small-data-set packing) and packed efficiently. Have a number of SDSPs available so that DFSMShsm can use others when one is filled.

4.4.1 ARCMMEXT overhead removal

The ARCMMEXT exit looks at already migrated data sets to see whether they should be moved on to another device/migration level. When Secondary Space Management builds the SMQE queue during the second phase, ARCMMEXT is started to confirm the data set migration eligibility. It is done within serialization scope and leads to unjustified delays because data sets might not be candidates for migration.

In DFSMShsm V1.13, the ARCMMEXT exit is started by Secondary Space Management during the initial MCD records scanning to identify candidates to put on queue (SMQE). This queue is processed on the second phase when determining whether DFSMShsm should migrate data sets to level 2.

4.4.2 Balanced SDSP selection algorithm

Another tuning option has been added in the way DFSMShsm selects the SDSPs. The target SDSP is currently selected based on ML1 volume ADDVOL sequence. The same SDSP or SDSPs are always selected first. This process always fills these preferred SDSPs, so you must reorganize them, which can leave additional SDSPs empty.

DFSMS V1.13 introduces a change in this approach. A new SDSP selection algorithm is used that ensures that the SDSPs are used evenly. This algorithm is based on SDSP freespace status. Freespace on the SDSP is calculated in two ways:

- During ADDVOL processing
- Every time an SDSP is closed (at completion of data processing)

Explanation: SDSP free space is calculated based on high allocated and high used relative byte addresses (RBAs). If the new CA reclaim feature is been used, free space includes reclaimed CAs.

Have a number of SDSPs allocated to DFSMShsm so that the distribution of the small data sets that are candidates for SDSP processing can happen across multiple SDSPs.

4.5 DFSMShsm RAS and usability improvements

DFSMS V1.13 introduces a number of usability and RAS improvements.

4.5.1 PDA trace now enabled by default during DFSMShsm startup

DFSMS V1.13 introduces a change in the default for DFSMShsm PDA processing. Problem determination aid (PDA) is a function that logs diagnostic information in storage and later writes it into a file. Previously the default was PDA=NO (PDA logging disabled). This default has been changed to YES.

If PDA=NO is specified in the ARCCMDxx member or during startup, the function is still disabled.

In an existing DFSMShsm environment where you did not specify a value in the PDA parameter, you must allocate PDA data sets. You then must add those PDA data sets to the DFSMShsm startup procedure that is migrating to DFSMS V1.13. Otherwise the startup will fail due to missing output files for offloading the PDA data.

If you do not want to use the PDA function and want a seamless transition to DFSMS V1.13, specify PDA=NO. Generally, however, use a PDA setting of YES because troubleshooting DFHSM errors will probably require you to provide PDA data as part of the documentation.

4.5.2 Fast Replication ARC1809I messages

A DFSMShsm Fast Replication backup initiated by issuing an FRBACKUP command to pair a primary volume with a secondary volume in the backup storage group might cause a problem. Target volumes that have already been processed are often re-examined, resulting in multiple ARC1809I return code 2 messages for the same volume. You can suppress ARC1809I messages by using a patch, but doing do means that **all** ARC1809I messages are left out.

To manage the behavior with the SETSYS command instead, DFSMS V1.13 introduced a new SETSYS parameter:

```
FASTREPLICATION(VOLUMEPAIRMESSAGES(YES|NO))
```

Description

Setting this parameter to YES limits the informational message to being issued once per target volume per source storage group. Changing the setting to NO causes the ARC1809I message to occur only once for each secondary volume.

The current patch to disable the occurrence of ARC1809I messages continues to be supported.

> **Consideration:** The default for VOLUMEPAIRMESSAGES is NO.

Compatibility and coexistence

Because the default for the new fast replication keyword is NO, a migration to DFSMS V1.13 does not require any mandatory changes. If the patching of the ARC1809I message is already used, the patch continues to work.

If you want changed DFSMShsm behavior on issuing the ARC1809I messages, run the VOLUMEPAIRMESSAGES(NO) SETSYS command.

If needed for diagnostic reasons, you can always change the parameter to YES to have the full list of ARC1809I error messages. Turn this support on and off as needed.

The QUERY SETSYS command has been updated to be able to display the status of the new keyword. See the update to the FASTREPLICATION command in Figure 4-1.

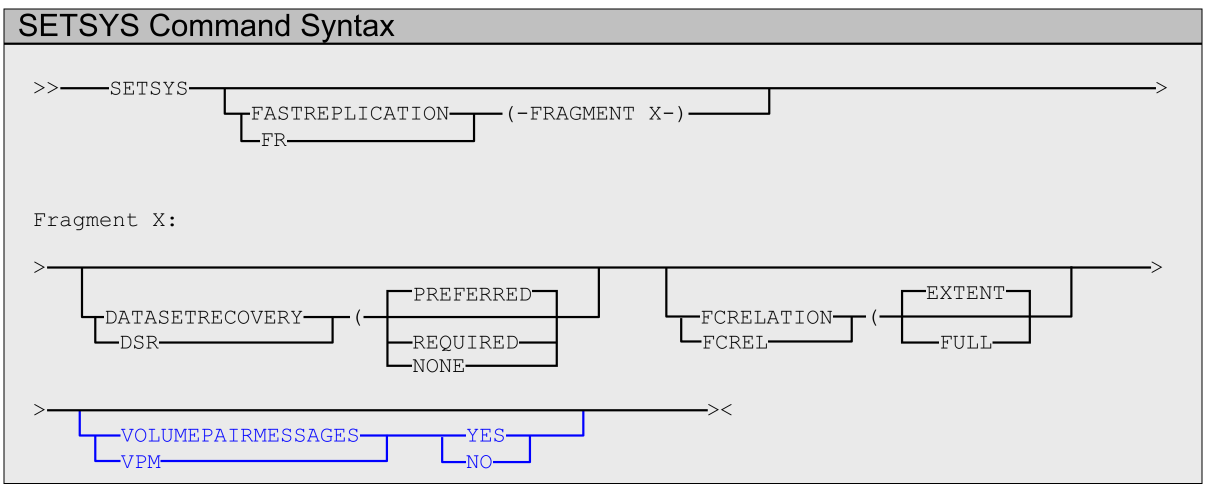


Figure 4-1 DFSMShsm new SETSYS parameter VOLUMEPAIRMESSAGES(YES!NO)

4.6 Release of recalls: DASD only

DFSMShsm V1.13 introduces the ability to enable DASD-only recalls (and deletes, which go together with recalls) after a HOLD RECALL() is issued. Enabling DASD-only recalls can now happen through the command RELEASE RECALL(DASD).

HOLD RECALL has these possible variations:

- HOLD RECALL(ALL)
- HOLD RECALL(TAPE)
- HOLD RECALL(TAPE(TSO))

If a HOLD of all recalls is issued, RELEASE of all held TAPE or TSO user recalls related to tape is not accepted. This process is possible only based on a previous specific hold on these two functions.

After a RELEASE RECALL(DASD), a verification is done of whether a HOLD RECALL(ALL) is currently active. If it is not, nothing happens. If a HOLD(ALL) status is active, it is converted to a HOLD RECALL(TAPE) status, leaving DASD-only recalls (and deletes) enabled.

The new parameter DASD is added to the RECALL command as shown in Figure 4-2.

SETSYS Command Syntax

```
>>──RELEASE──────────────────────────────────────────────────────────>

>─────────────────────────────────────────────────────────────────────><
      └─RECALL─┬──────────────────────────────┬──────────────────┘
               └─(──┬─TAPE──┬──────┬─┬──)──┘
                    │       └(TSO)─┘ │
                    └─DASD───────────┘
```

Figure 4-2 DFSMShsm recall command: DASD only

> **Remember:** There is no equivalent HOLD for DASD recalls only

4.6.1 Identification of originating host in a CRQ environment

DFSMShsm Common Recall queue is a shared queue that holds all recall requests in a sysplex. It is based on a shared CRQ list structure in the Coupling Facility (CF).

Description

Any member of the sysplex can process the recalls based on availability and priority. This configuration provides the benefit of using all the processing power in the sysplex instead of being limited to (current) resources in a single logical partition (LPAR).

The advantage of this configuration is that participating systems without resources such as a tape drive can still have tape recalls processed by another member of the sysplex. If a system is unavailable, recalls can be processed from the Common Recall queue. Additionally, change request (CRQ) enables more efficient usage of tape resources. It is able to stack recalls against any single tape volume because a single request making recalls is faster and more efficient.

All recall requests have two connected hosts:

- The host from where the recall was issued (originating host)
- The host where the recall is active (processing host)

A problem with a single recall entry in the Common Recall queue where you need to cancel a request requires you to cancel the request from the originating host. However a display such as QUERY COMMONQUEUE(RECALL) does not show the originating host, only the processing host. In a large sysplex, it can be quite time consuming to search for this information.

Example 4-4 shows output from a pre-DFSMS V1.13+.

Example 4-4 DFSMShsm Common Recall queue display without host information

```
ARC1543I RECALL MWE FOR DATASET DDAP.DHSTT.N.DDAACT.ACTCURK1.G0944V00, FOR USER
TM, REQUEST 00009826, WAITING TO BE PROCESSED ON A COMMON QUEUE,00001234 MWES
AHEAD OF THIS ONE
```

Searching for the host made diagnosis and handling of failing requests in a large DFSMShsm environment difficult. In DFSMS V1.13, the originating host is now included in the display. Additionally, the processing host is stored in a management work element (MWE) that is used for troubleshooting. See the output from QUERY COMMONQUEUE(RECALL) in Example 4-5.

Example 4-5 DFSMShsm Common Recall display with originating host information

```
ARC1543I type MWE FOR DATA SET name, FOR USER userid, REQUEST request_number,
ISSUED FROM HOST hostid, WAITING TO BE PROCESSED ON A COMMON QUEUE, nmwe MWES
AHEAD OF THIS ONE{,REQUEST ORIGINATED ON HOST hostid}
```

Compatibility and coexistence

Migrating to DFSMS V1.13 automatically starts the new support in environments where DFSMShsm Common Recall queue is enabled.

4.6.2 DFSMShsm ONLYIF enhancements

To enable DFSMShsm to share PARMLIBs between more DFSMShsm tasks, ONLYIF HSMHOST(x) statements were introduced.

Description

These statements allow you to have certain statements in a shared PARMLIB run only for a specific host. This support is limited to give only one host ID at a time. Also the implementation is limited to the statement that follows the ONLYIF statement, as shown in Example 4-6.

Example 4-6 DFSMShsm usage of ONLYIF: Old implementation

```
ONLYIF HSMHOST(A)
SETSYS ABSTART(2000 2200)
```

To allow for more flexible sharing of PARMLIBs between multiple DFSMShsm tasks, an extended support is now available in DFSMS V1.13. ONLYIF now has statements that indicate that a block of statements follows until closed by an END statement. The implementation looks like Example 4-7.

Example 4-7 DFSMShsm usage of ONLYIF: New implementation

```
ONLYIF HSMHOST(A,B)
BEGIN
SETSYS ABSTART(2000 2200)
SETSYS PRIMARYSPMGMTSTART(1700-2000)
END
```

You can now specify more than one DFSMShsm task in one ONLYIF statement. Also, it is possible to give a number of statements after the BEGIN statements has started an ONLYIF sequence. The END statement ends such a sequence.

Compatibility and coexistence

The previous functionality in the ONLYIF command still works along with the new support. Upgrade to DFSMS V1.13 in a mixed z/OS environment, however, requires compatibility program temporary fixes (PTFs) on the prior level systems.

4.6.3 AUDIT COPYPOOLSCONTROL for orphaned FRTV records

The DFSMShsm Fast Replication feature is based on a volume to volume IBM FlashCopy® from a primary pool to a copy pool. In a recovery situation, it is important that the integrity is in place so that recovery can happen without any problems. Experience has shown that target volumes are left with a pointer to one or more primary volumes, preventing these target volumes from being reused. This is also a potential risk in a recovery situation.

DFSMS V1.13 improves AUDIT COPYPOOLCONTROLS to detect all orphaned fast replication target volume (FRTV) records when a COPYPOOL is not specified on the command. Using a copy pool name on the audit command will not find all orphaned records. This is because the target volume found in error might not have been associated with the copy pool specified on the audit command. The FIXCDS commands are displayed in the output from the AUDIT command. Specifying FIX DFSMShsm runs FIXCDS DELETE on the detected invalid pointers. The error message that identifies the invalid pointers is an ERR 202 as shown in Example 4-8.

Example 4-8 Example of Audit COPY POOLS CONTROL message

```
*ERR 202 ORPHANED I (FRTV) RECORD FOUND FOR TARGET VOLUME <tgtVOLSER>, SOURCE
VOLUME <srcVOLSER>
```

Implementing this function improves integrity in the BCDS and mean less exposure for errors in the backup and recovery that uses the Fast Replication feature.

4.6.4 DFSMShsm ARC0570I patches

During auto function (AUTODUMP, AUTOBACKUP, or AUTOMIGRATION), message ARC0570I might be issued. The reason is that DFSMShsm is trying to obtain storage group and copy pool information. This information might not be there for valid reasons, but the ARC0570 message can be confusing. Message ARC0570 connected with return code 17 indicates that no DFSMS storage groups were defined, whereas return code 36 indicates that no copy pools are defined.

To suppress either or both of these message, two new patches come with DFSMS V1.13. For suppression of message ARC050I associated with return code 17 (no DFSMS storage groups defined), a patch is available as shown in Example 4-9.

Example 4-9 DFSMShsm patch for suppressing ARC0570I RC17

```
ARC0570I RC17 messages will be turned off using the following patch:
PATCH .MCVT.+297 BITS(....1...)
```

For suppression of message ARC0570I associated with return code 36 (no copy pools defined), use the patch shown in Example 4-10.

Example 4-10 DFSMShsm patch for suppressing ARC0570I RC36

```
ARC0570I RC36 messages will be turned off using the following patch:
PATCH .MCVT.+297 BITS(.....1..)
```

Tip: Messages can be turned on again by specifying ‘0’ instead of ‘1’ in the patches.

4.6.5 Change of default in FASTREPLICATION(DSR)

DFSMShsm fast replication recovery on a data set level can use a background FlashCopy to do the recovery. If a new copy pool backup becomes active using the same volume, the backup session fails with a message that indicates that the volume is already in use. However, the problem is hard to diagnose for the Storage Administrator because the available information is limited.

Description

To improve troubleshooting, DFSMS V1.13 introduced a new default on the FASTREPLICATION(DSR) command: FASTREPLICATION(DSR(NONE)). Previously the default was FASTREPLICATIION(DSR(PREFERRED)). With the new default and the use of traditional copy methods, the error message indicates a serialization problem, which is much easier to diagnose.

When FlashCopy is used, the error message indicates that the backup volume was already a FlashCopy source. With the new default, traditional copy is used. The advantage of using traditional copy is that any error message clearly indicates a serialization problem.

Compatibility and coexistence

The value for FASTREPLICATION issued on the recover command in the current implementation overrules any specification on the coded SETSYS value of the same parameter. If any value for the copy is not specified for the recover, the SETSYS specification is used. This approach will stay the same.

For native DFDSS, you cannot specify FASTREPLICATION(DSR(NONE) against a PPRC primary volume. However, DFSMShsm does not check this.

z/OS DFSMShsm Storage Administration has been updated to state that the options on ALLOWPPRCP: PRESERVEMIRRORNO, PRESERVEMIRRORPREFERRED, PRESERVEMIRRORREQUIRED do not go together with a FASTREPLICATION specification of NONE.

4.6.6 Change of DFSMShsm informational messages

DFSMShsm currently issues the messages ARC0704I, ARC0503I, and ARC0036I as informational messages. These messages address VTOC copy, dynamic allocation, and PDA write errors.

The message type has changed from informational (I) to error (E) in DFSMS V1.13. This change allows users to program their automated operations to react to them.

5

DFSMSrmm enhancements

The DFSMSrmm enhancements in z/OS V1.13 DFSMS provide improvements in the areas of usability and maintainability. This release includes these changes:

- New RETENTIONMETHOD(EXPDT)
- Excluding data sets from VRSEL processing
- Data set attribute COPYFROM function
- TVEXTPURGE extra days
- SEARCHDATASET extensions
- VRS last reference date
- Selective volume movement
- Last change details
- Support RETPD(93000)
- Dialog navigation enhancements

This chapter addresses how to specify the different retention methods for your volumes and to exclude some of your data sets from vital record processing. This chapter includes the following sections:

- Overview
- New RETENTIONMETHOD(EXPDT)
- Excluding data sets from VRSEL processing
- Data set attribute COPYFROM function
- TVEXTPURGE extra days
- SEARCHDATASET extensions
- VRS last reference date
- Selective volume movement
- Last change details
- Support RETPD(93000)
- Dialog navigation enhancements
- Migration considerations

5.1 Overview

In your enterprise, you probably store and manage removable media in several types of media libraries. For example, in addition to your traditional tape library, a room with tapes, shelves, and drives, you might have several automated, virtual, and manual tape libraries. You probably also have both on-site libraries and off-site storage locations, also known as vaults or stores.

With DFSMSrmm, a z/OS feature, you can manage your removable media as one enterprise-wide library across systems and Sysplexes. DFSMSrmm manages your installation tape volumes and the data sets on those volumes. DFSMSrmm also manages the shelves where volumes are in all locations except in automated tape libraries.

DFSMSrmm manages all tape media, such as cartridge system tapes and 3420 reels, as well as other removable media you define to it. For example, DFSMSrmm can record the shelf location for optical disks and track their vital record status. However, it does not manage the objects on optical disks.

5.2 New RETENTIONMETHOD(EXPDT)

Among the important decisions to be made when using DFSMSrmm is how to retain tape data sets and for how long. You might want to retain a data set for a specific period after it is created. Or you might want to retain it based on some event such as while the data set is cataloged, or retain it permanently. With z/OS V1.13, DFSMSrmm offers two retention methods, EXPDT and VRSEL, for retention of your volumes and data sets on your volumes. For each volume set, specify whether the volumes and data sets in that volume set are managed by the expiration date or by VRSEL retention vital record specification (VRS) policies.

Every volume has a retention method, either EXPDT or VRSEL. The default retention method specified in PARMLIB is used, unless one is assigned by the EDG_EXIT100 installation exit, the RMM TSO ADDVOLUME, or the CHANGEVOLUME subcommand.

When data is created, the storage administrator can select an appropriate retention method for a volume set. The storage administrator can select between these options:

VRSEL The VRSEL retention method can retain a volume beyond its expiration date by using a vital record specification to define its retention policy. The VRSEL retention method can also be used for setting a movement policy. Volumes and data sets managed by this retention method are subject to frequent VRSEL processing. In every run of VRSEL processing, your data sets and volumes are matched to the vital record policies. Therefore, the retention and movement management can change from one inventory management run to another.

EXPDT The EXPDT retention method is based on the expiration date and avoids VRSEL processing. As a result, the retention information must be set when a tape data set is created. You can also manually override an expiration date to release a volume before the original expiration date is reached. The movement of the volumes must be managed manually.

5.2.1 Specifying a retention method

The EDGRMMxx PARMLIB member has a new operand. This operand sets the system-wide default retention method for new tape volumes or tape volume sets created during Open/Close/end-of-volume (O/C/EOV) processing. It also sets the retention method for tape volumes added to the DFSMSrmm control data set (CDS).

5.2.2 Expiration date

One important parameter for both retention methods is the expiration date of the data set and the volume. The data set expiration date or retention period is determined when a new tape data set is created. The expiration date or retention period can be specified at multiple levels:

1. The default retention period RETPD specified in the DFSMSrmm PARMLIB member
2. The EXPDT or RETPD in the DFSMS data class if the data set is associated with a DFSMS data class
3. The JCL DD statement, which is set by using the EXPDT or RETPD keywords
4. The DFSMSrmm installation exit EDG_EXIT100

Remember: Changing the expiration date for a data set record that represents one part of a multivolume data set on volumes managed by the EXPDT retention method updates all the data set records for that data set.

5.2.3 Retention date

z/OS V1.13 DFSMSrmm provides enhancements to make tape management easier and improve administrator productivity. The retention date is displayed instead of the expiration date in the search results list if the volume or data set is VRS retained. Formerly, when a resource was retained by VRS, the search results list for volumes or data sets might include resources that had expired. Now you can more easily determine from the search results list why the resource is retained without viewing the volume and data set details.

5.2.4 Using the EDGRMMnn PARMLIB option RETENTIONMETHOD

Use this operand to set the system-wide retention method default for new tape volume sets. New tape volume sets can be created during O/C/EOV processing, or through DFSMSrmm commands. A tape volume set can be a multi-volume set, or a single tape volume. RETENTIONMETHOD can be abbreviated as RM as shown in Figure 5-1.

```
                          |—RETENTIONMETHOD(VRSEL)———————|
                          |                              |
▸——OPTION——————————————————————————————————————————————————————————◂◂
                          |                              |
                          |                  |—VRSEL—|   |
                          |—RETENTIONMETHOD(--———————————)—|
                            RM               |—EXPDT—|
```

Figure 5-1 EDGRMMnn RETENTIONMETHOD operand

The command has the following parameters:

- VRSEL: The VRSEL retention method can retain a volume beyond its expiration date by using a vital record specification to define a retention policy. The VRSEL retention method can also be used for setting a movement policy. Volumes and data sets managed by this retention method are subject to frequent VRSEL processing. In every run of VRSEL processing, the matching of your data sets and volumes to the vital record policies is done again. Therefore, the retention and movement management can change from one inventory management run to another. This method is the default.
- EXPDT: The EXPDT retention method is based on the expiration date and avoids VRSEL processing. As a result, the retention information can be known when a tape data set is created. You can also manually override an expiration date to release a volume before the original expiration date is reached. The movement of the volumes must be managed manually.

Use the RMM TSO LISTCONTROL subcommand with the OPTION operand to list the current setting. Figure 5-2 on page 53 shows the result of the LISTCONTROL subcommand that includes the new Retention method information.

```
System options:
PARMLIB Suffix  = 02
Operating mode  = P    Retention period: Default = 0       Maximum = NOLIMIT
                                         Catalog = 6     hours
                       Retention method: Method  = VRSEL
Control data set name      = RMM.CONTROL.DSET
Journal file data set name = RMM.JOURNAL.DSET
Journal threshold          = 75%              Journal transaction = NO
Catalog SYSID              = Notset
Scratch procedure name     = EDGXPROC
Backup procedure name      = EDGCDSBK
IPL date check  = N        Date format    = J       RACF support   = N
SMF audit       = 42       SMF security   = 42      CDS id         = SC70
MAXHOLD value   = 100      Lines per page = 54      System ID      = SC70
BLP             = RMM      TVEXT purge    = RELEASE Notify         = N
                                 days     = 0
Uncatalog       = Y        VRS job name   = 2       Message case   = M
MASTER overwrite= USER     Accounting     = J       VRS selection  = NEW
VRS change      = INFO     GDG duplicate  = BUMP    GDG cycle by = GENERATION
VRSMIN action   = INFO     VRSMIN count   = 1
VRSDROP action  = INFO     VRSDROP count  = 0              percent = 10
VRSRETAIN action= INFO     VRSRETAIN count= 0              percent = 80
EXPDTDROP action= INFO     EXPDTDROP count= 0              percent = 10
Disp DD name    = DISPDD   Disp msg ID    = EDG4054I
Retain by       = SET      Move by        = SET     CMDAUTH Owner  = NO
PREACS          = NO       SMSACS         = NO      CMDAUTH Dsn    = YES
Reuse bin       = CONFIRMMOVE                       Media name     = 3480
                                                    Local tasks    = 10
PDA: ON
 Block count    = 0        Block size     = 0       Log            = ON
SMSTAPE:
 Update scratch = YES      Update command = YES     Update exits   = YES
 Purge          = ASIS
Client/Server:
 Subsystem type = STANDARD Port           = 0
 Server                    Server tasks   = 0
  host name  =
  IP address =
```

Figure 5-2 LISTCONTROL OPTION output

5.2.5 EDG_EXIT100 retention method support

You can use the EDG_EXIT100 installation exit to set the retention method to be used for a new tape data. When you create a tape volume or tape volume set, or rewrite an existing set from the first file, you can override the system default retention method.

With z/OS V1.13, every volume has a retention method, either EXPDT or VRSEL. The default retention method specified in PARMLIB is used unless one is assigned by the EDG_EXIT100 installation exit, or by the ADDVOLUME or CHANGEVOLUME command.

Update the sample EDGUX100 exit module based on your retention requirements by performing these tasks:

1. Optionally, define the system default retention method. See the Parmlib Member EDGRMMxx OPTION command
2. Define the DFSMSrmm default and maximum retention periods by using OPTION RETPD and MAXRETPD
3. Update the sample EDGUX100 exit module based on your retention requirements:
   a. Copy the sample EDGUX100 exit module and use the copy as a base for your exit module.
   b. Update the exit module. Perform your processing only when the PL100_CAN_RETMET bit is set to B'1'. Set PL100_RETENTIONMETHOD to the value required, and ensure the PL100_SET_RETMET bit is set to B'1'. If you do not set a retention method, the system default retention method is used.
   c. Make any other changes required, such as setting or clearing the EXPDT. The sample EDGUX100 exit module includes an example of setting the retention method. To use the sample EDGUX100 exit module for this function, modify the table as shown in Figure 5-3. The order in which the table entries are listed is important. The exit scans the table until it finds the first entry where the job name, data set name and program name masks match the current request. You can change the priority of matching by changing the order of the table entries.

```
RMTAB  DS OF                             START OF RM TABLE
       SPACE 1
       DC    CL8'*'                      JOBNAME
       DC    CL44'RMMUSER.RMVRSEL.*'     data sets NAME
       DC    CL8'*'                      PROGRAM NAME
       DC    AL1(PL100_RM_VRSEL)         RETENTION METHOD VRSEL
       DC    XL3'00'                     RESERVED
       SPACE 1
       DC    CL8'*'                      JOBNAME
       DC    CL44'RMMUSER.RMEXPDT.*'     data sets NAME
       DC    CL8'*'                      PROGRAM NAME
       DC    AL1(PL100_RM_EXPDT)         RETENTION METHOD EXPDT
       DC    XL3'00'                     RESERVED
       SPACE 1
       DC    CL8'RM END'                 END OF RM TABLE MARKER
```

Figure 5-3 Sample RETENTIONMETHOD selection table

The command has the following parameters:

– JOBNAME: One-to-eight alphanumeric or national characters that include % and *.
  - **%**: Can be used to ignore a positional character in the job name.
  - *****: Can be used to ignore all remaining characters in the job name. A JOBNAME of * means that the entry applies to all jobs.

– DATA SET NAME: Can be up to 44 characters, following z/OS data set naming conventions that include % and *.
  - **%:** Can be used to ignore a positional character in the data set name.
  - *****: Can be used to ignore all remaining characters in the data set name. A data set name of * means that the entry applies to all data sets.

The use of the character * is not the same as in the generic data set names supported by DFSMSrmm for vital records specifications and search data set masks. Here the * works like the characters *.* might in a generic data set name mask.

– PROGRAM NAME: A value up to eight alphanumeric characters that include % and *.
  - %: Can be used to ignore a positional character in the program name.
  - *: Can be used to ignore all remaining characters in the program name. A program name of * means that the entry applies to all programs.
– Retention method: This can be either of the following settings:
  - PL100_RM_VRSEL: To assign the retention method VRSEL
  - PL100_RM_EXPDT: To assign the retention method EXPDT

To compile and link the updated EDGUX100 user exit, use the JCL shown in Figure 5-4.

```
//ASMCL    PROC SM=,LM=
//C        EXEC PGM=ASMA90,PARM='NODECK,XREF(SHORT),LINECOUNT(58)'
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//SYSLIB   DD   DSN=SYS1.MACLIB,DISP=SHR
//         DD   DSN=SYS1.MODGEN,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(7040,400)
//SYSUT2   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSUT3   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSIN    DD   DSN=RMM.ADDONS.SOURCE(&SM),DISP=SHR
//L        EXEC PGM=HEWL,PARM='LIST,MAP,XREF,RENT,REUS,',COND=(4,LT,C)
//SYSLMOD  DD   DSN=SYS1.SANDBOX.MIGLIB(&LM),DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//         PEND
//COMPLNKO EXEC ASMCL,LM=EDGUX100,SM=EDGUX100
```

Figure 5-4 Compiling and link updating EDGUX100 user exit

You can use the SETPROG EXIT command to control exits that are defined to the dynamic exits facility. To activate or change the current EDGUX100 user exit, use the command shown in Figure 5-5. This command replaces the old sequence you used before z/OS V1.12.

```
SETPROG EXIT,REFRESH,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
```

Figure 5-5 SETPROG command syntax to refresh a dynamic exit

The correct sequence to replace an existing exit, before z/OS V1.12, is shown in Figure 5-6.

```
MODIFY DFRMM,QUIESCE

SETPROG EXIT,DELETE,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100

SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100

MODIFY DFRMM,M=nn
```

Figure 5-6 SETPROG command syntax sequence before z/OS V1.12

The command has the following parameters:

- ADD: Adds an exit routine to an exit.
- REPLACE: Replaces an exit routine for an exit.
- DELETE: Deletes an exit routine from an exit.
- EXITNAME: The 1-16 character name of the exit.
- MODNAME: The 1-8 character name of the exit routine. If DSNAME is not specified, the system tries to locate the exit routine by using the link pack area (LPA), the LNKLST concatenation, and the nucleus.

Tip: DFSMSrmm provides a second sample for EDGUX100, called EDGCVRSX. It differs from the EDGUX100 sample in that the special date, retention method, VRSELEXCLUDE, and pooling function are table driven. You can change this table dynamically. For more information about using EDGCVRSX for EDGUX100, see the SAMPLIB member EDGCMM01 and the IBM Redbooks DFSMSrmm Primer.

5.2.6 Subcommands for RETENTIONMETHOD parameters

You can use the TSO RMM ADDVOLUME or CHANGEVOLUME subcommands with the RETENTIONMETHOD operand to set the retention method for a tape volume or tape volume set. After a retention method is defined for a non-scratch volume, it is not overridden by the system-wide default during OPEN output processing. Instead, it can be changed by installation exit EDG_EXIT100. Volumes in a set always assume the retention method of the first volume in the set.

Specify this operand for the first volume in a multivolume sequence. All other volumes added to the set assume the same retention method. The correct syntax to set the retention method to EXPDT or VRSEL is shown in Figure 5-7.

Attention: Authorization can be either of these settings:

- Based on STGADMIN.EDG.MASTER access, if the resource below is not defined
- UPDATE access to STGADMIN.EDG.CV.RM to allow any volume to be updated.

```
 |                      |—PARMLIB_option—|    |
                            |——RETENTIONMETHOD(————————————————————)—|
                            |                                        |
 ▶▶—— ADDVOLUME— VOLSER ——————————————————————————————————————————————————————◀◀
      AV                    |                                        |
                            |——RETENTIONMETHOD(————————————————————)—|
                               RM                |—EXPDT——————————|
                                                 |—VRSEL——————————|


                              |                    |—PARMLIB_option—|    |
                              |——RETENTIONMETHOD(————————————————————)—|
                              |                                        |
 ▶▶—— CHANGEVOLUME— VOLSER ———————————————————————————————————————————————————◀◀
      CV                      |                                        |
                              |——RETENTIONMETHOD(————————————————————)—|
                                 RM                |—EXPDT——————————|
                                                   |—VRSEL——————————|
```

Figure 5-7 RETENTIONMETHOD operand command syntax

The command has the following parameters:

- EXPDT: Sets the retention method for a tape volume set to be based on EXPDT. Data sets and volumes managed by this retention method are never processed by VRSEL inventory management.
- VRSEL: Sets the retention method for a tape volume set to be VRSEL. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to vital record specifications. If a match is found, the system determines whether the data set or volumes are retained by VRS.

Consideration: When you manually define or change a volume that is a start of a volume set, you can specify the retention method for the volume set. All other volumes added to the set are set to the same retention method. If you do not specify a retention method, DFSMSrmm uses the default retention method specified by the RETENTIONMETHOD option in EDGRMMxx.

5.2.7 Using the SEARCHVOLUME subcommand

Using the SEARCHVOLUME subcommand with the RETENTIONMETHOD operand specifies a list limited to volumes that have a specified retention method. Specify EXPDT to select volumes with the EXPDT retention method. Specify VRSEL to select volumes with the

VRSEL retention method. The new RETENTIONMETHOD operand syntax is shown in Figure 5-8. This option enables DFSMSrmm inventory management to attempt to match data sets and volumes to vital record specifications. If a match is found, the system determines whether the data set or volumes are retained by VRS.

```
▸▸— SEARCHVOLUME— VOLUME(VOLSER) ——————————————————————————————————————◂◂
    SV                                  |                              |
                                        |—RETENTIONMETHOD(———————————)—|
                                           RM              |—EXPDT—|
                                                            |—VRSEL—|
```

Figure 5-8 RETENTIONMETHOD operand command syntax

The command has the following parameters:

- EXPDT: Limits the search to volumes defined with the EXPDT retention method.
- VRSEL: Limits the search to volumes defined with the VRSEL retention method.

Example of searching for all volumes

Figure 5-9 shows how to use the RMM SEARCHVOLUME subcommand to get a list of all volumes owned by user SCHLUM, as shown in Figure 5-10.

> **Remember:** If you do not specify the RETENTIONMETHOD operand DFSMSrmm searches all volumes, regardless of the retention method assigned to a volume.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(SCHLUM)
```

Figure 5-9 SEARCHVOLUME without the RETENTIONMETHOD operand

Figure 5-10 shows the list of all six volumes owned by user SCHLUM.

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
VT0008 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0013 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0015 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0016 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0017 SCHLUM          2011/325   2011/334   VTFM001  1     M
VT0021 SCHLUM          2011/325   2011/334   VTFM001  1     M
EDG3012I 6          ENTRIES LISTED
```

Figure 5-10 SEARCHVOLUME result without using the RETENTIONMETHOD operand

Example of limiting a search by retention method

Figure 5-11 shows how to use the RMM SEARCHVOLUME subcommand with the RETENTENMETHOD(EXPDT) operand. In the example, it is used to get a list of all volumes owned by user SCHLUM that have a retention method of EXPDT.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(SCHLUM) RETENTIONMETHOD(EXPDT)
```

Figure 5-11 SEARCHVOLUME using the RETENTIONMETHOD(EXPDT) operand

The list of volumes assigned to owner SCHLUM with a retention method of EXPDT set in the volume record are shown in Figure 5-12.

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
VT0008 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0013 SCHLUM          2011/324   2011/333   VTFM001  1     M
EDG3012I 2          ENTRIES LISTED
```

Figure 5-12 SEARCHVOLUME result using the RETENTIONMETHOD(EXPDT) operand

Now you can use the LISTVOLUME subcommand to get all the details of one of the listed volumes as shown in Figure 5-13.

```
RMM LISTVOLUME VT0008
```

Figure 5-13 LISTVOLUME subcommand

Figure 5-14 shows the result of the LISTVOLUME subcommand.

```
Volume information:
Volume = VT0008   VOL1  =         Rack  =          Owner   = SCHLUM
  Type = LOGICAL         Stacked count  = 0        Jobname = TEST9999
  Worldwide ID =                                   WORM    = N
Creation:  Date = 2009/222    Time = 12:49:45  System ID = SC70
Assign:    Date = 2011/324    Time = 16:22:36  System ID = SC70
                                               User ID   = SCHLUM
Expiration date = 2011/333   Original      = 2011/333
         set by = OCE_JFCB
Retention date  =            Set retained  = NO
Retention method= EXPDT
         set by = OCE_DEF
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume status:            Hold = N   File 1 data sets seq = 1
Status = MASTER    Availability =                 Label = SL
Current label version =           Required label version =
Media information: VTFM
Density = IDRC  Type = VT3590G2  Format = VT3590G2  Compaction = YES
Special attributes   = NONE      Vendor =
Encryption Key Labels:                                             Method:
1=
2=
Action on release:
Scratch immediate  = N  Expiry date ignore = N
Scratch = Y  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Actions pending:
Scratch = N  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Storage group  =
Loan location  =          Account = 999,POK
Old loan loc   =
Description    =
Security class =          Description =
```

Figure 5-14 LISTVOLUME VT008 results

The command has the following parameters:

- Retention method: This field specifies which retention processing is to be used. Retention method can be changed only for the first volume in a multi-volume set and is assumed for all volumes of the set. Retention method cannot be changed for a SCRATCH volume unless you specify the STATUS field as well.
- EXPDT: The retention method for a tape volume set is only based on the expiration date. Data sets and volumes managed by this retention method are never processed by VRSEL inventory management.
- Set by: Shows where the retention method is coming from. This parameter has these options:
  - UNDEFINED: Not set.
  - CMD: Retention method set by TSO subcommand.
  - CMD_DEF: Default retention method applied during subcommand processing.
  - OCE_DEF: Default retention method applied during tape recording.

- OCE_EXIT: EDG_EXIT100 sets retention method during tape recording.
- LCS_DEF: Default retention method applied for system managed tapes when called from OAM installation exits.
- CNVT: Retention method set during conversion.
- EXPORT_DEF: Default retention method applied during export processing.
- INERS_DE: Default retention method applied during tape initialization.

Example of limiting a search by retention method

Figure 5-15 shows how to use the RMM SEARCHVOLUME subcommand with the RETENTENMETHOD(VRSEL) operand. The example discovers a list of volumes owned by user SCHLUM that have a retention method of VRSEL.

```
RMM SEARCHVOLUME VOLUME(*) LIMIT(*) OWNER(SCHLUM) RETENTIONMETHOD(VRSEL)
```

Figure 5-15 SEARCHVOLUME using the RETENTIONMETHOD(VRSEL) operand

The list of volumes assigned to owner SCHLUM with a retention method of VRSEL set in the volume record is shown in Figure 5-16.

```
Volume Owner    Rack   Assigned   Expiration Location Dsets St Act   Dest.
                       date       date
------ -------- ------ ---------- ---------- -------- ----- -- ----- --------
VT0015 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0016 SCHLUM          2011/324   2011/333   VTFM001  1     M
VT0017 SCHLUM          2011/325   2011/334   VTFM001  1     M
VT0021 SCHLUM          2011/325   2011/334   VTFM001  1     M
SCHLUM
EDG3012I 4          ENTRIES LISTED
```

Figure 5-16 SEARCHVOLUME result using the RETENTIONMETHOD(VRSEL) operand

Now you can use the LISTVOLUME subcommand (Figure 5-17) to get all the details of one of the listed volumes as shown in Figure 5-18 on page 62.

```
RMM LISTVOLUME VT0021
```

Figure 5-17 LISTVOLUME subcommand

Figure 5-18 shows the result of the LISTVOLUME subcommand.

```
Volume information:
Volume = VT0021   VOL1  =          Rack  =          Owner   = SCHLUM
  Type = LOGICAL         Stacked count  = 0         Jobname = TEST9999
  Worldwide ID =                                    WORM    = N
Creation:  Date = 2009/222    Time = 12:49:45  System ID = SC70
Assign:    Date = 2011/325    Time = 10:59:17  System ID = SC70
                                               User ID   = SCHLUM
Expiration date = 2011/334   Original      = 2011/334
         set by = OCE_JFCB
Retention date  =            Set retained  = NO
Retention method= VRSEL
         set by = OCE_DEF
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume status:            Hold = N   File 1 data sets seq = 1
Status = MASTER    Availability =                Label = SL
Current label version =           Required label version =
Media information: VTFM
Density = IDRC  Type = VT3590G2  Format = VT3590G2  Compaction = YES
Special attributes   = NONE      Vendor =
Encryption Key Labels:                                           Method:
1=
2=
Action on release:
Scratch immediate  = N  Expiry date ignore = N
Scratch = Y  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Actions pending:
Scratch = N  Replace = N  Return = N  Init = N  Erase = N  Notify = N
Storage group  =
Loan location  =          Account = 999,POK
Old loan loc   =
Description    =
Security class =          Description =
```

Figure 5-18 LISTVOLUME VT0021 results

The command has the following parameters:

- Retention method: This field specifies which retention processing is to be used:
  - VRSEL: The volume is managed by using the VRSEL expiration method. DFSMSrmm inventory management attempts to match data sets and volumes to VRSs. If a match is found, it determines whether the data sets or volumes are to be retained by VRS.
  - Set by: Shows where the retention method is coming from.
  - UNDEFINED: Not set.
  - CMD: Retention method set by TSO subcommand.
  - CMD_DEF: Default retention method applied during subcommand processing.
  - OCE_DEF: Default retention method applied during tape recording.
  - OCE_EXIT: EDG_EXIT100 sets retention method during tape recording.
  - LCS_DEF: Default retention method applied for system managed tapes when called from OAM installation exits.
  - CNVT: Retention method set during conversion.

– EXPORT_DEF: Default retention method applied during export processing.
– INERS_DE: Default retention method applied during tape initialization.

5.3 Excluding data sets from VRSEL processing

You can use the EDG_EXIT100 installation exit to exclude specific data sets from DFSMSrmm VRSEL processing as they are created or rewritten. You can specify this setting for any data set. However, DFSMSrmm ignores the request unless the data set is on a volume that is managed by the VRSEL retention method. The data set VRSELEXCLUDE attribute is set for all data sets on volumes managed by the EXPDT retention method, and is not affected by this support.

When DFSMSrmm excludes a data set from VRSEL processing, it ensures that the data set vital record attribute is reset and the retention date is set to current date. The matching VRS information is left unchanged. Internally, DFSMSrmm uses the VRSELEXCLUDE indication for data sets on volumes managed by the EXPDT retention method.

5.3.1 Subcommands for VRSELEXCLUDE parameters

All data in the DFSMSrmm inventory is managed by dynamic VRS policies. With z/OS V1.13, DFSMSrmm provides a new operand VRSELEXCLUDE for the RMM CHANGEDATASET and SEARCHDATASET subcommands.

5.3.2 Syntax format of the VRSELEXCLUDE operand

Use this operand to override DFSMSrmm VRSEL processing. You can specify this operand for any data set on a volume managed by the VRSEL retention method. If VRSELEXCLUDE(YES) is specified for a data set already retained as a vital record, its vital record attribute is reset. The retention date is set to the current date. The data set VRSELEXCLUDE attribute is set to YES for all data sets on volumes managed by the EXPDT retention method.

When a data set spans volumes, set the VRSELEXCLUDE attribute for each data set record. You need one data set record for each of the volumes the data set is on.

The correct syntax to set the vital record selection processing to YES or NO is shown in Figure 5-19.

```
▸▸— CHANGEDATASET– dsn –VOLUME(vol) ————————————————————————————————◂◂
    CD                  VO          |                              |
                                    |—VRSELEXCLUDE(——————————)–|
                                       VX           |–NO——|
                                                    |–YES–|
```

Figure 5-19 RETENTIONMETHOD operand command syntax

The command has the following parameters:

- NO: Ensures that a data set is included in VRSEL processing.
- YES: Excludes a data set from VRSEL processing.

Attention: Authorization can be either of these settings:

- Based on STGADMIN.EDG.MASTER access if the resource below is not defined
- UPDATE access to STGADMIN.EDG.CD.VX to allow any data set to be updated.

5.3.3 Using EDG_EXIT100 to exclude data sets from VRSEL support

A new option is provided by using EDG_EXIT100 to override DFSMSrmm VRSEL processing for specific data sets as they are created or rewritten. You can specify this option for any data set. However, DFSMSrmm ignores your request unless the data set is on a volume that is managed by VRSEL retention method. The data set VRSELEXCLUDE attribute is set for all data sets on volumes managed by the EXPDT retention method, and is not affected by this support. If a data set is already retained as a vital record, the vital record attribute is reset and the retention date set to the current date.

When DFSMSrmm accepts your request to exclude a data set from VRSEL, processing ensures that the data set vital record attribute is reset. The retention date is set to current date, and the matching VRS information is left unchanged.

You can use the EDG_EXIT100 installation exit to set the VRSELEXCLUDE attribute for VRSEL managed data sets. You can do so at OPEN time when you create a tape data set or rewrite existing data sets.

To update the sample EDGUX100 exit module based on your retention requirement, perform these tasks:

1. Copy the sample EDGUX100 exit module and use the copy as a base for your exit module.
   a. Update the exit module. Perform your processing only when the PL100_CAN_VRSELEXCLUDE bit is set to B'1'.
   b. Set PL100_SET_VRSELEXCLUDE bit to B'1' for data sets. If you do not request VRSELEXCLUDE, the default for the retention method is used. If the installation exit sets PL100_SET_VRSELEXCLUDE, any VRS management value set in PL100_VRS is ignored.
   c. You do not need to set the PL100_SET_VRSELEXCLUDE bit when you also request to set the retention method to EXPDT. DFSMSrmm always sets the VRSELEXCLUDE attribute for data sets managed by the EXPDT retention method.
2. Make any other changes required such as setting the retention method when creating the first file on the tape, or clearing the EXPDT.

The sample EDGUX100 exit module includes an example of setting the VRSELEXCLUDE attribute. To use the sample EDGUX100 exit module for this function, modify the table as shown in Figure 5-20.

The order in which the table entries are listed is important. The exit scans the table until it finds the first entry where the job name, data set name and program name masks match the current request. You can change the priority of matching by changing the order of the table entries.

```
VXTAB DS OF                         START OF VRSELEXCLUDE TABLE
      SPACE 1
      DC    CL8'*'                  JOBNAME
      DC    CL44'RMMUSER.VX.*'      data sets NAME
      DC    CL8'*'                  PROGRAM NAME
      DC    CL8'SCHLUM*'            JOBNAME
      DC    CL44'RMM.ADDONS.*'      data sets NAME
      DC    CL8'*'                  PROGRAM NAME
      DC    CL8'BACK%%%%'           JOBNAME
      DC    CL44'SSC.BACK*'         data sets NAME
      DC    CL8'ADRDSSU'            PROGRAM NAME
      SPACE 1
      DC    CL8'VX END'             END OF VX TABLE MARKER
```

Figure 5-20 Sample VRSELEXCLUDE selection table

The command has the following parameters:

- JOBNAME: One-to-eight alphanumeric or national characters that include % and *.
  - **%**: Can be used to ignore a positional character in the job name.
  - *****: Can be used to ignore all remaining characters in the job name. A JOBNAME of * means that the entry applies to all jobs.
- data sets NAME: Can be up to 44 characters that follow the z/OS data set naming conventions, including% and *.
  - **%:** Can be used to ignore a positional character in the data set name.
  - *****: Can be used to ignore all remaining characters in the data set name. A data set name of * means that the entry applies to all data sets.

    The use of the character * is not the same as in the generic data set names supported by DFSMSrmm for vital records specifications and search data set masks. Here the * works like the characters *.* does in a generic data set name mask.
- PROGRAM NAME: A value of up to eight alphanumeric characters that include % and *.
  - **%:** Can be used to ignore a positional character in the program name.
  - *****: Can be used to ignore all remaining characters in the program name. A program name of * means that the entry applies to all programs.

Use JCL to compile and link the updated EDGUX100 user exit, as shown in Figure 5-21.

```
//ASMCL    PROC SM=,LM=
//C        EXEC PGM=ASMA90,PARM='NODECK,XREF(SHORT),LINECOUNT(58)'
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//SYSLIB   DD   DSN=SYS1.MACLIB,DISP=SHR
//         DD   DSN=SYS1.MODGEN,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(7040,400)
//SYSUT2   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSUT3   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSIN    DD   DSN=RMM.ADDONS.SOURCE(&SM),DISP=SHR
//L        EXEC PGM=HEWL,PARM='LIST,MAP,XREF,RENT,REUS,',COND=(4,LT,C)
//SYSLMOD  DD   DSN=SYS1.SANDBOX.MIGLIB(&LM),DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSUT1   DD   UNIT=SYSDA,SPACE=(3520,400)
//SYSLIN   DD   DSN=RMM.ADDONS.OBJ(&LM),DISP=SHR
//         PEND
//COMPLNKO EXEC ASMCL,LM=EDGUX100,SM=EDGUX100
```

Figure 5-21 Compiling and link updating EDGUX100 user exit

To activate or change the current EDGUX100 user exit in V1R13, use the command shown in Figure 5-22.

```
SETPROG EXIT,REFRESH,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100
```

Figure 5-22 SETPROG command syntax

The correct sequence to replace an existing exit, before z/OS V1.12, is shown in Figure 5-23.

```
MODIFY DFRMM,QUIESCE

SETPROG EXIT,DELETE,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100

SETPROG EXIT,ADD,EXITNAME=EDG_EXIT100,MODNAME=EDGUX100

MODIFY DFRMM,M=nn
```

Figure 5-23 SETPROG command syntax sequence before z/OS V1.12

The command has the following parameters:

- ADD: Adds an exit routine to an exit.
- REPLACE: Replaces an exit routine for an exit.
- DELETE: Deletes an exit routine from an exit.
- EXITNAME: The 1-16 character name of the exit.
- MODNAME: The 1-8 character name of the exit routine. If DSNAME is not specified, the system tries to locate the exit routine by using the LPA, the LNKLST concatenation, and the nucleus.

Tip: DFSMSrmm provides a second sample for EDGUX100, called EDGCVRSX. It is different from the EDGUX100 sample. The special date, retention method, VRSELEXCLUDE, and pooling function are table driven. You can change that table dynamically. For more information about using EDGCVRSX for EDGUX100, see SAMPLIB member EDGCMM01 and the IBM Redbooks DFSMSrmm Primer.

5.3.4 Using the SEARCHDATASET subcommand

Using the SEARCHDATASET subcommand with the VRSELEXCLUDE operand lists data sets that have the specified VRSELEXCLUDE value. Specify YES to search for data sets that are excluded from VRSEL processing. Specify NO to ensure that a data set is included in VRSEL processing. The new VRSELEXCLUDE operand syntax is shown in Figure 5-24.

```
                                         |─VOLUME(*)────────────|
                 |─DSNAME(*)─────────|   |─OWNER(cmd_issuer_id)─|
                 |                   |   |                      |
▸▸── SEARCHDATASET──────────────────────────────────────────────────────▸
     SD          |                   |   |                      |
                 |─DSNAME(dsn_mask)─|    |─VOLUME(VOLSER)───────|
                   DS                    | VO                   |
                                         |─OWNER(owner_id)──────|
                                           OW
  |─FILESEQ(1)────────────────|
  |                           |
▸───────────────────────────────────────────────────────────────────────▸
  |                           |     |                           |
  |─FILESEQ(physical_file_seq)─|    |──VRSELEXCLUDE(──────────)─|
    SEQ                               VX           |─NO──|
                                                   |─YES─|
```

Figure 5-24 VRSELEXCLUDE operand command syntax

The command has the following parameters:

- NO: To search for data sets that are included in VRSEL processing.
- YES: To search for data sets excluded from VRSEL processing.

Example of searching for all volumes

Figure 5-25 shows how to use the RMM SEARCHDATASET subcommand to get a list of all data sets owned by user SCHLUM.

Remember: If you do not specify the VRSELEXCLUDE operand, DFSMSrmm searches all data sets, regardless of the vital record selection criteria set for a data set.

```
RMM SEARCHDATASET LIMIT(*) OWNER(SCHLUM)
```

Figure 5-25 SEARCHDATASET without the VRSELEXCLUDE operand

Figure 5-26 lists all 10 data sets owned by user SCHLUM.

```
Data set name                                Volume Owner    Create date Seq
-------------------------------------------- ------ -------- ----------- -----
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0008 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0013 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0015 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0016 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0017 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0021 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0022 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0023 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0024 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0025 SCHLUM   2011/325    1
EDG3012I 10         ENTRIES LISTED
```

Figure 5-26 SEARCHDATASET result without using the VRSELEXCLUDE operand

Example of searching for excluded volumes

Figure 5-27 shows using the RMM SEARCHDATASET subcommand to list all data sets owned by user SCHLUM that have the VRSELEXCLUDE set to YES.

```
RMM SEARCHDATASET LIMIT(*) OWNER(SCHLUM) VRSELEXCLUDE(YES)
```

Figure 5-27 SEARCHDATASET using VRSELEXCLUDE(YES) operand

Figure 5-28 shows the eight data sets that have the VRSELEXCLUDE set to YES.

```
Data set name                                Volume Owner    Create date Seq
-------------------------------------------- ------ -------- ----------- ----
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0008 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0013 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0017 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0021 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0022 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0023 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0024 SCHLUM   2011/325    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0025 SCHLUM   2011/325    1
EDG3012I 8          ENTRIES LISTED
```

Figure 5-28 SEARCHDATASET using VRSELEXCLUDE(YES) operand

Now you can use the LISTDATASET subcommand to get all the details of one of the listed data sets, as shown in Figure 5-29.

```
RMM LD 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG' VOL(VT0025)
```

Figure 5-29 LISTDATASET subcommand

Figure 5-30 shows the result of the LISTDATASET subcommand.

```
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume        = VT0025              Physical file sequence number = 1
Owner         = SCHLUM                          data sets sequence = 1
Create date   = 2011/325   Create time = 19:35:33 System ID        = SC70
Expiration date       = 2011/325   Original expir. date  =
         set by       = OCE_DEF
Block size            = 80         Block count           = 1
Data set size(KB)     = 1
Physical size(KB)     = 0          Compression           = 0.00
Percent of volume     = 0          Total block count     = 2
Logical Record Length = 0          Record Format         = F
Date last written     = 2011/325   Date last read        = 2011/325
Job name              = TEST9999   Last job name         = TEST9999
Step name             = STEP03     Last step name        = STEP03
Program name          = EOVTEST    Last program name     = EOVTEST
DD name               = OUT        Last DD name          = OUT
Device number         = 0406       Last Device number    = 0406
Management class      =            VRS management value  =
Storage group         =            VRS retention date    =
Storage class         =            VRS retained          = NO
Data class            =            Closed by Abend       = NO
                                   Deleted               = NO
VRSEL exclude         = YES        Catalog status        = UNKNOWN
Primary VRS details:
       Name           =
       Job name       =            Type                  =
       Subchain NAME  =            Subchain start date   =
Secondary VRS details:
       Value or class =
       Job name       =
       Subchain NAME  =            Subchain start date   =
Security Class   =          Description   =
BES key index    = 0

Last Change information:
Date             = 2011/325    Time = 20:25:03  System  = SC70
User change date = 2011/325    Time = 20:25:03  User ID = SCHLUM
```

Figure 5-30 LISTDATASET results

The command has the following parameters:

- Retention Date: The date a data set is no longer retained by the current VRS. For data sets not retained by a VRS, the retention date is either the date no longer retained by VRS, or null. This parameter has these possible values:
  - Calendar date: The date calculated by VRSEL processing in the currently selected date format.
  - WHILECATLG: Data set is retained by a VRS with WHILECATALOG.
  - CYCL/ccccc: Data set is retained by a VRS defined with CYCLES, where ccccc is the COUNT of cycles.

  - CATRETPD: Data set is retained by a VRS defined with WHILECATALOG and the data set is not cataloged, but the CATRETPD PARMLIB option applies.
  - PERMANENT: Data set is retained by a VRS defined with DAYS COUNT(99999).
- VRSEL exclude: This field specifies whether the data set is excluded from vital record processing. The VRSELEXCLUDE attribute is set to YES for all data sets on volumes managed by the EXPDT retention method. It can optionally be set to YES for any data set on a volume which is managed by the VRSEL retention method.

  When a data set spans volumes, set the VRSELEXCLUDE attribute for each data set record. You need one data set record for each of the volumes the data set is on.

  This parameter has these possible values:
  - YES: Excludes a data set from VRSEL processing
  - NO: Ensures that a data set is included in VRSEL processing

> **Tip:** If a data set is not the vital record selected, the retention date field is always empty.

Example of searching for non-excluded volumes

Figure 5-31 shows using the RMM SEARCHDATASET subcommand to list all data sets owned by user SCHLUM that have the VRSELEXCLUDE set to NO.

```
RMM SEARCHDATASET LIMIT(*) OWNER(SCHLUM) VRSELEXCLUDE(NO)
```

Figure 5-31 SEARCHDATASET using VRSELEXCLUDE(NO) operand

Figure 5-32 shows that only two data sets have the VRSELEXCLUDE set to NO.

```
Data set name                                Volume Owner    Create date Seq
-------------------------------------------- ------ -------- ----------- ----
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0015 SCHLUM   2011/324    1
RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0016 SCHLUM   2011/324    1
EDG3012I 2          ENTRIES LISTED
```

Figure 5-32 SEARCHDATASET using the VRSELEXCLUDE(NO) operand

Now you can use the LISTDATASET subcommand to get all the details of one of the listed data sets as shown in Figure 5-33.

```
RMM LD 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG' VOL(VT0015)
```

Figure 5-33 LISTDATASET subcommand

Figure 5-34 shows the result of the LISTDATASET subcommand.

```
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume        = VT0015               Physical file sequence number = 1
Owner         = SCHLUM                            data sets sequence = 1
Create date   = 2011/324   Create time = 19:41:58 System ID         = SC70
Expiration date       = 2011/333   Original expir. date  = 2011/333
         set by       = OCE_JFCB
Block size            = 80         Block count           = 1
Data set size(KB)     = 1
Physical size(KB)     = 0          Compression           = 0.00
Percent of volume     = 0          Total block count     = 1
Logical Record Length = 0          Record Format         = F
Date last written     = 2011/324   Date last read        = 2011/324
Job name              = TEST9999   Last job name         = TEST9999
Step name             = STEP03     Last step name        = STEP03
Program name          = EOVTEST    Last program name     = EOVTEST
DD name               = OUT        Last DD name          = OUT
Device number         = 044C       Last Device number    = 044C
Management class      =            VRS management value  =
Storage group         =            VRS retention date    = PERMANENT
Storage class         =            VRS retained          = YES
Data class            =            Closed by Abend       = NO
                                   Deleted               = NO
VRSEL exclude         = NO         Catalog status        = UNKNOWN
Primary VRS details:
       Name           = RMM.F13002.**
       Job name       =            Type                  = data sets
       Subchain NAME  =            Subchain start date   =
Secondary VRS details:
       Value or class =
       Job name       =
       Subchain NAME  =            Subchain start date   =
Security Class   =          Description   =
BES key index    = 0

Last Change information:
Date             = 2011/325    Time = 20:25:03  System  = SC70
User change date = 2011/325    Time = 20:25:03  User ID = SCHLUM
```

Figure 5-34 LISTDATASET results

The command has the following parameters:

The command has the following parameters:

- Retention Date: The date a data set is no longer retained by the current VRS. For data sets not retained by a VRS, the retention date is either the date no longer retained by VRS, or null. This parameter has these possible values:
  - Calendar date: The date calculated by VRSEL processing in the currently selected date format.
  - WHILECATLG: Data set is retained by a VRS with WHILECATALOG.
  - CYCL/ccccc: Data set is retained by a VRS defined with CYCLES, where ccccc is the COUNT of cycles.

  - CATRETPD: Data set is retained by a VRS defined with WHILECATALOG and the data set is not cataloged, but the CATRETPD PARMLIB option applies.
  - PERMANENT: Data set is retained by a VRS defined with DAYS COUNT(99999).
- VRSEL exclude: This field specifies whether the data set is excluded from vital record processing. The VRSELEXCLUDE attribute is set to YES for all data sets on volumes managed by the EXPDT retention method. It can optionally be set to YES for any data set on a volume which is managed by the VRSEL retention method.

  When a data set spans volumes, set the VRSELEXCLUDE attribute for each data set record. You need one data set record for each of the volumes the data set is on.

  This parameter has these possible values:

  - YES: Excludes a data set from VRSEL processing
  - NO: Ensures that a data set is included in VRSEL processing

5.3.5 Inventory Management VRSEL/EXPROC Processing

In release z/OS V1.13, the inventory management processing has the following changes:

- You do not need to run VRSEL processing unless any volumes are defined with the VRSEL retention method. Only EXPROC processing is required to handle expiration of all volumes managed by the EXPDT retention method.
- EXPROC processing provides a summary of volumes by retention method.
- The expiration date of volumes is set during OPEN processing. For volumes managed by the EXPDT retention method, no special considerations exist for open data sets. They are managed based on the volume EXPDT.
- For volumes managed by the EXPDT retention method, no special considerations exist for data sets closed by ABEND processing or that are DELETED. They are managed based on the volume EXPDT.
- Volumes managed by the EXPDT retention method are included only in the EXPDTDROP limit. VRSRETAIN and VRSDROP limits apply only to volumes managed by VRSEL retention method.

For inventory management (VRSEL), one new message EDG2246I has been added. For expiration processing (EXPROC), no new messages are created. However, existing messages are expanded and adapted with new/additional inserts as shown in Figure 5-35 on page 73.

```
EDG6001I INVENTORY MANAGEMENT STARTING ON 2011/326 AT 13:26:39 -
               PARAMETERS IN USE ARE
         DATEFORM(J),VRSEL,EXPROC,BACKUP(AMS)
EDG2309I THE PARMLIB OPTIONS CURRENTLY IN USE ARE
         VRSEL(NEW)
         VRSJOBNAME(2)
         VRSMIN(1,INFO)
         VRSCHANGE(INFO)
         VRSDROP(PERCENT(10),INFO) VRSRETAIN(PERCENT(80),INFO)
               EXPDTDROP(PERCENT(10),INFO)
         SMSTAPE(PURGE(ASIS) UPDATE(EXITS,SCRATCH,COMMAND))
         CATRETPD(6)
         UNCATALOG(Y)
         TPRACF(N)
         NOTIFY(N)
         SYSID(SC64)
         CATSYSID()
         RETAINBY(SET)
         MOVEBY(SET)
         GDG(CYCLEBY(GENERATION),DUPLICATE(BUMP))
EDG2229I NUMBER OF VRS RECORDS READ IS 3
EDG2238I NUMBER OF UNUSED VRS RECORDS IS 1
EDG2246I NUMBER OF data sets RECORDS EXCLUDED FROM VRSEL =        8   7%
EDG2242I INITIAL NUMBER OF VRS RETAINED VOLUMES        =       107  33%
EDG2244I NUMBER OF VRS RETAINED VOLUMES TO BE DROPPED  =         0   0%
EDG2243I INITIAL NUMBER OF NEWLY ASSIGNED VOLUMES      =         2   1%
EDG2245I NUMBER OF NEWLY ASSIGNED VOLUMES TO BE RETAINED=        0   0%
EDG2444I EXIT PROCESSING DISABLED FOR THIS EXPROC RUN -
               NO ACTIVE EXIT MODULE FOR EXIT EDG_EXIT200
EDG2427I INITIAL NUMBER OF EXPDT RETAINED VOLUMES      =         6   2%
EDG2428I NUMBER OF EXPDT RETAINED VOLUMES TO BE DROPPED =        2  33%
EDG2420I PHYSICAL VOLUMES READ                         =        30   9%
EDG2420I LOGICAL  VOLUMES READ                         =       300  91%
EDG2420I RM_VRSEL VOLUMES READ                         =       322  98%
EDG2420I RM_EXPDT VOLUMES READ                         =         8   2%
EDG2420I TOTAL    VOLUMES READ                         =       330 100%
EDG2421I PHYSICAL VOLUMES UPDATED                      =         4  13%
EDG2421I LOGICAL  VOLUMES UPDATED                      =         4   1%
EDG2421I RM_VRSEL VOLUMES UPDATED                      =         6   2%
EDG2421I RM_EXPDT VOLUMES UPDATED                      =         2  25%
EDG2421I TOTAL    VOLUMES UPDATED                      =         8   2%
EDG2435I PHYSICAL VOLUMES SELECTED FOR EXPROC          =        30 100%
EDG2435I LOGICAL  VOLUMES SELECTED FOR EXPROC          =       300 100%
EDG2435I RM_VRSEL VOLUMES SELECTED FOR EXPROC          =       322 100%
EDG2435I RM_EXPDT VOLUMES SELECTED FOR EXPROC          =         8 100%
EDG2435I TOTAL    VOLUMES SELECTED FOR EXPROC          =       330 100%
EDG2424I LOGICAL  VOLUMES SET PENDING RELEASE          =         4   1%
EDG2424I RM_VRSEL VOLUMES SET PENDING RELEASE          =         2   1%
EDG2424I RM_EXPDT VOLUMES SET PENDING RELEASE          =         2  25%
EDG2424I TOTAL    VOLUMES SET PENDING RELEASE          =         4   1%
EDG2425I PHYSICAL VOLUMES RETURNED TO SCRATCH          =         4  13%
EDG2425I LOGICAL  VOLUMES RETURNED TO SCRATCH          =         2   1%
EDG2425I RM_VRSEL VOLUMES RETURNED TO SCRATCH          =         4   1%
EDG2425I RM_EXPDT VOLUMES RETURNED TO SCRATCH          =         2  25%
EDG2425I TOTAL    VOLUMES RETURNED TO SCRATCH          =         6   2%
EDG2426I PHYSICAL VOLUMES - SCRATCH RECORDS WRITTEN    =         2  10%
EDG2426I RM_EXPDT VOLUMES - SCRATCH RECORDS WRITTEN    =         2  14%
EDG2426I TOTAL VOLUMES    - SCRATCH RECORDS WRITTEN    =         2   6%
EDG2429I MAIN INVENTORY MANAGEMENT UPDATES HAVE COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK VRSEL    COMPLETED SUCCESSFULLY
EDG2307I INVENTORY MANAGEMENT TASK EXPROC   COMPLETED SUCCESSFULLY
EDG6426I CONTROL data sets AND JOURNAL BACKUP SUCCESSFUL
EDG6901I UTILITY EDGHSKP COMPLETED WITH RETURN CODE 0
```

Figure 5-35 EDGHSKP messages

The command has the following parameters:

- EDG2420I *volume_type* VOLUMES READ = *number percent%*

  Explanation: DFSMSrmm issues this message when the number of volumes read is greater than zero.

  In the message text, the *volume_type* can be one of these settings:

  - PHYSICAL
  - LOGICAL
  - STACKED
  - RM_VRSEL
  - RM_EXPDT
  - TOTAL

  The *number* is the number of *volume_type* volumes read during inventory management. The *percent* is the percentage of total volumes read by DFSMSrmm inventory management processing.

  – System action: Processing continues.

  – Operator response: None.

  – Source: DFSMSrmm

  – Detecting Module: EDGMUPD

  – Routing Code: 11

  – Descriptor Code: 7

- EDG2421I *volume_type* VOLUMES UPDATED = *number percent%*

  Explanation: DFSMSrmm issues this message when the number of volumes updated is greater than zero.

  In the message text, the *volume_type* can be one of these settings:

  - PHYSICAL
  - LOGICAL
  - STACKED
  - RM_VRSEL
  - RM_EXPDT
  - TOTAL

  The *number* is the number of *volume_type* volumes that were updated in the DFSMSrmm control data set. The *percent* is the percentage of the *volume_type* volumes read by DFSMSrmm inventory management processing.

  – System action: Processing continues.

  – Operator response: None.

  – Source: DFSMSrmm

  – Detecting Module: EDGMUPD

  – Routing Code: 11

  – Descriptor Code: 7

- EDG2424I *volume_type* TOTAL VOLUMES, THIS RUN, SET PENDING RELEASE = *number percent%*

  Explanation: This message is issued for information only.
  In the message text, the *volume_type* can be one of these settings:

  - PHYSICAL
  - LOGICAL

- STACKED
- RM_VRSEL
- RM_EXPDT
- TOTAL

The *number* is the number of *volume_type* volumes set pending release in this run of inventory management. The *percent* is the percentage of *volume_type* volumes selected for EXPROC processing (see message EDG2435I).

– System action: Processing continues.

– Operator response: None.

– Source: DFSMSrmm

– Detecting Module: EDGMUPD

– Routing Code: 11

– Descriptor Code: 7

► EDG2425I *volume_type* TOTAL VOLUMES RETURNED TO SCRATCH = *number percent%*

Explanation: This message is issued for information only.

In the message text, the *volume_type* can be one of these settings:

- PHYSICAL
- LOGICAL
- STACKED
- RM_VRSEL
- RM_EXPDT
- TOTAL

The *number* is the number of *volume_type* volumes returned to scratch status after all release actions are completed. The *percent* is the percentage of *volume_type* volumes selected for EXPROC processing (see message EDG2435I).

– System action: Processing continues.

– Operator response: None.

– Source: DFSMSrmm

– Detecting Module: EDGMUPD

– Routing Code: 11

– Descriptor Code: 7

► EDG2426I *volume_type* TOTAL NUMBER OF SCRATCH RECORDS WRITTEN = *number percent%*

Explanation: DFSMSrmm issues this message to the MESSAGE file when EXPROC processing for system managed volumes requests deferred processing of volumes to scratch.

In the message text, the *volume_type* can be one of these settings:

- PHYSICAL
- LOGICAL
- STACKED
- RM_VRSEL
- RM_EXPDT
- TOTAL

The *number* is the number of system-managed *volume_type* volumes that are ready for returning to scratch on this system. A record has been written about these volumes to the EDGSPLCS output file. The *percent* is the percentage of *volume_type* volumes selected for EXPROC processing (see message EDG2435I).

– System action: DFSMSrmm inventory management processing continues.

– Operator response: None.

– System programmer response: Run the EDGSPLCS utility to process the scratch statements produced by EDGHSKP EXPROC processing for system managed volumes.

– Source: DFSMSrmm

– Detecting Module: EDGMUPD

- EDG2435I *volume_type* VOLUMES SELECTED FOR EXPROC = *number percent*%

  Explanation: DFSMSrmm issues this message to the MESSAGE file during inventory management EXPROC processing. This message is issued for information only.

  In the message text, the *volume_type* can be one of these settings:

  - PHYSICAL
  - LOGICAL
  - STACKED
  - RM_VRSEL
  - RM_EXPDT
  - TOTAL

  The *number* is the number of volume_type volumes read during inventory management. The *percent* is the percentage of total volumes read by DFSMSrmm inventory management processing.

– System action: DFSMSrmm inventory management processing continues.

– Operator response: None.

– System programmer response: None.

– Source: DFSMSrmm

– Detecting Module: EDGMUPD

5.4 Data set attribute COPYFROM function

You might need to copy or move tape data from one tape to another. You can move data by using a tape copy utility that uses DFSMSrmm services to correctly copy and restack tape data sets. This process preserves the data set attribute settings, and can optionally set retention periods for the source data set, copy data set, or both. If you use your own job control language (JCL) and the IEBGENER utility, you can use the RMM CHANGEDATASET subcommand with the COPYFROM operand. This subcommand copies the source data set attributes. However, a tape copy application can select the method used to communicate with DFSMSrmm.

After the target data set is created, the tape copy application can use the RMM TSO CHANGEDATASET subcommand with the COPYFROM operand. This command copies all applicable attributes from the source data set to the target data set. DFSMSrmm determines which attributes are to be copied.

During the creation of the target data set, tape copy application programs can use the EDG_EXIT100 installation exit. This exit copies all applicable attributes from the source data set to the target data set.

You can ensure that all applicable data set attributes are copied with DFSMSrmm COPYFROM support. This process avoids using multiple RMM subcommands to modify only the attributes supported using the subcommands.

5.4.1 EDG_EXIT100 Tape Copy application support

A tape copy application can use the EDGPL100 macro to start the installation exit EDG_EXIT100. This exit copies the data set attributes from the source data set to the copy data set during OPEN processing. EDG_EXIT100 can notify DFSMSrmm that the data set being created is being copied from another. During OPEN processing, the exit can identify the source data set from which DFSMSrmm will obtain all existing data set attributes. The attributes of this data set are used for the target data set. DFSMSrmm end-of-volume (EOV) processing ensures that the attributes are copied to all target data set records when the output data set becomes a multivolume data set.

5.4.2 Using the CHANGEDATASET COPYFROM subcommand

For a target data set that has already been created, a tape copy application can use the CHANGEDATASET COPYFROM subcommand. This subcommand copies all applicable attributes from the source data set to the target data set. DFSMSrmm determines which attributes are to be copied. Retention of the source data set can be specifically set. Retention of the target volumes and data sets can be selected at the volume set level, and even switched between VRSEL and EXPDT retention methods.

You can use the CHANGEDATASET COPYFROM subcommand to ensure that all applicable data set attributes are copied. This process avoids using multiple RMM subcommands to modify only some of the attributes.

The CHANGEDATASET COPYFROM subcommand identifies a single volume data set or any part of a multivolume data set. Validation is done to ensure that the source and target data sets have the same recording format and record length. You use the CHANGEDATASET subcommand once for each target data set record. For multivolume data sets, this means that you must issue the subcommand once for each volume the target data set is written on.

Attention: Authorization requires one of these settings:

- Based on STGADMIN.EDG.MASTER access, if the resource below is not defined.
- READ access to STGADMIN.EDG.CD.COPYFROM.*dsname* to copy attributes and update retention for identically named data sets.
- UPDATE access to STGADMIN.EDG.COPYFROM.*dsname* to copy attributes and update retention for any two data set records.

Figure 5-36 shows the syntax of the CHANGEDATASET subcommand with the new COPYFROM operands.

```
                                              |-FILESEQ(1)-----------------|
                                              |                            |
>>-- CHANGEDATASET- dsn -VOLUME(vol) -------------------------------------------->
     CD                                       |                            |
                                              |-FILESEQ(physical_file_seq)-|
                                                SEQ

                                    |-FILESEQ(1)-----------------|
                                    |                            |
> ----COPYFROM(- VOLUME(VOLSER) ------------------------------------------------->
                                    |                            |
                                    |-FILESEQ(physical_file_seq)-|
                                      SEQ

      |-DSNAME(dsn)---------|
      |                     |
> -------------------------------------------------------------------) ------><
      |                     |  |               |   |              |
      |-DSNAME(old_dsname)-|  |-RETPD(days)-|   |-VRSELEXCLUDE-|
        DSN                                          VX
```

Figure 5-36 CHANGEDATASET COPYFROM command syntax

The command has the following parameters:

- **COPYFROM**: Specifies a source data set whose attributes are to be copied when creating the metadata for the target data set. These suboperands can be used:
  - DSN(*olddsname*): Identifies the source data set record from which attributes are to be copied. You can optionally use a different data set name as the target. The default is that the *old_dsname* matches the data set name.

    > **Remember:** DFSMSrmm does not fold data set names to uppercase letters when you specify quoted data set names. When you specify data set names or data set name masks, be sure to specify the correct case for each character.

  - VOLUME(*oldvol*): Identifies the source data set record from which attributes are to be copied. There is no default value.

    > **Remember:** The volume processing is case-sensitive, so you must specify characters in uppercase characters.

  - FILESEQ(`oldseq`): Identifies the source data set record from which attributes are to be copied. It specifies the physical file sequence number. Use this operand to identify the relative position of the source data set on the old volume. The minimum allowable decimal value is 1. The maximum allowable decimal value is 65535. The default value is 1.

– RETPD(*days*): Causes DFSMSrmm to update the source data set record expiration date. By default, the source data set is not updated. The value can be 0 - 93000. There is no default value.
– VRSELEXCLUDE: Causes the source data set to be excluded from VRSEL processing.

Consideration: When you specify any other CHANGEDATASET subcommand operands, DFSMSrmm processes the COPYFROM operand first, then the additional operands. This process means that additional operands can specify data that overrides the attributes copied.

Data attributes not copied

Some data set attributes are not copied as shown in Table 5-1.

Table 5-1 Data attributes that are not copied

Attribute	CHANGEDATASET operand	Extract file field	REXX Variable
ABEND[a]	ABEND	RDABEND	EDG@ABND
Block count	BLKCOUNT	RDBLKCNT	EDG@BLKC
Block size	BLKSIZE	RDBLKSZ	EDG@BLKS
Catalog status	n/a	RDCAT	EDG@CTLG
Compression ratio	n/a	RDCOMP_RAT	EDG@CRAT
Data class name	DATACLASS	RDDCNAME	EDG@DC
Data set name	data_set_name	RDDSNAME	EDG@DSN
Data set sequence number	LABELNUMBER	RDLABNO	EDG@DSEQ
Data set size	n/a	RDDSSIZE, RDSIZE	EDG@DSS6
Device number	DEVNUM	RDUNITAD	EDG@DEV
End block ID	n/a	n/a	n/a
Last change date	n/a	RDLCDATE	EDG@LCDT
Last change system	n/a	RDLCSID	EDG@LCSI
Last change time	n/a	RDLCTIME	EDG@LCTM
Last change user	n/a	RDLCUID	EDG@LCID
EDG@LDEV	n/a	RDLDEVN	EDG@LDEV
Logical record length	LRECL	RDLRECL	EDG@LRCL
Owner	n/a	RDOWNDSN	EDG@OWN
Percentage of the volume	PERCENT CT	RDPERCENT	EDG@DPCT
Physical file sequence number	FILESEQ	RDDSNSEQ	EDG@FILE
Physical space used	n/a	RDPHYS_SIZE	EDG@PSZ6

Attribute	CHANGEDATASET operand	Extract file field	REXX Variable
Record format	RECFM	RDRECFM	EDG@RCFM
Start block ID	n/a	n/a	n/a
Storage class name	STORAGECLASS	RDSCNAME	EDG@SC
Storage group	n/a	RDSGNAME	EDG@SG
Tape encryption key	BESKEY	RDBESKEY	EDG@BESK
Total block count (increased size)	TOTALBLKCOUNT	RDTOTAL_BLKCNT	EDG@BLK6
Total block count	TOTALBLKCOUNT	RDTOTAL_BLKCNT	EDG@BLKT
Volume serial	VOLUME	RDVOLSER	EDG@VOL
VRSEL exclusion[b]	VRSELEXCLUDE	RDVEX EDG@VEX	RDVEX EDG@VEX

a. This attribute is copied only when it is set. The source setting is merged with the target setting.

b. This attribute is copied only when both the source and target volumes are managed by RM(VRSEL).

Retention of the target data set

In addition to specifying how the source data set is retained, you can also specify how the target data set is retained. Using the EDG_EXIT100 installation exit, select the retention method used for the target tape volume set. If the VRSEL retention method is used, you can also exclude individual data sets from VRSEL processing. For best results, ensure that the retention method of the target tape volume set matches that of the source.

When data set attributes are copied, all existing VRSEL-related attributes are copied. If the target data set is on a volume set with the VRSEL retention method, you can expect the same results as for the source data set. When both the source and target data set use the VRSEL retention method, the VRSELEXCLUDE attribute is copied. If switching from the VRSEL to the EXPDT retention method, ensure that the expiration date or retention period is set appropriately for each target data set.

5.4.3 Using EDGINERS to scan a volume

A program named EDGINERS can scan and detect when data set attributes are copied by a tape copy application. It issues message EDG6685I to help the storage administrator to understand why the mismatch is shown.

The EDGINERS JCL was used in Figure 5-37 to scan and compare the volume information with the DFSMSrmm information shown in Figure 5-38 on page 81.

```
//SCAN     EXEC PGM=EDGINERS
//SYSPRINT DD SYSOUT=*
//TAPE     DD UNIT=(VT3590,,DEFER)
//SYSIN    DD *
  SCAN VOLUME(VT0015)
//
```

Figure 5-37 JCL sample to scan a volume

Depending on the use of the COPYFROM operand, the recorded step name no longer matches the step information about the tape header as shown in Figure 5-38.

```
EDG6679I SCAN RESULTS:
 * * * * Device 0462, TAPE, VOLSER=VT0015
     VOL1 label =         VOL1VT0015
---------------------------------------------------------------------------------------------------------------------
------
Data set 0001            1...5...10...15...20...25...30...35...40...45...50...55...60...65...70...75...80
     HDR1 label =        HDR1VOLUMES.AND.CATLGVT001500010001     0113240113330000000IBM OS/VS 370
     HDR2 label =        HDR2F000800000000TEST9999/STEP03  P      00FD84
             * Tape mark
---------------------------------------------------------------------------------------------------------------------
------
        LBL VOLSER Dsname                                    Vseq Dseq  Crdate   Jobname  Step     RECF LRECL
BLKSZ
On vol   SL  VT0015                         VOLUMES.AND.CATLG 0001 00001 2011/324 TEST9999 STEP03   F
80
           Mismatch(*)                                                                    *
RMM data SL  VT0015        RMM.F13002.ON.TWO.VOLUMES.AND.CATLG 0001 00001 2011/324 TEST9999 STEPXXXX F
80
EDG6683I MISMATCH ON Step
EDG6685I data sets ATTRIBUTES WERE PREVIOUSLY CHANGED USING COPYFROM
EDG6678I VOLUME VT0015 SCAN SUCCESSFUL
EDG6631I UTILITY EDGINERS COMPLETED WITH RETURN CODE 0
```

Figure 5-38 EDGINERS scan result

The command has the following parameters:

- EDG6685I data sets ATTRIBUTES WERE PREVIOUSLY CHANGED USING COPYFROM

  Explanation: This message contains the results of the comparison between tape label contents and the information defined to DFSMSrmm. This comparison is created during a scan request to the operator console and SYSPRINT file.

  – System action: Processing continues.
  – Operator response: Use this message to help you understand the reason for the reported mismatches.
  – System programmer response: None
  – Storage Administrator Response: When reviewing the message contents for mismatches between the volume and DFSMSrmm information, consider that the DFSMSrmm information has been altered. It might have been altered by a tape copy application or by the DFSMSrmm CHANGEDATASET subcommand using COPYFROM.
  – Source: DFSMSrmm
  – Detecting Module: EDGINERS
  – Routing Code: 3,5,11

5.5 TVEXTPURGE extra days

With z/OS V1R12 DFSMSrmm, the parameter TVEXTPURGE had the options RELEASE, EXPIRE, and NONE. With z/OS V1.13, DFSMSrmm has a new option for the EXPIRE(*days*). If tapes are expired by using the EDGTVEXT HSM exit, extra days for retention can be defined only with no additional processing.

5.5.1 Use of extra days

You can use DFSMSrmm to release DFSMShsm tapes that are requested to be purged by DFSMShsm. You can also specify that DFSMSrmm retain a tape for a few days after its expiration date has been reached. By default, the expiration date protection for DFSMShsm tapes is done by DFSMShsm. DFSMShsm uses 1999/365 as the expiration date for permanent retention. To enable extra days retention for purged DFSMShsm tape volumes, use the TVEXTPURGE(EXPIRE(*days*)) option or set retention options in the vital record specifications. These specifications are used to retain the tape volumes.

5.5.2 EDGRMMxx PARMLIB member

This member specifies how DFSMSrmm processes volumes are purged by callers of EDGTVEXT or EDGDFSMShsm. The TVEXTPURGE operand specifies how you want to handle the release of DFSMShsm tape volumes, by using the new specification shown in Figure 5-39.

```
                              |─TVEXITPURGE(RELEASE)──────────────────|
                              |                                       |
▸ ──OPTION──────────────────────────────────────────────────────────────────────◂◂
                              |                                       |
                              |─TVEXITPURGE(--──────────────────────)─|
                                              |                     |
                                              |        |─0───|      |
                                              |─EXPIRE(───────────)─|
                                              |        |─days─|     |
                                              |─NONE────────────────|
```

Figure 5-39 EDGRMMnn TVEXTPURGE option

The command has the following parameters:

- EXPIRE(*days*): Use the EXPIRE(*days*) option to set the volume expiration date to the current date plus *days* for volumes to be purged. Use the EXPIRE(*days*) parameter when you use the EXPDT retention method, using days as a way to delay expiration of the volume. When using the VRSEL retention method, you can optionally use this operand in combination with vital record specifications that use the UNTILEXPIRED retention type. Apply EXPIRE(*days*) to set a new volume EXPDT. Then run VRSEL to extend retention by using the extra days retention type. For example, specify EXPIRE(0) when using a VRS with extra days retention. You can also use a non-zero EXPIRE(*days*) value and avoid using an extra days retention VRS.

  The *days* parameter is the number of days for which DFSMSrmm retains the volume before considering it for release. The value is a one to four digit decimal number and is added to today's date to compute the new expiration date. If the value exceeds the maximum retention period (MAXRETPD), it is reduced to the MAXRETPD value. The default value for days is 0.

  TVEXTPURGE(EXPIRE) is the same as TVEXTPURGE(EXPIRE(0)).

- NONE: DFSMSrmm takes no action for volumes to be purged.
- RELEASE: DFSMSrmm releases a volume to be purged according to the release actions set for the volume. You must run expiration processing to return a volume to scratch status. This parameter is the default.

You can specify that volumes purged from DFSMShsm are to be retained for a few extra days. This process allows you to be sure that purged volumes do not contain any data that might still be needed. DFSMShsm migration and backup volumes can be retained by EXPDT=99365. You can optionally set EXPDT to the current date or a future date when the volume is purged from DFSMShsm with EDGTVEXT.

You can release volumes or set the volume expiration date either to the current date or based on a number of days from the current date. The effect of setting the expiration date depends on the retention method (EXPDT or VRSEL) already specified for the volume.

5.6 SEARCHDATASET extensions

The SEARCHDATASET subcommand is used to create a list of data sets that match criteria you specify. You can specify a generic name by using the full set of filter masks. Specify a fully qualified name to restrict the search to data sets that match the name exactly. Specify the FILESEQ operand to restrict the search to data sets at a relative position on the volume. You can also use the JOBNAME operand to restrict the search to data sets that are created by a particular job name.

With z/OS V1.13 DFSMSrmm the SEARCHDATASET subcommand has been updated with new operands:

- CATALOG
- CRDATE
- DATACLASS
- EXPDT
- FORCE
- LASTCHANGEDATE
- LASTREFDATE
- MANAGEMENTCLASS
- NODATACLASS
- NOMANAGEMENTCLASS
- NOOEXPDT
- NOSTORAGECLASS
- NOSTORAGEGROUP
- OEXPDT
- READDATE
- RETDATE
- STORAGECLASS
- STORAGEGROUP
- VRSELEXCLUDE
- WRITEDATE

With this implementation, the SEARCHDATASET search is more efficient with many data sets.

Figure 5-40 shows the syntax of the CHANGEDATASET subcommand with the new date range options.

```
                                            |—VOLUME(*)————————————|
                  |—DSNAME(*)————————|      |—OWNER(cmd_issuer_id)—|
                  |                  |      |                      |
►►— SEARCHDATASET————————————————————————————————————————————————————————►
    SD            |                  |      |                      |
                  |—DSNAME(dsn_mask)—|      |—VOLUME(VOLSER)———————|
                    DS                      | VO                   |
                                            |—OWNER(owner_id)——————|
                                              OW
   |—FILESEQ(1)————————————————————|
   |                               |
►——————————————————————————————————————————————————————————————————————►
   |                               |
   |—FILESEQ(physical_file_seq)————|
     SEQ

►——————————————————————————————————————————————————————————————————————►
  |                      |   |                      |
  |—CRDATE———————————————|   |—EXPDT————————————————|
  |—CRDATE(-date_range-)—|   |—EXPDT(-date_range-)——|
    CRD                        E

►——————————————————————————————————————————————————————————————————————►
  |                               |   |                            |
  |—LASTCHANGEDATE————————————————|   |—LASTREFDATE————————————————|
  |—LASTCHANGEDATE(-date_range-)——|   |—LASTREFDATE(-date_range-)——|
    LASTC                               LASTR

►——————————————————————————————————————————————————————————————————————◄◄
  |                          |      |                            |
  |—READDATE—————————————————|      |—WRITEDATE——————————————————|
  |—READDATE(-date_range-)———|      |—WRITEDATE(-date_range-)————|
    REA                               W

   date_range
 |—————————————————————————————————————————————————————————————————————|
  |                            |    |                            |
  |        |—current date—|    |    |        |—current date—|    |
  |—START(-————————————————-)——|    |—END(-—————————————————-)———|
    S      |—start_date———|           E      |—end_date—————|
           |—relative—————|                  |—relative—————|
```

Figure 5-40 SEARCHDATASET command syntax updated with new operands

The command has the following parameters:

- CRDATE(*date_range*): Lists the data sets whose creation date matches the specified date criteria. CRDATE is mutually exclusive with the SINCE operand. The date_range criteria can be specified by using the START and END suboperands. The default is the current date.

- EXPDT(*date_range*): Lists the data sets whose current expiration date matches the specified date criteria. The date_range criteria can be specified by using the START and END suboperands.
- LASTCHANGEDATE(*date_range*): Lists the data sets whose last changed date matches the specified date criteria. The date_range criteria can be specified by using the START and END suboperands. The default is the current date.
- LASTREFDATE(*date_range*): Lists the data sets based on both the last read date and last write date, using the most recent of both dates. The most recent of the two values within the date range for a data set is selected. The date_range criteria can be specified by using the START and END suboperands. The default is the current date.
- READDATE(*date_range*): Lists the data sets whose last read date matches the specified date criteria. The date_range criteria can be specified by using the START and END suboperands. The default is the current date.
- WRITEDATE(*date_range*): Lists the data sets whose last write date matches the specified date criteria. The date_range criteria can be specified by using the START and END suboperands. The default is the current date.
- *date_range*: Lists the data sets whose creation date matches the specified date. In the following list *nn or cccc* can be one of the following options:
  - CRDATE
  - EXPDT
  - LASTCHANGEDATE
  - LASTREFDATE
  - READDATE
  - WRITEDATE

  - *cccc*DATE: Only data sets whose creation date is the current date are listed.
  - *cccc*DATE(START): Only data sets whose creation date is the current date are listed.
  - *cccc*DATE(START()): Only data sets whose creation date is the current date are listed.
  - *cccc*DATE(START(start_date)): Only data sets whose creation date is on or after the specified start date are listed, where start_date is either an absolute date or relative date.
  - *cccc*DATE(END): Only data sets whose creation date is the current date are listed.
  - *cccc*DATE(END()): Only data sets whose creation date is the current date are listed.
  - *cccc*DATE(END(*end_date*)): Only data sets whose creation date is on or before the specified end date are listed, where end_date is either an absolute date or relative date. Because START defaults to the current date, the specified end date equal to or greater than the current date when START is omitted.
  - CRDATE(START(*start_date*)END(*end_date*)): Only data sets whose creation date is within the range delimited by the specified start and end dates are listed. Both *start_date* and *end_date* are either an absolute date or relative date. The specified end date equal to or greater than the specified start date.

Each of the start_date and end_date values can be absolute or relative dates:

- *Absolute dates* are specified as either *yyyy/ddd* or *yyddd* format. For example, January 3, 2011 can be specified as 2011/003 or 11003.
- *Relative dates* are specified as a number of days, months, or years before the current date.
  - -0: Specifies the current day, current month, current year.
  - -n: Specifies that the date is n days before the current date.

- -nM: Specifies that the date is n months before the current month and the current day in the month is as the current date.
- -nY: Specifies that the date is n years before the current year and the current day in the year is as the current date.

The value range for n is 0 - 99999, with a required leading "-" and an optional suffix of **M** or **Y**. The default is the current date.

Examples to list data sets whose creation date is:

Today specify	SD CRDATE
Three days ago specify	SD CRDATE(START(-3) END(-3))
Before January 1, 2000 specify	SD CRDATE(START(0000/001) END(1999/365))
On or after January 2, 2005 Specify	SD CRDATE(START(2005/002))

Figure 5-41 shows the syntax of the SEARCHDATASET subcommand with the other new options.

```
                                            |—VOLUME(*)———————————|
                  |—DSNAME(*)———————|       |—OWNER(cmd_issuer_id)—|
                  |                 |       |                      |
▸▸— SEARCHDATASET————————————————————————————————————————————————————————▸
    SD            |                 |       |                      |
                  |—DSNAME(dsn_mask)—|      |—VOLUME(VOLSER)———————|
                    DS                      | VO                   |
                                            |—OWNER(owner_id)——————|
                                              OW
  |—FILESEQ(1)———————————————————|
  |                              |
▸——————————————————————————————————————————————————————————————————————▸
  |                              |
  |—FILESEQ(physical_file_seq)—|
    SEQ

▸——————————————————————————————————————————————————————————————————————▸
  |                     |   |                                   |
  |—CATALOG(-——————————)—|  |—DATACLASS(-—————————————————)—|
    CA       |—NO———————|   | DA         |—dataclass_name—|    |
             |—UNKNOWN—|    |                                   |
             |—YES——————|   |—NODATACLASS———————————————————————|

▸——————————————————————————————————————————————————————————————————————▸
  |                |   |                                            |
  |—FORCE(-———————)—|  |—MANAGEMENTCLASS(-———————————————————————)—|
    FO     |—NO——|     | M                |—managementclass_name—|  |
           |—YES—|     |                                            |
                       |—NOMANAGEMENTCLASS——————————————————————————|

▸——————————————————————————————————————————————————————————————————————▸
  |                 |   |                                         |
  |—OEXPDT(-———————)—|  |—STORAGECLASS(-——————————————————————)—|
  |        |—date—|  |  | STORAGEC     |—storageclass_name—|    |
  |        |—*————|  |  |                                         |
  |—NOOEXPDT-————————|  |—NOSTORAGECLASS——————————————————————————|

▸——————————————————————————————————————————————————————————————————————◂◂
 |                                         |   |                       |
 |—STORAGEGROUP(-————————————————————————)—|   |—VRSELEXCLUDE(-——————)—|
 | STORAGEG      |—storagegroup_name—|      |     VR            |—NO——|
 |                                         |                   |—YES—|
 |—NOSTORAGEGROUP——————————————————————————|
```

Figure 5-41 SEARCHDATASET command syntax updated with new operands

The command has the following parameters:

- CATALOG: Specifies to limit the search to data sets based on catalog status. Specify CATLG(UNKNOWN) to search for data sets that are not yet cataloged. Specify CATLG(YES) to search for data sets that are currently cataloged. Specify CATLG(NO) to list only data sets that are uncataloged. CATALOG can be abbreviated as CATLG.

If you do not specify the CATALOG operand, DFSMSrmm searches all data sets, regardless of the catalog status of the data set.

- DATACLASS: Specifies to limit the search to data sets with the specified data class name. A data class name is one-to-eight alphanumeric, national, or special characters. DATACLASS is mutually exclusive with NODATACLASS.

  If you do not specify the DATACLASS operand, DFSMSrmm searches all data sets, regardless of a data class is assigned to a data set.

- FORCE: Specifies to limit the search to data sets with the force flag set. This means that a CHANGE subcommand was used with the FORCE operand, and the requested change was only made because FORCE was specified. Specify FORCE(NO) to list only data sets where the force flag is not set. Specify FORCE(YES) to list only data sets where the force flag is set on.

  If you do not specify the FORCE operand DFSMSrmm searches all data sets, regardless if the force flag is set to a data set.

- MANAGEMENTCLASS: Specifies to limit the search to data sets with the specified management class name. A management class name is one-to-eight alphanumeric, national, or special characters. MANAGEMENTCLASS is mutually exclusive with NOMANAGEMENTCLASS.

  If you do not specify the MANAGEMENTCLASS operand, DFSMSrmm searches all data sets, regardless of a management class is assigned to a data set.

- NODATACLASS: Specifies to limit the search to data sets that do not have a data class. NODATACLASS is mutually exclusive with DATACLASS.
- NOMANAGEMENTCLASS: Specifies to limit the search to data sets that do not have a management class. NOMANAGEMENTCLASS is mutually exclusive with MANAGEMENTCLASS.
- NOOEXPDT: Specifies to limit the search to data sets that do not have an original expiration date. NOOEXPDT is mutually exclusive with OEXPDT.
- NOSTORAGECLASS: Specifies to limit the search to data sets that do not have a storage class. NOSTORAGECLASS is mutually exclusive with STORAGECLASS.
- NOSTORAGEGROUP: Specifies to limit the search to data sets that do not have a storage group. NOSTORAGEGROUP is mutually exclusive with STORAGEGROUP.
- OEXPDT: Lists the data sets whose original expiration date matches the specified date. If you use an *, DFSMSrmm returns data set information for all data sets that have any original expiration date. OEXPDT is mutually exclusive with NOOEXPDT.

  If you do not specify the OEXPDT operand DFSMSrmm searches all data sets, regardless of an original expiration date is assigned to a data set.

- STORAGECLASS: Limits the search to data sets with the specified storage class name. A storage class name is one-to-eight alphanumeric, national, or special characters. STORAGECLASS is mutually exclusive with NOSTORAGECLASS.

  If you do not specify the STORAGECLASS operand, DFSMSrmm searches all data sets, regardless of a storage class is assigned to a data set.

- STORAGEGROUP: Limits the search to data sets with the specified storage group name. A storage group name is one-to-eight alphanumeric, national, or special characters. STORAGEGROUP is mutually exclusive with NOSTORAGEGROUP.

  If you do not specify the STORAGEGROUP operand, DFSMSrmm searches all data sets, regardless of a storage group is assigned to a data set.

- VRSELEXCLUDE: Limits the search to data sets excluded (or not excluded) from VRSEL processing. Specify NO to search for data sets that are not excluded from VRSEL processing. Specify YES to search for data sets that are excluded from VRSEL processing. VRSELEXCLUDE can be abbreviated as VX.

If you do not specify the VRSELEXCLUDE operand DFSMSrmm searches all data sets, regardless of the vital record selection criteria set for a data set.

5.7 VRS last reference date

Over time, the number of VRSs can grow to a number that is hard to comprehend, especially for no longer used VRSs. With DFSMSrmm V1.13, you can list a VRS based on the last reference date.

5.7.1 SEARCHVRS subcommand specifications

The SEARCHVRS subcommand is used to create a list of vital record specifications. Figure 5-42 shows an overview of the SEARCHVRS subcommand with the two new operands, LASTREFDATE and LASTCHANGAEDATE. Use the start_date operand to specify a date range for your search. A date_range consists of a start date and an end date. Each date can be an absolute date in either *yyyy/ddd* or *yyddd* format, or it can be a relative value from which DFSMSrmm calculates the date.

```
▸▸─SEARCHVRS──────────────────────────────────────────────────────────▸
   SS        |                           | |                          |
             |─LASTCHANGEDATE────────────| |─LASTREFDATE──────────────|
             |─LASTCHANGEDATE(-date_range-)─| |─LASTREFDATE(-date_range-)─|
               LASTC                           LASTR

 date_range
|──────────────────────────────────────────────────────────────────────|
  |                      |     |                       |
  |       |─current date─|     |     |       |─current date─|   |
  |─START(-──────────────-)─|   |─END(-──────────────-)─|
    S     |─start_date───|       E    |─end_date─────|
          |─relative─────|            |─relative─────|
```

Figure 5-42   SEARCHVRS subcommand using the LASTREFDATE and LASTCHANGEDATE

The command has the following parameters:

- LASTCHANGEDATE(*date_range*): Lists the vital record specifications whose last changed date matches the specified date criteria. The date criteria can be specified by using the START and END suboperands.
- LASTREFDATE(*date_range*): Lists the vital record specifications based on their last reference date. The last reference date of the vital record specification record is the date of the last inventory management VRSEL run that used this VRS to retain a data set or volume.
- *date_range*: Lists the data sets whose creation date matches the specified date. In the following list *nn or cccc* can be one of these operands:
  - - LASTCHANGEDATE
  - - LASTREFDATE

  The default is the current date.

- *cccc*DATE: Only data sets whose creation date is the specified date are listed
- *cccc*DATE(START): Only data sets whose creation date is the specified date are listed
- *cccc*DATE(START()): Only data sets whose creation date is the specified date are listed
- *cccc*DATE(START(start_date)): Only data sets whose creation date is on or after the specified start date are listed, where start_date can be an absolute date or relative date.
- *cccc*DATE(END): Only data sets whose end date is the specified date are listed
- *cccc*DATE(END()): Only data sets whose end date is the specified date are listed
- *cccc*DATE(END(end_date)): Only data sets whose creation date is on or before the specified end date are listed, where end_date is either an absolute date or relative date. Because START defaults to the current date, the specified end date equal to or greater than the current date when START is omitted.
- CRDATE(START(start_date)END(end_date)): Only data sets whose creation date is within the range delimited by the specified start and end dates are listed. Both start_date and end_date are either an absolute date or relative date. The specified end date equal to or greater than the specified start date.

Each of the start_date and end_date values can be absolute or relative dates.

- *Absolute dates* are specified as either *yyyy/ddd* or *yyddd* format. For example, January 3, 2011 can be specified as 2011/003 or 11003.
- *Relative dates* are specified as a number of days, months, or years before the current date.
  - -0: Specifies the current day, current month, current year.
  - -n: Specifies that the date is n days before the current date.
  - -nM: Specifies that the date is n months before the current month and the current day in the month is as the current date.
  - -nY: Specifies that the date is n years before the current year and the current day in the year is as the current date.

The value range for n is 0 - 99999, with a required leading “-” and an optional suffix of **M** or **Y**. The default is the current date.

Examples to list data sets whose creation date is:

Today specify	SD CRDATE
Three days ago specify	SD CRDATE(START(-3) END(-3))
Before January 1, 2000 specify	SD CRDATE(START(0000/001) END(1999/365))
On or after January 2, 2005 Specify	SD CRDATE(START(2005/002))

5.7.2 RMM ISPF panel updates

Figure 5-43 shows the RMM ISPF panel, Display data sets VRS, with the new fields.

```
  Panel  Help
------------------------------------------------------------------------------
                        DFSMSrmm Display data sets VRS
Command ===>


Data set mask . : 'SCHLUM.RMMTEST.MOVE.**'                      GDG . : NO
     Job name mask . :

Count . . . : 99999               Retention type  . . . . . : CYCLES
                                  While cataloged . . . . . : YES
Delay . . . : 0   Days            Until expired . . . . . . : NO

Location  . . . . . : REMOTE
Number in location  : 1
Priority  . . . . . : 0
                                  Release options:
Next VRS in chain . :             Expiry date ignore  . . . : NO
    Chain using . . :             Scratch immediate . . . . : NO

Owner . . . . . : SCHLUM
Description . . : RETAIN AND MOVE data sets
Last reference  : 2011/326    13:26:39  ( YYYY/DDD  HH:MM:SS )
Delete date . . : 1999/365    ( YYYY/DDD )

Last Change information:
 Date . . . . . . . : 2011/326       Time  . : 13:26:40     System  : SC70
 User change date : 2010/289       Time  . : 20:45:17     User ID : MHLRES7
```

Figure 5-43 DFSMSrmm VRS Display data sets VRS panel

The command has the following parameters:

- Last Reference: Display the date and time of the last inventory management VRSEL run that used the VRS to retain a data set or volume.
- Last change information: Display these details of the most recent change to the record:
  - Date: The last change date.
  - Time: The last change time.
  - System: The ID of the system on which the last change occurred.
  - User change date: The date of the most recent change by a user, other than by a DFSMSrmm internal function.
  - User change time: The time of the most recent change by a user, other than by a DFSMSrmm internal function.

– User ID: The ID of the user who caused the most recent change. If the most recent change was made by DFSMSrmm processing, the ID starts with an asterisk (*). Internal IDs include these values:
  - OAM: DFSMSrmm system managed tape support.
  - HKP: Inventory management.
  - OCE: DFSMSrmm OPEN/CLOSE EOV support.

5.8 Selective volume movement

In addition to the existing capability based on enabled Stacked Volume Support for LOGICAL volumes, volume movement can now be controlled by location independently of volume type. Non-IBM virtual tape solutions needs another way to prevent volume movement driven by VRS.

5.8.1 Functionality

With the new LOCDEF operand AUTOMOVE(YES/NO), you can define locations that are not applicable for automated movement. When the volume's current location is defined in EDGRMMxx PARMLIB member with LOCDEF location AUTOMOVE(NO), DSTORE processing does not set the destination from the required location.

During inventory management DSTORE, DFSMSrmm validates the current location name for a volume and determines whether automated movement is required. If validation fails, no movement is initiated. If a location is not defined by using LOCDEF on the inventory management system, automated movement is started. All volumes can be manually moved by RMM subcommands.

Tip: Moves requested manually from an AUTOMOVE(NO) location to bin-managed storage locations are not started by DSTORE processing.

Figure 5-44 shows the correct command syntax for the LOCDEF AUTOMOVE operand.

```
                  |            |-YES-|    |
                  |-AUTOMOVE(--------)-|
                  |                       |
►—LOCDEF——————————————————————————————————————————————————————————◄◄
                  |                       |
                  |-AUTOMOVE(--------)-|
                               |-NO--|
```

Figure 5-44 The new LOCDEF AUTOMOVE option

The command has the following parameters:

- AUTOMOVE: Use this operand to identify libraries or storage locations that hold tape volumes that are not eligible for automated movement initiated by inventory management. During inventory management DSTORE processing, the volume's current location is validated against the matching LOCDEF. Consider this operand for libraries that contain virtual volumes. You can also use it for volumes that either cannot be moved or for which you do not want DFSMSrmm to initiate the movement.

- YES: DSTORE starts automated movement when the required location does not match the current location of the volume. This setting is the default.
- NO: DSTORE does not initiate automated movement

5.8.2 RMM LISTCONTROL command

There is a new field, AM(AUTOMOVE), in the output of the TSO RMM LISTCONTROL command for the location definition, as shown in Figure 5-45.

```
Location definitions:
Location Def Mgtype  Ltype  Priority AM Medianames
-------- --- ------  ------ -------- -- -----------------------------------
          N          AUTO     4800   Y
          N          MANUAL   4900   Y
DISTANT   N          STORE     200   Y
LOCAL     N          STORE     300   Y
REMOTE    N          STORE     100   Y
SHELF     N                   5000   Y
VTFM001   Y  NOBINS  HSTORE   2000   N  *
```

Figure 5-45 Location definitions

5.9 Last change details

You need an easy way to audit changed media without running the EDGAUD audit reports. With z/OS V1.13, the last change information is added to all list command output and to the dialog for all resources stored in the RMM CDS. This configuration reduces the need to run the EDGAUD audit reports.

The EDGAUD and EDGRPTD utilities and the EDGRRPTE exec produce information about your removable media library and storage locations. You can also get security trail information about volumes and data sets defined to DFSMSrmm. They also provide audit trail information about volumes, shelf assignments, and user activity.

The new information is added to all list command outputs for all resources stored in the Removable Media Manager (RMM) control data sets (CDSs). It is also added to all list, change, and delete panels. The last change information is available for the following information:

- Volumes
- Data sets
- Racks
- Bins
- Owners
- Products
- VRSes

You can use the LISTDATSET command, shown in Figure 5-46, to get the last change information.

```
   RMM LD 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG' VOL(VT0026)
```

Figure 5-46 RMM TSO LISTDATASET subcommand

Figure 5-47 shows the result of the command.

```
Data set name = RMM.F13002.ON.TWO.VOLUMES.AND.CATLG
Volume        = VT0026              Physical file sequence number = 1
Owner         = MHLRES7                         data sets sequence = 1
Create date   = 2011/326   Create time = 16:52:22 System ID         = SC70
Expiration date       = 2011/326   Original expir. date  =
         set by       = OCE_DEF
Block size            = 80         Block count           = 1
Data set size(KB)     = 1
Physical size(KB)     = 0          Compression           = 0.00
Percent of volume     = 0          Total block count     = 1
Logical Record Length = 0          Record Format         = F
Date last written     = 2011/326   Date last read        = 2011/326
Job name              = TEST9999   Last job name         = TEST9999
Step name             = STEP03     Last step name        = STEP03
Program name          = EOVTEST    Last program name     = EOVTEST
DD name               = OUT        Last DD name          = OUT
Device number         = 0407       Last Device number    = 0407
Management class      =            VRS management value  =
Storage group         =            VRS retention date    =
Storage class         =            VRS retained          = NO
Data class            =            Closed by Abend       = NO
                                   Deleted               = NO
VRSEL exclude         = YES        Catalog status        = UNKNOWN
Primary VRS details:
       Name           =
       Job name       =            Type                  =
       Subchain NAME  =            Subchain start date   =
Secondary VRS details:
       Value or class =
       Job name       =
       Subchain NAME  =            Subchain start date   =
Security Class   =          Description   =
BES key index    = 0

Last Change information:
Date             = 2011/326    Time = 16:52:23  System  = SC70
User change date =             Time =           User ID = *OCE
```

Figure 5-47 List data set output

Figure 5-48 shows details that indicate the last date when information was changed on that data set record. This example is the same data set as before, but using the RMM ISPF dialog.

```
                          DFSMSrmm data sets Details              Multi-Volume
Command ===>

Data set name . . : 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG'
Volume serial . . : VT0026      Physical file sequence number . . : 1
Owner . . . . . . : MHLRES7     data sets sequence number  . . . . : 1
                                                                   More:     +
Job name  . . . . : TEST9999
Step name . . . . : STEP03                  Record format . . . . : F
Program name  . . : EOVTEST                 Block size  . . . . . : 80
DD name . . . . . : OUT                     Logical record length : 0
Create date . . . : 2011/326    YYYY/DDD    Block count . . . . . : 1
Create time . . . : 16:52:22                Data set size (KB)  . : 1
System id . . . . : SC70                    Physical size (KB)  . : 0
                                            Compression . . . . . : 0.00
                                Total block count . . . : 1
Expiration date . : 2011/326    YYYY/DDD    Percent of volume . . : 0
 Set by . . . . . : OCE_DEF                 Device number . . . . : 0407
Original  . . . . :             YYYY/DDD

Last job name . . : TEST9999                Last DD name  . . . . : OUT
Last step name  . : STEP03                  Last device number  . : 0407
Last program name : EOVTEST
Date last read  . : 2011/326                VRS management value  :
Date last written : 2011/326                Management class  . . :
                                            Data class  . . . . . :
VRSEL exclude . . : YES                     Storage class . . . . :
Retention date  . :                         Storage group . . . . :
VRS retained  . . : NO

Security name . . :                         BES key index . . . . : 0
Classification  . :

Primary VRS details:  (Use MATCHVRS primary command to display matching VRSes)
 VRS name . . . :
 Job name . . . :                             VRS type  . . . . . :
 Subchain name  :                             Subchain start date :

Secondary VRS details:
 Value or class :
 Job name . . . :
 Subchain name  :                             Subchain start date :

Catalog status  . : UNKNOWN
Closed by Abend . : NO                      Deleted . . . . . . . : NO

Last Change information:
 Date . . . . . . : 2011/326      Time  . : 16:52:23     System  : SC70
 User change date :               Time  . :              User ID : *OCE
```

Figure 5-48 List data sets Details panel

The command has the following parameters:

- Last change information: Details of the most recent change to the record:
  - Date: The last change date.
  - Time: The last change time.
  - System: The ID of the system on which the last change occurred.
  - User change date: The date of the most recent change by a user, other than by a DFSMSrmm internal function.
  - User change time: The time of the most recent change by a user, other than by a DFSMSrmm internal function.
  - User ID: The ID of the user who caused the most recent change. If the most recent change was made by DFSMSrmm processing the ID starts with an asterisk (*). Internal IDs include these values:
    - OAM: DFSMSrmm system managed tape support
    - HKP: Inventory management
    - OCE: DFSMSrmm OPEN/CLOSE EOV support

5.10 Support RETPD(93000)

The number of days that DFSMSrmm retains the data set before considering it for release (the retention_period) can be a decimal number from 0 to 93000. This value is used in RETPD and MAXRETPD in all specifications in DFSMSrmm.

5.10.1 Command syntax

This value is set on the EDGRMMxx PARMLIB member, as shown in Figure 5-49.

```
             |-MAXRETPD(NOLIMIT)-------|      |-RETPD(5)-----------|
             |                         |      |                    |
▸—OPTION——————————————————————————————————————————————————————————————◂◂
             |                         |      |                    |
             |-MAXRETPD(--------------)-|      |-RETPD(-----------)-|
                         |-nnnnn--|                     |-nnnnn-|
```

Figure 5-49 EDGRMMnn PARMLIB options MAXRETPD and RETPD syntax

The command has the following parameters:

- MAXRETPD: Specifies the maximum retention period that a user can request for data sets on volumes. Specify NOLIMIT or a value of 0 - 93000 days. When a value of 0 - 93000 days is specified, the value is added to the current date to determine the maximum allowed expiration date. Specify NOLIMIT to use the dates 99365 or 99366, which mean to never expire. If the calculated date is 31 December 1999, the expiration date 1 January 2000 is used.

  MAXRETPD is always used to determine the volume expiration date. The volume expiration date can be ignored when EXPIRYDATEIGNORE is specified on all vital record specifications. MAXRETPD is important if the vital record specification specifies UNTILEXPIRED and the decision is based on the volume expiration date, or if the volume is managed by the EXPDT retention method. For the VRSEL retention method, the preferred method is to put the retention requirements all into the vital record specifications and make MAXRETPD=RETPD. If users are allowed to specify expiration and retention period to override vital record specifications, select a MAXRETPD that covers the maximum retention period they would like to enforce. If this forces 99365 to be reduced, define vital record specifications for any data that should be permanently retained, like DFSMShsm tape data.

Use MAXRETPD to set limits on the values that can be specified for EXPDT and RETPD. If the retention period or expiration date specified in the JCL or management class exceeds the MAXRETPD value, DFSMSrmm overrides it and uses the value in MAXRETPD to determine the expiration date. For data sets on volumes that use the EXPDT retention method, if the expiration date determined with the LASTREF data set attribute exceeds the MAXRETPD value, DFSMSrmm overrides it and uses the value in MAXRETPD to determine the expiration date. The volume label has the JCL-specified value because DFSMSrmm does not change tape labels or system control blocks. The control data set contains the JCL-specified value as well as the expiration date calculated from MAXRETPD. You can display the JCL-specified value for information only.

If the DFSMSrmm ISPF dialog or RMM TSO subcommands are used to specify a retention period or expiration date that exceeds the MAXRETPD value, DFSMSrmm fails the subcommand or panel request.

Default: MAXRETPD(NOLIMIT)

- RETPD: Specifies the default retention period for all new data sets on volumes. Specify a value of 0 - 93000 days. The specified value is added to the current date to determine the expiration date. Select a default retention for PARMLIB RETPD that is a small value. This setting ensures that all tape data created outside the service levels is released as soon as possible. The MAXRETPD value you specify in the PARMLIB limits the calculated expiration date.

  DFSMSrmm sets a default retention period by using these steps

  - If you specify RETPD or EXPDT, the value is used as the new expiration date of the volume. If you do not specify RETPD or EXPDT, DFSMSrmm uses the EXPDT or RETPD allocation attribute of a data class, if all these circumstances are true:
    - The Storage Management Subsystem is active
    - The data set is associated with a data class, either explicitly by the DATACLAS keyword on the JCL or implicitly by an automatic class selection routine
    - The data class has an EXPDT or RETPD allocation attribute
  - If you do not specify RETPD or EXPDT, DFSMSrmm uses the default retention period set in EDGRMMxx

  Whenever a new data set is written to tape, DFSMSrmm checks whether the volume's expiration date needs to be updated. This decision is based on whether the new data set has a longer expiration date than the volume on which it is written. DFSMSrmm gets the expiration date for a data set from the job file control block (JFCB) at open time. If there is a date in the JFCB, DFSMSrmm compares this date to the current expiration date for the volume. If the date in the JFCB allows the volume to be retained longer, DFSMSrmm uses that date to update the volume's expiration date.

  If there is no expiration date in the JFCB, DFSMSrmm uses the EDGRMMxx RETPD value to calculate the new expiration date. If the RETPD value allows the volume to be retained longer, DFSMSrmm uses that date to update the volume's expiration date.

You can set the date in the JFCB in these ways:

– RETPD and EXPDT keywords in the JCL
– Data class when the Storage Management Subsystem is active and the volume is system-managed
– Management class when the Storage Management Subsystem is active and the volume is system-managed
– A user program, by using the RDJFCB macro and the OPEN TYPE=J after modifying the JFCB
– Installation exits in use on your particular system

5.10.2 DFSMSrmm subcommands

The DFSMSrmm subcommands that allow the following specification are shown in Figure 5-50.

```
ADDDATASET
ADDVOLUME
CHANGEDATASET
CHANGEVOLUME
GETVOLUME
```

Figure 5-50 DFSMSrmm subcommands that allow RETPD to be specified

These commands have these parameters

- RETPD(*retention_period*): Specifies the number of days that DFSMSrmm retains the volume before considering it for release.
- *retention_period*: A decimal number from 0 to 93000.

5.11 Dialog navigation enhancements

The DFSMSrmm dialog now has more point-and-shoot fields in the Volume and data sets Detail panels. When using the ISPF GUI, point-and-shoot fields are displayed as push buttons.

The new primary commands CHAIN, CHAINVOLUME, CHAINDATASET, VL, and IL are supported. They produce the same results as the existing VL and IL line commands. You can abbreviate CHAINVOLUME to CHAINV or any other valid matching abbreviation. Similarly, you can abbreviate CHAINDATASET to CHAIND or any other valid matching abbreviation.

The CHAIN primary command on the volume panels implements CHAINV, and the data sets panels implement CHAIND.

5.11.1 Point-and-shoot fields

Point-and-shoot fields have the following characteristics:

- Point-and-shoot fields span the output field only. No prompt text is included.
- If a point-and-shoot action is not appropriate due to a zero value, no point-and-shoot action is taken. The message `Action not supported` is displayed instead.

- When the point-and-shoot action results in a search, the long message `Search in progress` is displayed.
- To easily see the fields that are enabled for point-and-shoot, you must customize their color, intensity, and highlighting. Issue the Interactive System Productivity Facility (ISPF) system command PSCOLOR from any ISPF command line. Then adjust the Point-and-Shoot panel element as wanted. For more information, see *z/OS V1.13.0 ISPF User's Guide Vol II,* SC34-4823.

You can adjust the point-and-shoot panel element as shown in Figure 5-51.

```
                       CUA Attribute Change Utility
Command ===>                                                  Defaults

Change colors, intensities, or highlights for panel attribute elements.
Enter the EXIT command to save changes or enter the CANCEL command to exit
without saving. To restore the defaults for a type, clear the field and
press Enter or select the Defaults point-and-shoot field to restore all
default settings for all types.

Panel Element                 Color         Intensity  Highlight
                                                              More:   -
Point-and-Shoot  . . . . . . . RED           HIGH       USCORE
PD Available Choices . . . . . WHITE         LOW        NONE
PD Unavailable Choices . . . . BLUE          LOW        NONE
Reference Phrase . . . . . . . WHITE         HIGH       NONE
Scroll Information . . . . . . WHITE         HIGH       NONE
Sel. Available Choices . . . . WHITE         LOW        NONE
Sel. Unavailable Choices . . . BLUE          LOW        NONE
Variable Output Info.  . . . . TURQ          LOW        NONE
Warning Message Text . . . . . YELLOW        HIGH       NONE
Warning Text . . . . . . . . . RED           HIGH       NONE
Work Area Separator Line . . . BLUE          LOW        NONE
```

Figure 5-51 ISPF settings for PSCOLOR command

Tip: After changing PSCOLOR settings, you see different field attributes in the Volume Details panel in the point-and-shoot fields.

New point-and-shoot fields on the volume details panel

Table 5-2 shows the point-and-shoot fields in the volume detail panel.

Table 5-2 Volume detail point-and-shoot fields

Field	Point-and-shoot action	RMM Dialog Displays
VOL1 VOLSER	LV vol1 ALL	Volume Details for VOL1 VOLSER
Rack number	LR number	Rack Details
Set retained	CHAINV primary command for volume chain	Volume search results list for the volume set
Expiration date	DATE OPTIONS fast path command	Dialog User Options

Field	Point-and-shoot action	RMM Dialog Displays
Availability	CHAINV primary command for volume chain	Volume search results list for the volume set
Owner	LO name	Owner Details
Security	LC SECLEVEL(security_class)	Security Classification Rules for the security level
Last changed by	LO name	Owner Details
Previous volume	LV prev_VOLSER	ALL Volume Details for previous VOLSER
Next volume	LV next_VOLSER	ALL Volume Details for next VOLSER
Volume sequence	CHAINV primary command for volume chain	Volume search results list for the volume set
Number of data sets	VOLUME(VOLSER) NOLIMIT	Data set search results list for all data sets on this volume
Actions pending	CONTROL ACTIONS fast path command	Volume Action Status list
Location	CONTROL LOCDEF fast path command	Location Definitions list
Bin number	Bin number LB number LOCATION(loc) MEDIANAME(namel*)	Bin Details
Product details	LP number [LEVEL(version)]	Product Details

New point-and-shoot fields on the data sets Details panel

Table 5-3 shows the point-and-shoot fields in the data sets Detail panel.

Table 5-3 Volume detail point-and-shoot fields

Field	Point-and-shoot action	RMM Dialog Displays
Volume VOLSER	LV VOLSER ALL	Volume Details for VOLSER
Owner	LO name	Owner Details
Physical file sequence number	SD VOLUME(VOLSER) NOLIMIT	Data set search results list for all data sets on this volume
Data set sequence number	CHAIND primary command for data set chain	Data set search results list for all data sets for all volumes in the volume set
Create date	DATE OPTIONS fast path command	Dialog User Options

5.11.2 Using the new ISPF primary commands

The CHAIN primary command on the volume panels implements CHAINV, and the data set panels implement CHAIND.

Using the CHAINVOLUME command

Use the CHAINVOLUME command to list all volumes in the multi-volume set. Display the DFSMSrmm Volume Details panel as shown in Figure 5-52, then issue the CHAINVOLUME command on the command line. If no volume chain exists, the returned list will be a single volume. You can abbreviate CHAINVOLUME to CHAINV, or you can use VL.

```
                       DFSMSrmm Volume Details - VT0017           Multi-Volume
Command ===> CHAINVOLUME

Volume . . . . . . : VT0017     VOL1 VOLSER : ______   Rack number : _______
Media name . . . . : 3592                              Status  . . : MASTER
                                                                    More:     +
Volume type  . . . : LOGICAL       Stacked count  . . . . . . . : 0
WorldWide ID . . . :                                   Worm . . : NO
Retention date . . :               Expiration date  . . . . . : 2011/334
Set retained . . . : NO              Set by . . . . . . . . . . : OCE_JFCB
Hold . . . . . . . : NO            Original expiration date . : 2011/334
Retention method . : VRSEL
  Set by . . . . . : OCE_DEF

Description  . . . :

Data set name  . . : 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG'
Media information  : VTFM
Media type . . . . : VT3590G2
Vendor . . . . . . :               Release actions:
Label  . . . . . . : SL              Return to SCRATCH pool . : YES
  Current version  :                 Replace volume . . . . . : NO
  Required version :                 Return to owner  . . . . : NO
Density  . . . . . : IDRC            Initialize volume  . . . : NO
Recording format . : VT3590G2        Erase volume . . . . . . : NO
Compaction . . . . : YES             Notify owner . . . . . . : NO
Attributes . . . . : NONE            Expiry date ignore . . . : NO
Availability . . . : _______________ Scratch immediate  . . . : NO

Enter SCROLL commands for more volume information, or END command to CANCEL.
```

Figure 5-52 DFSMSrmm Volume Details panel

Tip: All fields with different field attributes (such as underlining) are point-and-shoot fields.

The command has the following parameter:

- **CHAINVOLUME**: Returns a volume search results list that contains all volumes in the multi-volume set. You can use it from both the Volume and data sets Details panels. If no volume chain exists, the returned list is a single volume. The results are just like the VL line command.

Figure 5-53 shows the result of the CHAINVOLUME command issued on the Volume Details panel.

```
                   DFSMSrmm Volumes (Page 1 of 2)            Row 1 to 2 of 2
Command ===>                                                Scroll ===> CSR

Enter HELP or PF1 for the list of available line commands
Use the RIGHT command to view other data columns
   Volume          Assigned   Expir./    S                 Dest-    Tr- Data
S  serial Owner    date       Retn. date R Status  Location ination  ans sets
-- ------ -------- ---------- ---------- - ------- -------- -------- --- -----
   VT0008 MHLRES7  2011/324   2011/333     MASTER  VTFM001           N   1
   VT0013 MHLRES7  2011/324   2011/333     MASTER  VTFM001           N   1
```

Figure 5-53 CHAINVOLUME result panel

Using the CHAINDATASET command

Use the CHAINDATASET command to list all data sets in the multi-volume set. Display the DFSMSrmm data sets Details panel as shown in Figure 5-54, then issue the CHAINDATASET command on the command line. If no data set chain exists and no volume chain exists, the returned list contains a single data set. You can abbreviate CHAINDATASET to CHAIND or you can use IL.

```
                          DFSMSrmm data sets Details               Multi-Volume
Command ===> CHAINDATASET

Data set name . . : 'RMM.F13002.ON.TWO.VOLUMES.AND.CATLG'
Volume serial . . : VT0015      Physical file sequence number . . : 1
Owner . . . . . . : MHLRES7     data sets sequence number  . . . . : 1
                                                                 More:     +
Job name  . . . . : TEST9999
Step name . . . . : STEP03                  Record format . . . . : F
Program name  . . : EOVTEST                 Block size  . . . . . : 80
DD name . . . . . : OUT                     Logical record length : 0
Create date . . . : 2011/324    YYYY/DDD    Block count . . . . . : 1
Create time . . . : 19:41:58                Data set size (KB)  . : 1
System id . . . . : SC70                    Physical size (KB)  . : 0
                                            Compression . . . . . : 0.00
                                Total block count . . . : 1
Expiration date . : 2011/333    YYYY/DDD    Percent of volume . . : 0
 Set by . . . . . : OCE_JFCB                Device number . . . . : 044C
Original  . . . . : 2011/333    YYYY/DDD

Last job name . . : TEST9999                Last DD name  . . . . : OUT
Last step name  . : STEP03                  Last device number  . : 044C
Last program name : EOVTEST
Date last read  . : 2011/324                VRS management value  :
Date last written : 2011/324                Management class  . . :
                                            Data class  . . . . . :
VRSEL exclude . . : NO                      Storage class . . . . :
Retention date  . : PERMANENT               Storage group . . . . :
VRS retained  . . : YES
```

Figure 5-54 DFSMSrmm Volume Details panel

Tip: All fields with different field attributes (such as underlining) are point-and-shoot fields.

The command has the following parameter:

- **CHAINDATASET**: Returns a volume search results list that contains all data sets in the multi-volume set. You can use it from both the data sets and Volume Details panels. If no volume chain exists, the returned list contains a single data set. The results are just like the IL line command.

Figure 5-55 shows the result of the CHAINDATASET command issued on the data sets Details panel.

```
                 DFSMSrmm data sets (Page 1 of 2)              Row 1 to 2 of 2
Command ===>                                                  Scroll ===> CSR



Enter HELP or PF1 for the list of available line commands
Use the RIGHT command to view other data columns
                                                Volume          File
S  data sets name                                serial Owner    seq
-- -------------------------------------------- ------ -------- -----
   RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0008 MHLRES7  1
   RMM.F13002.ON.TWO.VOLUMES.AND.CATLG          VT0013 MHLRES7  1
```

Figure 5-55 CHAINDATASET result panel

Using the CHAIN command

When issued from the Volume Details panel, CHAINVOLUME is handled as described in “Using the CHAINVOLUME command” on page 101. When issued from the data sets Details panel, CHAINDATASET command is handled as described in “Using the CHAINDATASET command” on page 102.

5.12 Migration considerations

Before you use DFSMSrmm z/OS V1.13, note the following considerations:

- There are no new migration considerations.
- z/OS releases before z/OS V1.13 require the PTF for coexistence (APAR OA32984) to be installed before using the new functions on z/OS V1.13.
- A new DFSMSrmm report, EDGRT18, includes data sets and volumes other than those that are scratch. REPORT18 is split by retention method, and sorted by data set name, create date, and create time.
- A new column was added to the VRSRETN report to include the data set VRSELEXCLUDE attribute.
- A new column was added to the EXPDROP report to include the retention method.
- There are no new migration health checks shipped for z/OS DFSMSrmm V1.13.

6

VSAM enhancements

This chapter addresses DFSMS 1.13 enhancements that apply mainly to Virtual Storage Access Method (VSAM) record-level sharing (RLS) implementations. If you are an experienced SMS/ISMF user, go directly to 6.2, “VSAM new facilities in DFSMS V1.13” on page 111.

DFSMS R1.13 has these main VSAM enhancements:

- Buffer management facility (BMF) enhancements. For more information, see “Enhanced BMF performance in DFSMS 1.13” on page 111.
- New storage class option to disconnect the RLS cluster from buffering. For more information, see “Storage class option to disconnect RLS cluster from buffers” on page 114.
- VSAM OPEN first time failure memory dump data capture. For more information, see “VSAM OPEN first time failure data capture” on page 115.

This chapter includes the following sections:

6.1 VSAM RLS basic concepts

Before moving to the new VSAM enhancements in DFSMS V1.13, the following is a review of VSAM RLS.

6.1.1 VSAM RLS to implementing data sharing

Record-level sharing is a method of managing VSAM buffer pools for VSAM clusters. Unlike nonshared resources (NSR), local shared resource (LSR), and global shared resource (GSR), the buffers are kept in the coupling facility cache and can be accessed across all systems. As expected, each buffer in the buffer pool contains a data control interval (CI) or an index CI. The objectives of buffering include:

- For random read accesses, avoid I/O operations by having CI read hits in such buffers. VSAM always does a synchronous I/O operation for a random write.
- For sequential accesses, buffering makes the I/O operation more efficient by reducing its number and so the task programs total I/O connect time.

RLS allows any number of task programs that run in different MVS systems in your sysplex to share existing VSAM clusters. This sharing allows RLS to provide full read and write data integrity by using these objects:

- A focal point for all I/O requests, such as the SMSVSAM address space (one per each MVS)
- Coupling facility structures such as one lock structure (IGWLOCK00) and several cache structures
- VSAM RLS shared control data set (SHCDS)

The RLS serialization is done at the logical record level, in contrast with other VSAM buffering modes where it is done at the control interval level. This process provides for a better RLS granularity. However, to implement recoverable VSAM clusters accessed concurrently by online and batch, in addition to RLS, your installation must have its own back out log. The back out log is for batch updates, and uses Transactional VSAM (TVS).

The major objective of VSAM RLS is continuous availability (24 / 7). All Multiple Virtual Storage (MVS) systems that access the same VSAM clusters are identical where data is concerned. If one of them fails, or there is planned shutdown, the VSAM data is still accessible from other surviving MVS systems in the same sysplex. Also, keeping CIs in the cache structures in the coupling facility can improve VSAM performance by avoiding I/O operations.

Figure 6-1 shows a general view of the VSAM RLS components.

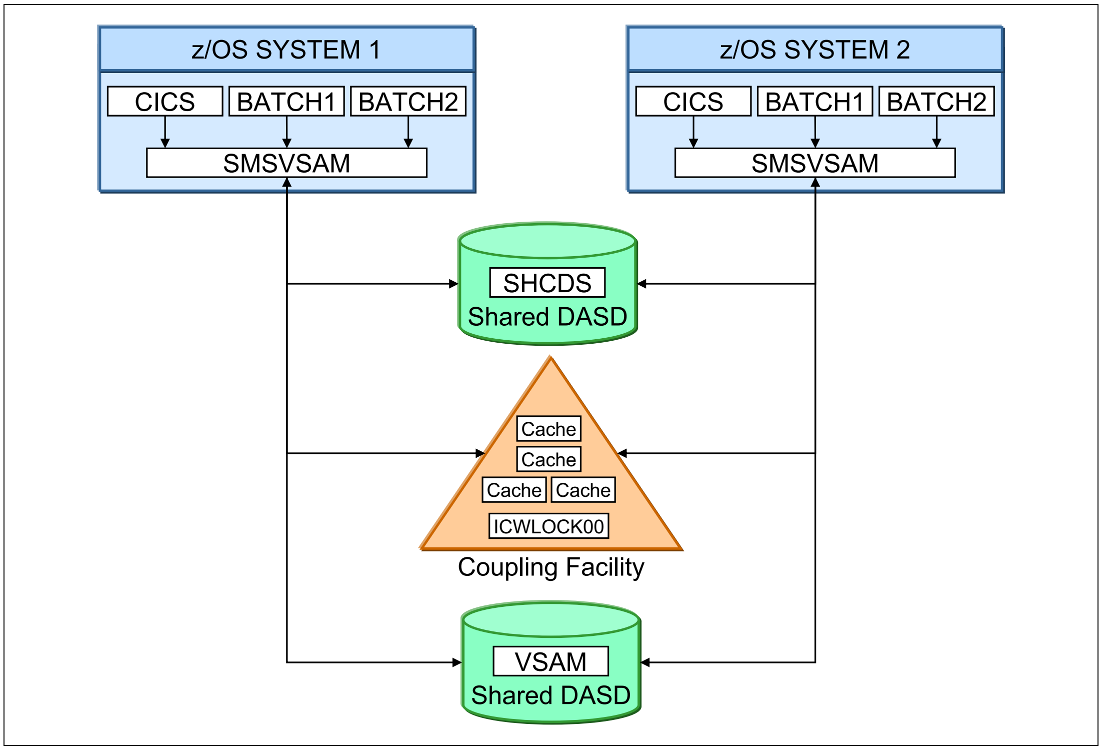


Figure 6-1 VSAM RLS Implementation

VSAM RLS does not introduce new types of VSAM data sets. Instead, it introduces a new way of accessing existing data sets. Apart from the need to open data sets in RLS mode, the same VSAM record management interfaces used for other buffering techniques are used for RLS. These techniques include get, put, point, and erase.

You can specify the RLS mode in the MACRF parameter of the ACB macro that you use to open the data set in your task program. Or you can specify the RLS mode in the keyword RLS in the DD card that points to the VSAM cluster in job control language (JCL). The same cluster can be accessed in RLS mode and non-RLS mode.

RLS mode can be used in these circumstances:

- For all types of VSAM organization, as such KSDS, RRDS, VRRDS, and ESDS VSAM clusters.
- With an alternate Index (AIX) PATH access for KSDS and ESDS, if in extended format
- With extended addressability (clusters larger than 4 GB)
- With spanned logical record format
- For clusters that support data compression
- With sequential data striping (beginning with z/OS 1.12).

You can use VSAM RLS to access a cluster from just one MVS system. In this instance, the cluster can be shared concurrently by several tasks programs in the same z/OS with read and write integrity. This approach produces better granularity than using share options one or two.

6.1.2 Buffer management facility (BMF) concept

BMF enhancements in DFSMS V1.13 include those described in this section.

RLS SMSVSAM buffer pools can be in these locations:

- Data spaces owned by the SMSVSAM address space, each one up to 2 GB addresses (31-bit).
- SMSVSAM address space private area (above the bar). You must declare IGDSMSxx option for these buffers, as follows:

```
Above the 2 GB Bar . . . : YES
```

The CI population of such buffers is managed by BMF, a VSAM RLS component, by using a least recently used (LRU) algorithm. There is NOT a sequential look ahead and a sequential CI discard for any type of RLS access. So avoid using RLS for sequential processing for performance considerations.

To avoid DASD I/O operations, BMF uses sophisticated LRU algorithm targeting. This targeting keeps the most referenced CIs, or the best selection of CIs of direct (random) access in the buffer pool. BMF introduces a dynamic LRU algorithm by varying the number of buffers in the buffer pools and, therefore, the number of loaded CIs. This number of buffers is affected in the following ways:

- Increases when a new buffer is needed by a read incoming CI. It is up to the installation to define an upper limit.
- Decreases when a CI is deleted or stolen time driven by the BMF aging routine, the one that implements the LRU algorithm.

BMF buffer aging (stealing) algorithm before DFSMS V1.13

The least referenced CIs are stolen by an LRU BMF aging routine when necessary. This routine must efficiently answer these questions:

- When to steal?
- Which CIs are to be stolen?
- How many CIs are to be stolen?

The installation influences the BMF aging algorithm by setting limits through the IGDSMSxx Parmlib member options, such as the following and those shown in Example 6-1:

- RLS_MAX_POOL_SIZE: The maximum size of the SMSVSAM local buffer pool or private area address space and data space. BMF avoids exceeding the buffer pool size you specify although more storage might be temporarily used. The default is 100 MB.
- RLSAboveTheBarMaxPoolSize: Specifies the total size of the buffer pool that is above the 2 GB bar at SMSVSAM address space for these environments:
  - All MVS systems
  - Each MVS system is referenced by name in this option. See Example 6-1. The default is zero.

Example 6-1 IGDSMSxx BMF options

```
RLSABOVETHEBARMAXPOOLSIZE({(ALL,size)})
RLSABOVETHEBARMAXPOOLSIZE({(sysname1,size1[;sysname2,size2[...;sysname32,size32]])})
```

The behavior of the BMF algorithm depends on the location of the CI buffers. The behavior varies if the CI buffers are in the SMSVSAM data space (31-bit) or in the address space 64-bit buffers (above the bar).

Continue by providing the answers for the three BMF stealing routine questions.

When to steal

The trigger to steal is the number of occupied buffers that are approaching the limit. Periodically BMF verifies the total buffer pool size utilization and compares it to the RLS_MAX_POOL_SIZE and RLSAboveTheBarMaxPoolSize values.

After this comparison, the BMF stealing routine runs control interval (CI) stealing, which runs with a dynamic frequency. At higher frequencies, more CIs are stolen per time and their measured unreferenced pattern is less exact. Additionally, more processor cycles are spent. Depending on frequency, the BMF stealing routine has four operational modes:

- Normal mode
- Maintenance mode
- Accelerated mode
- Panic mode.

Those modes are addressed in “BMF stealing routine details for 31-bit (data spaces)” on page 109.

Which CIs to steal

The age of the control interval in the buffer is determined by the time interval since it was last referenced by the BMF stealing routine. All CIs are investigated in a round robin approach where the CI buffers keep a static position in the queue:

- For the 64-bit buffer pool (address space above the bar), the age is measured in minutes. The BMF algorithm uses a time stamp instead of a counter to currently track aging of above the bar CI buffers.
- For the 31-bit buffer pool (data space), this age is measured in unreferenced interval count (UIC) seconds. The UIC is roughly the number of seconds a CI is without a reference or stolen. It can be thought as equivalent to the time interval between BMF stealing routines. The UIC of a CI in a buffer is reset to zero when the CI is referenced. The BMF stealing routine investigates every CI in a circular fashion. The CIs that are stolen are the ones with a UIC greater than the goal UIC.

How many CIs to steal

To answer this question, the BMF stealing routine initially sets a goal for the UIC. It changes this goal according to the operational mode of the BMF LRU algorithm it is in. Any CI that has a UIC greater than the goal UIC is eligible for being released, or stolen, to create room for another CI. This goal (target) UIC is the Initial_Free_UIC. Thus the criteria to stop stealing is not connected with an amount of still available buffers at the buffer pool, but with a minimum UIC value.

BMF stealing routine details for 31-bit (data spaces)

Only the four modes of BMF stealing routine for 31-bit buffers are addressed because DFSMS 1.13 improved only 31-bit buffering. For more information about 64-bit buffering, see *VSAM Demystified*, SG24-6105. The BMF stealing routine has these operational modes:

- Normal mode: The BMF algorithm is in Normal mode when the 31-bit CI buffer total amount is less than 80% of the RLS_MAX_POOL_SIZE. There is no stealing in Normal mode for these reasons:
  - Buffers are released that contain only Invalid and paged out CIs.
  - Each CI in the buffers has its UIC increased by the amount of elapsed seconds since the last time the routine passed by.
  - CIs can stay in buffers indefinitely when Normal mode is on.

- Maintenance mode: The BMF algorithm enters Maintenance mode when the 31-bit buffer total size is between 80% and 120% of RLS_MAX_POOL_SIZE. Maintenance Mode has these characteristics:
  - Initial_Free_UIC is decreased by 1, which indicates that the stealing criteria is becoming more severe.
  - All CIs with UIC greater than Initial_Free_UIC are automatically stolen.
  - Each non-stolen CI in the buffers has its UIC increased by the number of elapsed seconds since the last time the BMF stealing routine passed by this CI.
- Accelerated mode: The BMF algorithm enters Accelerated Mode when the 31-bit buffer total size is greater than 120% and less than two times the value of RLS_MAX_POOL_SIZE. This mode has these characteristics:
  - Initial_Free_UIC is decreased by 4, which indicates that the stealing criteria is becoming even more severe.
  - CIs with UIC greater than Initial_Free_UIC are stolen.
  - Each not stolen CI in the buffers has its UIC increased by the amount of elapsed seconds since the last time.
  - CI requests for new buffers are first served by stolen buffers.
  - If there are no buffers to steal, a Getmain request for new storage buffers is done.
- Panic mode: The BMF algorithm enters Panic mode when the 31-bit buffer pool total size is greater than two times the RLS_MAX_POOL_SIZE value, or 1728 MB. This mode has the following characteristics:
  - Initial_Free_UIC is reduced by 8, which indicates that the stealing criteria is very severe.
  - CIs with UIC greater than Initial_Free_UIC are stolen.
  - Each not stolen CI in the buffers has its UIC increased by the number of elapsed seconds since the last time.
  - Requests for new buffers are first served by stolen buffers.
  - If there are no buffers to steal, the request is put to sleep (task in wait state) until the next BMF stealing routine run.

The installation has no control over how each mode works. The only variable that affects the mode type is the RLS_MAX_POOL_SIZE value.

Figure 6-2 shows the RMF Monitor III RLSSC report with an example of one of the MVS systems (SC63). The example exceeds by two times the goal size for the below 2 GB (31-bit) buffer. Therefore, its LRU has reached Panic Mode. The 31-bit local buffer pool was decreased to 10 MB to see a practical effect of a local buffer being too undersized. Therefore SC63 reached 22 MB, which twice the size of the goal.

```
RMF V1R12     VSAM   LRU Overview   -  SANDBOX         Line 1 of 12
Command ===>                                                Scroll===> CSR

Samples: 120       Systems: 4     Date: 05/13/11   Time: 19.14.00   Range: 120   Sec

MVS           Avg CPU -  Buffer   Size -    Accel     Reclaim   ------  Read   ------
System          Time      Goal     High      %           %       BMF%    CF%    DASD%

SC63
  Below 2 GB    0.734     10 M     22 M     0.0        100      73.5   10.3    16.2
  Above 2 GB    0.092    500 M     10 M     0.0        0.0       0.0    0.0     0.0
SC64
  Below 2 GB    0.022     10 M      0 M     0.0        0.0       0.0    0.0     0.0
  Above 2 GB    0.023    500 M      0 M     0.0        0.0       0.0    0.0     0.0
SC65
  Below 2 GB    0.035     10 M     16 M     100        0.0       0.0    0.0     0.0
  Above 2 GB    0.028    500 M      0 M     0.0        0.0       0.0    0.0     0.0
SC70
  Below 2 GB    0.528     10 M     16 M     100        0.0      73.5   10.3    16.2
  Above 2 GB    0.089    500 M     10 M     0.0        0.0       0.0    0.0     0.0
```

Figure 6-2 VSAM RLS Activity by Storage Class - Sysplex Total View

6.2 VSAM new facilities in DFSMS V1.13

DFSMS 1.1.3 includes the following new facilities:

- Enhanced BMF performance
- A storage class option to disconnect RLS cluster from buffers
- VSAM Open first time failure date capture

6.2.1 Enhanced BMF performance in DFSMS 1.13

Three new BMF enhancements improve performance in your z/OS systems:

- Using time stamps rather than UIC for tracking CI buffer aging in the SMSVSAM data space buffer pools (31-bit)
- Placing CI buffers on the top of the BMF Read buffer chain when referenced
- Modifying BMF LRU buffer clean up to process only CI buffers at the end of CI buffer chains

BMF replacing UIC by time stamps for 31-bit CI buffers

As seen in 6.1.2, “Buffer management facility (BMF) concept” on page 108, the BMF routine maintains CI buffer pool population by stealing, when needed, aged CIs in buffers. This process uses an LRU algorithm. Before DFSMS V1.13, the data space buffer pool LRU routine used an UIC to track CI aging in buffers.

In DFSMS 1.13, the UIC is replaced with a time stamp. This time stamp eliminates the need for a data space BMF stealing routine to increment the UIC of each CI in buffers on regular

LRU cycles. As the UIC, the time stamp is reset when the CI buffer is referenced. The use of the time stamp is the same strategy implemented before DFSMS 1.13 for 64-bit buffers. The CI to be stolen is the one associated with the highest time stamp.

Implementing time stamps to track CI buffer aging in the data space buffer pools eliminates the frequent runs of the data space LRU cleanup routine. This routine is needed to increment UIC values.

With this UIC by time stamp replacement, the criteria Initial_Free_UIC used to decide which CIs are stolen is replaced by a sufficiently aged criteria. This criteria is the current time stamp minus the last reference time stamp. It has the following values, depending on the BMF stealing routine mode:

- Normal Mode. If the occupied buffer pool is less than 80%, the BMF stealing routine does not run at all. The aged criteria value is not needed.
- Maintenance Mode. If the occupied buffer pool is between 80% and 100% of the RLS_MAX_POOL_SIZE), the aged criteria is one hour.
- Accelerated Mode. If the occupied buffer pool is between 100% to 120% of the RLS_MAX_POOL_SIZE), the aged criteria is 5 minutes.
- Panic Mode. If the occupied buffer pool is greater than 120% of the RLS_MAX_POOL_SIZE), the aged criteria is still 5 minutes. However, the BMF stealing routine runs through the CI buffers more frequently.

Placing buffers on the top of the BMF read buffer

This improvement complements the previous UIC time stamp replacement. In releases before DFSMS V1.13, all the CI buffers were searched in a round robin approach. The CI buffers kept a static position in the queue.

With this enhancement, referenced CI buffers are moved to the front of the CI buffer read chain, ordering the chain by buffer age. When a requested CI is found in a buffer, the time stamp is reset. In addition, it is placed at the front of the appropriate read chain as the CI x in the top of the chain. See Figure 6-3 on page 113 for details.

In DFSMS V1.13, the chains of CI buffers are kept in age order. The oldest time stamps are on the end, newest on the front. When the BMF stealing routine is searching for possible reclaims, it starts at the end. It stops searching when it finds a time stamp newer than the aged criteria.

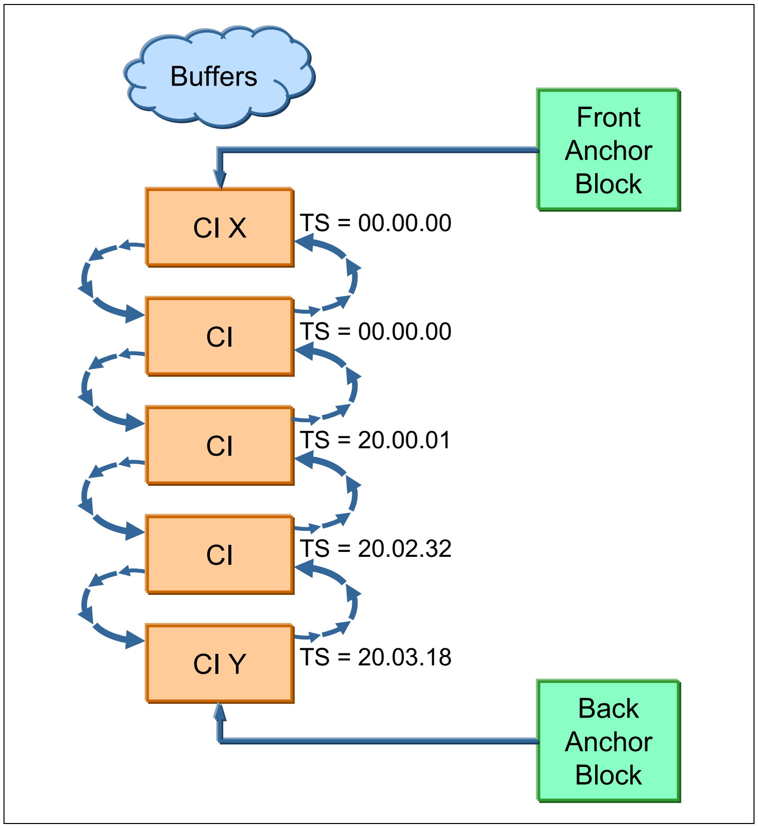


Figure 6-3 The CI buffer chain

The BMF stealing routine is enhanced in both types of buffer pools to search the read chains for CI buffers to steal. Note CI Y in Figure 6-3, which has a time stamp of 20.03.18. Stealing of a particular read chain ends when a CI buffer that is sufficiently aged for stealing is found. Age is the current time stamp minus the last reference time stamp. This process significantly reduces the amount of processor resources devoted to LRU activities for clusters with many buffers. Moving CI buffers to the front of the Read chain when referenced ensures that CI buffers that are aged are at the end of these chains.

This functioning is internal and not apparent to customers, except that they benefit from performance improvements by saving processor cycles, mainly with large data sets.

Modifying LRU buffer cleanup

Modifying the BMF stealing routine to process only CI buffers at the end of read chains reduces the BMF LRU cycle run time.

The major objective of these BMF modifications is to save BMF processor cycles and to produce a more precise LRU algorithm. When migrating to DFSMS 1.13, the AVG processor TIME column at RMF Monitor III has a perceived decrease (Figure 6-4). This column is the average processor time spent by BMF LRU processing during each reporting interval (milliseconds).

```
                     RMF V1R11 VSAM LRU Overview   - SYSPLEX            Line 1 of 2
Command ===>                                                          Scroll===> HALF

Samples: 120       Systems: 2     Date: 11/28/09   Time: 13.25.00   Range: 120   Sec

MVS           Avg CPU -  Buffer   Size -    Accel     Reclaim    ------   Read   ------
System          Time      Goal     High       %          %         BMF%    CF%    DASD%

SYS4
  Below 2 GB    0.023      MAX      1 M      0.0        0.0        0.0    0.0     0.0
  Above 2 GB    3.543      MAX      1 M      0.0        0.0       97.5    0.0     2.5
SYS5
  Below 2 GB    4.457      MAX      1 M      0.0        0.0        0.0    0.0     0.0
```

Figure 6-4 RLSLRU RMF Monitor III report

Compatibility and coexistence

There are no compatibility and coexistence issues associated with implementing BMF improvements; and there are no external parameters to be declared by the installation.

6.2.2 Storage class option to disconnect RLS cluster from buffers

This enhancement also improves performance in your z/OS systems. DFSMS V1.13 includes a new storage class option (SCDDCLS) to disconnect an RLS cluster from the buffer pool as soon as the RLS cluster is closed. Disconnect here means to free up all the buffers that contain CIs from the closed RLS cluster.

Description

Before DFSMS 1.13, the RLS cluster remained connected for approximately 10 minutes after a close. With the new storage class option, you can eliminate the time delay to immediately release valuable buffer pool storage. This process can be done for those clusters that you do not intend to quickly reopen in RLS buffering mode.

This local z/OS function in a shared VSAM RLS cluster is accessed by several MVS systems, such as MVS1 and MVS2. The last close issued on MVS1 system triggers a disconnect on MVS1 local buffers if SCDDCLS was declared. The cluster remains opened and connected with MVS2 local buffers.

The SCDDCLS attribute indicates whether the cluster is disconnected immediately upon last closing the cluster in the z/OS system or stay connected for a short time. When set to YES, the cluster is immediately disconnected, as soon as the cluster is last closed. When the value is NO, the sphere stays connected for a short time after the last CLOSE. The default value is NO, the same as releases before DFSMS V1.13.

Figure 6-5 describes the storage class option Disconnect Sphere at Close.

```
 Panel  Utilities  Scroll  Help
----------------------------------------------------------------------
DGTICSC2                     STORAGE CLASS DISPLAY        Page 2 of 2
Command ===>

CDS Name  . . . . . : IBMUSER.MYSCDS
Storage Class Name  : SC1

Guaranteed Space  . . . . . . . . . : NO
Guaranteed Synchronous Write  . . . : NO
Multi-Tiered SGs  . . . . . . . . . :
Parallel Access Volume Capability : NOPREF
Cache Set Name  . . . . . . . . . . :
CF Direct Weight  . . . . . . . . . :
CF Sequential Weight  . . . . . . . :
Lock Set Name . . . . . . . . . . . :
Disconnect Sphere at CLOSE  . . . . : NO




Use UP Command to View previous Page;
Use HELP Command for Help; Use END Command to Exit.
```

Figure 6-5 ISMF storage class definition

AMS DCOLLECT collects the information of this new attribute and records it in record type SC. This is a benefit for customers if their environment is often short of buffer pool space and the same clusters are not reopened.

Prior releases offer more benefits for customers running applications that repeatedly close and reopen the same clusters in RLS mode. Turning on this feature might even degrade the performance because BMF must reprime all the CI buffers it needs when reopening the same clusters. Customers can mix these two approaches because different clusters can belong to different storage classes.

Compatibility and coexistence

Decide whether to migrate clusters to the disconnect option at each cluster basis and at each z/OS basis. Do not use this function if the cluster will be opened soon.

6.2.3 VSAM OPEN first time failure data capture

This enhancement improves RAS in your z/OS systems. DFSMS 1.13 release includes a mechanism for taking an SVC memory dump at VSAM open when logic errors are encountered. The running task does not abend. After the memory dump is generated, VSAM goes on starting and passing an open return code to the application. This function applies to any type of VSAM buffering method, RLS or not.

Description

Before DFSMS 1.13, IBM support had to go through the console message IEC161I to find the location of non-abending task open logical errors. After the location was determined, a slip trap was set with the specific location and sent to the customer to generate a memory dump. This process was time-consuming and not a first-time data capture.

Some of the open logical problems that might cause the described behavior:

- OPEN a catalog data set in a non-zero key
- GETMAIN error
- Error occurred on an IEFDDSRV call to obtain UCB addresses
- Enqueue resource failed

The installation sees a memory dump being taken when VSAM open encounters internal logic errors. If the reason that causes the memory dump cannot be determined, open a problem management record (PMR) for further assistance.

In this new implementation, in addition to the current IEC161I pp-pdf message, two other messages are written to the console as shown in Example 6-2. These messages list the failing load module name and the specific offset, thus simplifying the task of problem determination and time spent.

Example 6-2 The two messages

```
IEA045I AN SVC DUMP HAS STARTED AT TIME=12.15.44 DATE=04/12/2010 840 FOR ASID
(002A) QUIESCE = YES
IEA794I SVC DUMP HAS CAPTURED: 721 DUMPID=001 REQUESTED BY JOB (VPFFDC01) DUMP
TITLE=VSAM O/C/EOV FFDC DUMP - IDA0192Z + 0000521A RC=020CCC=053
```

In future releases, IBM plans to extend this function to VSAM close and VSAM end-of-volume (EOV) routines.

Compatibility and coexistence

The following publications are updated to include this functionality:

- *z/OS DFSMSdfp Diagnosis*, GY27-7618
- *z/OS MVS System Messages Volume 7 (IEB - IEE)*, SA22-7637.

7

Catalog enhancements

This chapter addresses changes to the catalog environment. DFSMS V1.13 introduces improvements to catalog settings, scalability, and flexibility. DFSMS V1.13 also adds functions to dynamically change the catalog environment to improve availability.

This chapter includes the following sections:

- Introduction
- New CATALOG PARMLIB member
- Alias number constraint relief
- Catalog: VVDS expansion
- Catalog RAS: Replace catalog pseudo close with VSAM close
- IDCAMS enhancements

7.1 Introduction

This chapter assumes that you know Catalog Management basics. This chapter explains the changes in this new release to Catalog Services, and align those changes with current features. It also explains how to implement new functions along with their associated benefits and considerations.

7.2 New CATALOG PARMLIB member

Before DFSMS V1.13, definitions to the Catalog environment had to happen through the SYSCATxx member in SYS1.NUCLEUS or through the LOADxx member in SYS1.PARMLIB. If a LOADxx member was found, a SYSCATxx member would not be considered. This implementation has limited flexibility because the available 80 characters in the PARMLIB data set were almost used up. Also, because changes to the catalog setup would often require an IPL, few could be made dynamically. However, a few dynamic changes could be performed through MODIFY CATALOG commands. A more dynamic way of changing the catalog setup and adding new functions to the catalog environment was needed.

7.2.1 Description

In DFSMS V1.13 a new PARMLIB member with the naming convention of IGGCATxx can be added to the concatenation, holding parameters exclusively for the catalog environment. To make this PARMLIB member active, code a CATALOG=xx statement in the IEASYSxx member in PARMLIB. This parameter points to one or more IGGCATxx PARMLIB members with the suffix specified. These members are processed in the order given. If you do not specify CATALOG= xx, IGGCAT00 is searched in the PARMLIB concatenation. If that is not found, then the defaults are used. For a sample CATALOG=xx specification in IEASYSxx, see Example 7-1.

Example 7-1 IEASYSxx pointer to IGGCATxx member

```
CATALOG=aa (specifies one member)
CATALOG=(aa,bb,..) (specfies more members)
```

A few rules apply for IGGCATxx:

- A catalog member must have a valid parameter as shown in Table 7-1 on page 119.
- Input ends at the end of PARMLIB member (no other termination is needed).
- Commas and blanks are regarded as delimiters.
- Delimiters are not needed between parameters (a right parenthesis is sufficient between parameters).
- Comments can be specified between parameters, beginning with '/*' and ending with '*/'.

An example of coding the new IGGCATxx member is shown in Example 7-2.

Example 7-2 Sample IGGCATxx member

```
VVDSSPACE(40,40)
NOTIFYEXTENT(85)
TASKMAX(360)
```

Using IGGCATxx is optional. If CATALOG=xx is specified in the IEASYSxx member, the IGGCATxx member takes precedence over LOADxx or SYSCATxx member. If an IGGCATxx member was not defined, the default specifications are used. Likewise, if one or more catalog members are empty, a message is displayed and the process continues with the next member. If you specified a parameter more than once, the last specification is used.

Use the new command **D IPLINFO, CATALOG** to display the catalog settings.

The parameters can be specified in the IGGCATxx member, as shown in Table 7-1. They do not need to start in column 1, but must be located between columns 1 and 71. Using this new design, setting and changing parameters will carry over well in future releases.

> **Tip:** The IGGCATxx parameters are processed at initial program load (IPL) time or when catalog address space (CAS) is restarted.

Table 7-1 IGGCATxx parameters

Parameter	Description
VVDSSPACE (pri,sec)	Specification of primary and secondary allocation of the VSAM volume data set (VVDS) in tracks. The default is 10 tracks. Previously, this specification was set dynamically by using a MODIFY CATALOG command, but was in effect only until next IPL. Now it will always be in effect after IPL, if set.
TASKMAX(nn)	The value for the maximum # of CAS service tasks that can be active concurrently. The default is 180, the minimum is 24, and the maximum is 360. Previously the value was specified in SYS1.NUCLEUS(SYSCATxx). If specified in IGGCATxx, this value takes precedence over the SYSCATxx specification. MODIFY CATALOG can be used to change the value of TASKMAX. However, it can only decrease the value or to reset it to 90% of the value given in the SYSCATxx member. MODIFY changes last only until the next IPL. If IGGCATxx is used to define this parameter, it is also initiated after an IPL.
NOTIFYEXTENT(nn)	Notification percentage level of a catalog running full (default is 80%). The warning level can be changed by a dynamic MODIFY CATALOG command, but will in that case only last until the next IPL. This IGGCATxx specification will be active after IPL also.
DELFORCEWNG (YES|NO)	To prevent inadvertent forced deletion of data sets or VSAM volume record (VVR) entries, use IDCAMS to issue a write to operator with reply (WTOR) to the master console. Perform this process before proceeding with a delete command.
DSNCHECK (YES|NO)	Catalog data set name checking does not allow data sets to be cataloged that contain data set qualifiers that begin with a numeric character. Turn it off during cataloging of numeric tape data sets.
SYMREC (YES|NO)	Allows unauthorized programs to write symptom records to the logrec data set, to a job log, or both. This process can be through the use of the SYMREC macro. The SYMREC authorization exit (ASREXIT) must be in effect.
UPDTFAIL (YES|NO)	Enables/disables the message IEC390I from been issued when a Virtual Storage Access Method (VSAM) update request against a catalog abnormally terminates.
VVRCHECK (YES|NO)	Enables/disables the enhanced VVR record checking feature during VVDS I/O. More extensive checks of VVR content are made before writing and immediately following reading. This setting is intended primarily for situations when records are being corrupted
DELRECOVWNG (YES|NO)	Enables/disables the ability to issue a warning message when the DELETE UCAT RECOVERY command is issued.
EXTENDEDALIAS (YES|NO)	Enables/disables the ability to create extension records for user-catalog aliases.

7.2.2 Compatibility and coexistence

This support is not apparent because no changes are required when upgrading to DFSMS V1.13 if you want to continue with the previous implementation. If you want, you can specify a CATALOG=xx parameter in the IEASYSxx member in SYS1.PARMLIB. Or you can populate an IGGCATxx member with the parameters you want.

7.3 Alias number constraint relief

Currently the number of alias pointers to a user catalog is limited to approximately 3,500. When reached, this limit can increase the need for defining more user catalogs. This configuration increases the amount of administration needed for backup and maintenance of catalogs.

7.3.1 Description

The reason for the limited number of alias pointers is that the maximum record size of the catalog is 32K (32768 bytes). The User Catalog Connector record contains the volume the user catalog is on and all aliases associated with the user catalog. This maximum length can be exhausted depending on the length of the individual aliases, and on the usage of multi-level aliases.

DFSMS V1.13 removes this constraint by enabling a new catalog record type V, Catalog Connector Extension record, to hold extensions to the connector record. The maximum number of 255 extension records raises the limit from the current 3,500 to 500K aliases. After it is created, the extension record exists forever along with the catalog connector record, even if aliases in the extension record get deleted.

A new command, **`F CATALOG,ENABLE(EXTENDEDALIAS),`** is provided to enable the new support. After the command is issued, a flag is set and the alias constraint relief is active. The support is disabled by default, but you can also disable the feature by issuing this command:

```
F CATALOG,DISABLE(EXTENDEDALIAS)
```

The support can also be enabled or disabled with the new PARMLIB member IGGCATxx parameter **`EXTENDEDALIAS(YES|NO)`**.

7.3.2 Compatibility and coexistence

In a shared environment, all MVS systems must be at DFSMS V1.13. Prior level systems cannot recognize the extension records.

However, a system before z/OS V1.13 can access a catalog that has user catalog connector extension records built by a z/OS R13 system or higher. Toleration program temporary fixes (PTFs) are supplied that allow the aliases to be included in the alias search. The PTFs also prohibit EXPORT DISCONNECT from systems before z/OS V1R13 for those user catalog connectors that have user catalog connector extension records.

DEFINE ALIAS allows new aliases on systems before z/OS V1.13 only if the alias fits in the existing user catalog connector record. For systems before z/OS V1.13, EXPORT of user catalogs does not move user catalog connector extension records. IMPORT *on pre-z/OS V1R13 systems* that use a portable data set created on a V1.13 might be unable to DEFINE all of the aliases. Existing error processing reports that these aliases might not be defined.

7.4 Catalog: VVDS expansion

Bigger logical volumes now call for more attention to VVDS allocation size. Extended address volumes (EAVs) require larger logical volumes. Ensure that the allocation sizes of the VVDS are large enough to store all the data sets that might be allocated on a single volume.

7.4.1 Description

Currently the capacity for the VVDS is limited to ‘FFFF’X (65535) control intervals (CIs). Each CI can point to one or more data sets. However, for large volumes, combined with small data set sizes, the limited number of CIs can prevent you from fully using the volume. The maximum size of the VVDS is currently 364 cylinders (5460 tracks).

DFSMS V1.13 introduces a size change that enables a VVDS to store ‘FFFFF’X (1048575) CIs, a 16 times increase in capacity. The VVDS allocation size in cylinders can now reach 5825 cylinders (87375 tracks).

7.4.2 Compatibility and coexistence

All systems in a shared environment need to be on the DFSMS V1.13 level to upgrade to full support. Migration support is not apparent in the sense that the old and new implementations work together. This functions when VVDSs shared between systems are under the previous allocation size. Lower-level systems are able to read and use a R1.13 VVDS below the ‘FFFF’X’ limit. These prior level systems can read, delete, insert, and update VVRs beyond the previous limit of ‘FFFF’X CIs with the toleration authorized program analysis report (APAR) for expanded R1.13 VVDSs.

A DFSMS V1.13 system can extend a VVDS created by a lower-level system beyond the x’FFFF’ limit. However, a VVDS cannot be extended to the higher number of CIs by a system using a lower level of DFSMS. This limitation is true even if the VVDS was created by a DFSMS V1.13 with this support.

7.5 Catalog RAS: Replace catalog pseudo close with VSAM close

In DFSMS releases before V1.13, the closing of a catalog is not done by a normal VSAM close, which creates an SMF record type 64. Instead Catalog Management does a pseudo close of the catalog by invalidating the access method control block (ACB). To create an SMF record for catalog closes, this process has been changed. In DFSMS V1.13, the closing of the catalog is now a standard VSAM close. It creates two System Management Facilities (SMF) records: One at the closure of the index component and one at the closure of the data component. It is now possible to see jobname Catalog in the SMF record type 64.

7.6 IDCAMS enhancements

The following IDCAMS enhancements are described in detail:

- AMS LISTCAT NOIMBED and NOREPLICATE removal
- AMS LISTCAT with CDILVL
- AMS: Delete UCAT WTOR

7.6.1 AMS LISTCAT NOIMBED and NOREPLICATE removal

These two index options (IMBED and REPLICATE) use physical 3390 characteristics that no longer exist with RAID controllers such as the IBM DS8000®. Starting with DFSMS 1.5, the options are no longer valid in IDCAMS DEFINE. However, data sets already defined before DFSMS 1.5 with such options are still supported.

DFSMS 1.11 contains a new Health Check warning message that detects such attributes for catalogs. Starting in DFSMS 1.12, when these data sets are migrated or dumped by DFDSS or DFHSM, these attributes disappear at restore or recall.

Currently, when VSAM data sets are listed by using AMS LISTCAT, IMBED or NOIMBED, and REPLICATE or NOREPLICATE, attributes are displayed and printed. At DFSMS 1.13, the AMS LISTCAT command no longer shows either NOIMBED or NOREPLICATE when displaying attributes for a particular data set. Instead LISTCAT displays only the IMBED and REPLICATE attributes of a data set.

7.6.2 AMS LISTCAT of catalog CDILVL

There are two options in the LISTCAT verb of IDCAMS to filter the catalog entities to be listed:

- ENTRIES(entryname [entryname...])
- LEVEL(level)

See Example 7-3 for an example of the filtering.

Example 7-3 Example of using ENTRIES and LEVEL for filtering a LISTCAT

```
Suppose a catalog contains the following names:
1. A.A.B
2. A.B.B
3. A.B.B.C
4. A.B.B.C.C
5. A.C.C
6. A.D
7. A.E
8. A
If ENTRIES(A.*) is specified, entries 6 and 7 are listed.
If ENTRIES(A.*.B) is specified, entries 1 and 2 are listed.
If LEVEL(A.*.B) is specified, entries 1, 2, 3, and 4 are listed.
If LEVEL(A) is specified, entries 1, 2, 3, 4, 5, 6, and 7 are listed.
```

DFSMS 1.13 introduces a new parameter with the options CDILVL and NOCDILVL. It applies only to filter LEVEL option of LISTCAT. CDILVL specifies that DATA and INDEX objects in CLUSTERs and AIXs be listed if at least one of the three objects matches the LEVEL pattern. That is, with the CDILVL option specified at LEVEL, LISTCAT displays all components in a cluster and alternate index (AIX) when at least one name matches the LEVEL pattern. This name can be the base, the data, the index, or the cluster. The options have these differences:

- NOCDILVL specifies that only the objects whose patterns match the LEVEL pattern are listed. NOCDILVL is the default.
- CDILVL specifies that DATA and INDEX objects in clusters and AIXs are listed if at least one name matches the LEVEL pattern.

7.6.3 AMS: DELETE UCAT WTOR

With DFSMS 1.13 support, IDCAMS now issues a confirmation WTOR message (IDC1999I) before deletion is done when running a DELETE UCAT RECOVERY. This message is similar to the existing confirmation WTOR issued in a DELETE UCAT FORCE scenario.

To enable or disable this message, use the modify (F) commands shown in Example 7-4.

Example 7-4 Enabling and disabling the WTOR

```
F CATALOG,ENABLE(DELRECOVWNG)
F CATALOG,DISABLE(DELRECOVWNG)
```

By default, the WTOR is disabled. The message can also be enabled or disabled by using the PARMLIB member IGGCATxx parameter **DELRECOVWNG (YES|NO)**.

8

zHPF enhancements

In Enterprise Disk DS8700 and DS8800 microcode Release 6.2, non-VSAM access methods (BSAM, QSAM, and BPAM) support zHPF, improving the overall I/O and processor performance. Additionally, DS8K 6.2 allows Media Manager to issue zHPF channel programs suited for DB2 List Prefetch. Many of the new features have been made available in R11 and R12 through program temporary fixes (PTFs).

This chapter covers the basics of zHPF. It also includes a description of the new enhancements of zHPF and the Basic Access Method (BAM) use of zHPF channel program commands.

This chapter includes the following sections:

- zHPF background
- BAM usage of zHPF
- zHPF protocol enhancements
- SAM internal trace facility

8.1 zHPF background

IBM System z® High Performance FICON® (zHPF) is an enhancement to the Fibre Channel connection (FICON) protocol. It reduces the number of information units exchanged between a channel and the controller during an I/O operation. Currently, sending small blocks of data over FICON involves additional handshaking between the channel engine and the Fibre Channel (FC) adapter in the control unit. Now zHPF has reduced the processor usage of this process. zHPF allows the collapsing of command chains and data-chained channel command word (CCW) strings into a single command called the transport control word (TCW). This configuration provides a substantial performance improvement in data transfer, especially in online environments.

In 2007, the IBM Storage and IBM Systems Technology Groups, along with z/OS development, combined to provide zHPF channel programming support. z/OS support shipped in DFSMS V1.9. In 2012, these development groups provided support for these major enhancements to the z/OS channel programming protocol:

- Media Manager support for ILR in zHPF channel programs
- Media Manager support for BiDirectional (BiDi) zHPF channel programs (requires new channel hardware)
- Media Manager support for format writes in zHPF channel programs
- BAM (BSAM, QSAM, BPAM) support for zHPF channel programs for non-extended format data sets.

There is also z/OS support for enabling, disabling, and detecting zHPF support.

8.1.1 zHPF support in System z architecture

Enhancements have been made to System z architecture and the FICON protocol. IBM introduced High Performance FICON, known as zHPF, with the IBM z10™ processor. It is further enhanced by the new channels in the IBM zEnterprise® System (zHPF requires certain control unit and z/OS levels). zHPF is a channel I/O architecture that replaces Modified Indirect Data Address Words (MIDAWs). Just as MIDAWs improved the efficiency of the channel subsystem, zHPF goes further towards greatly improved channel efficiency. The zHPF architecture encompasses new protocols in the communication between the channel and the control unit. It is further enhanced by a new channel program that is implemented by media manager. zHPF is transparent to DB2, just as MIDAWs were. Currently, not all media manager I/Os can use zHPF. I/Os that access discontiguous pages are ineligible, as are format-writes. On the z10, I/Os that read or write more than 64K are ineligible, but this restriction is removed on the zEnterprise system.

These enhancements optimize I/O for online transaction processing workloads. When used by the FICON channel, z/OS, and the DS8000 control unit, zHPF can help reduce processor usage and improve performance in these ways:

- The maximum number of I/Os can be improved by up to 100% for small data transfers that are able to use zHPF.
- Production workloads with a mix of data transfer sizes can see up to 30 - 70% of FICON I/Os by using zHPF, saving 10-30% channel utilization.
- Sequential I/O transferring less than a single track size (48 KB per I/O) can also benefit.
- Data accessed by DB2, PDSE, VSAM, and extended format SAM can benefit from zHPF and the new channel programs built by Media Manager.

zHPF support has these requirements:

- Only available on IBM System z10®, on FICON Express2 and Express4
- z/OS V1R8 and later (or V1R7 with lifecycle extension 5637-A01)
- IBM DS8000 Release 4.1 (LMC level 5.4.1.xx, bundle version 64.1.x.x) or later
- Priced license feature with a monthly maintenance charge

8.1.2 New channel program in zHPF

In zHPF, all the traditional CCWs are replaced by a single TCW as shown in Figure 8-1. The DS8700 and DS8800 DASD control units support both the traditional channel program that consists of CCW chains and the newer zHPF channel program. CCW chains are also called a command mode channel program. The newer zHPF channel program is called a transfer mode channel program. The TCW points to the Transport Command Control Block (TCCB). This configuration allows multiple channel commands to be sent to the control unit as a single entity instead of a stream of individual commands.

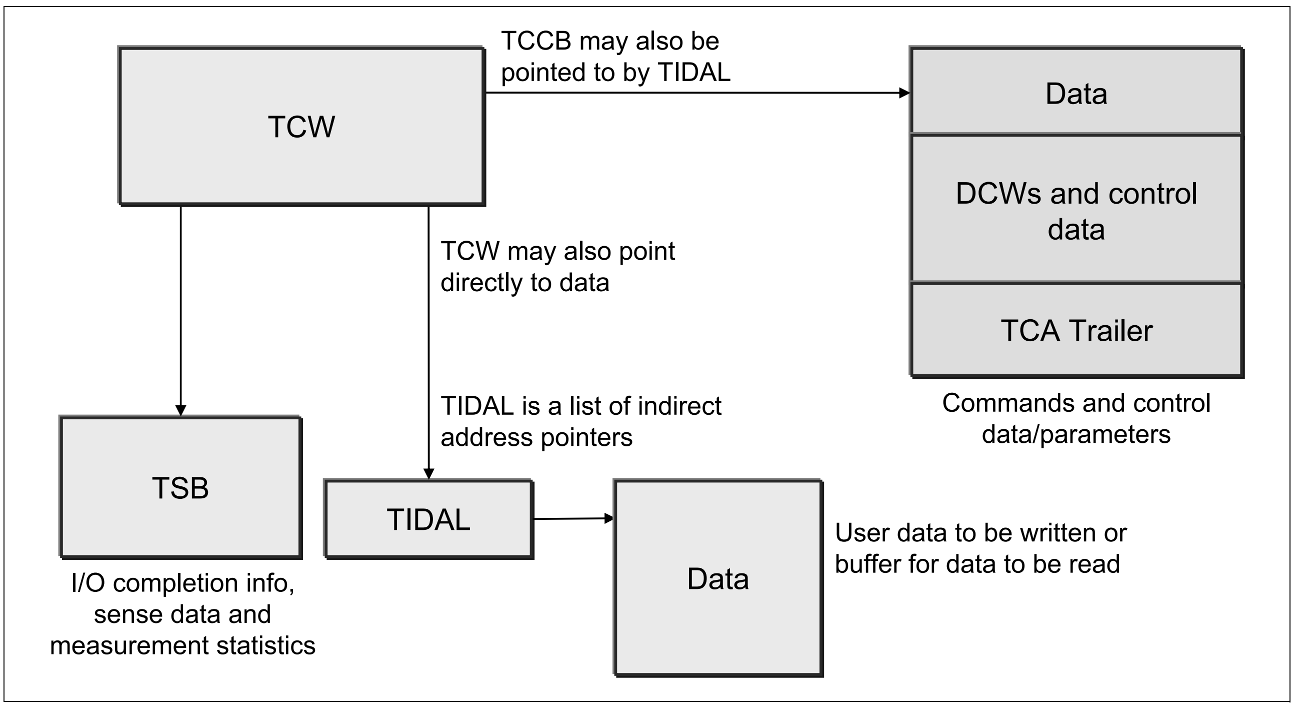


Figure 8-1 zHPF control blocks

To use zHPF, a chargeable High Performance FICON feature for System z is needed on the DS8000 (feature 7092). After the hardware supports zHPF, the function is easily activated. ZHPF=YES must be set in the IECIOSxx member in SYS1.PARMLIB (default for this parameter is ‘NO’). This parameter can also be set dynamically with a SETIOS command.

To display status of the function, you can issue the display command **D IOS,ZHPF.** Example 8-1 shows an unsupported processor.

Example 8-1 Output from D IOS,ZHPF on unsupported processor

```
IOS630I 14.10.15 ZHPF FACILITY 600
HIGH PERFORMANCE FICON FACILITY NOT SUPPORTED BY PROCESSOR
```

Example 8-2 shows the results of the same command on a supported processor.

Example 8-2 Issuing D IOS,ZHPF on supported processor

```
D IOS,ZHPF
IOS630I 21.17.10 ZHPF FACILITY 677
HIGH PERFORMANCE FICON FACILITY IS ENABLED
```

Support level can also be displayed at the device by using the **D M=DEV(uuuu)** command as shown in Example 8-3.

Example 8-3 Output from display matrix: D M=DEV(a000)

```
IEE174I 15.37.28 DISPLAY M 367
DEVICE A000   STATUS=ONLINE
CHP                   B8   BC
ENTRY LINK ADDRESS    ..   ..
DEST LINK ADDRESS     0D   0D
PATH ONLINE           Y    Y
CHP PHYSICALLY ONLINE Y    Y
PATH OPERATIONAL      Y    Y
MANAGED               N    N
CU NUMBER             A000 A000
MAXIMUM MANAGED CHPID(S) ALLOWED:  0
DESTINATION CU LOGICAL ADDRESS = 00
SCP CU ND         = 002107.951.IBM.75.0000000AN901.0006
SCP TOKEN NED     = 002107.900.IBM.75.0000000AN901.0000
SCP DEVICE NED    = 002107.900.IBM.75.0000000AN901.0000
HYPERPAV ALIASES CONFIGURED = 14
FUNCTIONS ENABLED = MIDAW,ZHPF
```

8.1.3 Media Manager and zHPF

Media Manager is an I/O driver in z/OS. It is logically located between the access method and the Input/Output Supervisor (IOS) during the execution of an I/O operation. The Media Manager interfaces are used by various components of the z/OS operating system and vendor products. These products run on z/OS to access certain z/OS data sets and their metadata.

In zOS V1.9, the Media Manager was modified to support the building and execution of zHPF channel programs. The Media Manager builds and runs channel programs that conform to the zHPF channel program format under these circumstances:

- The device supports the zHPF channel program protocol
- The request (or channel program to be built) conforms to the limitations imposed by zHPF protocol

Media Manager has these limitations:

- It can run read or update write operations, but not both in a single channel program
- Only contiguous records can be accessed
- There are limitations on the amount of data to be transferred and the number of contiguous tracks to be accessed

New support in the DS8000 control unit removes some of these limitations. The two line items addressed in this spec support these updates to the zHPF protocol. The two line items for Media Manager are:

- Imbedded Locate Record (ILR) Support
- z/OS List Prefetch (BiDi)

In addition, new support in the DS8000 implements format write support for zHPF Channel programs. This support is enabled as part of the DS8000 support for SAM zHPF channel programs.

Media manager converts the channel program written by the access method (with CCWs) into a zHPF TCW. This conversion has been done since z/OS 1.9 because not all the access methods are aware of zHPF. It occurs if the ZHPF=YES parameter is set in the IECIOSxx. Such conversion might not be possible if the device does not support zHPF protocols. The conversion is not possible in the following cases:

- Reads and updates are possible, but cannot happen within the same channel program
- Only contiguous records are to be accessed
- Limited channel program size
- No data chaining, as used by Large Blocks support
- No Suppress Length Indicator (SLI)
- No Program Controlled Interrupts (PCI), as used by address space manager (ASM)
- No Suspend Resume, as used by ASM
- No Status Modifier logic (used to skip transfer in channel (TIC) CCW in count key data (CKD))
- No dynamic appending
- No initial status interruption (z bit)

To discover if some of these limitations are lifted in the future, Media Manager always detects if the hardware supports the requested function. Thus, access methods that do not start Media Manager (such as zFS, DB2, EXCP users, and DFSort) cannot use zHPF.

8.2 BAM usage of zHPF

Basic Access Method (BAM) is a set of access methods that includes BSAM, BPAM, and QSAM. An access method writes channel programs to be started by the channel for the I/O protocol dialog with the controller.

BAM uses Media Manager to run I/O to extended format sequential data sets and partitioned data sets extended (PDSEs). It builds its own channel programs for all other types of sequential data sets and partitioned data sets (PDSs). In V1R13 (and in R11 and R12 by using SPE), BAM builds zHPF channel programs for most functions that involve non-extended format sequential data sets and PDSs.

zHPF architecture allows the more efficient packaging (chaining command and data packages). Small block I/Os benefit from this configuration.

This support is not apparent to the application and is based on changes in the DS8000 microcode only.

If the device is not zHPF capable, BAM discovers this and builds conventional CCWs instead. When a supported I/O is scheduled, BAM builds the channel program that supports zHPF for these data sets types:

- Basic format sequential data sets (which cannot be larger than 64K tracks)
- Large format sequential data sets
- Members within partitioned data sets (not entire PDSEs, which use Media Manager)

For VSAM and PDSEs, Media Manager is still involved in building channel programs.

8.2.1 Implementing BAM

To implement this BAM feature, IBM DS8000 6.2 licensed code is required. Activation requires update to IGDSMSxx member in SYS1.PARMLIB. A new parameter, **`SAM_USE_HPF(YES)`,** must be specified. If this parameter is specified, BAM starts using the new TCW. Hardware support must be available and zHPF must be set to 'YES' in the IECIOSxx member. This function can also be activated through the SETSMS command by issuing **`SETSMS SAM_USE_HPF(YES)`**.

The status of the function can be displayed by using the command **`D SMS,OPTIONS`**.

Remember: SAM_USE_HPF parameter defaults to 'YES' in V1R13, which is a change from R11 and R12.

8.2.2 Migration

Implementation is seamless because BAM uses traditional CCWs until support is enabled by using these settings:

- An appropriate microcode level in the DS8000 (level R6.2)
- HPF settings in IECIOSxx and IGDSMSxx members in SYS1.PARMLIB

8.3 zHPF protocol enhancements

The first deployment of zHPF protocol had some limitations as described in 8.1.3, "Media Manager and zHPF" on page 128. Support in the DS8000 (license code R6.2) addresses some of these limitations.

8.3.1 Imbedded Locate Record support

Locate record is the name of a CCW, and in zHPF protocol is a function described in the TCCB. It indicates to the controller the location of the physical record (block) to be transported during the I/O operation. This location includes the cylinder number (CC), the track (head) number (HH), and the physical record number (R).

The TCCB has all the information that the controller needs to start the I/O. The channel sends it to the controller. The TCCB has at least one Device Command Word (DCW), with commands equivalent to former CCWs such as Define Extent, Locate Record, Read, or Write (Figure 8-2).

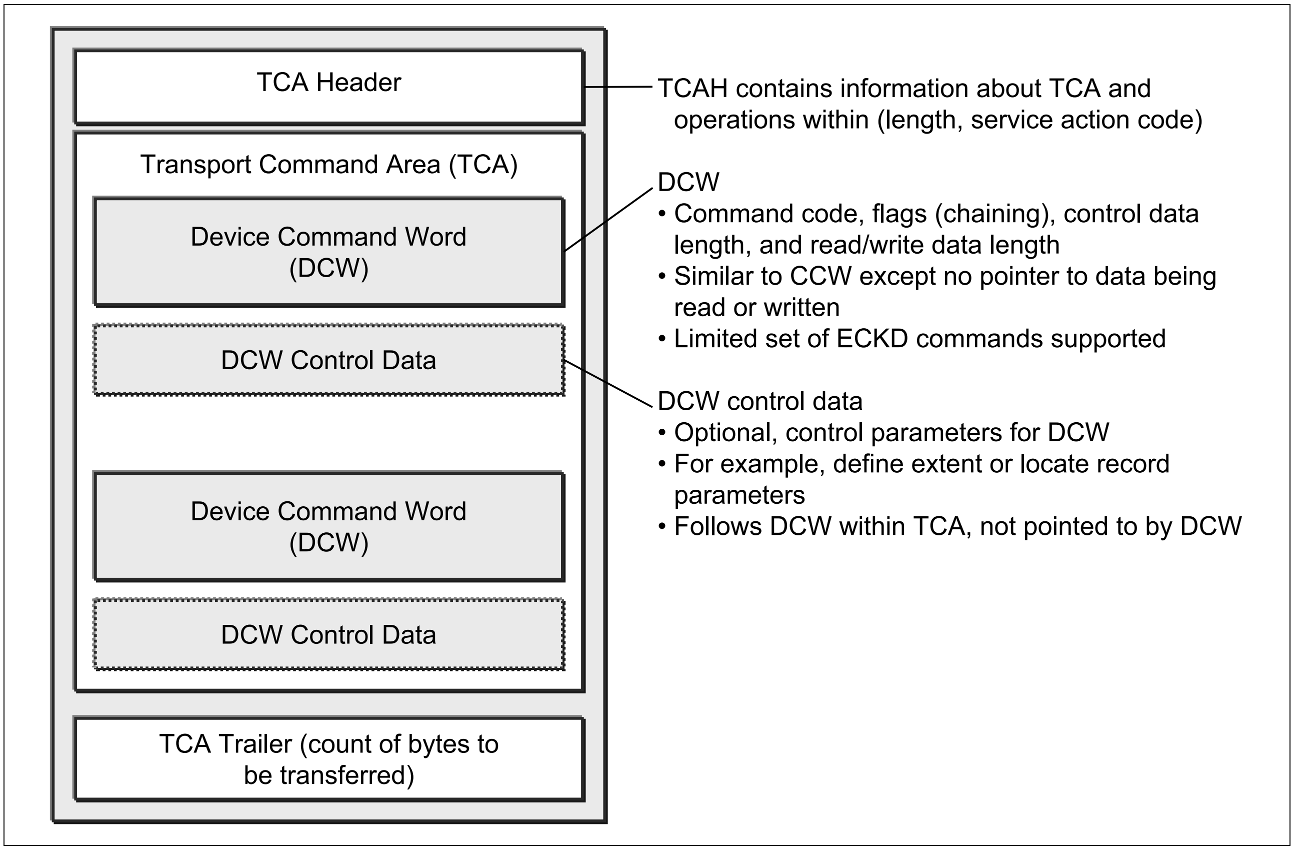


Figure 8-2 TCCB layout

A TCCB contains one or more DCWs. The TCCB is limited in size to 240 bytes, so the number of DCWs is also limited. Each DCW defines the location for the I/O. Each DCW also defines the operation (read, update write, or format write) to be run at that location.

> **Tip:** BiDi support as described in 8.3.2, "IOS list prefetch (BiDi)" on page 131 eliminates this limitation.

8.3.2 IOS list prefetch (BiDi)

The limited size of a TCCB previously meant that there were a maximum number of searches that could be contained in one channel program. If this number was exceeded, a new channel program had to be built by Media Manager.

The DS8000 is updated to support IOS List Prefetch, also called bidirectional (BiDi) zHPF channel programs. This support enables Media Manager to build an extension TCCB (called TCAX) and be able to build a TCCB that is larger than 240 bytes. BiDi requires specific channel hardware.

8.3.3 Format write support

New zHPF support in the DS8000 implements format write support for zHPF channel programs. The channel program has these sets of write operations:

- Format writes that store a count and data of the physical record. These are used for sequential writes, such as in QSAM and DB2 (for formatting table spaces).
- Modified writes that store only the data portion of the physical record. These are used for random writes such as in DB2.

This support is enabled as part of the DS8000 support for SAM zHPF channel programs. Media Manager already supported format writes, and now BAM in DFSMS V1.13 supports them as well.

8.4 SAM internal trace facility

This DFSMS 1.13 release provides a trace facility to improve the serviceability of the BSAM, QSAM, and BPAM access methods. This trace facility can be used by IBM service personnel to create a trace table that records a chronology of internal events.

9

PDSE enhancements

This chapter provides information about the DFSMS V1.13 enhancements to improve the diagnostic files. These files are used for problem determination tasks in the partitioned data set extended (PDSE)-related code (a component of DFSMSdfp). This chapter includes the following sections:

- PDSE recovery background
- PDSE enhancements summary
- PDSE diagnostic command operands
- Validation of PDSE data sets
- System procedure library as a PDSE

9.1 PDSE recovery background

A PDSE has many advantages when compared with partitioned data set (PDS) organization:

- No need to compress the directory
- Faster directory search
- Better space management for directory and members
- Better integrity and performance when a PDSE data set is shared among program tasks
- More intelligent caching (64 bit storage for directories and hiperspace for members)

Figure 9-1 shows the design of the PDSE directory and members.

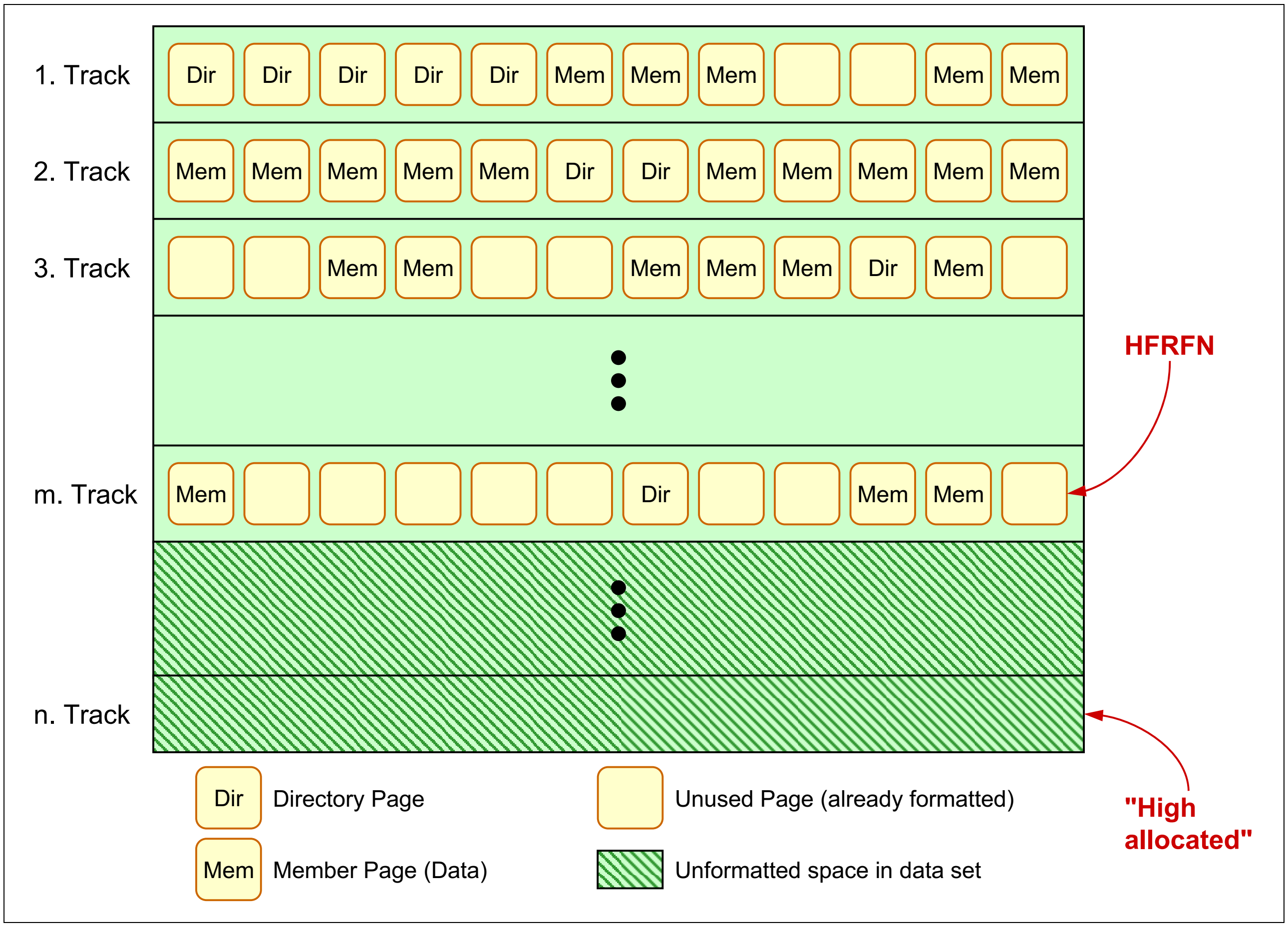


Figure 9-1 PDSE directory and members design

However to implement such functions, the internal complexity is greater than a PDS. To handle this implementation (before DFSMS V1.13) a non-restartable address space was introduced, the SMSPDSE, with the following properties:

- It receives the PDSE I/O requests from clients' address spaces through cross memory. That is, the requesting task connects to a PDSE data set through the SMSPDS address space.
- It issues Getmains for common storage area (CSA) subpools to store internal PDSE data sets control blocks.
- It owns storage and hiperspaces used for buffering.
- It owns ENQs, latches, and locks to ensure data integrity in PDSE data sets.

Although protected by the ESTAE mechanism (to decrease the probability of abends), the SMSPDSE address space is non-restartable. The reason is PDSE data sets are concatenated at the LNKLST and an address space cannot be created without using library lookaside (LLA) global connections. This configuration creates a situation where a failure (hang, deadlock, out of storage) during the recovery of this address space might cause all PDSE data sets to be inaccessible. You might need an initial program load (IPL) to free them.

To address this problem (before DFSMS V1.13), in addition to the SMSPDSE address space, a new restartable address space was introduced, SMSPDSE1. It does not contain control block information about the LNKLIST. SMSPDSE1 reduced the impact of problems in the use of PDSE data sets if they stopped functioning properly, particularly in a sysplex. As many functions as possible are handled by SMSPDSE1.

The SMSPDSE1 AS is restartable. It provides connections to, and processes requests for, those PDSE data sets that are not part of global connections associated with SMSPDSE. Global connections are used for PDSEs in the link list. The reliability and availability of PDSEs are improved by eliminating the need to IPL a system due to a hang, deadlock, or out-of-storage condition in PDSE address spaces.

However, SMSPDSE1 is not intended to be routinely restarted. Some amount of storage will be lost each time it is restarted. Depending on the nature of the problem, an IPL might still be required to completely solve a problem. However, the IPL can be scheduled for a more convenient time.

SMSPDSE1 is created if IGDSMSxx initialization parameters in a Parallel Sysplex coupled systems environment are set as follows:

```
PDSESHARING(EXTENDED)
PDSE_RESTARTABLE_AS(YES).
```

9.2 PDSE enhancements summary

DFSMS V1.13 provides the following new features for PDSE data sets:

- Additional diagnostic operands to display the PDSE connections on the DISPLAY SMS command or to REFRESH the cached directory pages on the VARY SMS command. When a PDSE data set is opened, the opening task is connected to the PDSE data set in the PDSE or PDSE1 address space. Therefore, the status of connections is important.
- A utility called IEBPDSE to verify the status of one or more PDSEs. If a PDSE is damaged, it might affect the PDSE or PDSE1 address space and affect multiple users. The IEBPDSE utility can assist in troubleshooting PDSE problems.

9.3 PDSE diagnostic command operands

In z/OS V1.13, a new DISPLAY SMS command operand and a new VARY SMS command operand are provided. These operands aid PDSE diagnostic software or refresh PDSE in-storage data. Details about these commands are provided in this section.

> **Attention:** These commands are the result of requests for implementation from PDSE level 2 in the IBM Support Center. Issue them only under the direction of the IBM Support Center when an error has occurred.

Although PDSE and PDSE1 errors are not common, they can be serious when errors occur in the PDSE code, or due to some catastrophic system failure. The results often extend beyond the job or user who encountered the error. Two new operator commands simplify the troubleshooting of PDSE errors. In some cases, the repair of the PDSE and PDSE1 structures or the selection of which element of the system needs to be terminated is also determined. These following commands are commonly used when the processing for one or more PDSEs or PDSE1s is unresponsive.

The commands before DFSMS V1.13 are:

```
VARY SMS,PDSE | PDSE1, ANALYSIS command (called the ANALYSIS command)
VARY SMS,PDSE | PDSE1, FREELATCH command (called the FREELATCH command)
```

With DFSMS V1.13, the following **D SMS,PDSE** and **V SMS,PDSE** commands have new operands to aid PDSE diagnostic systems.

```
D SMS,PDSE<1>,CONNECTIONS,DSN(pdsename)<,VOL(VOLSER)>
V SMS,PDSE<1>,REFRESH,DSN(pdsename)<,VOL(VOLSER)>
```

The commands can be issued against the PDSE or its address space.

9.3.1 D SMS command PDSE CONNECTIONS operand

The CONNECTIONS operand is useful in determining which jobs are affected when an error associated with a PDSE occurs. You can then determine whether an IPL or restart of the PDSE address space must be done immediately, or a cancellation of the involved jobs.

The **D SMS,PDSE** command (and the **D SMS,PDSE1** command, if the PDSE1 environment is enabled) with the CONNECTIONS operand can be issued, as shown in Example 9-1. The DSN operand is required, but the VOL operand is only required if the particular PDSE is not cataloged. You must know in advance the data set involved in the eventual hang, through previous console messages.

Example 9-1 DISPLAY SMS, PDSE command CONNECTIONS operand

```
D SMS,PDSE,CONNECTIONS,DSN(pdsename)<,VOL(VOLSER)> and
D SMS,PDSE1,CONNECTIONS,DSN(pdsename)<,VOL(VOLSER)>
```

The output in Figure 9-2 is the result of issuing the following command:

```
D SMS,PDSE1,CONNECTIONS,dsn(DB9j9.SDSNload)
```

In this case all the connections are with the PDSE1 address space instance.

```
IGW051I PDSE CONNECTIONS Start of Report(SMSPDSE1) 260 ---------data set
name--------------------- -----vsgt------DB9J9.SDSNLOAD   01-SBOX0J-002D0A
--asid-- --name-- --tcb--- -open-
  00A0   CZSR014S 007FF890 Input    0096   AZSR014S 007FF890 Input    0094
AZSR014S 007FF890 Input    0085   DB9JADMT 007FF890 Input    0082   DB9JDIST
007FF890 Input    007C   DB9JDBM1 007FF890 Input    0075   DB9JMSTR 007FF890
Input
PDSE CONNECTIONS End of Report(SMSPDSE1)
```

Figure 9-2 DISPLAY SMS, PDSE1 CONNECTIONS output example

The output in Figure 9-3 is the result of issuing the following command:

```
D SMS,PDSE,CONNECTIONS,dsn(DB9j9.SDSNload)
```

In this case, there are no connections to the DB9j9.SDSNLOAD data set. It might be possible to resolve the problem by restarting the PDSE1 instance without abending the connected address spaces (jobs). If the restart is successful, an IPL can be avoided.

If connections are shown in the PDSE address space instance resolution, an IPL might be required to resolve the issue. The IPL is needed because the PDSE task is not restartable.

```
IGW051I PDSE CONNECTIONS Start of Report(SMSPDSE ) 287
++ No Connections found for VSGT:01-SBOXOJ-002D0A
DSN: DB9J9.SDSNLOAD  PDSE CONNECTIONS
End of Report(SMSPDSE )
```

Figure 9-3 DISPLAY SMS, PDSE CONNECTIONS output example

9.3.2 V SMS PDSE refresh operand

The REFRESH operand is useful in discarding potentially bad cached data for a PDSE data set after an error. Use this command to discard cached PDSE or PDSE1 directory pages. This process ensures that the next access to the specified PDSE uses the data directly from the device.

The REFRESH operand relates to only the data set specified, not to the whole PDSE environment. It is possible, depending on the use of a data set, for it to be associated with the PDSE and with the PDSE1 environment.

The V SMS PDSE command with the REFRESH operand can be issued against the PDSE, the PDSE1 address space environment, or both (Figure 9-4). The DSN operand is required, but the VOL operand is only required if the particular PDSE is not cataloged.

```
V SMS,PDSE,REFRESH,DSN(pdsename)<,VOL(VOLSER)>
or
V SMS,PDSE1,REFRESH,DSN(pdsename)<,VOL(VOLSER)>
```

Figure 9-4 VARY SMS, PDSE command REFRESH operand

The output in Figure 9-5 is the result of issuing the following command:

```
V SMS,PDSE1,REFRESH,dsn(DB9j9.SDSNload)
```

The message IGW052I does not itself identify which PDSE environment it relates to.

```
IGW052I The cached directory blocks for PDSE DB9J9.SDSNLOAD have been discarded
```

Figure 9-5 VARY SMS,PDSE1,RESFESH command output

The output in Figure 9-6 is the result of issuing the following command:

```
V SMS,PDSE,REFRESH,dsn(DB9j9.SDSNload)
```

The message IGW052I is the same as for the case when the PDSE1 environment was specified.

```
IGW052I The cached directory blocks for PDSE DB9J9.SDSNLOAD have been discarded
```

Figure 9-6 VARY SMS,PDSE,REFRESH command output

9.3.3 PDSE diagnostic command operand compatibility and coexistence

The new CONNECTIONS and REFRESH operands are not available in releases before DFSMS V1.13. No compatibility changes are required on prior releases.

9.4 Validation of PDSE data sets

The IEBPDSE utility can verify the structural integrity of a PDSE in the same way as the EXAMINE command does with VSAM KSDS data sets. This process minimizes the possibility of backing up broken data sets. Apart from diagnosis situations, it can also be used to do the following checks:

- Validate critical data sets, and data sets that are backed up frequently.
- Start the PDSE validation utility in a separate JCL step before or after key operations, such as backing up a data set. Starting the IEBPDSE usually does not have any significant impact on performance.

9.4.1 IEBPDSE validation utility

In DFSMS V1.13, the new IEBPDSE utility can be used to validate a PDSE before or after copying the data set to a new location. The utility provides output in a message data set that contains informational messages, the results of the validation check, and any error messages. The utility runs a number of checks on a specified PDSE data set or set of concatenated PDSE data sets. These checks verify that requirements for directory structures and other rules are met.

The utility also provides output in the SYSPRINT data set, optionally including a message for each structure and the count of pages and records in the directory. The PDSE validation utility can be run in job control language (JCL). Like most utilities, IEBPDSE can also be run from TSO if SYSLIB is allocated to a PDSE.

Using the IEBPDSE utility with JCL

To start the tool, specify PGM=IEBPDSE on the EXEC statement in JCL. Name the data set or data sets to be validated on the SYSLIB DDNAME.

The following JCL statements are needed to run the IEBPDSE utility:

- EXEC statement: This statement starts the PDSE validation utility by using PGM=IEBPDSE. The PARM keyword can be specified.

  ```
  PARM=[DUMP|NODUMP]
  ```

If the DUMP option is specified, the PDSE validation utility issues an ABEND in the PDSE address space. This process occurs when an error is found in the analysis of the PDSE, and results in an SVC memory dump. The default value for PARM is NODUMP if PARM is not specified.

- SYSPRINT: This DD statement defines a sequential output message data set. If this statement is omitted, the output is displayed in the job log. The block size for the SYSPRINT data set must be a multiple of 121. If not, the job step ends with a return code of 8.

> **Consideration:** The documentation states that the SYSPRINT data set must have a block size of 121. However, this limit did not seem to be enforced, and a change was made dynamically at execution time. When the LRECL was set to 133 without specifically setting the BLKSIZE on a data set, the result was a data set with LRECL=BLKSIZE=131. However, the job completed successfully. Because the documented blocksize is 121, it can be enforced after future maintenance.

- SYSLIB: Defines the PDSE data set or data sets accessed by the PDSE utility when running the operations specified on the control statements. The DSNAME parameter is required. To validate more than one data set, concatenate additional data sets to the first that is defined on SYSLIB.

  If any of the data sets defined in SYSLIB are not in PDSE format, they are skipped without any message. However, their presence does not cause an error.

> **Remember:** The PDSE validation utility does not validate the data in the members or the structure of the members.

JCL examples

These job examples demonstrate the use of IEBPDSE.

Example 1

The JCL in Figure 9-7 shows how to run IEBPDSE to validate one PDSE data set and send the results to SYSPRINT.

```
//MHLRES2P JOB (999,POK),'MHLRES2',CLASS=A,MSGCLASS=T,
//             NOTIFY=&SYSUID,TIME=1440,REGION=6M
/*JOBPARM S=*
//STEPCHK  EXEC PGM=IEBPDSE
//SYSPRINT DD SYSOUT=A
//SYSLIB   DD DISP=SHR,DSN=MHLRES2.IEBPDSE.TEST1
```

Figure 9-7 PDSE JCL to validate one PDSE data set

The output in Example 9-2 shows execution of the job. Note the message `IGW700I PDSE Directory Validation Successful` followed by the name of the data set and statistics about the data set.

Example 9-2 Output from the IEBPDSE job in Figure 9-7

```
J E S 2  J O B  L O G  --  S Y S T E M  S C 6 4  --  N O D E  W T S C P

17.41.28 JOB10221 ---- FRIDAY,    23 SEP 2011 ----
17.41.28 JOB10221  IRR010I  USERID MHLRES2  IS ASSIGNED TO THIS JOB.
17.41.28 JOB10221  ICH70001I MHLRES2  LAST ACCESS AT 15:21:25 ON FRIDAY, SEPTEMBER 23, 2011
```

```
17.41.28 JOB10221  $HASP373 MHLRES2P STARTED - INIT 9    - CLASS A - SYS SC64
17.41.28 JOB10221  IEF403I MHLRES2P - STARTED - TIME=17.41.28 - ASID=009B - SC64
17.41.28 JOB10221  -                                         --TIMINGS (MINS.)--
17.41.28 JOB10221  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
17.41.28 JOB10221  -MHLRES2P          STEPCHK     00     56    .00    .00    .00   1175   0
17.41.28 JOB10221  IEF404I MHLRES2P - ENDED - TIME=17.41.28 - ASID=009B - SC64
17.41.28 JOB10221  -MHLRES2P ENDED.  NAME-MHLRES2              TOTAL CPU TIME=   .00  TOTAL
17.41.28 JOB10221  $HASP395 MHLRES2P ENDED
------ JES2 JOB STATISTICS ------
  23 SEP 2011 JOB EXECUTION DATE
            6 CARDS READ
           46 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
            3 SYSOUT SPOOL KBYTES
         0.00 MINUTES EXECUTION TIME
        1 //MHLRES2P JOB (999,POK),'MHLRES2',CLASS=A,MSGCLASS=T,                JOB10221
          //             NOTIFY=&SYSUID,TIME=1440,REGION=6M                     00020000
          /*JOBPARM S=*                                                         00021000
          IEFC653I SUBSTITUTION JCL - (999,POK),'MHLRES2',CLASS=A,MSGCLASS=T,NOTIFY=MHLRES
        2 //STEPCHK  EXEC PGM=IEBPDSE                                           00030000
        3 //SYSPRINT DD SYSOUT=A                                                00040000
        4 //SYSLIB   DD DISP=SHR,DSN=MHLRES2.IEBPDSE.TEST1                      00050000
ICH70001I MHLRES2  LAST ACCESS AT 15:21:25 ON FRIDAY, SEPTEMBER 23, 2011
IEF236I ALLOC. FOR MHLRES2P STEPCHK
IEF237I JES2 ALLOCATED TO SYSPRINT
IGD103I SMS ALLOCATED TO DDNAME SYSLIB
IEF142I MHLRES2P STEPCHK - STEP WAS EXECUTED - COND CODE 0000
IEF285I   MHLRES2.MHLRES2P.JOB10221.D0000101.?          SYSOUT
IGD104I MHLRES2.IEBPDSE.TEST1                         RETAINED,  DDNAME=SYSLIB
IEF373I STEP/STEPCHK /START 2011266.1741
IEF032I STEP/STEPCHK /STOP  2011266.1741
        CPU:     0 HR  00 MIN  00.04 SEC    SRB:     0 HR  00 MIN  00.00 SEC
        VIRT:    12K  SYS:   300K  EXT:        4K  SYS:    10160K
IEF375I  JOB/MHLRES2P/START 2011266.1741
IEF033I  JOB/MHLRES2P/STOP  2011266.1741
        CPU:     0 HR  00 MIN  00.04 SEC    SRB:     0 HR  00 MIN  00.00 SEC
IGW700I PDSE Directory Validation Successful
DSN:MHLRES2.IEBPDSE.TEST1
ADPages:175 IXRecords:6778
NDPages:23 IXRecords:2265
AD ND Tree Nodes:2265
```

Example 2

The JCL in Figure 9-8 shows how to start IEBPDSE to validate three PDSE data sets. It also shows sending the results to the job log because there is no SYSPRINT DD. One of the data sets (SYS1.LINKLIB) is not a PDSE, but is included to show that it is ignored without a problem.

```
//MHLRES2P JOB (999,POK),'MHLRES2',CLASS=A,MSGCLASS=T,
//             NOTIFY=&SYSUID,TIME=1440,REGION=6M
/*JOBPARM S=*
//STEPCHK  EXEC PGM=IEBPDSE
//SYSLIB   DD DISP=SHR,DSN=MHLRES2.IEBPDSE.TEST1
//         DD DISP=SHR,DSN=SYS1.LINKLIB
//         DD DISP=SHR,DSN=CEE.SCEERUN2
```

Figure 9-8 PDSE JCL to validate multiple PDSE data sets

The output in Example 9-3 shows execution of the job. Note the message `IGW700I PDSE Directory Validation Successful` followed by the name of the data set and statistics about the data set MHLRES2.IEBPDSE.TEST1. This output is then followed by the same message and statistics for data set CEE.SCEERUN2.

There are no messages that concern SYS1.LINKLIB, which was included in SYSLIB, but it is not a PDSE.

> **Remember:** Data set MHLRES2.IEBPDSE.TEST1 was copied from CEE.SCEERUN2 including all members. There are slight differences between the two data sets, which reflects the allocation of the data sets but does not indicate a problem.

Due to the absence of the SYSPRINT DD, the messages are sent to the job log.

Example 9-3 Output from the IEBPDSE job in Figure 9-8

```
J E S 2  J O B  L O G  --  S Y S T E M  S C 6 4  --  N O D E  W T S C P L X 2

17.51.26 JOB10222 ---- FRIDAY,    23 SEP 2011 ----
17.51.26 JOB10222  IRR010I  USERID MHLRES2  IS ASSIGNED TO THIS JOB.
17.51.26 JOB10222  ICH70001I MHLRES2  LAST ACCESS AT 17:41:28 ON FRIDAY, SEPTEMBER 23, 2011
17.51.26 JOB10222  $HASP373 MHLRES2P STARTED - INIT 9    - CLASS A - SYS SC64
17.51.26 JOB10222  IEF403I MHLRES2P - STARTED - TIME=17.51.26 - ASID=009B - SC64
17.51.26 JOB10222  +IGW700I PDSE Directory Validation Successful
17.51.26 JOB10222  +DSN:MHLRES2.IEBPDSE.TEST1
17.51.26 JOB10222  +ADPages:175 IXRecords:6778
17.51.26 JOB10222  +NDPages:23 IXRecords:2265
17.51.26 JOB10222  +AD ND Tree Nodes:2265
17.51.26 JOB10222  +IGW700I PDSE Directory Validation Successful
17.51.26 JOB10222  +DSN:CEE.SCEERUN2
17.51.26 JOB10222  +ADPages:128 IXRecords:6796
17.51.26 JOB10222  +NDPages:23 IXRecords:2265
17.51.26 JOB10222  +AD ND Tree Nodes:2265
17.51.26 JOB10222  -                                         --TIMINGS (MINS.)--
----PA
17.51.26 JOB10222  -JOBNAME  STEPNAME PROCSTEP    RC   EXCP    CPU    SRB  CLOCK   SERV  PG
PAGE
17.51.26 JOB10222  -MHLRES2P          STEPCHK     00     73    .00    .00    .00   2285   0
0
```

```
17.51.26 JOB10222  IEF404I MHLRES2P - ENDED - TIME=17.51.26 - ASID=009B - SC64
17.51.26 JOB10222  -MHLRES2P ENDED.  NAME-MHLRES2              TOTAL CPU TIME=   .
17.51.26 JOB10222  $HASP395 MHLRES2P ENDED
------ JES2 JOB STATISTICS ------
  23 SEP 2011 JOB EXECUTION DATE
            7 CARDS READ
           66 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
            4 SYSOUT SPOOL KBYTES
         0.00 MINUTES EXECUTION TIME
        1 //MHLRES2P JOB (999,POK),'MHLRES2',CLASS=A,MSGCLASS=T,
          //              NOTIFY=&SYSUID,TIME=1440,REGION=6M
          /*JOBPARM S=*
          IEFC653I SUBSTITUTION JCL - (999,POK),'MHLRES2',CLASS=A,MSGCLASS=T,NOTIF
        2 //STEPCHK  EXEC PGM=IEBPDSE
        3 //SYSLIB   DD DISP=SHR,DSN=MHLRES2.IEBPDSE.TEST1
        4 //         DD DISP=SHR,DSN=SYS1.LINKLIB
        5 //         DD DISP=SHR,DSN=CEE.SCEERUN2
ICH70001I MHLRES2  LAST ACCESS AT 17:41:28 ON FRIDAY, SEPTEMBER 23, 2011
IEF236I ALLOC. FOR MHLRES2P STEPCHK
IGD103I SMS ALLOCATED TO DDNAME SYSLIB
IEF237I 9202 ALLOCATED TO
IEF237I 9282 ALLOCATED TO
IGW700I PDSE Directory Validation Successful
DSN:MHLRES2.IEBPDSE.TEST1
ADPages:175 IXRecords:6778
NDPages:23 IXRecords:2265
AD ND Tree Nodes:2265
IGW700I PDSE Directory Validation Successful
DSN:CEE.SCEERUN2
ADPages:128 IXRecords:6796
NDPages:23 IXRecords:2265
AD ND Tree Nodes:2265
IEF142I MHLRES2P STEPCHK - STEP WAS EXECUTED - COND CODE 0000
IGD104I MHLRES2.IEBPDSE.TEST1                        RETAINED,  DDNAME=SYSLIB
IEF285I   SYS1.LINKLIB                                 KEPT
IEF285I   VOL SER NOS= Z1DRA1.
IEF285I   CEE.SCEERUN2                                 KEPT
IEF285I   VOL SER NOS= Z1DRA2.
IEF373I STEP/STEPCHK /START 2011266.1751
IEF032I STEP/STEPCHK /STOP  2011266.1751
        CPU:     0 HR  00 MIN  00.08 SEC    SRB:     0 HR  00 MIN  00.00 SEC
        VIRT:     8K  SYS:   300K  EXT:        4K  SYS:    10208K
IEF375I  JOB/MHLRES2P/START 2011266.1751
IEF033I  JOB/MHLRES2P/STOP  2011266.1751
        CPU:     0 HR  00 MIN  00.08 SEC    SRB:     0 HR  00 MIN  00.00 SEC
```

Example 3

The JCL in Figure 9-9 shows how to start IEBPDSE. The command validates one PDSE data set and specifies that a memory dump be taken if an error is found.

```
//MHLRES2P JOB (999,POK),'MHLRES2',CLASS=A,MSGCLASS=T,
//             NOTIFY=&SYSUID,TIME=1440,REGION=6M
/*JOBPARM S=*
//STEPCHK  EXEC PGM=IEBPDSE,PARM=DUMP
//SYSPRINT DD SYSOUT=*
//SYSLIB   DD DISP=SHR,DSN=MHLRES2.IEBPDSE.TEST1BAD
//         DD DISP=SHR,DSN=MHLRES2.IEBPDSE.TEST1
```

Figure 9-9 PDSE JCL to validate a set of PDSE data sets with PARM=DUMP

The output in Figure 9-10 shows execution of the job. Note the message `IGW700I PDSE Directory Validation Successful` followed by the name of the data set and statistics about the data set MHLRES2.IEBPDSE.TEST1BAD. This output is then followed by the same message and statistics for data set MHLRES2.IEBPDSE.TEST1.

Neither data set was considered bad by IEBPDSE, so no memory dump was generated.

Remember: Data set MHLRES2.IEBPDSE.TEST1BAD was copied from MHLRES2.IEBPDSE.TEST1 including all members. In this case, unlike what happened with Example 2, the statistics are identical between the two data sets.

```
IEF033I  JOB/MHLRES2P/STOP  2011266.1904
        CPU:     0 HR  00 MIN  00.08 SEC    SRB:     0 HR  00 MIN  00.00 SEC
IGW700I PDSE Directory Validation Successful
DSN:MHLRES2.IEBPDSE.TEST1BAD
ADPages:175 IXRecords:6778
NDPages:23 IXRecords:2265
AD ND Tree Nodes:2265
IGW700I PDSE Directory Validation Successful
DSN:MHLRES2.IEBPDSE.TEST1
ADPages:175 IXRecords:6778
NDPages:23 IXRecords:2265
AD ND Tree Nodes:2265
```

Figure 9-10 Output from job in Figure 9-9

Return codes

The process can generate the following return codes:

00 (X'00') Successful completion.

04 (X'04') Slightly damaged PDSE. The data set can be opened normally, but has some form of corruption. Currently, the only instance of a “slightly” damaged PDSE is when the free space list marks free pages as used. This process wastes space that could normally be reclaimed. This error does not prevent you from opening the PDSE, but you should copy the PDSE to a new data set.

08 (X'08') Corrupted PDSE.

12 (X'0C') PDSE could not be opened.

16 (X'10') Input data sets not PDSEs (no PDSEs found during SYSLIB concatenation).

Using the IEBPDSE utility with TSO

To run IEBPDSE under TS,O the data sets to be verified must be allocated to the SYSLIB DDNAME. The program must then be run from SYS1.LINKLIB.

Example of execution of IEBPDSE with TSO

Figure 9-11 shows the TSO command that can be used to run IEBPDSE.

```
alloc fi(syslib) da('cee.sceerun2') shr
call 'SYS1.linklib(iebpdse)'
```

Figure 9-11 Example command to run IEBPDSE on TSO

The output in Figure 9-12 shows the result of running the commands (one after the other) listed in Figure 9-11.

```
IGW700I PDSE Directory Validation Successful
DSN:CEE.SCEERUN2
ADPages:128 IXRecords:6796
NDPages:23 IXRecords:2265
AD ND Tree Nodes:2265
***
```

Figure 9-12 IEBPDSE output on TSO

9.4.2 IEBPDSE Compatibility and coexistence

The IEBPDSE program is not available in code releases before DFSMS V1.13.

9.5 System procedure library as a PDSE

A system procedure library as a PDSE data set is not new in z/OS V1R13. The information is included here as a reminder of this support.

Every library that contains cataloged procedures must be either a PDS or PDSE. PDSE support (the PDSE SMSPDSE and SMSPDSE1 address spaces) is initialized during nucleus initialization program (NIP) processing. Because SYS1.PROCLIB and all data sets in the MSTJCLxx proclib concatenation are referenced only during Master Scheduler Initialization (MSI), these data sets can be PDSE format. MSI occurs after NIP processing is complete.

10

EAV: New 1 TB support

The maximum size for an extended address volume (EAV) when it was first introduced was 223 GB. Support for even larger volumes is now available, enabling creation of up to 1 TB logical volumes on the DS8K control unit.

One reason for larger volumes is the UCB/UCW address constraint (up to 64K primary devices per logical partition) and the need for simpler management. This update covers the considerations needed for the new 1 TB maximum size EAV. This new EAV feature is available for no extra charge.

This chapter includes the following sections:

- Background
- Support for larger volumes
- Migration
- Enhanced FTP support

10.1 Background

An EAV is a logical volume that exceeds the previous limitation of 65520 cylinders. The first EAV support came out with DFSMS V1.10.

The initial EAV architecture supported a volume size of up to 223 GB per volume. Only certain Virtual Storage Access Method (VSAM) data sets could be stored above the 65520 cylinder line. This line is also called the cylinder-managed area. The types of data sets supported have increased in DFSMS V1.11 and DFSMS V1.12. In V1.13, all data sets are allowed in the cylinder-managed area.

The implementation of 1 TB 3390 volumes makes using parallel access volumes (PAVs) even more important for DASD I/O performance. However, remember that for integrity reasons, the DS8K does not run certain parallel activities:

- Writes to the same extent
- Writes together with reads to the same extent

10.2 Support for larger volumes

z/OS V1.13 (or SPE for z/OS V1R12 OA28553), along with an update to the DS8K microcode, provides support for volumes larger than 223 GB (262,668 cylinders). The new support enables a maximum size of 1 TB for a logical volume on a DS8K (1,182,006 cylinders), more than four times the previous maximum. This chapter addresses considerations you need to following when migrating these volumes. This is a minor change, compared to moving to the new architecture when it was released originally.

10.3 Migration

This section addresses the migration steps to implement 1 TB EAV 3390 volumes.

10.3.1 Creating a 1 TB EAV

For 1 TB EAV volumes, you need either a DS8700 or a DS8800 and the appropriate level of microcode. You need enough space (free extents) on the DS8K to be able to create the volume type.

Before creating the volumes, consider whether you need these 1 TB EAV volumes, and how they would fit into your existing environment with lower capacity devices. The distribution of larger volumes on the logical storage subsystem (LSS) control units can be an issue. Testing has shown a considerable increase in planned and unplanned IBM HyperSwap® elapsed time using larger (or slower) devices in a HyperSwap ready environment. This increase is due to staging of larger amounts of metadata. Therefore, distribute your large volumes on as many LSSs as possible.

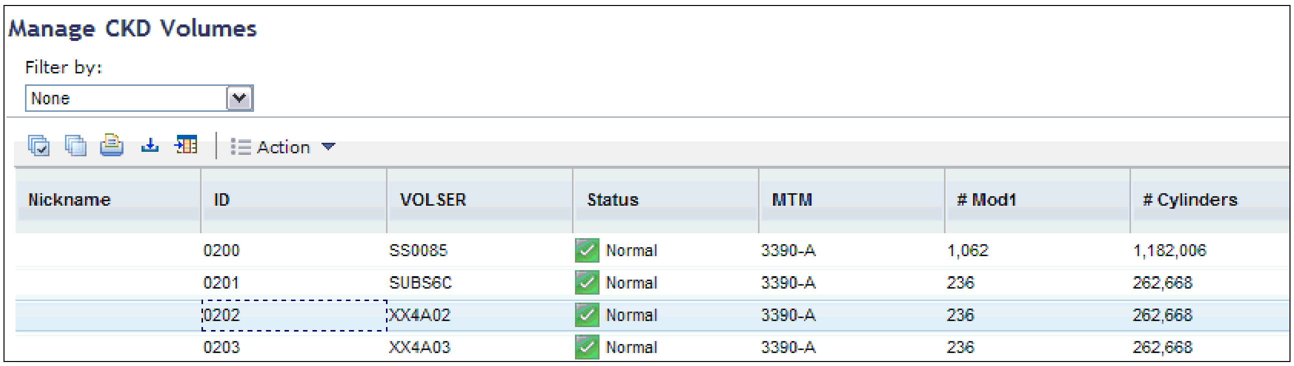
You can create the volume from scratch on the DS Storage Manager with the actual needed size specified. Alternatively, you can choose to upgrade some of your existing volumes to 1 TB EAVs, for example by using the nondisruptive DS8000 Dynamic Volume Expansion feature (DVE). This example demonstrates how to upgrade an existing EAV volume of the previous maximum size (262668 cylinders) to the new Extended Address Volume with 1 TB support.

The volume attributes for the device in scope for this upgrade can be seen in a DEVSERV display in Figure 10-1.

```
IEE459I 12.27.41 DEVSERV QDASD 547
 UNIT VOLSER SCUTYPE DEVTYPE       CYL  SSID SCU-SERIAL DEV-SERIAL EFC
04A02 XX4A02 2107941 2107900    262668  49D2 0175-XC901 0175-XC901 *OK
****       1 DEVICE(S) MET THE SELECTION CRITERIA
****       0 DEVICE(S) FAILED EXTENDED FUNCTION CHECKING
```

Figure 10-1 Example of DEVSERV display of previous maximum size EAV volume

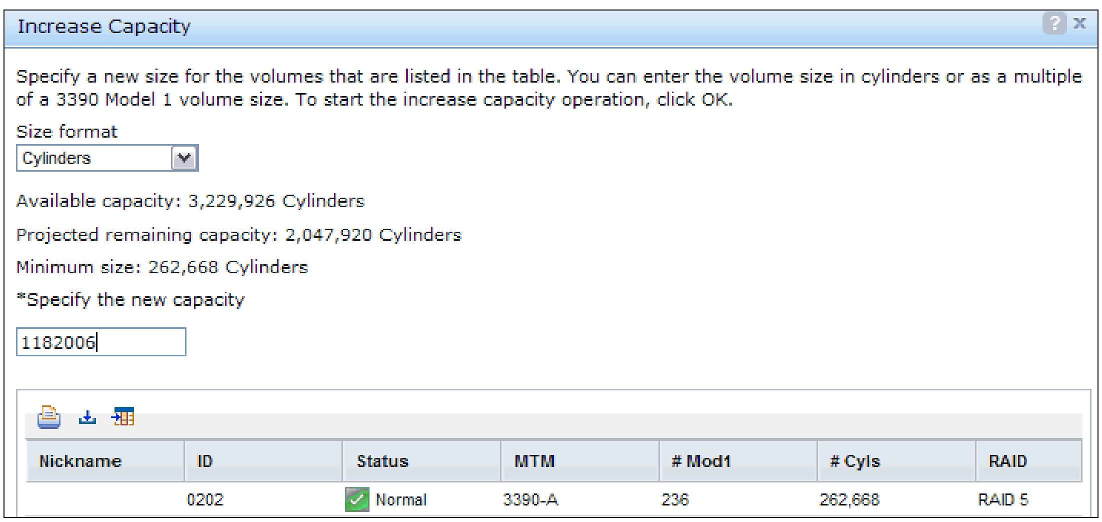
Also, from the DS8K Storage Manager, the volume characteristics look like Figure 10-2 (3390-A for EAV size).



Nickname	ID	VOLSER	Status	MTM	# Mod1	# Cylinders
	0200	SS0085	Normal	3390-A	1,062	1,182,006
	0201	SUBS6C	Normal	3390-A	236	262,668
	0202	XX4A02	Normal	3390-A	236	262,668
	0203	XX4A03	Normal	3390-A	236	262,668

Figure 10-2 Display of EAV in the DS8K Storage Manager

The Increase Capacity option in the DS8K Storage Manager is selected to upgrade the EAV volume. Increase the size to the new maximum limit of 1 TB (11820069 cylinders) as shown in Figure 10-3.



Nickname	ID	Status	MTM	# Mod1	# Cyls	RAID
	0202	Normal	3390-A	236	262,668	RAID 5

Figure 10-3 Increase Capacity window in DS8K Storage manager

The DS8K Storage Manager shows these capacities:

- The available capacity (3,229,926 cylinders)
- The remaining capacity after increasing the volume in scope (2,047,920 cylinders)
- The current capacity of the volume (minimum size = 262,668 cylinders)

The new capacity is specified (1,182,006 cylinders). The update to the EAV volume is run and displays the new capacity as shown in Figure 10-4.

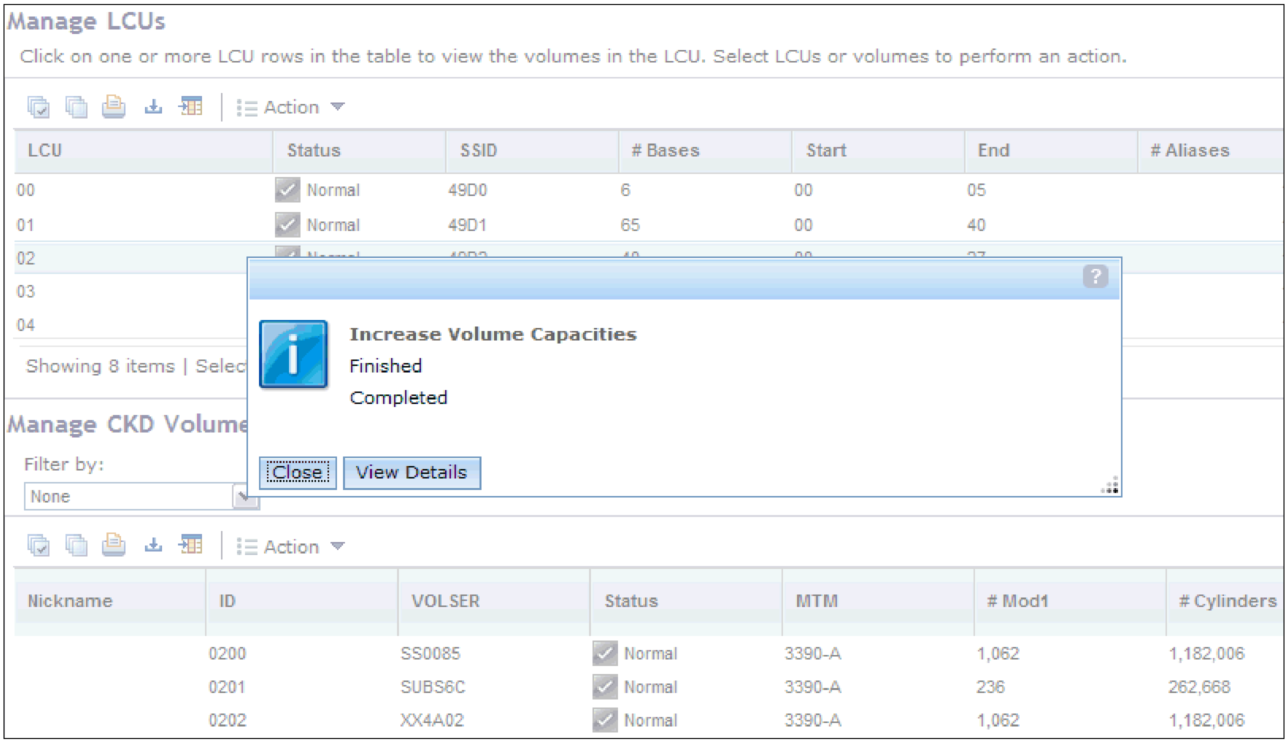


Figure 10-4 Capacity upgrade to maximum size EAV completed

Also, as seen from z/OS, the volume capacity changed (Figure 10-5). A DEVSERV display shows the capacity upgrade to the maximum size: 1,182,006 cylinders.

```
RESPONSE=J80
 IEE459I 12.30.24 DEVSERV QDASD 207
  UNIT VOLSER SCUTYPE DEVTYPE       CYL  SSID SCU-SERIAL DEV-SERIAL EFC
 04A02 XX4A02 2107941 2107900   1182006  49D2 0175-XC901 0175-XC901 *OK
 ****       1 DEVICE(S) MET THE SELECTION CRITERIA
 ****       0 DEVICE(S) FAILED EXTENDED FUNCTION CHECKING
```

Figure 10-5 DEVSERV display of EAV volume maximum size

10.3.2 Automatic DVE REFVTOC

Before V1R13, a DVE sent a write to operator with reply (WTOR) to the console. The WTOR asked if this upgrade should be allowed, and would wait for a reply. Also, you would be notified to run an ICKDSF REFVTOC to make the new capacity available and visible.

In DFSMS V1.13, the ICKDSF run happens automatically through a DEVMAN address space task, resulting in the REFVTOC. Nevertheless, DEVMAN sends informative write to operator (WTO) messages to the console about the execution of the DVE function.

You can find the messages in the Syslog (or Operlog) on all logical partitions connected to the DS8K (Example 10-1). However, they cannot be found in the DEVMAN sysout because the JCT is non-displayable.

Example 10-1 DEVMAN actions after a requested Dynamic Volume Expansion in DS8K

```
IEA019I 4A02,XX4A02,VOLUME CAPACITY CHANGE,OLD=0004020C,NEW=00120936
ICK502I   BUILDIX FUNCTION STARTED
IEC604I VTOC CONVERT ROUTINE ENTERED ON 4A02,XX4A02,DOS,DEVMAN
ICK503I   4A02 REQUEST RECEIVED TO CONVERT VTOC TO IXFORMAT
ICK504I   4A02 VTOC FORMAT IS CURRENTLY OSFORMAT, REQUEST ACCEPTED
ICK503I   4A02 REQUEST RECEIVED TO CONVERT VTOC TO IXFORMAT
ICK504I   4A02 VTOC FORMAT IS CURRENTLY OSFORMAT, REQUEST ACCEPTED
ICK513I   4A02  BUILDIX PROCESSING COMPLETED:  VTOC IS NOW IN IXFORMAT
DMO0054I 4A02,XX4A02,REFVTOC 430
**** DEVMAN ************************************************************
```

However, to have this dynamic DEVMAN support enabled, a parameter must be set in the DEVSUPxx member in SYS1.PARMLIB: ENABLE(REFVTOC) as shown in Figure 10-6.

```
 BROWSE    SYS1.PARMLIB(DEVSUP00) - 01.17            Line 00000000 Col
********************************* Top of Data *************************
/*  DEVSUPXX MEMBER FOR OS/390 INTEGRATION TEST PLEX.
/*    DEFAULTS FOR MEDIA, ERROR AND PRIVATE ARE OVERRIDDEN
/*    TO ALLOW PARTITIONING OF THE 3494 & 3495 TAPE LIBRARIES.
/*
COMPACT = YES,            /* DATA STORED IN COMPACTED FORMAT.
VOLNSNS = YES,            /* 36 TRK. TAPE CAN BE OVERWRITTEN TO 18 TRK.
MEDIA1  = 0011,           /* -
MEDIA2  = 0012,           /* - UNIQUE CATAGORIES FOR TAPES IN THE
MEDIA3  = 0013,           /* -   3494 & 3495 THAT BELONG TO THE
MEDIA5  = 0015,           /* - 3592 UNIQUE CATEGORY
ERROR   = 001E,           /* -   OS/390 INTEGRATION TEST PLEX.
PRIVATE = 001F,           /* -
ENABLE(REFVTOC),          /* - AUTOMATIC VTOC AND VTOC INDEX REBUILD
```

Figure 10-6 Dynamic DEVMAN support

If you want to verify that your DEVMAN feature is activated, use the REPORT option in the DEVMAN modify command as shown in Example 10-2.

Example 10-2 DEVMAN current enabled support

```
F DEVMAN,REPORT
DMO0030I DEVICE MANAGER REPORT 322
**** DEVMAN **********************
* FMID:  HDZ1D10
* APARS: 11.12    11.11
* OPTIONS:  REFVTOC   REFUCB
* NO SUBTASKS ARE ACTIVE
**** DEVMAN **********************
```

In this case, the support is enabled. If the support is not there, add the parameter to the DEVSUPxx member. You can wait for the next IPL or activate the support through a DEVMAN command: **F DEVMAN,ENABLE(REFVTOC)**.

Figure 10-7 shows how the new larger size EAVs are presented in Interactive Storage Management Facility (ISMF) and through Interactive System Productivity Facility (ISPF).

```
                               VOLUME LIST
                                                   Entries 53-63 of 63
Enter Line Operators below:                        Data Columns 3-8 of 43

    LINE        VOLUME FREE       %    ALLOC      FRAG   LARGEST     FREE
    OPERATOR    SERIAL SPACE      FREE SPACE      INDEX  EXTENT      EXTENTS
   ---(1)----   -(2)-- ---(3)---  (4)- ---(5)---  -(6)-  ---(7)---   --(8)--
                ARC040 87583564K    40   127384M    252  15351572K       102
                ARC041 87425635K    40   127538M    240  18650978K        91
                ARC042 86439659K    40   128501M    259   8732841K        92
                ARC043 86388142K    40   128551M    236  11974974K        79
                ARC044 85249494K    39   129663M    262   7834739K        95
                ARC045 85993320K    39   128937M    229  14538134K        77
                ARC046 ---------   ---  ---------   ---  ---------   -------
                ARC047   382882M    40   575234M    254  55952933K       426
```

Figure 10-7 ISMF volume list, with a maximum size EAV at the bottom

ISMF reports free space and allocated space in megabytes to avoid overflow of the fields.

The ISPF data set list (3.4) shows similar information in tracks and cylinders. Remember that this volume was upgraded from a 3390-9 volume. The volume table of contents (VTOC) therefore inherits the size (140 tracks in this case) from this volume as shown in Figure 10-8.

```
Menu  RefList  RefMode  Utilities  Help
                      VTOC Summary Information
 Volume . : ARC047

 Unit . . : 3390                  Free Space

  VTOC Data                       Total              Tracks         Cyls
  Tracks  . :         140         Size  . . :     7,085,299      472,351
  %Used . . :          25         Largest . :     1,011,150       67,410
  Free DSCBS:       5,269         Free
                                  Extents . :     426

  Volume Data                     Track Managed      Tracks         Cyls
  Tracks . :    17,730,090        Size  . . :       680,719       45,379
  %Used  . :            60        Largest . :       114,840        7,656
  Trks/Cyls:            15        Free
                                  Extents . :      50
 Command ===> 
```

Figure 10-8 VTOC summary information panel

10.3.3 Considerations on VTOC and VVDS size

When moving to a 1 TB size volume, consider the size of the VTOC, VTOC index, and VSAM volume data set (VVDS). Compared to the previous maximum size volume, this architecture is more than four times this size. Most volumes actually are considerably smaller, even compared to the previous maximum limit.

Size of VTOC and VTOC index

Copying to the new 1 TB EAV from an existing volume results in the same VTOC size as the one on the originating volume. The volume remains the same whether you use DSS or IBM TDMF®. This configuration is insufficient for a volume with a substantially larger number of data sets.

To avoid this limitation, define the volume as an entirely new volume and size the VTOC, VTOC index, and VVDS according to the new volume size. Alternatively if you copy the volume to the 1 TB EAV, resize the VTOC by using the ICKDSF EXTVTOC (n) function. Use the EXTINDEX for the VTOC index. This process allows you to expand the VTOC or VTOC index to the value in tracks you specified in the bracket. For the value to be valid, it must expand the current track size of the VTOC. Example 10-3 shows how to extend the VTOC and the VTOC index in their current location.

Example 10-3 Example of expanding VTOC and VTOC index by using ICKDSF

```
REFORMAT DDNAME(VOLDD) VERIFY(MLH123) EXTVTOC(200) EXTINDEX(16)
```

If there is no room to expand the VTOC in its current location, use this keyword in ICKDSF to move and allocate a bigger VTOC elsewhere on the volume:

```
NEWVTOC(cylinder,head/ANY,n)
```

The parameters for this keyword specify the start location of the new VTOC and the size (n) in tracks. The new VTOC allocation must be in one contiguous extent. Use REFVTOC if any system shows an incorrect value after copying to a new volume, or changing the VTOC or VTOC index location and size. Example 10-4 shows moving a VTOC to a new location and expanding it.

Example 10-4 Example of moving a VTOC to a new location with a size of 120 tracks

```
REFORMAT NEWVTOC(ANY,120) VERIFY(MHL123) UNITADDRESS(1234)
```

The following example shows an upgrade of an existing 3390-9 volume to an EAV volume, by using DVE feature in the DS8000. A display of the volume in an ISPF data set list shows a volume of the standard size of 10017 cylinders (Figure 10-9). The volume has a VTOC size of 120 tracks before the DVE.

```
  Menu  RefList  RefMode  Utilities  Help
                          VTOC Summary Information
 Volume . : XX4B00

 Unit . . : 3390

  Volume Data           VTOC Data             Free Space   Tracks    Cyls
  Tracks . :  150,255   Tracks  . :       120  Size  . . : 150,119  10,007
  %Used  . :        0   %Used . . :         1  Largest . : 150,105  10,007
  Trks/Cyls:       15   Free DSCBS:     5,997  Free
                                               Extents . :       2


 Command ===> _
```

Figure 10-9 ISPF display of a standard size 3390-9 volume

After an upgrade with DVE, the volume is larger, but the VTOC is the same size. Remember that this configuration is not changed automatically. A VTOC of a certain size might fit the volume it came from. However, when you move it to a volume that is (as in this case) 16 times larger, change the VTOC accordingly.

As mentioned, the VTOC must be extended by using one of the two options: EXTVTOC or NEWVTOC. The example uses the NEWVTOC option because there was no space to extend the VTOC in its current location. In comparison to EXTVTOC, NEWVTOC requires the volume to be taken offline, which might be a problem in an environment that needs continuous availability. As an alternative, create some free space in connection with the current VTOC by moving data sets to another location on the volume.

Expand the VTOC to 200 tracks in its new location. The volume was put offline on all LPARs in the sysplex and the NEWVTOC action was done by using ICKDSF as shown in Example 10-5.

Example 10-5 Sysout from an ICKDSF move of VTOC to new location

```
0 REFORMAT NEWVTOC(ANY,200) VERIFY(XX4B00) UNITADDRESS(4B00)
-ICK00700I DEVICE INFORMATION FOR 4B00 IS CURRENTLY AS FOLLOWS:
-          PHYSICAL DEVICE = 3390
-          STORAGE CONTROLLER = 2107
-          STORAGE CONTROL DESCRIPTOR = E8
-          DEVICE DESCRIPTOR = 0E
-          ADDITIONAL DEVICE INFORMATION = 4A00003C
-          TRKS/CYL = 15, # PRIMARY CYLS = 262668
0ICK04000I DEVICE IS IN SIMPLEX STATE
 ICK00091I 4B00 NED=002107.900.IBM.75.0000000XC901
-ICK03091I EXISTING VOLUME SERIAL READ = XX4B00
0ICK01520I THE VTOC-INDEX WAS DELETED
0ICK01314I VTOC IS LOCATED AT CCHH=X'000A 0000' AND IS   200 TRACKS.
-ICK00001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
0          13:31:51    10/12/11
```

After a successful execution, put the volume online again. Before re-creating the indexed VTOC with the BUILDIX command with the IXVTOC parameter, view the results with an ISPF data set list (option 3.4). You see the message: “Free space error” as shown in Figure 10-10. You have not rebuilt the index for the VTOC, so the volume was converted to an OSVTOC. This process did not update the free space chain. The chain is updated during the next allocation or when you rebuild the index.

```
  Menu  RefList  RefMode  Utilities  Help
 ------------------------------------------------------------------------------
                        Data Set List Utility          Free space error
                                                             More:     +
   blank Display data set list             P Print data set list
       V Display VTOC information         PV Print VTOC information

 Enter one or both of the parameters below:
   Dsname Level . . . ______________________________________________
   Volume serial  . . XX4B00
```

Figure 10-10 ISPF showing free space errors after moving VTOC

To retrieve the VTOC information, do an ICKDSF refresh of the VTOC to have the space information updated. See the input for this ICKDSF function in Example 10-6.

Example 10-6 ICKDSF refresh of VTOC

```
REFORMAT DDNAME(VVVVVV) VERIFY(VVVVVV) REFVTOC
```

The VTOC and the volume can now be displayed through ISPF as shown in Figure 10-11.

```
Menu  RefList  RefMode  Utilities  Help
                        VTOC Summary Information
  Volume . : XX4B00

  Unit . . : 3390                Free Space

   VTOC Data                     Total             Tracks        Cyls
   Tracks  . :        200        Size  . . :    3,939,819     262,653
   %Used . . :          1        Largest . :    2,957,220     197,148
   Free DSCBS:      9,997        Free
                                 Extents . :        3

   Volume Data                   Track Managed     Tracks        Cyls
   Tracks . :     3,940,020      Size  . . :      982,599      65,505
   %Used  . :             0      Largest . :      982,450      65,496
   Trks/Cyls:            15      Free
                                 Extents . :        2
  Command ===> _
```

Figure 10-11 EAV volume after extension of VTOC in new location

The VTOC size has changed to 200 tracks. An IEHLIST listing of the volume shows the changed location of the VTOC.

> **Remember:** For devices that contain more than 64K tracks, there are special considerations for VTOC placement. The existing IBM software uses relative track addressing or the track-track-record (TTR) format to process the VTOC. This configuration restricts the highest address that can be referenced as a VTOC track to be 64K-1. Because of this limitation, the entire VTOC must be located within the first 64K tracks. That is, the VTOC must end before cylinder 4369 (X'1111') head 1.

To be able to calculate the VTOC and VTOC index, see Appendix C in *Device Support Facilities (ICKDSF) User's Guide and Reference*, GC35-0033. A table in this appendix shows the following information:

- How many DSCBs can be in one track
- How many index records one track can hold on the individual device types

Based on this information, do a calculation of the VTOC size. The example has a maximum size EAV volume (1182006 cylinders) with a VTOC size of 140 tracks. For this type of volume, you can have 50 DCSBs per VTOC track. Therefore, you can have (140*50) = 7000 data sets on the volume before the VTOC sets the limit for allocating more data sets. To fill a maximum size EAV by allocating 7000 data sets, you need an average data set size of 1182006/7000, or approximately 170 cylinders per data set. If your average data set size is less than this number, adjust your VTOC size accordingly.

Use a similar track calculation for the VTOC index. It is based on the number of data sets expected to be allocated on the volume and the average size of these data set names. This calculation is slightly more complex, but gives the number of tracks needed in the ICKDSF initialization of the volume or on an INDEX create. However, updating a VTOC index size is also more dynamic and less disruptive than increasing a VTOC size.

Considerations on VVDS size

The default size for a VVDS is 10 tracks primary and 10 tracks secondary. For a large volume such as a 1 TB EAV, calculate the expected number of data sets based on an expected average size. Use this calculation to estimate the size of the VVDS. *Managing Catalogs,* SC26-7409, has estimates on how many bytes each individual data set on this volume will require in the VVDS. The space required depends on the data set type (VSAM key-sequenced, linear, or sequential). So an estimate on the expected distribution between VSAM and non-VSAM data sets must be calculated along with the number of data sets. This process can establish an average size that is true for all large-size EAVs in your environment.

Table 10-1 from *Managing Catalogs* shows by data set how many bytes you need to store the information for all data sets on a volume.

Table 10-1 Estimates in bytes on how much space each data set type will occupy in a VVDS

Data Set Type	SMS-managed Volume	Non-SMS-managed Volume
VSAM key-sequenced data set or alternate index	530	480
VSAM entry-sequenced or relative record data set	370	320
VSAM linear data set	340	290
Non-VSAM data set	100	0

Based on the table, calculate the VVDS size need for a maximum sized SMS-managed EAV volume. The example volume is populated with 10000 data sets distributed into the following categories:

- 2000 VSAM key-sequenced data sets
- 2000 VSAM entry-sequenced data sets
- 1000 VSAM-linear data sets
- 5000 data sets as non-VSAM

Calculating this size based on Table 10-1, you need a primary allocation of the VVDS of 47 tracks, far above the default of (10,10) tracks. Size the VVDS according to the new larger volumes. Even the default secondary allocation of 10 tracks meets larger requirements than the 10 tracks that the default primary allocation provides.

Consideration: The Index data set must be allocated within the first 65,520 cylinders on a volume, and cannot exceed 64K-1 tracks in size.

10.3.4 DSS changes

The DFSMSdss logical data set restore has changed to support data sets with a size of more than 16,777,215 tracks. When DSS is about to restore a VSAM data set of more than 16,777,215 tracks, the allocation is changed to cylinders instead of tracks. This change avoids overflowing the field used to specify the size.

10.3.5 DFSMShsm changes

No external updates are made to DFSMShsm. Only internal fields are updated to support the larger volumes.

Space calculations for data sets migrated, backed up, or backed up using aggregate backup and recovery support (ABARS) from z/OS V1.10 to z/OS V1.13 might show a wrong value when being recalled or recovered. If some fields exceed capacity, error messages are generated showing the reason. For more information about this error, see APAR OA30632.

10.3.6 DFSORT support

DFSORT supports DSNTYPE LARGE for SORTIN and SORTOUT up to the new maximum size supported. When Basic Format data sets are used for SORTWK, the size limit is 65534 tracks (blocks). Using LARGE format data sets for SORTWK, the maximum usable size is 1,048,574 tracks. The Large Format data sets can be allocated larger, but the excess allocation will not be used.

10.3.7 DSNTYPE large format considerations

Large Format data sets (DSNTYPE=LARGE) used to have a size limit equal to the volume size. Due to limitations in the access method, these data sets now reach another limit of 16,777,215 tracks (1,118,481 cylinders) instead of the volume size. Earlier, the size of the data set was not checked, as the volume size was the limit.

The DADSM Create function has changed to handle a new diagnostic code ‘x’044E008E’. This code indicates when the limit of a Large Format sequential data set is reached. This diagnostic code is added to return code 216 (x’D8’) with the text: “The primary amount exceeded 16777215 tracks for DSNTYPE=LARGE” as shown in Figure 10-12 in ISPF.

```
DATA SET AZEEMM.EAVTB.TEST NOT ALLOCATED+
IGD17051I ALLOCATION FAILED FOR DATA SET
AZEEMM.EAVTB.TEST
, PRIMARY SPACE EXCEEDS 16,777,215 TRKS
***
```

Figure 10-12 IGD17051I message

In a space constraint relief allocation, DFSMS does not fail during an attempt to allocate more than the maximum size of a data set with a DSNTYPE of LARGE. Instead the allocation is adjusted to a valid allocation size. For a best fit allocation where the request exceeds the maximum size, the request is adjusted to the maximum value for this type of data set. If not available, a space allocation that can be recognized on the volume is substituted.

DADSM Extend has changed to limit DSNTYPE LARGE to extend over the maximum size for this data set type. DADSM Extend sets a return code of 4 and diagnostic code x'04210011' for 'basic' PS > 65535 tracks or for DSNTYPE=LARGE > x'FFFFFF' tracks. Example 10-7 shows an out-of space situation.

Example 10-7 Example of DSNTYPE LARGE not being able to extend

```
IEC030I B37-04,IFG0554A,COWDENKE,GEN,GENOUT,08C0,INDA81,04210011,COWDENK
```

10.3.8 ICKDSF changes

The initialization program ICKDSF has changed to support initialization and reformatting of 1 TB volumes of 1,182,006 cylinders. The support is applied through PTF for ICKDSF Release 17 (UK70554). PTFs for other platforms are (IBM z/VM®) UK70553 and (IBM z/VSE®) UK70555. Initializing a maximum size EAV looks like the ICKDSF job sysout in Example 10-8.

Example 10-8 ICKDSF example of initializing a maximum size EAV

```
ICK00700I DEVICE INFORMATION FOR 0F40 IS CURRENTLY AS FOLLOWS:
          PHYSICAL DEVICE = 3390
          STORAGE CONTROLLER = 2107
          STORAGE CONTROL DESCRIPTOR = E8
          DEVICE DESCRIPTOR = 0A
          ADDITIONAL DEVICE INFORMATION = 4A00003C
          TRKS/CYL = 15, # PRIMARY CYLS = 1182006
```

In the DS8000 control unit capacity is formatted in extents (1 GB on Open Systems and 1113 cylinders on CKD DASD). To optimize capacity, you can create volumes in increments of 1113 cylinders. ICKDSF now has this awareness, so it fails initialization of the volume if this criteria is not met as shown in Example 10-9.

Example 10-9 ICKDSF initializing an EAV, with a size not being a multiple of 1113 cylinders

```
ICK00700I DEVICE INFORMATION FOR 0F41 IS CURRENTLY AS FOLLOWS:
          PHYSICAL DEVICE = 3390
          STORAGE CONTROLLER = 2107
          STORAGE CONTROL DESCRIPTOR = E8
          DEVICE DESCRIPTOR = 0A
          ADDITIONAL DEVICE INFORMATION = 4A00003C
          TRKS/CYL = 15, # PRIMARY CYLS = 1177553
ICK31065I INVALID DEVICE SIZE: INIT COMMAND NOT SUPPORTED ON THIS DEVICE
```

The reason for this failure is that the primary cylinder allocation is not a multiple of 1113 cylinders (one extent in the DS8000).

10.3.9 DEVSERV PATHS and QDASD changes

The output display for QDASD and DEVSERV must support the new larger volume size. A maximum size volume looks like the QDASD display in Example 10-10.

Example 10-10 Sample display for QDASD on a large-size EAV volume

```
09.36.34 SYSTEM1          IEE459I 09.36.34 DEVSERV QDASD 625
 UNIT VOLSER SCUTYPE DEVTYPE       CYL  SSID SCU-SERIAL DEV-SERIAL EFC
00F41 IN7996 2107921 2107900   1182006  2606 0113-03261 0113-03261 *OK
```

The corresponding display for DEVSERV PATHS is shown in Example 10-11.

Example 10-11 DEVSERV PATHS display of a maximum size EAV volume

```
18.09.58 SYSTEM1          IEE459I 18.09.58 DEVSERV PATHS 600
 UNIT DTYPE  M CNT VOLSER  CHPID=PATH STATUS
      RTYPE  SSID CFW TC   DFW PIN DC-STATE CCA DDC       CYL CU-TYPE
00F4A,3390A ,0,000,EV9LIA,00=+ 04=+
      2107  201F  Y  YY.  YY.  N  SIMPLEX   3A        1182006 2107
```

```
************************ SYMBOL DEFINITIONS ***************************
O = ONLINE                          + = PATH AVAILABLE
```

All other support tools must already be in place. This larger size requires one more digit than in the previous support.

10.3.10 XRC changes

A larger volume requires a larger bitmap. One bit per track is required to hold the information about which tracks to copy. This process can require up to 2 MB of storage. Further internal changes to XRC have been made to support the larger EAV volumes.

10.4 Enhanced FTP support

From z/OS V1.13, data sets can be stored above the base addressing space on disk. The following data sets are also supported in the cylinder-managed area:

- Physical sequential basic
- Large format data sets
- PDS and PDSE data sets
- GDG data sets.

Because of this support, z/OS FTP can allocate and populate data sets.

10.4.1 Upgrade considerations

Make sure that programs that calculate and report data on a volume can support the larger size of an EAV volume. The program might have fields overrun due to larger amount of data accumulated for one volume.

10.4.2 Compatibility and coexistence

Support of 1 TB EAV volumes requires software and hardware support. z/OS V1.13 has the software support. For releases before zOS V1.13, only zOS V1.12 receives a small programming enhancement (SPE) (OA28553) that enables support at this level.
Example 10-12 shows the messages you receive when you try to bring a 1 TB EAV online without the required support.

Example 10-12 Messages when missing EAV support

```
IEE763I NAME= IECDINIT CODE= 0000000001000886
IEA434I DEVICE ONLINE IS NOT ALLOWED, 1182006 CYLS (MAX IS 262668)
IEE764I END OF IEE103I RELATED MESSAGES
```

The DS8000 microcode must be at these levels:

- IBM System Storage® DS8700: level 7.6.2.xx.xx (bundle version 76.20.xxx.xx) or later
- IBM System Storage DS8800: level 7.6.2.xx.xx (bundle version 86.20.xxx.xx) or later

11

DFSMSoam

This chapter describes DFSMSoam function that has been enhanced in z/OS V1R13. This chapter includes the following sections:

- OAM file system support
- OAM usability and reliability enhancements

11.1 OAM file system support

This new function in z/OS V1R13 introduces object access method (OAM) support for a new file system sublevel in the OAM storage hierarchy. It provides the following functions:

- An additional "disk" destination is provided in the OAM storage hierarchy. The existing hierarchy can consist of disk (implemented through DB2 tables on DASD), optical, and tape.
- Change from fast DASD to slower DASD
- Additional storage hierarchy targets, such as slow DASD
- Non-DB2 disk storage in the OAM storage hierarchy

11.1.1 z/OS V1R13 support

This new support provides additional flexibility in constructing the OAM storage hierarchy. For example, you can reuse older or slower DASD devices for zFS file system storage. Additionally, storage costs can be reduced by using less expensive disk in an NFS file server for NFS file system storage. Another consideration is using the file system sublevel as a form of "cache" when implementing the OAM Recall to Disk function.

> **Attention:** A program temporary fix (PTF) for z/OS V1R13 coexistence with APAR OA33022 must be installed on any pre-V1R13 level systems before starting OAM.
>
> OAM on pre-V1R13 level systems will not process objects in the file system sublevel.

In z/OS V1R13, the Disk level of the OAM storage hierarchy is now composed of these levels:

- Disk sublevel 1, which is the existing DB2 sublevel that uses DB2 tables
- Disk sublevel 2, which is the new file system sublevel

The new file system sublevel is a destination for primary objects stored as files in the z/OS UNIX file system hierarchy by using one of these file systems:

- zFS on native attached DASD
- NFS with a wide variety of storage options and technologies on network-attached NFS file servers

In an OAM environment, object storage groups allow the storage administrator to define an object storage hierarchy. The object storage hierarchy classifies storage areas according to location and, therefore, according to retrieval response time. Each object storage hierarchy must contain an object directory that contains control information about each object.

In z/OS V1R13, the OAM storage hierarchy can consist of these objects:

- DB2 object storage tables on DASD
- A file system (zFS or NFS)
- Optical storage
- Tape storage

> **Consideration:** Careful and complete planning is critical for a successful implementation of the file system capability within OAM. A number of considerations must be coordinated between the z/OS UNIX System Services environment and the OAM implementation. For more information, see *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426.

11.1.2 Installation considerations

The following topics introduce several installation considerations when you upgrade to z/OS V1R13 and use the new OAM file system support.

Attention: Run the CBRSMR1D SAMPLIB job if you are upgrading from any release earlier than z/OS V1R13. Whether you intend to take advantage of the new function, you must modify and run this job. It adds the File System Delete Table to the OAM Configuration Database. However, do not perform this step at initial installation. Perform this step for upgrade purposes only.

DB2 bind jobs

Make sure to run the appropriate DB2 bind jobs. This process is described in the *DFSMS migration action*s chapter in *z/OS V1R13 Migration*, GA22-7499. It is also covered in *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support,* SC35-0426.

The OAM bind jobs always include CBRPBIND, and typically include CBRHBIND and CBRABIND. For more information about the required upgrade actions and a complete upgrade checklist, see *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support,* SC35-0426.

File system security considerations

Security configuration for the file system is important to allow the OAM address space to access directories and files in the z/OS UNIX file system hierarchy. The Security Server (RACF) or equivalent security product must be configured. It must provide both a UNIX System Services group (with an associated group ID) and user (with an associated user ID) for the OAM started procedure.

When using the Security Server (RACF), a definition in the STARTED class is the preferred method for assigning identities to started procedures. These procedures include starting the OAM address space. For more information about the steps required to configure the Security Server (RACF) for the OAM started procedure, see *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426. The Resource Access Control Facility (RACF) configuration must be completed before beginning the steps listed under “OAM object storage group implementation”.

OAM object storage group implementation

For each OAM object storage group, perform these steps:

- Create a file system (zFS aggregate or NFS server definition).
- Create a mount point directory in the z/OS UNIX file system hierarchy.
- Mount the file system at a mount point directory and perform the following tasks for the directory:
  - Change owner/group to uid/gid for OAM address space.
  - Change permissions to '700' (rwx - only OAM address space).
- Create an OAM “sentinel” file in the file system at mount point and perform the following tasks for the file:
  - Change owner/group to uid/gid for OAM address space.
  - Change permissions to '600' (rw - only OAM address space).

- Add a SETDISK statement in the CBROAMxx PARMLIB member.
- Create or update an SMS storage class and ACS routines.
- Activate the SCDS.

Figure 11-1 illustrates the steps to prepare the UNIX file system hierarchy. The steps are to be completed for each object storage group in which you will use the OAM file system support.

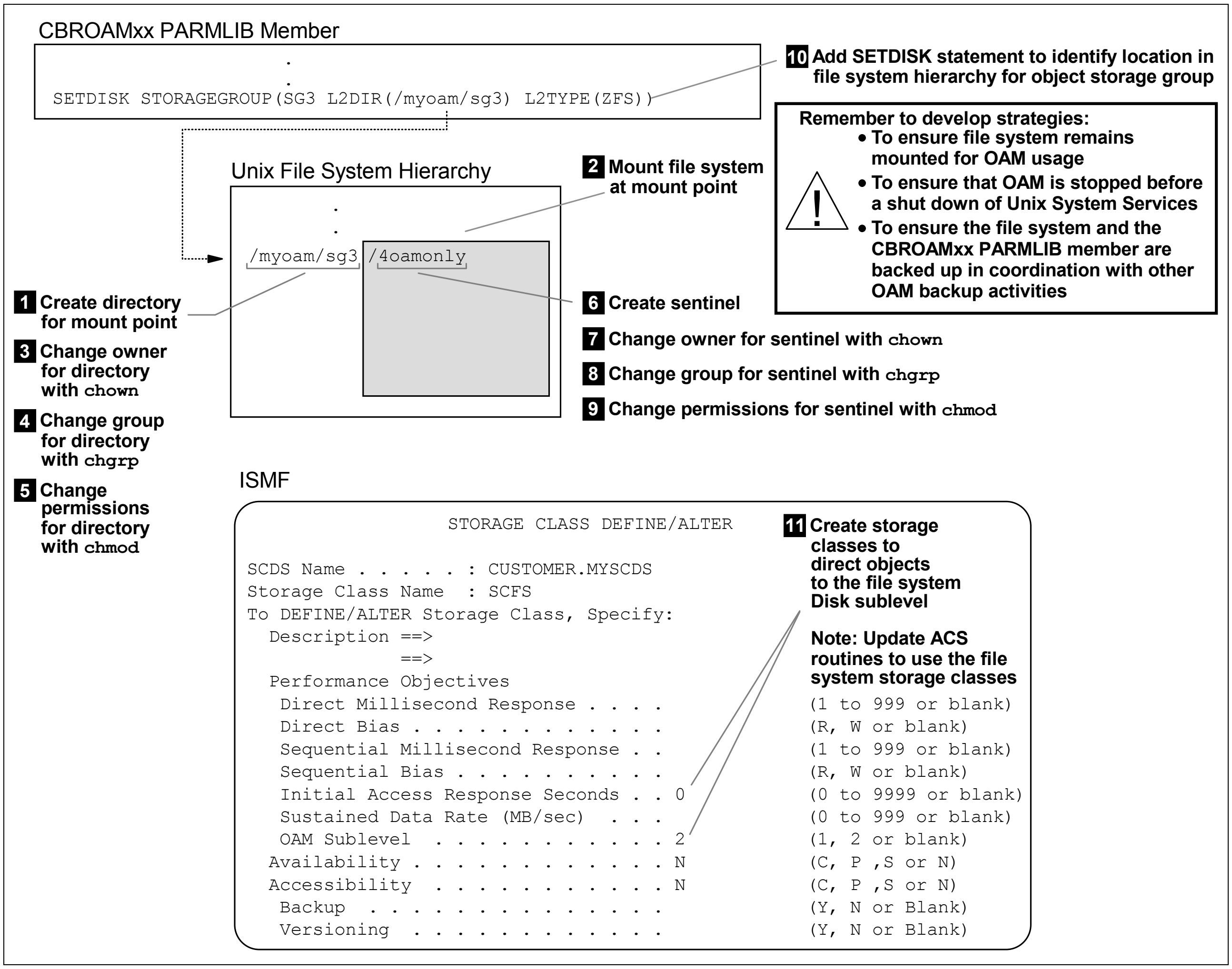


Figure 11-1 File system configuration summary

CBROAMxx PARMLIB member

The following changes have been made in the CBROAMxx PARMLIB member:

- A new SETDISK statement is available.

  The CBROAMxx PARMLIB member contains one or more SETDISK statements to configure the file system sublevel of the disk level in the OAM storage hierarchy. For each object storage group in which a file system sublevel is defined, a SETDISK statement must be specified. This statement provides the file system type and the file system directory to be used for the storage group. OAM uses these values to store objects in, and retrieve objects from, the file system. The file system type and file system directory for the object storage group must be carefully selected because these are static values and cannot be changed.

> **Attention:** The SETDISK statement is the only mechanism in OAM used to communicate the file system type and file system directory for the storage group. The file system type and file system directory specified is not recorded by OAM within DB2 database tables. The SETDISK statement must therefore continue to exist to provide access to objects in the file system sublevel.
>
> Therefore, backing up the CBROAMxx PARMLIB member to preserve these critical SETDISK statements must be included in your backup strategy. If a symbolic link is used, include the value of the symbolic link in your backup strategy.

- A new configuration option for the existing SETOPT statement in the CBROAMxx PARMLIB member is available to specify "Automatic Access to Backup" for file system errors.
- A new configuration option for the existing SETOSMC statement in the CBROAMxx PARMLIB member is available to specify a disk sublevel for "Recall to Disk":
  - 1 for the existing DB2 sublevel
  - 2 for the new file system sublevel

SMS storage class

Each object that OAM stores is associated with a storage class and a management class. OAM uses the storage class to determine the initial placement of an object in the OAM object storage hierarchy. OAM also uses the storage class to determine the correct placement of the object when the OSMC storage management cycle processes that object. OAM uses the Initial Access Response Seconds (IARS) parameter in the storage class. It uses this parameter to determine whether a primary copy of an object is stored on disk (DB2 tables or a file system) or on removable media (optical or tape). If the IARS parameter in the storage class that is assigned to the object is zero, the primary copy of the object is stored on disk. If the IARS parameter is nonzero, the primary object is stored on removable media.

Objects are directed to the file system sublevel of the OAM storage hierarchy by using the SMS storage class construct. Using ISMF, create a storage class or modify an existing storage class. Specify an `Initial access response seconds` value of 0 and an `OAM sublevel` value of 2 to direct objects to the file system sublevel

DB2 considerations

The OAM configuration database (CBROAM) contains configuration information related to the target destinations for objects that include tape volumes, optical libraries, drives, slots, and volumes. CBROAM also identifies objects to be ultimately deleted by OAM from optical volumes and the file system. It is a DB2 database, and contains a new table.

The File System Delete table contains one row for each object that is waiting to be deleted from the file system. Objects to be deleted from the file system can be deferred by adding them to a delete table in the File System for later physical deleting. The same will be the case for uncommitted writes (stores) to the File System. The name of this DB2 table is FSDELETE.

Also relative to DB2, note the following points:

- The existing ODINSTID field in the OAM Object Directory now can contain a value to identify unique instances of OAM files in the file system sublevel.
- The existing ODLOCFL field in OAM Object Directory now can contain new values:
  - 'E' when the object is in a new file system sublevel
  - '2' when the object is recalled to a new file system sublevel

11.2 OAM usability and reliability enhancements

Many changes have been made in z/OS V1R13 regarding OAM usability:

- Wildcard usage with the **F OAM,S,STORGRP** command
- Extend object expiration beyond 27 years
- Dynamic update of SGMAXTAPERETRIEVETASKS and SGMAXTAPESTORETASKS settings
- Improved media migration
- Enhanced OAM messages for specific DB2 errors
- SMF counter scalability
- CTICBR00 PARMLIB member
- CBR9875I recycle candidates display enhancement

11.2.1 Wildcard usage with the F OAM,S,STORGRP command

Operators must enter a separate command for each object or object backup storage group they want to process. To reduce keystrokes, the **F OAM,START,STORGRP,group-name** command can now accept a wildcard asterisk (*) to replace zero (0) or more characters in the group name. This feature enables installations with multiple object or object backup storage groups to start multiple storage group processes with a single command invocation.

The **F OAM,S,STORGRP,groupname** command has been enhanced to support a single asterisk as the last or only character in the group name. The following command starts processing for all object or object backup storage groups defined in the ACDS that have group names that start with GROUP:

```
F OAM,S,STORGRP,GROUP*
```

The following command starts processing for all object or object backup storage groups defined in the ACDS:

```
F OAM,S,STORGRP,*
```

Tip: The **F OAM,S,OSMC** command can be used to start processing of all object storage groups, but it ignores object backup storage groups.

11.2.2 Extend object expiration beyond 27 years

In previous releases, the maximum expiration criteria specified through SMS management class definitions (other than NOLIMIT) was 9999 days (roughly 27 years).

With z/OS V1R13, objects can still be retained FOREVER (or NOLIMIT). However, the 9999 day maximum associated with management class retention limit, Expire after Date/Days and Expire after Days Non-usage, has been expanded to 93000 days. Additionally, the maximum number of days specified through the RETPD and EVENTEXP keywords on the OSREQ API has also been expanded from 32767 to 93000.

SMS retention period

To use the expanded SMS retention period available in z/OS V1R13, you can set higher values in the following SMS Management Class attributes through ISMF:

- The retention limit
- Expire after Days Non-usage
- Expire after Days/Date

OSREQ keywords

Application programmers can optionally set higher values in the following OSREQ keywords:

- RETPD
- EVENTEXP

Table 11-1 shows the maximum values that are supported by OAM running on z/OS V1R13 and pre-R13 level systems.

Table 11-1 New maximum values for retention periods

Attribute	z/OS V1R13	Pre-V1R13
Management Class Retention Limit	93000	9999
Expire after Days Non-Usage	93000	9999
Expire after Days/Date	93000	9999
OSREQ RETPD	93000	32767
OSREQ EVENTEXP	93000	32767

11.2.3 SGMAXTAPESTORETASKS and SGMAXTAPERETRIEVETASKS

To change the distribution of tape drives allocated for OAM object and object backup storage groups, modify the SGMAXTAPESTORETASKS and SGMAXTAPERETRIEVETASKS values in the CBROAMxx PARMLIB member. Then restart OAM.

With z/OS V1R13, there is a mechanism to alter these values dynamically. The values specified for the SETOAM keyword in SGMAXTAPERETRIEVETASKS and SGMAXTAPESTORETASKS can be dynamically changed through the **`F OAM,UPDATE,SETOAM`** operator command. No restart of the OAM address space is required after issuing this command.

The following operator commands have been updated with this release:

- The **`F OAM,UPDATE,SETOAM,scope,SGMAXTPS`** command can be used to dynamically update the SGMAXTAPESTORETASKS associated with the specified storage groups.
- The **`F OAM,UPDATE,SETOAM,scope,SGMAXTPR`** command can be used to dynamically update the SGMAXTAPERETRIEVETASKS associated with the specified storage groups.

Consideration: This support applies to SGMAXTAPERETRIEVETASKS and SGMAXTAPESTORETASKS keywords specified at a storage group level.

The MAXTAPERETRIEVETASKS and MAXTAPESTORETASKS keywords specified at a global level are still not dynamically changeable.

11.2.4 Improved media migration

Processing tape or optical volumes with many collections can take a significant amount of time. This time is between when the **MOVEVOL** command is issued for a volume and the time of the first write to a new volume.

The OAM media migration utility, MOVEVOL, was enhanced to provide better performance characteristics for certain scenarios:

- The MOVEVOL algorithm was changed to no longer process objects on a collection by collection basis, thus allowing for more efficient processing and data movement. The changes in MOVEVOL can result in reduced time to migrate objects off a volume that contains objects from multiple OAM collections. The performance improvements are most significant when the source volume contains objects from many OAM collections.
- Additionally, before this support, running MOVEVOL on one member of an OAMplex resulted in measurable processor usage on “idle” members in the OAMplex. This usage is in reaction to XCF messages broadcast by the “active” member. With this support, the frequency of the broadcast messages from the active member is significantly reduced. This support results in much lower processor usage on the idle systems.

To move objects from a source volume, issue the following **MOVEVOL** command. *VOLSER* is the volume serial of the source volume from which objects are to be moved.

```
F OAM,START,MOVEVOL,VOLSER
```

11.2.5 Enhanced OAM messages for specific DB2 errors

OAM currently issues generic messages that display DB2 SQL codes when a DB2 error is encountered. The systems programmer must convert the hex return or reason code into a negative decimal SQL code and then look up the codes in DB2 manuals.

With z/OS V1R13, new messages are issued containing additional information for “common” SQL codes. This process can save the operator and storage administrator the trouble of having to derive the SQL codes and looking them up in the DB2 manuals.

11.2.6 SMF counter scalability

Some 4-byte counter fields in SMF Type 85, subtypes 32-35 and 87 that contain kilobyte values can overflow as workloads and tape capacity increase.

With z/OS V1R13, new 8-byte counter fields have been added to SMF Type 85, subtypes 32-35 and 87 to protect against potential overflow. The new 8-byte counters contain values in bytes. This enhancement avoids inaccuracies due to counter overflow (the 4-byte counters contain X'FFFFFFFF' if an overflow condition is detected).

The new 8-byte counters provide more granularity. They contain a number of bytes as opposed to a number of kilobytes in the old 4-byte fields.

The following OAM Subtype 32-35 counters are 4-byte fields that can overflow. A value of X'FFFFFFFF' in one of these fields indicates that an overflow was detected. The new 8-byte fields introduced in z/OS V1R13 supersede these 4-byte fields.

```
ST32PDWK  ST32PDRK  ST32PDDK  ST32POWK  ST32PORK   ST32PODK  ST32PTWK  ST32PTRK
ST32PTDK  ST32BOWK ST32BORK  ST32BODK  ST32BTWK  ST32BTRK  ST32BTDK ST32B2OWK
ST32B2ORK ST32B2ODK ST32B2TWK ST32B2TRK   ST32B2TDK ST32RCLK  ST32PUWK  ST32PURK
ST32PUDK
```

The following OAM Subtype 87 counters are 4-byte fields that can overflow. A value of X'FFFFFFFF' in one of these fields indicates that an overflow was detected. New 8-byte fields are introduced in z/OS V1R13 that supersede these 4-byte fields.

`ST87NKBW ST87NKBR`

11.2.7 CTICBR00 PARMLIB member

Installations had to copy the CBRCTI00 PARMLIB member to define OAM default trace options through the PARMLIB member. The member had to be copied from SYS1.SAMPLIB into the PARMLIB with a rename to CTICBR00.

With z/OS V1R13, OAM now ships the CTICBR00 PARMLIB member directly in PARMLIB. Therefore, the copy or rename step is no longer required, which can simplify OAM installation and migration.

11.2.8 RECYCLE candidates display enhancement

When an **F OAM,START,RECYCLE** command is issued, the Recycle candidates display message CBR9875I. This message is followed by a list of up to 40 volumes that meet the criteria specified by the **RECYCLE** command. This list is generated and sent to the hardcopy SYSLOG. The total number of volumes that meet the criteria for the **RECYCLE** command is not displayed.

With z/OS V1R13, the message line at the end of the Recycle candidates display shows the total number of volumes that met the criteria specified in the **RECYCLE** command. This configuration enables installation to better plan for tape recycle.

12

zSeries file system

zFS is not a part of DFSMS, but is used by it. This chapter provides an introduction to the functionality of the zFS file system. It also describes the modifications applied to zFS by DFSMS V1.13. If you are experienced with zFS, skip to 12.3, “New zFS functions in DFSMS V1.13” on page 175. For more information about zFS, see *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580.

This chapter includes these sections:

- Summary of the zFS enhancements
- zFS background information
- New zFS functions in DFSMS V1.13

12.1 Summary of the zFS enhancements

The following DFSMS V1.13 modifications affect zFS:

- zFS automatic takeover of disabled aggregates.

  In z/OS V1.13, zFS can automatically recover disabled aggregates when possible in both single-system and sysplex environments. This process works when multiple z/OS systems are running in zFS sysplex-aware mode. It is intended to eliminate the need to recover the file system manually before applications close and reopen the files to regain access to them. For more information, see 12.3.1, “zFS automatic takeover of disabled aggregates” on page 175.

- zFS internal restart.

  In z/OS V1.13, zFS maintains existing connections to zFS file systems while recovering from internal errors when possible. This configuration is intended to provide less-disruptive recovery from most internal zFS problems. It is designed to allow applications with open files to try file system operations again after zFS recovery is completed. For more information, see 12.3.2, “zFS internal restart” on page 176.

- zFS client direct I/O.

  zFS in z/OS V1.13 introduces the new Direct I/O function. This function allows all z/OS members of a sysplex to run zFS file system read and write I/O operations directly. This configuration is expected to yield substantial performance gains for z/OS systems not possible with zFS file owning systems. It does so by eliminating the performance impacts of zFS file owning systems. For more information, see 12.3.3, “zFS direct I/O” on page 177.

12.2 zFS background information

This section provides background about the functionality of z/OS Distributed File Service IBM zSeries® file system (zFS). It focuses on the basic concepts that are affected by the zFS modifications at DFSMS V1.13.

12.2.1 zFS highlights

zFS is a z/OS UNIX System Services (z/OS UNIX) physical file system. zFS is similar to a z/OS access method such as Virtual Storage Access Method (VSAM) with a key-sequenced data set (KSDS) organization. They are similar because a file system also deals with the way data is organized and accessed. zFS is a file system that can be used in addition to (or in replacement of) the original hierarchical file system (HFS) under z/OS UNIX. zFS file systems contain files and directories that can be accessed transparently (no difference from HFS) with z/OS UNIX application programming interfaces (APIs).

zFS has better serviceability and better performance than HFS because it is structured more like the z I/O architecture. Internally, it is a VSAM linear data set. Figure 12-1 shows the FILESYSYSTYPE TYPE(ZFS) parameter in BPXPRMxx in PARMLIB to define a zFS file system. Also, you can see the JCL procedure for activating the zFS address space.

- BPXPRMxx FILESYSTYPE statement
  - Defines zFS as a physical file system (PFS)

```
FILESYSTYPE TYPE(ZFS)
   ENTRYPOINT(IOEFSCM)
   ASNAME(ZFS)
```

ZFS PROC

```
//ZFS    PROC REGSIZE=0M
//ZFSGO EXEC PGM-BPXVCLNY,REGION=&REGSIZE,TIME=1440
//*STEPLIB   DD DISP=SHR,DSN=IOE.SIOELMOD
//IOEZPRM DD DSN=IOE.PARMLIB(IOEFSPRM),DISP=SHR
// PEND
```

Figure 12-1 Using the zFS physical file system

Alternatively, zFS file systems can be mounted into the z/OS UNIX hierarchy along with other file system types (for example, HFS, TFS, AUTOMNT, and NFS). Mounting a file system means to connect it to another file system as shown in Figure 12-2.

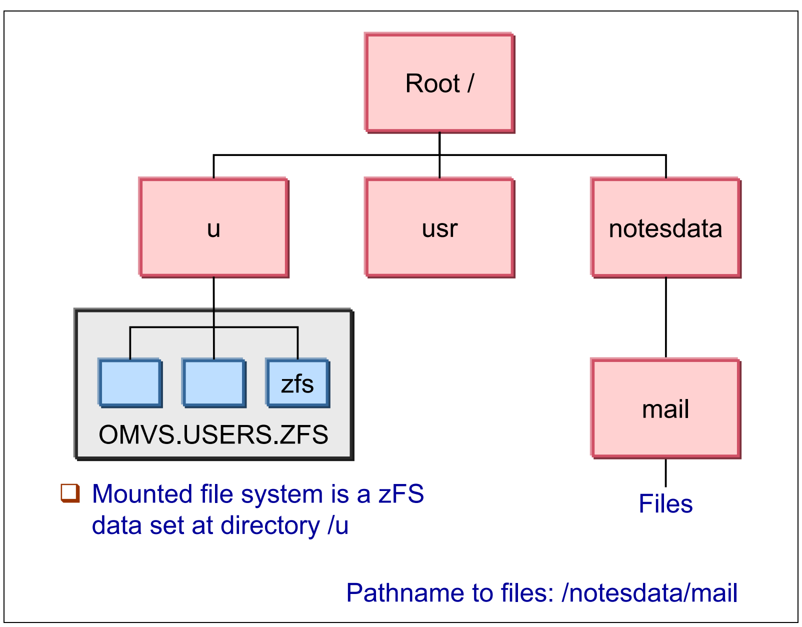


Figure 12-2 zFS file system mounted at the HFS file system

IBM announced that HFS is being functionally stabilized. As soon as possible, migrate your HFS file systems to zFS.

12.2.2 zFS aggregate concept

A zFS aggregate is a VSAM linear data set (VSAM LDS) that contains zFS file systems. An aggregate can have only one VSAM LDS, but it can contain an unlimited number of zFS file systems. The name of the aggregate is the same as the VSAM LDS name. Sufficient space must be available on the zFS aggregate 3390 volumes, as set by the IDCAMS DEFINE command. End-of-volume (EOV) routine decides when to allocate on these candidate volumes during any extension of a primary (or secondary) allocation areas.

Specifying an aggregate

A VSAM LDS greater than 4 GB can be specified. You must use the extended format and extend the addressability capability in the data class of the VSAM LDS cluster. After the aggregate is created, you must format the aggregate before any zFS file systems can exist in it. A zFS file system is a named entity in a zFS aggregate. zFS aggregates have these types:

- Compatibility mode aggregates as shown in Figure 12-3.
- Multi-file system aggregates, which are being phased out and should not be used.

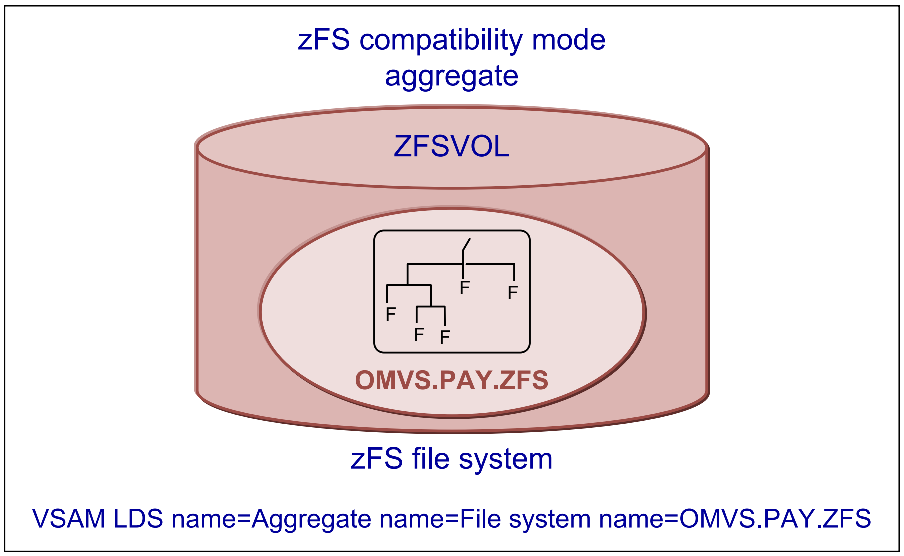


Figure 12-3 zFS aggregate in compatibility mode

A compatibility mode aggregate can contain only one zFS file system, making this type of aggregate more like an HFS file system. This system is flagged in the aggregate when it is created. The name of the file system is the same as the name of the zFS aggregate, which is the same as the VSAM LDS cluster name.

Diagnosing disabled aggregates in DFSMS V1.12

In this example, an internal error is detected by zFS, which is an abend with a completion code of 2C3. It is detected on an aggregate that is mounted with read/write (R/W) capability. Before DFSMS V1.13, zFS would attempt to isolate the failed aggregate rather than taking the full zFS file system down. As a result, zFS might mark an aggregate unavailable and issue this message:

```
IOEZ00422E Aggregate PLEX.JMS.AGGR001.LDS0001 disabled for writing
```

This message is in addition to a memory dump and possibly zFS trace information. To troubleshoot, the administrator can contact IBM service, providing the memory dump and the trace. They would also supply any other information that is useful for diagnosing the problem, such as what was running on the z/OS system when the problem occurred.

When an aggregate is disabled, applications that require this data cannot write or read to the aggregate. However, other aggregates that are not involved in the failure remain available.

An aggregate that has been disabled might be corrupted. zFS might have had an internal problem that disabled the aggregate to avoid writing anything incorrect into the aggregate. However, because this is an internal failure, zFS cannot ensure that the aggregate has no internal inconsistencies. To be sure that the aggregate is internally consistent, run the IOEAGSLV utility against the disabled aggregate.

Even though the aggregate is disabled, z/OS UNIX continues to display the aggregate mounted as R/W. To determine whether the aggregate is marked as disabled, use one of the following commands:

- **`zfsadm lsaggr`** to produce the output in Figure 12-4.

```
OMVS.DB2V9.SDSNMQLS.D110202                    SC63      R/W
OMVS.DB2V9.SDSN5HFS.D110202                    SC63      R/W
OMVS.DB2V9.SDSNWORF.D081006                    SC63      R/W
OMVS.DB2V9.SDSNWORF.D080701                    SC63      R/W
OMVS.DB2V9.SDSNMQLS.D071020                    SC63      R/W
OMVS.SC64.WAS7642.CONFIG.ZFS                   SC64      R/W
OMVS.DB2V9.SDSNWORF.D090701                    SC63      R/W
OMVS.DB2V9.SDSN5HFS.D071020                    SC63      R/W
OMVS.DB2V9.SDSNWORF.D090126                    SC63      R/W
OMVS.DB2V9.SDSNWORF.D070705                    SC63      R/W
OMVS.DB2V9.SDSNMQLS.D100722                    SC63      R/W
OMVS.DB2V9.SDSN5HFS.D100722                    SC63      R/W
OMVS.DB2V9.SDSNWORF.D070806                    SC63      R/W
```

Figure 12-4 Output from the zfsadm lsaggr command

- **`zfsadm aggrinfo`** to produce the output in Figure 12-5.

```
OMVS.SC63.WAS61PB3.CONFIG.HFS (R/W COMP): 136257 K free out of total 339840
OMVS.OAM.HFS (R/W COMP): 7036 K free out of total 7200
IMS11B.JMK1106.HFS (R/W COMP): 4823 K free out of total 25200
BB36163.SBBOHFS (R/W COMP): 310246 K free out of total 2347200
BB27064.SBBOHFS (R/W COMP): 237968 K free out of total 2376000
OMVS.TWS820.TWSCE2E.HFS (R/W COMP): 6350 K free out of total 7200
TWS.V8R6M0.SEQQHFS (R/W COMP): 14500 K free out of total 79200
MQ701.V101020.SCSQOJS.HFS (R/W COMP): 8348 K free out of total 165600
CICSTS41.HFS (R/O COMP): 134535 K free out of total 158400
HFS.ZOSR1C.Z1CRA1.XML (R/O COMP): 129685 K free out of total 1862640
HFS.ZOSR1D.Z1DRA1.XML (R/O COMP): 128261 K free out of total 1861200
OMVS.AEAGENT.HFS (R/W COMP): 7036 K free out of total 7200
RDZ801.D110520.ZFS (R/O COMP): 39364 K free out of total 136800
TWS.TWSABIN.RVA.ZFS (R/W COMP): 22617 K free out of total 41040
TWS.TWSAWRK.RVA.ZFS (R/W COMP): 289316 K free out of total 802800
```

Figure 12-5 Output from the zfsadm aggrinfo command

The disabled aggregate is unavailable until it is manually unmounted and mounted again by using this TSO command:

```
unmount filesystem('TWS.TWSABIN.RVA.ZFS’) remount(samemode)
```

12.2.3 Sharing zFS file systems

With shared zFS file systems, tasks programs have read/write access to file systems that are mounted on other z/OS system in the same sysplex. After a file system is mounted by a sysplex participating z/OS system, that file system is accessible by any other sysplex participating z/OS system. You cannot mount a shared zFS file system, so it is restricted to just one of those z/OS systems in a sysplex.

Consider a Parallel Sysplex that consists of three systems, SC63, SC64, and SC65 as shown in Figure 12-6:

- A user logged on to any system can change the file systems mounted on /u, and those changes are visible to all systems.
- The system programmer who manages maintenance for the Parallel Sysplex can change entries in both /etc file systems from either system.

The sysplex couple data set (CDS) contains the sysplex-wide mount table and information about all participating systems and all mounted file systems in the Parallel Sysplex. Shared zFS/ HFS file systems must be defined in the BPXPRMxx PARMLIB member.

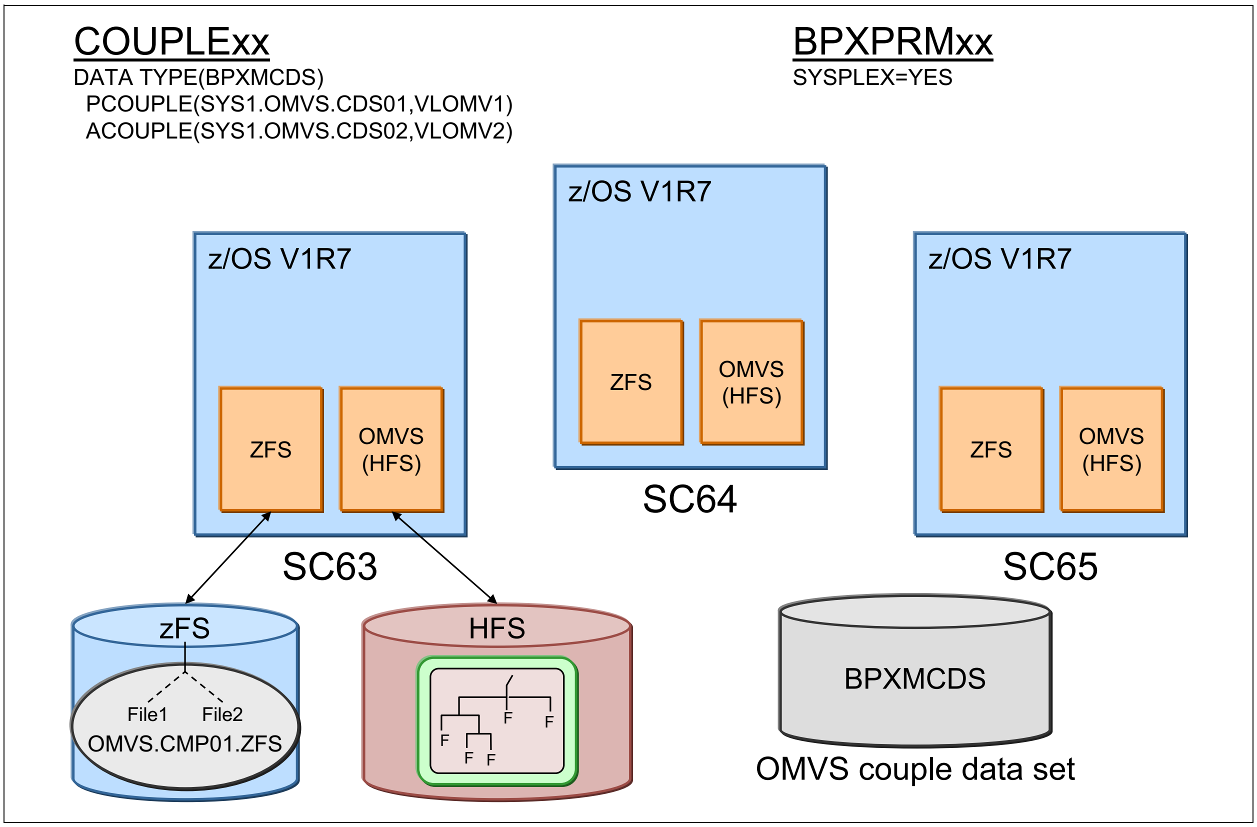


Figure 12-6 Sharing zFS file systems

The sharing support allows task programs accesses to file systems in R/W mode. However, before DFSMS V1.11, implementation of zFS/HFS data sharing in a Parallel Sysplex was under these rules:

- You can have many z/OS systems that share a file system in read-only mode.
- When zFS runs sysplex-aware (for read/write) on all systems, a read/write mounted sysplex-aware file system is locally mounted on all systems.

- There is still a z/OS UNIX owning system, but there is no z/OS UNIX function shipping to the owner.
- Requests from applications on any system are sent directly to the local zFS on each system. This configuration means that the local zFS is responsible for determining how to access the file system.
- One of the systems is known as the zFS owning system. This is the system where all I/O to the file system is done. zFS uses function shipping to the zFS owning system to access the file system.
- Each zFS client system has a local cache where it keeps the most recently read file system information. Therefore, in many cases when the data is still in the cache, zFS can avoid the zFS function shipping and satisfy the request locally.
- If one z/OS system has read/write access to a file system, the other z/OS systems must send requests to the z/OS owner through a cross-system coupling facility (XCF) signaling mechanism. This process also includes receiving and sending data. This traffic across XCF paths might impair the zFS access time and cause performance problems in the Parallel Sysplex.

Figure 12-7 shows three z/OS systems in a Parallel Sysplex that shares a pre-R11 environment. System 2 has R/W access to the zFS file system. Write requests are sent to System 2 using XCF.

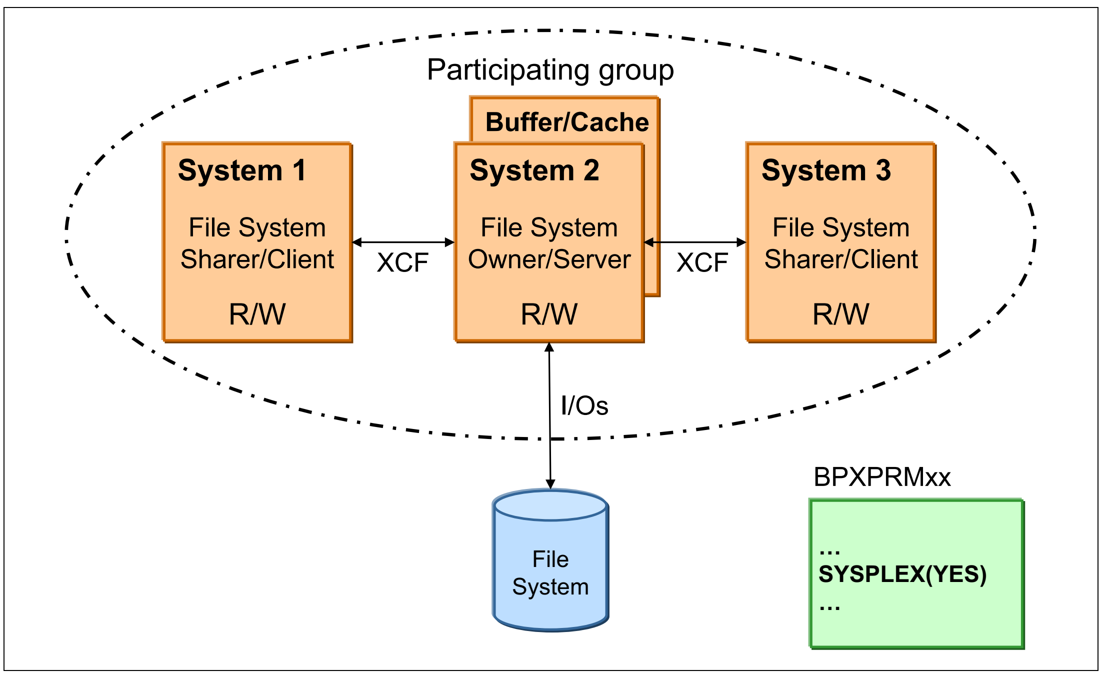


Figure 12-7 Use of XCF at zFS sharing for write requests in pre-V1.11 releases

12.3 New zFS functions in DFSMS V1.13

This section describes the enhancements in DFSMS V1.13 that apply to zFS.

12.3.1 zFS automatic takeover of disabled aggregates

This function can improve the availability of your z/OS systems that use the zFS file system.

In DFSMS V1.13, zFS can automatically recover disabled aggregates in both single-system and sysplex environments when multiple systems are running in zFS sysplex-aware mode. This process is intended to eliminate the need to recover the zFS file system manually before applications close and reopen the files to regain access to them.

zFS can internally remount any zFS file systems that were locally mounted, without requiring any z/OS UNIX support. If the failed file system is shared, zFS can run one of these tasks:

- Request that another z/OS system in the shared file system environment take over the aggregate (if it is sysplex-aware)
- Attempt an internal remount in the same mode

This action should recover and automatically re-enable the aggregate. For more information about the sysplex-aware file system, see 12.2.3, “Sharing zFS file systems” on page 174.

In summary, with zFS at the DFSMS V1.13 level, this implementation works as follows.

- zFS automatic re-enablement of disabled aggregates occurs automatically if zFS has disabled this aggregate. Therefore, zFS can recover a disabled aggregate without administrator intervention.
- zFS ownership of a disabled zFS sysplex-aware file system might change during automatic re-enablement of the disabled aggregate.
- As in previous releases, the installation and the applications might see failures as a result of zFS disabling an aggregate. Run the salvager utility (IOEAGSLV) against the aggregate at your earliest convenience.

Automatic re-enablement of a disabled aggregate that becomes disabled again is tried up to three times. If automatic re-enablement fails all three time, the zFS file system must be manually unmounted and mounted.

In this situation you can use one of the following commands, just as you could before this new function was introduced:

```
TSO: unmount filesystem('hering.test.zfs') remount(samemode)
sudo rexx: s "unmount HERING.TEST.ZFS" mtm_samemode
sudo /usr/sbin/unmount -f HERING.TEST.ZFS
```

Example 12-1 shows the messages issued upon automatic re-enablement.

Example 12-1 Messages on automatic re-enablement

```
IOEZ00422E Aggregate HERING.TEST.ZFS disabled
IOEZ00548I Requesting that SC75 takeover aggregate HERING.TEST.ZFS
IOEZ00388I Aggregate takeover being attempted for aggregate HERING.TEST.ZFS
IOEZ00044I Aggregate HERING.TEST.ZFS attached successfully.
```

12.3.2 zFS internal restart

This new function improves the continuous availability of your zFS file systems. In z/OS V1R13, zFS maintains existing connections to file systems while recovering from internal errors when possible. This configuration is intended to provide less-disruptive recovery from most internal zFS problems. It is designed to allow applications with open files to try file system operations again after zFS recovery is complete. You can use this command to see whether a zFS internal restart has taken place:

```
MODIFY ZFS,QUERY,STATUS
```

12.3.3 zFS direct I/O

This implementation significantly improves the performance of your z/OS systems, if you are sharing zFS file systems among z/OS systems in a sysplex.

zFS in DFSMS V1.13 introduces the new Direct I/O function. This function allows all z/OS members of a Parallel Sysplex to run zFS file system read and write I/O operations directly. This configuration is expected to yield substantial performance gains for z/OS systems that would not have been zFS owning systems in the prior design. It avoids the performance impacts to z/OS systems caused by zFS owning systems.

If one z/OS system has read/write access to a file system (the owner of the file system), the other z/OS systems are able to run I/O reads directly or use local buffer hits. The coherency of the buffer pool is ensured by a token mechanism. This process happens if the zFS file system is defined as sysplex-aware (SYSPLEXx=FILESYS) in PARMLIB member IOEZPRMxx. It must also be mounted as RWSHARE, either by using RWSHARE MOUNT PARM or IOEFSPRMxx's sysplex_filesys_sharemode=rwshare. Using the SYSPLEX=FILESYS and the RWSHARE mounting options allows you to individually choose which zFS read/write file systems are sysplex-aware, and which are not.

When all z/OS systems are at DFSMS V1.13, zFS can directly read and write sysplex-aware zFS file systems from all z/OS systems in the shared file system environment. This configuration makes zFS file system ownership much less important. There is less concern where applications run in a sysplex shared file system environment.

Figure 12-8 on page 178 shows that, if the zFS file system is mounted sysplex-aware, any client zFS can directly access the file system. This access allows reading and writing data, and also partially reading metadata. This configuration is also true with the file system FS2. Metadata update requests are always sent to the zFS owning system.

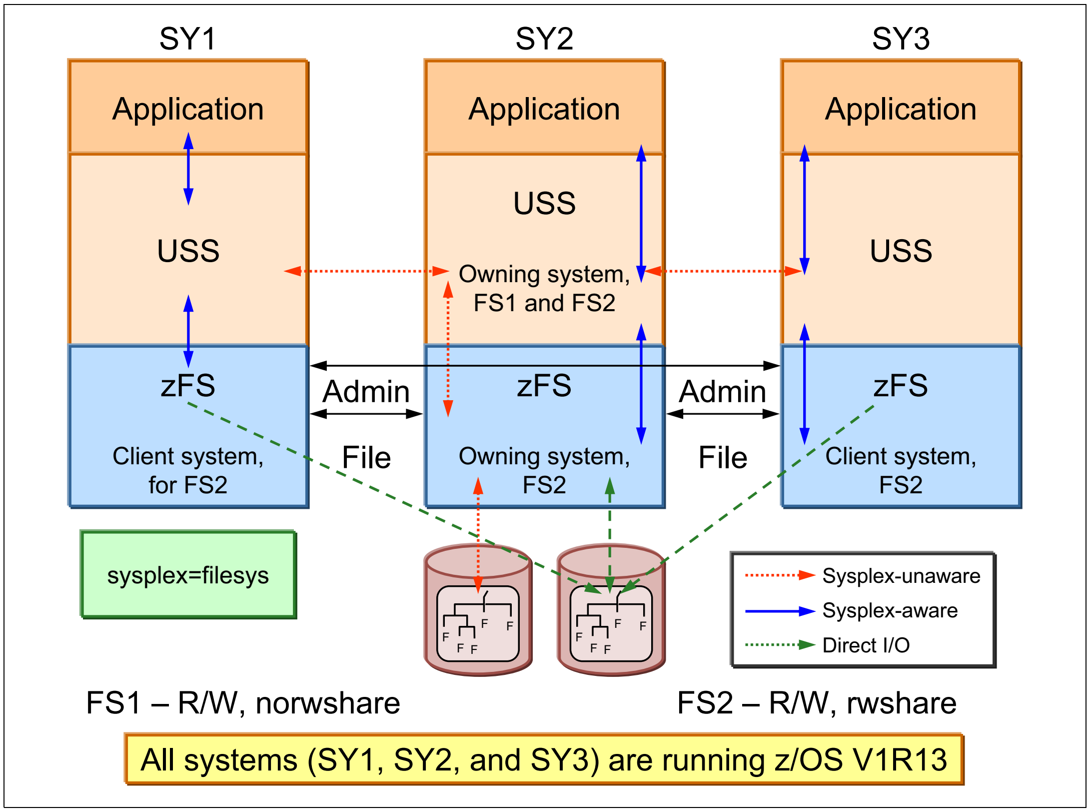


Figure 12-8 zFS sharing in DFSMS V1.13

13

SDM enhancements

This chapter addresses the System Data Mover changes introduced in DFSMS V1.13 as they relate to extended remote copy (XRC) and Peer-to-Peer Remote Copy (PPRC).

This chapter includes the following sections:

13.1 XRC time stamp suppression

Time Stamp suppression support (in 2008) addressed a problem with integrity between time stamps in the Application Systems and the Extended Remote Copy Data Mover Systems. Previously, you needed to have the time of day (TOD) clock in Data Mover Systems set to an earlier time than the Data Mover systems to avoid compromising data. Data integrity was still exposed with this approach with the manual setting of TOD.

The support provided a way of suppressing time stamps in the channel programs. It can be done on the Data Mover systems only. It should not under any circumstance be done in the Application systems. This suppression was done by modifying a bit in a sys1.nucleus module. However, this process was not in line with Change Control standards in most customer environments. It was desirable to have a more standardized way of suppressing the time stamping.

So in DFSMS V1.13, the XRC Time Stamp Support adds a parameter to PARMLIB member ANTXIN00 PARMLIB: **`SuppressTime Stamp(NO|YES)`**. If this parameter has a value of YES in the startup, a bit is set. This bit is always checked by an exit whenever a time stamp is about to be written, such as when set time stamp suppression is done. The default value for SuppressTime Stamp is NO.

> **Attention:** Do not run time stamp suppression on application systems. It is only appropriate on systems that are not updating data.

The preferred value of SuppressTime Stamp is NO. Use it only on System Data Mover systems, IBM Geographically Dispersed Parallel Sysplex™ (GDPS®) controlling logical partitions (LPARs), or other systems not sharing a common time reference.

To change or refresh the value of SuppressTime Stamp, change the setting to the wanted one. Then cancel ANTAS000 to put the change into immediate effect.

If the parameter is set to YES, message ANTX8179 is issued at startup to notify the operator of the setting. Additionally, message ANTX8030W is issued after a number of non-time stamped I/Os. A new PARMLIB parameter, Notime stampCount, decides whether this warning is issued.

If no ANTXIN00 member is present at the time of startup, the value SuppressTime Stamp(NO) is active.

Defaults remain the same as before DFSMS V1.13. If a change in behavior is needed, the new PARMLIB definitions can be implemented. Change from the APAR OA24780 zap option to this PARMLIB option.

13.2 Concurrent Copy PARMLIB support

Concurrent Copy backup is a feature in DFSMSdss (also called DSS) that can take a point-in-time backup. It quiesces the application involved for only a short while, making data almost 100% accessible and ensuring recoverability.

When the Concurrent Copy backup function is initiated, DSS identifies the tracks involved and calls System Data Mover (SDM) to set up the session. SDM creates a bitmap picture of the tracks to be backed up. The backup will start after releasing the temporary enqueue on the data involved. Backup is now logically complete, but continues in the background until all tracks are backed up. Updates to the tracks in the bitmap are allowed. But before this process happens, the track is copied to a cache sidefile in the control unit and written to the DSS backup file. This process is important because the contents of these tracks in the bitmap reflects the point-in-time backup requested.

An identified problem with Concurrent Copy can be backing up data while heavy (batch) updating goes on in parallel to this data. This problem forces SDM to continuously offload the tracks about to be updated to cache and from there to the DSS output file.

To meet these issues with Concurrent Copy backup, two new keywords have been added to the ANTMIN00 member in SYS1.PARMLIB.

- *CCREADHAEAD* allows you to adjust the default number of buffers according to individual customer needs. Setting this parameter to 32 provides 32 additional 64 K buffers (2 MB) as fixed storage for the IOS. This configuration enables the backup to run more efficiently. The default value is 3.
- During peaks, *CCATTNTHROTTLE*(YES!NO) allows you to reduce parallel processing at a controller level. If set to YES, it makes the overall CC session operate more efficiently. A value of YES limits the number of parallel processes per controller session to two, as shown in Example 13-1.

Example 13-1 SYS1.PARMLIB(ANTMIN00) example using the new CC parameters

```
Startup -
     CCREADAHEAD(5) -
     CCATTNTHROTTLE(YES) -
     Hlq(SYS1) /* High level qualifier for VCC data sets */
FLAG NAME(SMFVCC) ACTION(OFF)
```

Besides setting the new parameters through PARMLIB, they can also be changed by using modify command to ANTMIN. Set new buffers for CCREADAHEAD by using this command:

```
F ANTMAIN,CCREADAHEAD 20 /* Session will obtain (20*64kb) as the buffer area */
```

Changing to the new configuration

The new support is not apparent to previous releases. If tuning is needed, implement the new PARMLIB parameters. Otherwise, do not change CCReadAhead parameter unless you face a performance problem with Concurrent Copy.

The problem can have one of these symptoms:

- Increased elapsed time on the Concurrent Copy jobs
- Increased contention for ANT.* latch resources (as shown in a **D GRS,C** display)
- Concurrent Copy jobs fail with 00000008-00000013 return/reason code.

In general, do the changes incrementally and watch the influence of each change. If no visible effect results, return to the previous setting. In addition, always monitor any change over time. There can be an initial benefit, but it might be followed by a degradation. When possible, use a test environment before putting a change into production.

Compatibility and coexistence

Toleration support is provided for the lower-level systems, enabling them to recognize the new PARMLIB values and also allow ANTMAIN initialization to happen without errors.

13.3 MaxTotalReader Task

This support addresses the contents of APAR OA28400. When the multiple reader support was implemented, it introduced the new parameters AllowEnhancedReader and MaxTotalReaderTasks. These parameters are used to specify the use of enhanced multiple reader support. They also limit the logical subsystem (LSS) and storage control session ID (SCID) numbers in one XRC session.

If AllowEnhancedReader(NO) is set, the value for MaxTotalReaderTasks defaults to 80. If the value for AllowEnhancedReader is set to YES, the MaxTotalReaderTasks is 32. However, MaxTotalReaderTasks still used the value of 80. In DFSMS V1.13, this configuration has been changed. AllowEnhancedReader(YES) uses the default of 32, whereas AllowEnhancedReaders(NO) has a default of 40.

Check the number of readers in your XRC session. If there are fewer than the new default of 40, adjust the MaxTotalReaderTask to a number that is equal to or greater than the number of sessions you have.

The number 40 is considered a good limit as performance is better at, or under, this limit. If you are beyond 40, consider splitting this session in two sessions with fewer utility volumes.

13.4 XRCSTART error handling

APAR OA31999 describes a problem after an XRCSTART. XRCSTART might fail and issue messages like those shown in Example 13-2.

Example 13-2 Sample error messages after an XRCSTART

```
ANTX5002E C1A0,******,0008,0028,004D,0000
ANTX5002E C1A0,******,0008,0028,004D,0000
ANTX5002E C1A0,******,0008,0028,004D,0000
```

This problem is caused by the logic previously used when reestablishing clustering. A fix in DFSMS V1.13 has changed this process so that it can handle this type of error in a coupled environment.

13.5 XRC query filter option

The STA() parameter on the XQUERY VOLUME PACE command has the new parameter BK2 added. Currently STA(BLK) shows blocked volumes in the XRC QUERY display. These volumes are marked with device blocking and long busy. The list allows you to identify bottlenecks that affect your overall availability and recoverability. However, this list does not contain all the volumes contributing to application impact.

To improve the identification of contributors to application impact, The new BK2 parameter is added to STA. This parameter lists all the blocked volumes with a higher residual count than 64 (times the value of their WrtPacingResidualCnt value). The type must be one of these types:

- Blocked
- Write paced
- Donotblock

To have the XPM batch exception monitor generate the new volume pace report, PTF UA25536 (with subsequent fixes UA27043 and UA34928) must be added. The XPM monitor currently has a parameter PACE with possible values of Y or N. If N is used, **XQUERY V(ALL) STATUS(BLK) DETAIL** is used. If the value is Y, **XQUERY V(ALL) STATUS(BLK) PACE** is used.

With the new support, the parameter 2 is added, which results in the following display command being issued:

```
XQUERY V(ALL) STATUS(BK2) PACE
```

If you want to take advantage of the new parameters, you need to start the implementation. However, they are not needed at the time of upgrade to DFSMS V1.13. The XPM batch exception monitor needs the PTFs mentioned previously applied to support the new implementation.

The new parameter STA(BK2) is recognizable on prior level systems with planned toleration support, enabling them to produce the new report.

13.6 PPRC linkinfo query

Before DFSMS V1.13, it was not possible to extract control unit information through the **CQUERY** command. You had to find out this information from the customer engineer (CE) and document it for further use.

Support in DFSMS V1.13 adds this function for 2105 and 2107 control units. This function is enabled by adding CQUERY PATHS(LNK) when using the ANTRQST API or CQUERY LINKINFO using TSO. These displays return link information between the primary and secondary control unit, indicating the adapter IDs to be used for specification of links in the CESTPATH command.

The CQUERY PATHS needs either of these keywords: SWWNN(wwnn), which is the world wide node name, or SDEVN(devn). If both are specified, SDEVN is ignored. An optional keyword SSUBCHSET can be provided. If this keyword is specified without SDEVN, it is ignored.

For CQUERY LINKINFO, the mutually exclusive keywords SWWNN(wwnn) and SDEVN can be specified. CQUERY PATHS also has the optional keyword SSUBCHSET. However, if this keyword is specified without SDEVN, it is ignored.

Table 13-1 shows TSO CQUERY parameters.

Table 13-1 PPRC TSO command changes

Parameter	Description
LINKINFO	PPRC displays all potential connectivity (for Fibre Channel ports) from the control unit that holds the DEVN (or SWWNN) specification. The target adapter ports are displayed for the control unit that holds the SDEVN specified volume.
SDEVN(device-number)	Device number in the secondary control unit that is used for the query operation.

Parameter	Description
SWWNN(world wide node name)	World wide node name of the secondary control unit in 8 byte hexadecimal.
SSUBCHSET(subchannel_set)	The secondary control units subchannel set in which the command is displayed. Defined as specified in the hardware configuration definition (HCD). Values are: ‘0’ = subchannel 0. ‘1’ = subchannel 1. If specified with SDEVN, this parameter is ignored.

Use this query when setting up new PPRC relationships instead of having to rely on manually collected information. Example 13-3 shows the results from this query.

```
cquery devn(cc20) linkinfo sdevn(dc00)
```

Example 13-3 Sample cquery

```
ANTP0259I CQUERY FORMATTED LVL 1
LINKINFO REPORT
************** PPRC REMOTE COPY CQUERY - LINKINFO ******************
*                                      DEVN SCH       WWNN         *
*                                      ---- --- ----------------   *
* PRIMARY      SECONDARY          PRI: CC20   0 5005076303FFC766   *
* ADAPTER      ADAPTER            SEC: DC00   0 5005076303FFC752   *
* IDS     STAT IDS                                                 *
* ------- ---- -------------------------------------------------   *
*   0103    0  0103                                                *
*   0302    3                                                      *
*   0233    0  0233                                                *
*   0033    0  0033                                                *
*   0303    0  0303                                                *
********************************************************************
ANTP0001I CQUERY COMMAND COMPLETED FOR DEVICE CC20. COMPLETION CODE: 00
```

14

Interactive Storage Management Facility (ISMF) enhancements

This chapter provides some introductory information about the functionality of the ISMF and the modifications that apply to SMS/ISMF in DFSMS V1.13. The introductory information helps you to understand these modifications. However, if you are experienced with SMS/ISMF, skip to 14.3, "SMS/ISMF modifications in DFSMS V1.13" on page 188.

This chapter includes the following sections:

- Summary of SMS/ISMF enhancements in DFSMS V1.13
- SMS/ISMF basic concepts
- SMS/ISMF modifications in DFSMS V1.13

14.1 Summary of SMS/ISMF enhancements in DFSMS V1.13

These SMS/ISMF enhancements are provided with DFSMS V1.13:

- A new SMS PARMLIB parameter

  Error messages generated during ISMF processing are passed back to the caller, which is responsible for externalizing these messages. In DFSMS V1.13, this approach has been modified so that ISMF externalizes its own error messages to the hardcopy and job logs. For more information, see 14.3.1, "New SMS PARMLIB parameter" on page 188.

- Increased retention period to larger than 9999 days

  With this support, the maximum retention period is now increased to 93000 days, which is about 254 years. For more information, see 14.3.2, "Data set retention period to larger than 9999 days " on page 190.

- Updated volume space statistics

  In certain situations, ISMF shows more space information about a data set. For more information, see 14.3.3, "Providing updated volume space statistics " on page 192.

- Modification to SMS to handle space requests greater than 2 terabytes

  Several z/OS and DFSMS components have restrictions about the maximum size of a data set. In DFSMS V1.13, some of these restrictions are relieved. For more information, see 14.3.4, "Support for data set space greater than 2 terabytes " on page 192.

- An operator command to display SMS CSECT ID information

  This command provides more problem determination data about SMS failures. For more information, see 14.3.5, "Operator command to display SMS CSECT ID information " on page 194.

- Miscellaneous RAS improvements.

  For more information, see "14.3.6, "Miscellaneous RAS improvements " on page 195.

14.2 SMS/ISMF basic concepts

Storage management subsystem (SMS) is a set of techniques that allow storage administrators to manage storage resources such as tape, optical, and solid state.

14.2.1 ISMF overview

Interactive Storage Management Facility (ISMF) is a component of DFSMSdfp that allows you to define and maintain policies to manage your storage resources.

These policies help to improve the usage of storage devices and to increase levels of service for user data. Minimal effort is required from users. More specifically, the storage administrator uses ISMF to define the following items:

- Data classes
- Storage classes
- Management classes
- Storage groups
- Aggregate groups
- Copy pools
- Automatic class selection (ACS) routines

These SMS interfaces are used by storage administrators to implement SMS policies:

- ISMF panels, where user selected definitions and parameters are stored in a source control data set (SCDS)
- PARMLIB member IGDSMSxx

14.2.2 Retention period and expiration date

The expiration date is used to protect a new data set against accidental deletion and to manage tape volumes. When the expiration date elapses, the data set can be deleted or overwritten by another data set.

Declaring the expiration date

An expiration date can be declared in job control language (JCL) and at the data set associated data class and management class SMS constructs. It is enforced by the JCL allocation function, TSO/E, OAM, DFSMShsm, and DFSMSrmm.

You can declare the expiration date by using these methods:

- Use the expiration date (EXPDT) parameter to specify the expiration date for a new data set.

  EXPDT= {yyddd} / {yyyy/ddd}. Expiration dates of 99365, 99366 and 99999 are considered logically “never-scratch” dates. The expiration date is stored (YYDDDD) in control blocks, and the year is offset from the year 1900. Therefore, an expiration date of FF016D is the highest value that is possible (2155-12-31).
- Use the retention period (RETPD) parameter to specify the retention period for a new data set.

  RETPD=**nnnn**, specifies the retention period, in days. The **nnnn** is one through four decimal digits (0 - 9999). z/OS adds **nnnn** to the current date to produce an expiration date.

z/OS time limits

There are two z/OS time limits that affect the data set expiration time figure:

- z/OS relies on the processor time-of-day (TOD) clock for keeping the time of day, which is used in several functions within the operating system. One of them is to determine when a data set expiration date expires. A TOD clock provides a high-resolution measure of real time that indicates the date and time of day. The cycle of the clock is approximately 143 years.

  The TOD clock is initialized by z/OS or by a Server Time Protocol (STP) panel in the HMC hardware console. The base value is 1900-01-01. The TOD wraparound time is approximately 2043-01-01. Currently no data sets can expire later than this date.
- Allocation and Scheduler (JES) routines convert the retention period to an expiration date in the hexadecimal format of YYDDDD. In this format, YY is the year (0-255) and the DDDD is the days. Because the year is offset from 1900, an expiration date of FF016D (year 2155) is the highest value that is possible.

Combining both time limits, the maximum expiration date is 2041, and not 2155.

With SMS, the JCL parameters override the expiration date defined in the data class parameter for the data set.

With SMS, both the expiration date specified on EXPDT and defined in the data class for an SMS-managed data set can be limited by a maximum expiration date. This date is defined in the management class for this data set.

14.2.3 Data set space limitations

Several z/OS and DFSMS components have restrictions on the maximum size of a data set. This number applies to a total data set capacity that includes the space in all DASD volumes on which a multivolume data set is located. These restrictions are related to the current space definitions in internal control blocks fields:

- MVS Allocation has a limit in the neighborhood of 16 million 3390 cylinders, or about 12.8 TB.
- ISMF stores the data set size in a signed fullword field. The data set size limit that can be handled is X '7FFFFFFF' KB. This limit translates to roughly 39 million 3390 tracks, or 2.5 million 3390 cylinders, or about 2 TB.

The trend is for larger data sets fueled by large customers who are starting to use the EAV support in DFSMS V1.13. In DFSMS V1.13, each 3390 volume can reach 1,182,006 cylinders.

14.2.4 Job File Control Block (JFCB)

The JFCB is a control block associated with an allocated data set. The source of its contents comes from certain data definition (DD) statement parameters or their equivalent in a Dynalloc macro. When in a DD statement, a Job Entry Subsystem (JES) Interpreter/Converter routine creates and passes it to an Initiator. The Initiator builds the JFCB in the scheduler work area (SWA), usually above the 16 MB line.

The source parameters for the JFCB are the ones that describe, at execution time, certain fields of a data control block (DCB) or an application control block (ACB) in a DD statement. JFCB contents are used by the Open routine to fill and sometimes override some of these DCB and ACB fields.

14.3 SMS/ISMF modifications in DFSMS V1.13

This section addresses the enhancements to SMS/ISMF in DFSMS V1.13.

14.3.1 New SMS PARMLIB parameter

This parameter can improve the RAS of your z/OS systems. SMS is a service that delivers storage service functions to requesting clients, such as TSO. Error messages generated during SMS processing are passed back to the caller, which is responsible for externalizing these messages.

In DFSMS V1.13, in specific situations, this approach has been modified so that SMS externalizes its own error messages to the hardcopy and job logs. Examples include DELETE and RENAME processing.

The purpose of this enhancement is to provide the installation with an option to control the issuance of these DELETE/RENAME messages. Depending on your installation choice, SMS either issues these messages or passes them back to the caller. For this configuration, a new parameter is added to the IGDSMSxx member of SYS1.PARMLIB. This parameter has the following format:

```
SUPPRESS_DRMSGS (YES|NO)
```

The default is NO. If this new parameter is not supplied, the DELETE/RENAME messages are NOT suppressed by SMS. Specification of YES suppresses the issuance of these messages to the hardcopy and job logs by SMS. This parameter in no way affects the issuance of these messages by the caller. Some callers of SMS issue these messages on their own, whereas others do not.

The SCOPE of this parameter is the entire MVS system. A facility is provided to modify this parameter between IPLs or issuances of the SET SMS command. Example 14-1 shows a partial list of IGDSMSxx parameters, including the SUPRESS_DRMSGS command.

Example 14-1 SET SMS command output

```
-D SMS,OPTIONS
            IGD002I 10:20:53 DISPLAY SMS 148
            ACDS     = SYS1.SMS.ACDS
            COMMDS   = SYS1.SMS.COMMDS
            ACDS LEVEL = z/OS V1.13
            INTERVAL = 15           DINTERVAL = 150
            SMF_TIME = YES          CACHETIME = 3600
            CF_TIME = 1800          PDSE_RESTARTABLE_AS = YES
....
....
            GDS_RECLAIM = YES       DSSTIMEOUT = 0
            BLOCKTOKENSIZE = NOREQUIRE       FAST_VOLSEL = ON
            USEEAV = YES            BREAKPOINTVALUE = 21
            OAMPROC =               SUPPRESS_DRMSGS = YES

--- After the SETSMS SUPPRESS_DRMSGS(NO) command.
            -D SMS,OPTIONS
             IGD002I 10:23:46 DISPLAY SMS 145
             ACDS     = SYS1.SMS.ACDS
             COMMDS   = SYS1.SMS.COMMDS
             ACDS LEVEL = z/OS V1.13
             INTERVAL = 15           DINTERVAL = 150
             SMF_TIME = YES          CACHETIME = 3600
             CF_TIME = 1800          PDSE_RESTARTABLE_AS = YES
             PDSE_BMFTIME = 3600     PDSE1_BMFTIME = 3600
             PDSE_LRUTIME = 60       PDSE1_LRUTIME = 50
             PDSE_LRUCYCLES = 15    PDSE1_LRUCYCLES = 200
....
....
            PDSE_BUFFER_BEYOND_CLOSE = NO   PDSE1_BUFFER_BEYOND_CLOSE = NO
             GDS_RECLAIM = YES       DSSTIMEOUT = 0
             BLOCKTOKENSIZE = NOREQUIRE       FAST_VOLSEL = ON
             USEEAV = YES            BREAKPOINTVALUE = 21
             OAMPROC =               SUPPRESS_DRMSGS = NO
```

Figure 14-1 shows a partial listing of the parameters contained in the IGDSMSxx member of PARMLIB.

```
PDSE_MONITOR=(YES|NO,interval,duration
PDSE1_MONITOR=(YES|NO,interval,duration)
PDSE_DIRECTORY_STORAGE=nnnnM
PDSE1_DIRECTORY_STORAGE=nnnnM
PDSE_BUFFER_BEYOND_CLOSE={YES|NO}
PDSE1_BUFFER_BEYOND_CLOSE={YES|NO}
GDS_RECLAIM={YES|NO}     DSSTIMEOUT=nnnn
BLOCKTOKENSIZE={REQUIRE|NOREQUIRE}     FAST_VOLSEL={ON|OFF}
USEEAV={YES|NO}      BREAKPOINTVALUE={nnnnn|10}
OAMPROC=procname
OAMTASK=taskid
DB2SSID=ssid
CA_RECLAIM={NONE|DATACLAS}
SUPPRESS_DRMSGS={YES|NO}
```

Figure 14-1 IGDSMSxx parameters (partial)

14.3.2 Data set retention period to larger than 9999 days

This enhancement improves the usability and security of your z/OS systems. Currently the data set and tape volume retention period is limited to 9999 days, which is about 27 years. However, there are legal requirements that a document be retained for longer than 27 years. With this support, the maximum retention period is now increased to 93000 days, which is about 254 years.

However, the currently imposed maximum date of 2155-12-31 is still in force, as are the no-expire dates of 99365, 99366 and 99999. For more information, see 14.2.2, “Retention period and expiration date” on page 187. Therefore, due to compatibility reasons the maximum effective retention period is less than 93000 days. Because the year is offset from the year 1900, an expiration date of FF016D is the highest value that is possible. If you use a retention value of 93000, where the calculated expiration date is beyond the year 2155, the maximum value of FF016D is used instead.

In DFSMS V1.13, the data set retention period is supported by JCL, TSO/E, DFSMShsm, and DFSMSrmm. It is increased from the current limit (up to 9999 days or approximately 27 years) to 93000 days or approximately 254 years. This higher limit is intended to make it easier to retain data for longer periods of time.

This section addresses the effects of making the retention period (and so the expiration date) larger than 9999 days for each DFSMS or z/OS component. For more information, see Table 14-1 on page 192.

OAM support

OAM is not subject to the currently imposed maximum date of 2155-12-31, and therefore the full 93000 day retention limit is applicable to OAM objects.

With this support, the object expiration criteria can be expanded to 93000 days from create. The optional RETPD keyword on the OSREQ STORE and CHANGE functions is expanded to accommodate the new management class retention limit of 93000 introduced in z/OS V1.13. Additionally, OAM recognizes the new management class expiration maximum values of 93000 days for Expire after Date/Days and Expire after Days Non-usage.

ISMF support

Several ISMF parameters in data class and management class and their related fields are incremented to allow a retention period of 93000 days:

- The Retention Period and the Expired Days After Creation values on the ISMF management class definition panel
- The Retention Period value on the ISMF ACS Test Case, and Data Class panels

The Data Class DEFINE / ALTER panel (DGTDCDC1) is modified to have the field Retpd or Expdt accept a value up to 93000:

```
Retpd or Expdt  . . . . . _________ (0 - 93000, YYYY/MM/DD, or blank)
```

The Management Class DEFINE / ALTER r panel (DGTDCMC1) is modified to have these fields accept a value of up to 93000:

- Expire after Days Non-usage . . NOLIMIT (1 - 93000 or NOLIMIT)
- Expire after Date/Days . . . . . NOLIMIT (0 - 93000, yyyy/mm/dd, or NOLIMIT)
- Retention Limit . . . . NOLIMIT (0 - 93000 or NOLIMIT)

ACS Test Case DEFINE /ALTER panel (DGTDFFL3) is modified to have the field Retpd accept a value up to 93000. The field is expanded to five digits.

IDCAMS

IDCAMS converted its retention-period-to-date routines in both AMS DEFINE and the ALTER commands. This new limit allows up to 254 years instead of 27 years. However, the currently imposed maximum date of 2155-12-31 is still in force, as are the no-expire dates of 99365, 99366, and 99999.

As result of this support, the parameter FOR (days) of the DEFINE and ALTER commands accepts the maximum number of days of 93000 instead of 9999. If the number is 0 through 92999, the object is retained for the specified number of days. If the number is 93000 or 9999, the object is retained indefinitely.

The IDCAMS DCOLLECT record type MC is changed to record a new 3-byte field for retention period.

DFSMShsm

HSM supports the changes to the expiration date described previously. In summary, when DFSMShsm sets 99365 as the expiration date to manage its tapes, 99365 means permanent retention or to never expire. If you choose to use DFSMShsm expiration date protected tape volumes, DFSMShsm sets the date 99365. This setting prevents DFSMSrmm from considering the volumes for release at any time. You must specify the MAXRETPD(NOLIMIT) operand to ensure that DFSMSrmm recognizes the 99365 date.

DFSMSrmm

You also use the DFSMSrmm PARMLIB MAXRETPD operand value to reduce the expiration date for all volumes that include DFSMShsm volumes. If you want to reduce the 99365 permanent retention expiration date, specify the MAXRETPD with a value of 0 - 9999 days.

If you want to avoid the processor needs of VRSEL processing for tape volumes managed by DFSMShsm (and similar applications), use the EXPDT retention method. If you require DFSMSrmm to manage the movement of DFSMShsm volumes, use DFSMSrmm vital record specifications instead of the 99365 permanent retention date to retain DFSMShsm volumes.

Table 14-1 shows the new maximum values for retention periods.

Table 14-1 New maximum values for retention periods

Attribute	z/OS V1.13	Pre-V1.13
Management class retention limit	93000	9999
Management class expiration attributes	93000	9999
Expire after days non-usage	93000	9999
Expire after days/date	93000	999
OSREQ RETPD	93000	32767
OSREQ EVENTEXP	93000	32767

TSO/E

Two TSO/E commands were modified to support a retention period larger than 9999 days:

- ALLOCATE allows decimal numbers of 0 - 93000 for the RETPD keyword, instead of 0 - 9999. The message prompts IKJ56700A and IKJ56701I, translations of the message text, and the HELP panel associated with the ALLOCATE command are updated to reflect the change.
- ATTRIB allows decimal numbers of 0 - 93000 for the RETPD keyword on the ATTRIB command, instead of 0 - 9999. The message prompts IKJ56700A and IKJ56701I, translations of the message text, and the HELP panel associated with the ATTRIB command must be updated to reflect the change.

14.3.3 Providing updated volume space statistics

This improvement enhances the serviceability of your z/OS systems by providing richer statistics about volume space.

To implement this improvement, the volume space updated indicator (VLDVSUP) is turned On in the Active configuration. It is turned on when the volume is varied online or when SMS is called by CVAF for a change to the volume space information.

SMS is modified to issue LSPACE for volumes when the Storage Group volume list is being requested and VLDVSUP is On. As a consequence, when ISMF is started by the LISTSYS command, the following information data is displayed:

- Total capacity in MB
- Amount of free space in MB
- Largest free extent in MB for the entire volume

  If the volume is an EAV volume, the space data is returned for the entire volume and for the track managed space.

14.3.4 Support for data set space greater than 2 terabytes

This enhancement improves continuous availability and serviceability in your z/OS systems.

The main objective of this enhancement is to increase the data set space capacity. For more information, see 14.2.3, “Data set space limitations” on page 188. ISMF and DADSM are modified to accomplish this goal.

SMS changes

SMS implements a much higher data set size limit in DFSMS V1.13 by converting FIXED(31) fields to a FIXED(63) fields. The entire FIXED(63) field is not used because other MVS components that provide space information to SMS have much smaller limits. Therefore, effectively the new limit is around 1,000 times larger, and this amounts to roughly 2500 million of 3390 cylinders.

This data set size increase corrects several availability problems before DFSMS V1.13.

SMS used to have a smaller data set size limit. SMS got space information from other components such as MVS Allocation, IDCAMS, DFSMSdss, and DFSMShsm. These components can send in space quantities whose maximum values exceeded the capacity of fields defined in SMS control blocks. When these overflow conditions were encountered, SMS issued the IGD17351I error message and failed the allocation as shown in Example 14-2.

In the worst cases, the overflow was not detected. This problem was caused by the presumption that the space quantities required to cause that the overflow were far greater than would ever be seen in practical situations.

Example 14-2 igd17351I message

```
IGD17351I SPACE REQUESTED IS TOO LARGE. ALLOCATION FAILED FOR DATA SET dsn
```

DADSM changes

DADSM provides support that allows callers such as SMS to specify space in MB rather than KB.

The DADSM CREATE verb updates the JFCB interface. This update allows primary and secondary space unit of allocation in MB, with additional precision in KB and bytes. For compatibility reasons, there is a new bit DACIMBRQ to indicate that the unit is in MB for both the primary and secondary quantities. The primary and secondary quantities are passed in a 4-byte variable each. Additionally, any residual values (for primary and secondary) in KB and bytes are passed in two 2-byte variables.

The current format-1 DSCB keeps the secondary quantity in a 2-byte variable (DS1SCXTV) when the request is in bytes, KB, or MB. There is an additional compacted factor of 256 or 65536. It continues to do so without considering any precision in KB or bytes remaining from the secondary quantity.

Related messages

The following are some observations about messages due to a larger data set capacity in DFSMS V1.13:

- Message IGD17351I is still issued when the converted space in megabytes exceeds the limit imposed by other components.
- For DD or dynamic defined VSAM data set, SMS issues IGD17351I when the converted space in megabytes exceeds x'FFFFFF'. This process occurs because the space fields in the space field remain 3 bytes long.
- Message IGD17351I is also issued at the conclusion of JCL Space parameters LIKE= and DATA CLASS merge processing if the space computed is greater than '7FFFFFFF' MB. This size represents the maximum value that can be passed into DADSM.
- Existing message IGD17051I might be issued during the processing of SMS-managed non-VSAM data sets. This message signals a failure in DADSM when the requested space quantity exceeds the maximum allowed on a single volume. Because

SMS-managed data sets can be spread over multiple volumes during space constraint relief processing, SMS treats this particular DADSM failure as one that can be tried again. However, there are some SMS-managed allocations that cannot be spread over multiple volumes. In these cases, SMS issues the IGD17051I message as soon as the error is detected. It runs this process rather than go through a lengthy try again process that will eventually fail anyway.

14.3.5 Operator command to display SMS CSECT ID information

This enhancement addresses RAS in your z/OS systems.

For problem determination, you often need to obtain virtual storage address, offset, and maintenance level of a failing SMS CSECT at the key IGDZLLA load module. Currently the installation can use IBM tools such as AMBLIST, CODEZAP, or SMS IPCS module map formatter to obtain this module-related information. However, these tools usually provide only partial information and require the user to manually derive other information, which can be inefficient and error-prone. To facilitate this process, SMS provides a new keyword, SMSMOD(modname) on the existing D SMS command. This keyword allows the installation to display the module-related information of the SMS load module IGDZILLA. This module contains lots of SMS CSECTs at LLA. The command syntax is:

```
D SMS,SMSMOD(modname)
```

> **Tip:** Currently PDSE supports the command **`D SMS,{PDSE|PDSE1},MODULE(modname)`** to display PDSE or PDSE1 modules. TO avoid confusion, this new SMS command defines a new keyword SMSMOD for the user to specify SMS module name.

When this display command is entered with a valid module name (CSECT ID), SMS issues a new version of IGD002I message. This message is shown in Example 14-3 at ❶.

```
D SMS,SMSMOD(IGDVTSCR)
```

When this display command is entered with a module name that is not in the IGDZILLA load module, SMS issues a new version of IGD004I. This message is shown in Example 14-3 at ❷.

```
D SMS,SMSMOD(BADMOD)
```

Example 14-3 Display SMS output

```
            -D SMS,SMSMOD(IGDVTSCR)
             IGD002I DISPLAY SMS MODULE ON 192 ❶
             A zOSV1R13 SYSTEM
             MODULE   MODULE   OFFSET   PTF      COMPILED MODULE
             NAME     ADDRESS  IGDZILLA LEVEL    DATE     ID PARM
             -------- -------- -------- -------- -------- -----------------------
             IGDVTSCR 07A8EC08 00106C08 NONE     03/18/11 03/07/11HDZ1D10 K1D0986
...
...
            -D SMS,SMSMOD(BADMOD)
             IGD004I COMMAND REJECTED 212 ❷
             MODULE NAME BADMOD IS NOT FOUND IN SMS LOAD MODULE IGDZILLA
```

14.3.6 Miscellaneous RAS improvements

This section addresses other improvements that affect the RAS and performance of your z/OS systems.

Critical path performance trace

As the complexity of configuration management increases, additional problem prevention and optimization code must be integrated into the product to ensure optimized performance. The SMSDATA trace is enhanced to record the execution duration of each subroutine. This trace helps measure the code performance and pinpoint areas that still have room for performance improvement. This is an internal enhancement.

Loops optimization

Currently, some SMS subroutines run loops that process volumes and storage groups. These loops are enhanced and consolidated to improve performance. One of them is IGDOPST1, which runs storage group and volumes association processing.

Including the ACDS level in the output of D SMS command

Before DFSMS V1.13, the D SMS and D SMS,OPTIONS operator commands generated two different versions of the IGD002I message. Both versions of this message list information that is extracted from the active IGDSMSxx member. However, one piece of information that was not displayed is the level of the currently active ACDS configuration, namely the z/OS version and release.

This new feature enhances both variants of the DISPLAY SMS command to put out the ACDS level of the configuration and all currently displayed fields. The version number is obtained from the current active ACDS. There might be cases when the version number of the ACDS is not available to be printed out. If this error happens, the ACDS level is listed as UNAVAIL.

The format of the notice and the result of the command is shown in Example 14-4.

```
ACDS LEVEL = z/OS Vn.nn|UNAVAIL
```

Example 14-4 Checking the level of ACDS

```
-D SMS
 IGD002I 10:23:11 DISPLAY SMS 143
 SCDS = SYS1.SMS.MHLRES3.SCDS
 ACDS = SYS1.SMS.ACDS
 COMMDS = SYS1.SMS.COMMDS
 ACDS LEVEL = z/OS V1.13
 DINTERVAL = 150
 REVERIFY = NO
 ACSDEFAULTS = NO
```

Checking CDS for VSAM linear data set

The SMS control data sets (ACDS and SCDS) must be defined as VSAM linear data sets (LDSs). If it is not a VSAM LDS, an abend occurs during data-in-virtual (DIV) processing that demands the LDS data set type, and memory dumps are produced. This abend generally happens when the configuration is activated or saved by a SETSMS command, although other scenarios are possible. However, even with the abend message and memory dumps, there is no clear message that tells the user what happened.

This problem can be fixed by detecting whether the CDS is a VSAM LDS and by delivering clearer information:

- When activating a CDS during IPL, SMS issues a Catalog Search Interface for the SMS configuration data set to identify the type of data set. If the CDS is not a VSAM LDS, SMS issues a specific error message (IGD091I) as shown in Example 14-5. The action that is requested fails, and abends and memory dumps are eliminated.

  Example 14-5 IGD091I message

  ```
  IGD091I  CATALOG SEARCH INTERFACE FAILED FOR cds_type ('dsname'). ERROR RETURN
  CODE IS rc REASON CODE IS rsnc
  ```

- When a SETSMS command is submitted, a clearer message (IGD090) is produced if the CDS is not a VSAM LDS as shown in Example 14-6. The following SETSMS commands have this modification:
  - SETSMS ACDS(dsname)
  - SETSMS SCDS(dsname)
  - SETSMS COMMDS(dsname)
  - SETSMS SAVEACDS(dsname)
  - SETSMS SAVESCDS(dsname)
  - SETSMS COPYSCDS(scds_dsn, acds_dsn)

  Example 14-6 IGD090 message

  ```
  IGD090I  cds_type ('dsname') RECORD ORGANIZATION IS NOT VSAM LDS (LINEAR)
  ```

Changing CDS from NOREUSE to REUSE

Define the SMS CDS data sets as REUSE. Before DFSMS V1.13, SMS Health Check verified whether the CDS is defined as REUSE. If the CDS is defined as NOREUSE, SMS Health Check issues a message that provides users with information to alter the data set to REUSE manually. It does not alter the data set option automatically. If the system does not have IBM Health Checker activated, SMS does not check the REUSE/NOREUSE option for the CDS.

This improvement ensures that when the ACDS or COMMDS is activated, even without IBM Health Checker activated, SMS takes over the role of the Health Check. SMS verifies whether the CDS is defined with REUSE option. If the CDS detects NOREUSE by either SMS or IBM Health Checker, SMS attempts to convert the CDS defined as NOREUSE to REUSE. One of the following messages is issued depending on whether IBM Health Checker is activated.

If IBM Health Checker is activated on the system, the following messages are issued to both the IBM Health Checker message log and system console:

- If CATALOG ALTER from NOREUSE to REUSE failed for any reason:

  ```
  IGDH1011E CHECK(IBMSMS,SMS_CDS_REUSE_OPTION) detected cds_type ('dsname') not
  defined with the REUSE option.
  ```

  SMS attempted to alter the CDS to REUSE but failed. Catalog Return Code is rc Reason Code is rsnc IGG0CLxx

- If CATALOG ALTER from NOREUSE to REUSE succeeded:

  ```
  IGDH1012E CHECK(IBMSMS,SMS_CDS_REUSE_OPTION) detected cds_type ('dsname') not
  defined with the REUSE option.
  ```

  SMS successfully altered the CDS to REUSE.

Without IBM Health Checker activated on the system, the following messages are issued to system console:

- If CATALOG ALTER from NOREUSE to REUSE succeeded:

```
IGD093I cds_type ('dsname') NOT DEFINED WITH THE REUSE OPTION AND HAS BEEN
AUTOMATICALLY ALTERED TO REUSE
```

- If CATALOG ALTER from NOREUSE to REUSE failed:

```
IGD094I  CATALOG ERROR WHILE ALTERING cds_type ('dsname') FROM NOREUSE TO
REUSE.  RETURN CODE IS rc REASON CODE IS rsnc IGG0CLxx
```

15

Enhancements to other components

This chapter describes changes to other components and features of DFSMS V1.13. These include DADSM, the Health Checker, GDPS DS8K Synergy, compression, IEBCOPY, and ICKDSF.

This chapter includes these sections:

- DADSM enhancements
- Health Checker update
- GDPS/PPRC HyperSwap DS8K Synergy
- Compression and virtual constraint relief
- IEBCOPY performance and APF authorization
- Update to ICKDSF

15.1 DADSM enhancements

DFSMS V1.13. introduces changes to DADSM/CVAF:

- Volume contention
- Device availability
- Device summary
- IPCS

15.1.1 Reduction in volume contention

CVAF and EXCP device support exits have been changed to reduce volume contention. These changes fall into the following areas:

1. The macro CVAFFILT reads five tracks of DSCBs at a time to make read operations more effective. This totals 250 records (five tracks with 50 DSCBs each). This process improves DFSMShsm performance.
2. CVAF has been upgraded from writing 15 VTOC index records to 255 VIRs in one channel program. This is a substantial reduction in writes (17 times).
3. EXCP VTOC I/O in the IECDSCAN scan exit utility has been changed. It only uses extents in the Define Extent CCW that will actually be accessed by the channel program. Multiple allegiance support can work optimally if the precise read/write extents are given. The define extent in the channel program must also define exactly the extents that will be used in the IO operation. Only overlap in defined extents in the channel programs that consist of both read and write intent must queue up. I/O to change the VTOC requires an enqueue on SYSVTOC.
4. A CLCL instruction is used instead of TRT when processing VTOC index record maps. Using the CLCL instruction results in a 50% reduction for long strings (compared to using TRT), and is also better for shorter strings. CLCL instructions can process one single VIR, whereas a TRT requires multiple instructions to process a VIR. The VTOC function that searches for the first free or used bit in a VIR has been changed to use CLCL.
5. The macro CSVQUERY is now used to obtain addresses of load modules for DADSM, CVAF, and Device Support during NIP processing. This process replaces the previous LOAD/DELETE macro sequence (IEAVNP16 and IECVINIT). Overall this process reduces the time spent during an initial program load (IPL).

The overall benefit of these changes is improved performance in these areas:

- Reduced processor time in the user address spaces.
- Reduced device busy time and improved throughput for VTOC processing

15.1.2 Device support availability

DADSM and CVAF have been changed to improve availability in two areas: DADSM and DEVMAN.

1. DADSM and CVAF support dynamic refresh of their load modules (adds, deletes, and updates), reducing the need for an IPL. An exit routine, CSVDYLPA, has been added for doing appropriate updates to the SVC table, because any listening exits are also reestablished.

You can perform an update with an operator command. Use SETPROG with the LPA statement or the SET PROG=xx command to activate any updates (Example 15-1).

Example 15-1 Sample SETPROG to refresh two modules in LPA

```
SETPROG LPA,ADD,DSNAME(SYS1.LPALIB),MODNAME(IGGDADSM,IGC0013I),ADDALIAS
```

2. The ADDALIAS parameter initiates the ALIAS process.

DEVMAN code is changed so that the ASID for DEVMAN can be reused after a restart. This change removes the risk of this task using up all available address space identifiers (ASIDs) in the system.

15.1.3 Device support simplification

To meet the simplification objective, changes have been made to DADSM, CVAF, and Device Support, resulting in the following improvements:

1. Dynamic exits are supported for DADSM pre- and post- processing exits. The DADSM functions create, extend, scratch, partial release, and rename all have pre- and postprocessing routines (IGGPRE00 and IGGPOST0). You can obtain control through them before and after DADSM processing. With this support, dynamic exit services are used to define a dynamic pre-exit and post-exit, and associate them with the existing exits as exit routines. This configuration makes it possible to replace, add, or remove pre-processing or post- processing exits without needing to IPL.

   > **Remember:** The exit addresses for DADSM IGGPRE00_EXIT and IGGPOST0_EXIT remain set, so programs that use this interface are unaffected.

   No action is needed regarding upgrade. You can change the pre-exit and post-exit name if you forget to update and activate the PROGxx member with these new names. Example 15-2 shows how to connect an exit routine to an exit.

   Example 15-2 ADD exit routine to EXIT

   ```
   SETPROG EXIT,ADD,EXITNAME=IGGPRE00_EXIT,MODNAME=IGGPRE01
   ```

2. When command rejects are detected by device support, you can capture additional diagnostic data due to an enhancement in the DASD ERP component. This feature uses the VARY SMS ERRORINJ command and the IGWCRITP macro (internal interfaces). It allows the customer, assisted by IBM Support, to request a diagnostic symptom record that includes a memory dump. The diagnostic record shows information about the error and the environment, and is similar to current DASD ERP reports. To capture the error, a SLIP is set with the address of the command reject code and the job is rerun.
3. The AOM component trace has been changed so that trace records are kept in a large data space within the DEVMAN address space. Trace records originally were kept in a small storage buffer. Therefore, the relevant trace data will most likely be captured in a diagnostic memory dump, reducing the need for a trace output writer. AOM trace support also eliminates trace records for events that are no longer useful.
4. New device support for health check on tape libraries has been added to report on errors during IPL. This health check routine is designed to include more device support health checks in the future. The routine is named DMO_TAPE_LIBRARY_INIT_ERRORS. A report shows an explanation and suggests a resolution. For more information about this check, see *IBM Health Checker for z/OS User's Guide,* SA22-7994. Also see *MVS System*

Messages & Codes, Volume 4, SA22-7634, Appendix A. Example 15-3 shows messages that can be displayed in the report.

Example 15-3 Sample device support Health Check report

```
DMOH0101I CHECK(DMO_TAPE_LIBRARY_INIT_ERRORS) ran successfully and found no
exceptions.
DMOH0102I text..
DMOH0104E CHECK(DMO_TAPE_LIBRARY_INIT_ERRORS) determined that library device
initialization
errors occurred during IPL.
DMOH0105I The check is not applicable in the current environment because there
are no tape libraries defined.
```

5. Tape Library device support has been changed to detect when mount completion is signaled with an invalid attention. When the error is detected, device support records a LOGREC record that includes the following message:

   ```
   LIBRARY MOUNT COMPLETE WAS REPORTED WITH AN 80 ATTENTION FOR DEVICE XXXX.
   ATTENTION TYPE 85 IS EXPECTED. THE MOUNT WAS   SUCCESSFULLY COMPLETED BY ERROR
   RECOVERY. SHOULD THE ERROR PERSIST, CONTACT HARDWARE SUPPORT.
   ```

6. A tape interrupt has been changed to write a software record when a tape mount completes with a x80 'Attention interrupt' instead of a x85 'State Change Interrupt'. Besides writing an entry to LOGREC, the tape unit control block (UCB) is made available using a software input/output supervisor (IOS) interface. Previously during hardware outages, taking a generalized trace facility (GTF) trace might take a long time, extending the outage. This support provides the information needed to diagnose and solve any problems more quickly.
7. Module IGX00039 has been changed and IGX01039 has been deleted to simplify reliability, availability, and serviceability (RAS).
8. The DEVMAN CTRACE has been changed so that buffers are automatically flushed after the disconnect of an external writer. You no longer need a command for this function.

15.1.4 Shipping AOM and DMO IPCS in MIGLIB

AOM and DMO IPCS modules are changed so that they are shipped in MIGLIB.

15.2 Health Checker update

This section addresses a DFSMS update related to health checks, and also provides a summary of what is currently available for DFSMS and storage-related health checks.

15.2.1 The Health Checker function

The IBM Health Checker examines the health of selected system components at regular intervals. It monitors settings, looks for single points of failure in the setup, and issues warnings based on its findings. You can find the settings for the Health Checker in SYS1.PARMLIB member HZSPRMxx. To make changes there, use SDSF statements or the command **`F hzsproc`** for dynamic updates. You can activate, deactivate, or update health checks by using **`F hzsproc`**.

Example 15-4 lists examples of operating the health checker.

Example 15-4 Examples of operating Health Checker by command

```
S hzsproc /* Start health checker */
F hzsproc,RUN,CHECK=(ibmhsm,*) /* Run a group of health checks */
F hzsproc,RUN,CHECK=(ibmsms,sms_cds_reuse_option) /* Run individ. health check */
F hzsproc,UPDATE,CHECK=(ibmsms,sms_cds_reuse_option),severity=(med) /*Update */
```

The interval between checks is set by the INTERVAL parameter or by a policy. All health checks have a severity (LOW, MEDIUM, or HIGH) defined on the check. You can set this severity to suit installation needs. Currently, 500 health checks are available, but this section focuses only on the DFSMS related checks.

New DFSMS health check

Device support Health Check on tape libraries has been added to the z/OS Health Checker reporting on tape library errors that occur during IPL. This health check routine, DMO_TAPE_LIBRARY_INIT_ERRORS, was designed to include more device support health checks in the future. It is the only new DFSMS related Health Check added in DFSMS V1.13. The report shows explanations and suggested remedies. For more information about this check, see *IBM Health Checker for z/OS User's Guide,* SA22-7994. The following messages might be displayed in the report. For more information, see *MVS System Messages & Codes, Volume 4,* SA22-7634, Appendix A.

```
DMOH0101I CHECK(DMO_TAPE_LIBRARY_INIT_ERRORS) ran successfully and found no
exceptions.
DMOH0102I text.
DMOH0104E CHECK(DMO_TAPE_LIBRARY_INIT_ERRORS) determined that library device
initialization errors occurred during IPL.
DMOH0105I The check is not applicable in the current environment because there are
no tape libraries defined.
```

Overview of current DFSMS related health checks

The list of current DFSMS and storage-related health checks spans multiple areas. These areas include catalog, tape libraries, DFHSM, DFRMM, zFS, Systems Managed Buffering, DFSMS itself, VSAM, and RLS. The following list shows currently available health checks.

- CATALOG_IMBED_REPLICATE
- DMO_TAPE_LIBRARY_INIT_ERRORS.
- HSM_CDSB_BACKUP_COPIES
- HSM_CDSB_DASD_BACKUPS
- HSM_CDSB_VALID_BACKUPS
- PDSE_SMSPDSE1
- ZOSMIGV1R10_RMM_REJECTS_DEFINED
- ZOSMIGV1R10_RMM_VOL_REPLACE_LIM
- ZOSMIGV1R10_RMM_VRS_DELETED
- ZOSMIGV1R11_RMM_DUPLICATE_GDG
- ZOSMIGV1R11_RMM_REXX_STEM
- ZOSMIGV1R11_RMM_VRSEL_OLD
- SMB_NO_ZFS_SYSPLEX_AWARE

- ZOSMIGREC_SMB_RPC.
- SMS_CDS_REUSE_OPTION
- SMS_CDS_SEPARATE_VOLUMES.
- VSAM_INDEX_TRAP
- VSAMRLS_CFCACHE_MINIMUM_SIZE
- VSAMRLS_CFLS_FALSE_CONTENTION
- VSAMRLS_DIAG_CONTENTION
- VSAMRLS_QUIESCE_STATUS
- VSAMRLS_SHCDS_CONSISTENCY
- VSAMRLS_SHCDS_MINIMUM_SIZE
- VSAMRLS_SINGLE_POINT
- ZOSMIGV1R11_ZFS_INTERFACELEVEL
- ZOSMIGREC_ZFS_RM_MULTIFS
- ZOSMIGV1R11_ZFS_RM_MULTIFS
- ZOSMIGV1R13_ZFS_FILESYS

For more information about health checks, see *IBM Health Checker for z/OS: User's Guide,* SA22-7994.

SMS health check function from SDSF

To access the IBM health checker from the TSO SDSF primary menu, type **CK** in the command line or select **CK** from the displayed list (Figure 15-1). SDSF is a separately priced program.

```
HQX7780 -----------------  SDSF PRIMARY OPTION MENU  --------------------------
COMMAND INPUT ===> CK                                           SCROLL ===> CSR

DA    Active users                      INIT  Initiators
I     Input queue                       PR    Printers
O     Output queue                      PUN   Punches
H     Held output queue                 RDR   Readers
ST    Status of jobs                    LINE  Lines
                                        NODE  Nodes
LOG   System log                        SO    Spool offload
SR    System requests                   SP    Spool volumes
MAS   Members in the MAS                NS    Network servers
JC    Job classes                       NC    Network connections
SE    Scheduling environments
RES   WLM resources                     RM    Resource monitor
ENC   Enclaves                          CK    Health checker
PS    Processes
                                        ULOG  User session log
END   Exit SDSF
```

Figure 15-1 Accessing Health Checker with CK

If you want to filter by CheckOwner, enter **FILTER CHECKOWNER EQ IBMDMO**. Figure 15-2 shows the new health check on tape libraries on startup after IPL.

```
SDSF HEALTH CHECKER DISPLAY  (ALL)                     LINE 44-61 (489)
COMMAND INPUT ===>                                            SCROLL ===> CSR
PREFIX=ANDRE*  DEST=(ALL)  OWNER=*  SYSNAME=*
NP   NAME                             CheckOwner       State              Statu
S    DMO_TAPE_LIBRARY_INIT_ERRORS     IBMDMO           ACTIVE(ENABLED)    SUCCE
     GRS_AUTHQLVL_SETTING             IBMGRS           ACTIVE(ENABLED)    EXCEP
     GRS_CONVERT_RESERVES             IBMGRS           ACTIVE(DISABLED)   GLOBA

.......... more health checks ...
```

Figure 15-2 New health check on tape libraries on startup after IPL

In Figure 15-2, note the headings: Name, CheckOwner, State, and Status. You can also scroll to the right for more headings. The health checks are grouped as shown in the column CheckOwner. You can activate or run a health check with a command against the individual health check or group. You can verify at the far right side that this health check ran successfully. Select the check DMO_TAPE_LIBRARY_INT_ERRORS by typing **S** under selection **NP.** Figure 15-3 shows the verification results.

```
SDSF OUTPUT DISPLAY DMO_TAPE_LIBRARY_INIT_ERRORS  LINE 0      COLUMNS 02- 81
 COMMAND INPUT ===>                                          SCROLL ===> CSR
******************************** TOP OF DATA**********************************
CHECK(IBMDMO,DMO_TAPE_LIBRARY_INIT_ERRORS)
START TIME: 09/21/2011 12:42:11.448711
CHECK DATE: 20100128  CHECK SEVERITY: LOW

DMOH0101I CHECK(DMO_TAPE_LIBRARY_INIT_ERRORS) ran successfully and found
no exceptions

END TIME: 09/21/2011 12:42:11.449351  STATUS: SUCCESSFUL
```

Figure 15-3 Results of health check verification

To demonstrate a health check with exceptions, select SMS_CDS_SEPARATE_VOLUMES, as shown in Figure 15-4. In this case, an exception was found, explained, and a recommendation to improve the setup is offered.

```
CHECK(IBMSMS,SMS_CDS_SEPARATE_VOLUMES)
START TIME: 10/06/2011 13:50:46.469308
CHECK DATE: 20090303  CHECK SEVERITY: MEDIUM

* Medium Severity Exception *

IGDH1001E CHECK(IBMSMS,SMS_CDS_SEPARATE_VOLUMES) detected the
ACDS (SYS1.SMS.ACDS)
and COMMDS (SYS1.SMS.COMMDS)
allocated on the same volume.

  Explanation:  As a best practice, an ACDS/COMMDS must reside on a
    volume, accessible from all systems in the SMS complex. To ease
    recovery in case of failure, the ACDS should reside on a different
    volume than the COMMDS. Also, you should allocate a spare ACDS on a
    different volume. The control data set (ACDS or COMMDS) must reside
    on a volume that is not reserved by other systems for a long period
    of time because the control data set (ACDS or COMMDS) must be
    available to access for SMS processing to continue.
```

Figure 15-4 Health check with exceptions

15.2.2 Upgrade and coexistence considerations

No particular upgrade action is required. The IBM Health Checker is an established, standard product that adds new health checks automatically. The follow-up can be automated through your automation package, based on the severity settings you choose for your environment. For more information about receiving email alerts, see the Health Checker documentation at:

http://www.redbooks.ibm.com/redpapers/pdfs/redp4590.pdf

15.3 GDPS/PPRC HyperSwap DS8K Synergy

This chapter provides the latest updates on GDPS/PPRC HyperSwap DS8K Synergy. It addresses Packages 1 and 2.

15.3.1 Objectives

IBM has announced an improvement in the synergy between host software and the DS8K disk product. Improvements will be released in stages. At the time of writing, two packages have been announced.

Package 1 enables a GDPS policy split, allowing primary and secondary control unit failures to be treated individually as related to policy setting. This support came out with the DS8000 Licensed Code 5.1 and GDPS 3.7 SPE in 3.Q2010. z/OS supports DFSMS V1R9 and later.

Consideration: This support is for DS8700 and DS8800 only, and will not support earlier DS8K models.

Package 2 is the main topic in this document, but package 1 is also addressed for reference and background. Package 2 introduces an improved recovery of HyperSwap enabled configurations, including Summary Event Notification. GDPS/PPRC HyperSwap DS8K Synergy support is applicable for the GDPS/PPRC HM, GDPS/PPRC, GDPS/MzGM, and GDPS /MGM products.

This improvement is planned in stages. At the time of writing, two packages have been announced.

> **Restriction:** At this time, GDPS support for Summary Event Notification for PPRC Suspends (PPRCSUM) is available. However, it is restricted. Do not enable PPRCSUM in DEVSUPxx until the restriction is lifted. Review the GDPS PSP from time to time. When the restriction is lifted, the PSP includes information about any additional maintenance that might be required before PPRCSUM is enabled.

15.3.2 DS8K Synergy: Package 1 contents

Protection of the GDPS/PPRC environments is based on a freeze policy that is activated when a control unit or site failure occurs. Before the DS8K synergy support, this process was based on three alternative policy settings:

- FREEZE=GO or FREEZE=SWAP,GO

  Allows applications to continue after a failure. Customer accepts a potential data loss. An unplanned HyperSwap to the secondary devices might happen if HS is enabled, but, basically the event suspends the mirroring.

- FREEZE=STOP or FREEZE=SWAP,STOP.

  For customers who cannot accept loss of data. A failure on the primary control unit results in an unplanned HyperSwap, if enabled, making it possible for the applications to continue processing. A failure in the secondary site control unit causes a FREEZE=STOP scenario

- FREEZE=COND

  For customers who need both a GO policy in the event of a false freeze and a STOP policy in the event of a disaster. GDPS/PPRC traps message IEA491E to determine the reason for the failure.

Most GDPS FREEZES are false freezes, and a high percentage of these are triggered by failures in the secondary control unit.

The support introduced with the DS8K Synergy Package 1 provided an improved granular policy setting. The policy is now split up into two parts: Primary disk subsystem failure processing and PPRC suspend failure processing.

PRIMARYFAILURE parameter

PRIMARYFAILURE, a new parameter, handles the processing if a primary disk failure occurs. The parameter has three options:

- SWAP (HyperSwap)
- GO (FREEZE & GO)
- STOP(FREEZE & STOP)

A secondary parameter can be added to the policy. It is started if the primary operand decision cannot be run. This situation might happen if a HyperSwap cannot be done because HyperSwap is disabled. The secondary parameter can be STOP or GO. An example of a policy is SWAP,STOP. The messages IOS002A and IEA491E are captured during failures.

PPRCFAILURE

The new PPRCFAILURE parameter handles mirroring problems on the primary or secondary control unit. It separates primary write I/O failures from PPRC mirroring problems. Specifying PPRCFAILURE=COND (replaces FREEZE=COND) can reduce the number of so-called false freezes. The options for this parameter are.

- GO(FREEZE & GO)
- STOP(FREEZE & STOP)
- COND: If the secondary is determined to be the source of the problem, a FREEZE&GO is run. If not, then a FREEZE&STOP is run.

A new query capability has been enabled along with this new support. This query helps GDPS decide whether the failure is a false freeze event, or is a situation for disaster recovery and react accordingly.

For more information about these parameters, see the *GDPS Implementation and Customization Guide* that is shipped with the product.

15.3.3 Package 2: Improved DS8K Synergy

This release includes improved DS8K recovery in HyperSwap Enabled Environments and two new functions as part of the DS8K Synergy Package 2. The required microcode level is listed at the end of this section.

PPRC summary support

The improvement involves the Controller aggregating PPRC state and presenting this to the host as a summary for the logical control unit (LCU). This process allows host software to reduce system resources:

- Reduced Console services by reporting PPRC state: One or two IEA075I messages per LCU instead of one IEA494I message per device
- Reduced the number of I/Os needed to collect PPRC state: A few channel programs per LCU instead of two channel programs per device
- Reduced processor usage associated with writing SMF 22 Records

This function reduces the resources used when a PPRC freeze occurs in an environment that spans multiple logical control units. Instead of issuing one IEA494I message per device, this function suppresses those messages and replaces them with one or two IEA075I messages per LCU. Message IEA075I message summarizes the PPRC state of all devices in a logical control unit, when one or more devices in a PPRC relationship is suspended.

> **Tip:** Message IEA0494I continues to be presented at a device level for other types of PPRC state transitions, such as full duplex.

Example 15-5 shows a IEA075I message that indicates a PPRC suspended state,

Example 15-5 IEA075I message on PPRC suspended state

```
14.50.10 SYSTEM1 IEA075I PPRC SUMMARY,SSID=8106,
DEVICE NED=2107.941.IBM.75.0000000TW551.0600,
SUSPENDED=002,PPRC=002,TOTAL=008,
REASON=0A,SUSPENDED, FREEZE COMMAND
```

A new parameter was added in z/OS V1.13 to DEVSUPxx member in SYS1.PARMLIB: PPRCSUM (YES!NO). This parameter activates the support to replace IEA494I messages with IEA075I messages at the LCU level. It also summarizes the PPRC state on all devices in an LCU.

The default at z/OS V1.13 installation is NO, meaning that the IEA494I messages display per device in a PPRC suspend scenario. You can also activate this new support after IPL by using the following commands. They notify each LCU to enable or disable PPRC Summary.

```
F DEVMAN,ENABLE(PPRCSUM)
```

or

```
F DEVMAN,DISABLE(PPRCSUM)
```

GDPS has been updated for PPRC Summary Support and uses the summary features. Be careful to not enable PPRC Summary until you install the appropriate level of GDPS support.

Storage Controller Health Message

Storage Controller Health Message status, a new feature in the DS8700 and DS8800, sends alerts at the LCU level when resources are not available or under service. This process occurs for both primary and secondary control units. Also, a new IEA074I message is scheduled.

If the problem has sufficient severity, GDPS might, based the scenario, take action such as initiating a HyperSwap. One scenario might be server (cluster) offline due to service-alerted as shown in Example 15-6.

Example 15-6 IEA074I message due to a server offline

```
IEA074I STORAGE CONTROLLER HEALTH,MC=cc,TOKEN=dddd,SSID=xxxx,
DEVICE NED=tttt.mmm.ggg.pp.ssssssssssss.uuuu,
SERVER OFFLINE DUE TO SERVICE
```

Another alert through message IEA074I might be triggered by a service or initial microcode load (IML) ending, as shown in Example 15-7.

Example 15-7 IEA074I message triggered by service or IML ending

```
IEA074I STORAGE CONTROLLER HEALTH,MC=cc,TOKEN=dddd,SSID=xxxx,
DEVICE NED=tttt.mmm.ggg.pp.ssssssssssss.uuuu,
DUAL SERVERS ONLINE
```

Example 15-8 shows details about the individual fields in the IEA074I message.

Example 15-8 IEA074I message details

```
cc = Message Code
dddd = Unique value for this occurrence
Note: Message presented for each LSS will contain the same token.
tttt.mmm.ggg.pp.ssssssssssss = Storage Facility Image serial number
¢ tttt = machine type
¢ mmm = model
¢ ggg = manufacturer
¢ pp = plant of manufacturer
¢ ssssssssssss = box sequence number
xxxx = SSID for this subsystem
```

Compatibility and coexistence

The support introduced in the DS8K Synergy Package 2 is shipped with the following levels of microcode:

- IBM System Storage DS8700 - level 7.6.2.xx.xx (bundle version 76.20.xxx.xx), or later
- IBM System Storage DS8800 - level 7.6.2.xx.xx (bundle version 86.20.xxx.xx) or later

To enable the PPRC Summary Support and Storage Controller Health Message, z/OS V1.13 needs additional program temporary fixes (PTFs). See APARs OA37935 and OA377756.

For more information, see “Related publications” on page 337.

15.4 Compression and virtual constraint relief

Compression Services includes a change that supports virtual constraint relief in Virtual Storage Access Method (VSAM), record-level sharing (RLS), and sequential access method (SAM).

15.4.1 VSAM, RLS, and SAM

Compression Services has changed in DFSMS V1.13. You can put compression dictionaries used by Compression Services above the 2 GB bar in private storage, if available. If storage is not available, the dictionaries are placed under the bar but over the 16 MB line. This process applies for VSAM, RLS, and SAM.

With RLS users using more compressed data sets, there is a risk that RLS might run out of virtual storage. This change reduces that risk.

15.4.2 SAM access method only

SAM allows shadow buffers above the bar. The interfaces to the compress and extract functions are changed so that they can specify 64-bit addresses for input and output.

> **Consideration:** This change requires users to make changes even if they operate above the bar.

SAM data sets are expected to use more virtual storage, so this change is positioning for the future.

15.5 IEBCOPY performance and APF authorization

This release addressed IEBCOPY performance and updated the product. It also removed APF authorization to make usage more flexible, especially for other callers.

15.5.1 APF authorization removal

Starting with DFSMS V1.13, IEBCOPY no longer needs to be authorized program facility (APF)-authorized. Before V1.13, this configuration was a requirement both for users and for callers using IEBCOPY as a subroutine. An example is SME binder and assembler.

You can remove the need to use a Program Controlled Interrupt (PCI), which is a performance feature from earlier releases. By converting from EXCP/VR to EXCP, you can remove the need for APF authorization of the IEBCOPY module and for the calling programs.

You can now store IEBCOPY and callers in a non-APF-authorized library.

15.5.2 IEBCOPY improved performance

Before DFSMS V1.13, IEBCOPY used 24 bit buffer processing in channel programs. In this release, IEBCOPY performance has been improved by a change in the channel programs used. IEBCOPY now uses track channel command words (CCWs) to read an entire track at a time (input output buffers (IOBs) and CCWs remain in 24 bit storage).

A minor RAS improvement enhances the IEBCOPY ESTAE routine to capture a better memory dump of storage before being released.

15.5.3 Upgrading

When upgrading to DFSMS V1.13, you can use the backup version, IEBCOPYO, if you discover a critical problem with the new version. This version still has the undocumented alias name of IEBDSCPY. The name applies to IEBCOPYO only, and will be removed in future releases.

Be careful not to switch these modules, or names, because maintenance is targeted to both the old and the new module. Changes can complicate maintenance.

> **Tip:** IEBCOPY is still delivered as APF-authorized, and seems to run faster with the authorization.

15.6 Update to ICKDSF

SUBCHSET, subchset_identifier is a new parameter added to ICKDSF. Specifying SUBCHSET always requires the unit parameter. The parameter is designed to point to a subchannel set other than the normal default (0). The subchannel set must be specified as a decimal character, as shown in Example 15-9.

Example 15-9 Initializing a volume by using subchannel set

```
INIT UNITADDRESS(uuuu) VERIFY(*NONE*) VOLID(VOL123) -
SUBCHSET(1)
```

A

Code samples DFSMSoam V1.13

This appendix contains sample code to analyze object access method (OAM) System Management Facilities (SMF) records (Type 85). This code can be useful when implementing OAM from DFSMS V1.13 to verify the processing. You do not need to verify the processing, but these code samples might be useful to review the activity and performance during OAM activity.

This appendix includes these sections:

- A.1, “Listing of OAM functions and SMF record subtypes” on page 214 lists the OAM functions and their corresponding SMF record type 85 subtypes.
- A.2, “Running the SMF analysis programs” on page 215 describes routines that view the contents of OAM SMF records along with JCL to run the examples.
- A.3, “Building the SMF85 programs” on page 262 details the code for the sample programs and how to set them up. After a program is set up, this process does not have to be done again for that program. All processing can be done by using the Part 1 examples.

The code samples are available for download. See Appendix B, “Additional material” on page 335 for instructions.

A.1 Listing of OAM functions and SMF record subtypes

The OAM functions and the corresponding SMF record type 85 subtypes are listed in Table A-1.

Table A-1 SMF record to OAM function cross-reference

SMF Type 85 subtype and functions that generate the data	SMF data listing program
1 OSREQ ACCESS 2 OSREQ STORE 3 OSREQ RETRIEVE 4 OSREQ QUERY 5 OSREQ CHANGE 6 OSREQ DELETE 7 OSREQ UNACCESS 8 OSREQ STOREBEG 9 OSREQ STOREPRT 10 OSREQ STOREEND	SMF85TA
32 OSMC STORAGE GROUP PROCESSING 33 OSMC DASD SPACE MANAGEMENT PROCESSING 34 OSMC OPTICAL DISK RECOVERY UTILITY 35 OSMC MOVE VOLUME (MOVEVOL) UTILITY	SMF85TH
36 OSMC SINGLE OBJECT RECOVERY UTILITY	SMF85TQ
37 OSMC LIBRARY SPACE MANAGEMENT UTILITY	Not provided because no OPTICAL devices are available for testing
38 OSMC SINGLE OBJECT RECALL UTILITY	SMF85TO
39 OSMC IMMEDIATE BACKUP COPY	SMF85TI
40 OSMC COMMAND RECYCLE	SMF85TJ
64 LCS OPTICAL DRIVE VARY ONLINE 65 LCS OPTICAL DRIVE VARY OFFLINE 66 LCS OPTICAL LIBRARY VARY ONLINE 67 LCS OPTICAL LIBRARY VARY OFFLINE	Not provided because no OPTICAL devices are available for testing
68 LCS OPTICAL CARTRIDGE ENTRY 69 LCS OPTICAL CARTRIDGE EJECT 70 LCS OPTICAL CARTRIDGE LABEL 71 LCS OPTICAL VOLUME AUDIT 72 LCS OPTICAL VOLUME MOUNT 73 LCS OPTICAL VOLUME DEMOUNT	Not provided because no OPTICAL devices are available for testing
74 LCS OPTICAL WRITE REQUEST 75 LCS OPTICAL READ REQUEST 76 LCS OPTICAL (LOGICAL) DELETE REQUEST 77 LCS OPTICAL (PHYSICAL) DELETE REQUEST	Not provided because no OPTICAL devices available for testing
78 LCS TAPE WRITE REQUEST 79 LCS TAPE READ REQUEST 88 LCS TAPE LOGICAL DELETE REQUEST	SMF85TW
80 LCS TAPE LIBRARY VARY ONLINE 81 LCS TAPE LIBRARY VARY OFFLINE	SMF85TR

SMF Type 85 subtype and functions that generate the data	SMF data listing program
82 LCS TAPE LIBRARY VOLUME ENTRY 83 LCS TAPE LIBRARY VOLUME EJECT 84 LCS TAPE LIBRARY VOLUME AUDIT 85 LCS TAPE LIBRARY VOLUME MOUNT 86 LCS TAPE LIBRARY VOLUME DEMOUNT	SMF85TU
87 OAM OWNED TAPE VOLUME DEMOUNT	SMF85TS
90 LCS File System Write Request 91 LCS File System Read Request 92 LCS File System Physical Delete Request 93 LCS File System Physical Delete Request for uncommitted Store cleanup	SMF85TP

A.2 Running the SMF analysis programs

SMF data can be analyzed to validate system operation if the appropriate records are being generated by SMF. This configuration must be enabled in PARMLIB member SMFPRM00 by requesting that record type 85 be collected.

SMF is a system function, so parameters in SYS1.PARMLIB member SMFPRMxx need to be examined to check that record type 85 is being collected. Most installations suppress some records.

Example A-1 shows a statement that would not exclude type 85 records, while excluding several other records. All other types would be included.

Example A-1 SMFPRMxx example

```
SYS(NOTYPE(4,5,16:20,34,35,40,60,62,64,67:69,99),
```

The following sections detail programs that analyze the SMF type 85 records. Job control language (JCL) for running each program is described in the documentation for each program. Sample JCL is also provided in the ITSO FTP repository to run all the programs after they are built. For more information, see A.3.1.6, “SMF85 Program sample job to run all programs” on page 266.

A.2.1 Program SMF85TA SMF record type 85 subtypes 1-10

OAM writes SMF Record type 85 subtypes 1/2/3/6/7/8/9/10 to document the OSREQ macros used in batch jobs and TSO OSREQ command activity that starts the OSREQ macro. The functions that generate the different SMF type 85 subtype records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TA is included that scans the SMF type 85 subtypes 1-10 records and formats the records of activity. The program itself and how to construct it is documented in A.3.2, “SMF Record type 85 subtype 1-10 data display program SMF85TA” on page 266.

Figure A-1 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

If you do not want output from all the types that the program can process, change the SMF record and subtype selection statement to include only those you want. For example, change OUTDD(OUTDD,TYPE(85(1:10))) to OUTDD(OUTDD,TYPE(85(2,3,4,5,6))) to exclude subtypes 1 and 7-10.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=&SYSUID
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1  <--- CHANGE AS NEEDED
//*         OR REPLACE BY DATA SETS WITH PREVIOUSLY EXTRACTED DATA
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(01:10)))
/*
//  EXEC PGM=SMF85TA
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TA.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-1 SMF85TA SMF data listing JCL

The Flags field, as shown in the output, reflects the flag bits as mapped by the flag fields in the SMF records corresponding to the particular subtype. SMF record type 85 subtype 1 is used to map subtype records 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10.

Attention: Do not use the contents of the CBRSMF macro as presented here. When you assemble the program, or want to refer to the macro, it can be found in SYS1.MACLIB.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. In this case you need fields prefixed with ST1 for the bulk of the fields, then ST2 through ST10 for differences from ST1.

Figure A-2 shows a customized version of the commands from example CBRSAMIV from SYS1.PARMLIB.

```
//STEP1    EXEC PGM=IKJEFT01,REGION=0M
//STEPLIB  DD DSN=DB9D9.SDSNEXIT,DISP=SHR
//         DD DSN=DB9D9.SDSNLOAD,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSTSPRT DD   SYSOUT=*
//SYSTSIN  DD   *
 OSREQ STORE    ANDRECS.SG3.OAMTES9 MHLRES2.SMF85TP.PDS.OAMTES9 -
 LENGTH(000035456)
 OSREQ QUERY    ANDRECS.SG3.OAMTES9 MHLRES2.SMF85TP.PDS.OAMTES9
 LISTCAT ENTRIES('ANDRECS.SG3.OAMTES9') ALL
 OSREQ CHANGE   ANDRECS.SG3.OAMTES9 MHLRES2.SMF85TP.PDS.OAMTES9 RP(1)
 OSREQ QUERY    ANDRECS.SG3.OAMTES9 MHLRES2.SMF85TP.PDS.OAMTES9
 OSREQ RETRIEVE ANDRECS.SG3.OAMTES9 MHLRES2.SMF85TP.PDS.OAMTES9 -
 COMPARE VIEW(PRIMARY)
 OSREQ DELETE   ANDRECS.SG3.OAMTES9 MHLRES2.SMF85TP.PDS.OAMTES9
/*
```

Figure A-2 Example job to demonstrate SMF data records from OSREQ

When the OSREQ sample job runs, it generates SMF records. Example A-2 shows selected lines of output from running program SMF85TA. Data that is commented on is highlighted in bold and with a larger font size.

When the OSREQ STORE is processed, the object has a TOKEN assigned to it. This TOKEN is used for activities that relate to the object as shown on the line `TTOK/TOK`, in this case *D6E2D4C97F5DDA50*. The OSREQ STORE at 2011286/21:26:09.036 received a non-zero return code and reason code *00000004/04020480*. This reason code is advisory, and indicates that this is the first use of Collection `ANDRECS.SG3.OAMTES9`.

When the OSREQ STORE is processed, the object has an instance identifier assigned to it as shown on the line `STOUT/OLRD/NLRD/INST`, in this case *06070E0B*. This instance is shown in the SMF subtype 90-93 records. An example of this instance is shown in A.2.7, “Program SMF85TP SMF record type 85 subtype 90-93” on page 246. It is a disk sublevel 2 file system store, as opposed to a disk sublevel 1 DB2 store.

The full set of SMF records relating to the OSREQ STORE example is shown in Example A-2 to illustrate what happened. After initial observation of the process, you can reduce the output by not processing subtypes 1 (ACCESS) and 7(UNACCESS).

Example A-2 SMF85TA output

```
SMF TYPE 85 SUBTYPE 1-10 RECORDS
SMF85TA  SMFDTE/TME:      2011286/21:26:08.084
STYPE/FUNC:               1/OSREQ ACCESS
COLN/OBJN:                                                        /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000000 /00000000 /00000000
STOK/RC2:                                /00000000
STOUT/OLRD/NLRD/INST:     00000000/         /         /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:09.036
STYPE/FUNC:               2/OSREQ STORE
COLN/OBJN:                ANDRECS.SG3.OAMTES9                     /MHLRES2.SMF85TP.PDS.OAMTES9
```

```
SGN/SCN/MCN/LEN/:         SG3     /SG3     /MCOBDASD/00000035456
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000004 /04020480 /80110000
STOK/RC2:                                /00000004
STOUT/OLRD/NLRD/INST:     00000000/          /          /06070E0B

SMF85TA  SMFDTE/TME:      2011286/21:26:09.036
STYPE/FUNC:               7/OSREQ UNACCESS
COLN/OBJN:                                                       /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000000 /00000000 /00000000
STOK/RC2:                                /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:10.038
STYPE/FUNC:               1/OSREQ ACCESS
COLN/OBJN:                                                       /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000000 /00000000 /00000000
STOK/RC2:                                /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:10.038
STYPE/FUNC:               4/OSREQ QUERY
COLN/OBJN:                ANDRECS.SG3.OAMTES9                    /MHLRES2.SMF85TP.PDS.OAMTES9
SGN/SCN/MCN/LEN/:         SG3     /SG3     /MCOBDASD/00000000001
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000000 /00000000 /00000000
STOK/RC2:                                /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:10.042
STYPE/FUNC:               7/OSREQ UNACCESS
COLN/OBJN:                                                       /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000000 /00000000 /00000000
STOK/RC2:                                /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:11.045
STYPE/FUNC:               1/OSREQ ACCESS
COLN/OBJN:                                                       /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000000 /00000000 /00000000
STOK/RC2:                                /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:11.045
STYPE/FUNC:               5/OSREQ CHANGE
COLN/OBJN:                ANDRECS.SG3.OAMTES9                    /MHLRES2.SMF85TP.PDS.OAMTES9
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:             /  /00000000 /00000000 /00000000
STOK/RC2:                                /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000
```

```
SMF85TA  SMFDTE/TME:      2011286/21:26:11.045
STYPE/FUNC:               7/OSREQ UNACCESS
COLN/OBJN:                                                              /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 0000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:           /  /00000000 /00000000 /00000000
STOK/RC2:                               /00000000
STOUT/OLRD/NLRD/INST:     00000000/         /         /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:12.047
STYPE/FUNC:               1/OSREQ ACCESS
COLN/OBJN:                                                              /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 0000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:           /  /00000000 /00000000 /00000000
STOK/RC2:                               /00000000
STOUT/OLRD/NLRD/INST:     00000000/         /         /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:12.047
STYPE/FUNC:               4/OSREQ QUERY
COLN/OBJN:                ANDRECS.SG3.OAMTES9                          /MHLRES2.SMF85TP.PDS.OAMTES9
SGN/SCN/MCN/LEN/:         SG3     /SG3     /MCOBDASD/00000000001
TTOK/TOK:                 0000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:           /  /00000000 /00000000 /00000000
STOK/RC2:                               /00000000
STOUT/OLRD/NLRD/INST:     00000000/         /         /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:12.051
STYPE/FUNC:               7/OSREQ UNACCESS
COLN/OBJN:                                                              /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 0000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:           /  /00000000 /00000000 /00000000
STOK/RC2:                               /00000000
STOUT/OLRD/NLRD/INST:     00000000/         /         /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:14.052
STYPE/FUNC:               1/OSREQ ACCESS
COLN/OBJN:                                                              /
SGN/SCN/MCN/LEN/:                 /        /        /00000000000
TTOK/TOK:                 0000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:           /  /00000000 /00000000 /00000000
STOK/RC2:                               /00000000
STOUT/OLRD/NLRD/INST:     00000000/         /         /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:14.053
STYPE/FUNC:               4/OSREQ QUERY
COLN/OBJN:                ANDRECS.SG3.OAMTES9                          /MHLRES2.SMF85TP.PDS.OAMTES9
SGN/SCN/MCN/LEN/:         SG3     /SG3     /MCOBDASD/00000000001
TTOK/TOK:                 0000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:           /  /00000000 /00000000 /00000000
STOK/RC2:                               /00000000
STOUT/OLRD/NLRD/INST:     00000000/         /         /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:14.055
STYPE/FUNC:               3/OSREQ RETRIEVE
COLN/OBJN:                ANDRECS.SG3.OAMTES9                          /MHLRES2.SMF85TP.PDS.OAMTES9
SGN/SCN/MCN/LEN/:         SG3     /        /        /00000035456
TTOK/TOK:                 0000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:       ******/  /00000000 /00000000 /80010000
```

```
STOK/RC2:                         /00000000
STOUT/OLRD/NLRD/INST:     00000000/0001-01-01/2011-10-13/06070E0B

SMF85TA  SMFDTE/TME:      2011286/21:26:14.056
STYPE/FUNC:               7/OSREQ UNACCESS
COLN/OBJN:                                                 /
SGN/SCN/MCN/LEN/:              /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:          /  /00000000 /00000000 /00000000
STOK/RC2:                         /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:16.057
STYPE/FUNC:               1/OSREQ ACCESS
COLN/OBJN:                                                 /
SGN/SCN/MCN/LEN/:              /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:          /  /00000000 /00000000 /00000000
STOK/RC2:                         /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:16.058
STYPE/FUNC:               6/OSREQ DELETE
COLN/OBJN:                ANDRECS.SG3.OAMTES9                      /MHLRES2.SMF85TP.PDS.OAMTES9
SGN/SCN/MCN/LEN/:         SG3  /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:          /  /00000008 /40060202 /00000000
STOK/RC2:                         /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000

SMF85TA  SMFDTE/TME:      2011286/21:26:16.058
STYPE/FUNC:               7/OSREQ UNACCESS
COLN/OBJN:                                                 /
SGN/SCN/MCN/LEN/:              /        /        /00000000000
TTOK/TOK:                 00000000000000000000000000000000/D6E2D4C97F5DDA50
VSN/VMT/RC/RS/FLGS:          /  /00000000 /00000000 /00000000
STOK/RC2:                         /00000000
STOUT/OLRD/NLRD/INST:     00000000/          /          /00000000
```

Figure A-3 through Figure A-8 on page 226 show the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 1-10.

```
ST1      DSECT                   SUBTYPES 1 - 10               @P7C
ST1COLN  DS    CL44' '           COLLECTION NAME
ST1OBJN  DS    CL44' '           OBJECT NAME
ST1SGN   DS    CL8' '            STORAGE GROUP NAME
ST1SCN   DS    CL8' '            STORAGE CLASS NAME
ST1MCN   DS    CL8' '            MANAGEMENT CLASS NAME
ST1OFF   DS    BL4'0'            OFFSET FOR PARTIAL OBJECT
*                                RETRIEVE (SUBTYPE 3), ZERO FOR
*                                ALL OTHERS.
ST1LEN   DS    BL4'0'            LENGTH,
*                                SUBTYPE 1 - UNUSED
*                                SUBTYPE 2 - LENGTH OF OBJECT STORED
*                                SUBTYPE 3 - NUMBER OF BYTES RETRIEVED
*                                SUBTYPE 4 - NUMBER OF QEL ELEMENTS
*                                            RETURNED.
*                                SUBTYPE 5 - UNUSED
*                                SUBTYPE 6 - LENGTH OF OBJECT DELETED
*                                SUBTYPE 7 - UNUSED
*                                SUBTYPE 8 - LENGTH OF OBJECT STORED
*                                SUBTYPE 9 - LENGTH OF OBJECT PART
*                                SUBTYPE10 - LENGTH OF OBJECT STORED
ST1TTOK  DS    CL16' '           OSREQ TRACKING TOKEN, SUPPLIED
*                                WITH TTOKEN KEYWORD ON OSREQ
*                                MACRO
ST1TOK   DS    CL8' '            OSREQ ACCESS TOKEN
ST1VSN   DS    CL6' '            VOLUME SERIAL NUMBER
ST1VMT   DS    CL2' '            VOLUME MEDIA TYPE
ST1RC    DS    BL4'0'            OSREQ RETURN CODE, IN REGISTER 15
*                                FOLLOWING OSREQ MACRO
ST1RS    DS    BL4'0'            OSREQ REASON CODE, IN REGISTER 15
*                                FOLLOWING OSREQ MACRO
ST1FLGS  DS    BL4'0'            PROCESSING FLAGS. MEANING
*                                DEPENDENT ON RECORD SUBTYPE.
ST1STOK  DS    CL16              OSREQ STOKEN.  Valid for subtypes
*                                8, 9, and 10.                 @L7A
ST1RC2   DS    BL4'0'            OSREQ Return Code 2.  Valid for
*                                subtypes 2, 3, and 10         @L7A
ST1STOUT DS    BL4'0'            STIMEOUT in seconds   SUB=8     @L8A
ST1OLRD  DS    CL10' '           Last refer date: OLD  SUB=3,5   @L8A
ST1NLRD  DS    CL10' '           Last refer date: NEW  SUB=3,5   @L8A
ST1INST  DS    BL4'0'            Instance ID. For SUB 2,3,6,10   @L9A
*                                                              @L7A
***********************************************************************
*                                                                     *
```

Figure A-3 SMF record type 85 subtype 1-10 from CBRSMF macro (1 of 6)

```
*                                                                      *
* SUBTYPE 1 - OSREQ ACCESS FLAGS                                       *
*                                                                      *
************************************************************************
ST1FLGS0 EQU   X'80'            When on,IADDRESS was specified on
*                               the OSREQ request                @L9A
************************************************************************
*                                                                      *
* SUBTYPE 2 - OSREQ STORE FLAGS                                        *
*                                                                      *
************************************************************************
ST2FLGS0 EQU   X'80'            OBJECT STORE TO Disk           @L9C
ST2FLGS1 EQU   X'40'            OBJECT STORE TO OPTICAL
ST2FLGS2 EQU   X'20'            OBJECT STORE TO TAPE
ST2FLGS3 EQU   X'10'            UNUSED
ST2FLGS4 EQU   X'08'            UNUSED
ST2FLGS5 EQU   X'04'            WHEN ON, THE OSREQ STORE
*                               REQUEST RESULTED IN THE MOUNTING
*                               OF A SHELF-RESIDENT REMOVABLE
*                               MEDIA VOLUME (TAPE OR OPTICAL)
*                               BY A HUMAN OPERATOR. ONLY VALID
*                               IF BIT 1 OR 2 IS ON.
ST2FLGS6 EQU   X'02'            WHEN ON, THE OSREQ STORE
*                               REQUEST RESULTED IN THE MOUNTING
*                               OF A LIBRARY-RESIDENT REMOVABLE
*                               MEDIA VOLUME (TAPE OR OPTICAL)
*                               BY A HUMAN OPERATOR. ONLY VALID
*                               IF BIT 1 OR 2 IS ON.
ST2FLGS7 EQU   X'01'            WHEN ON, THE OSREQ STORE
*                               REQUEST WAS SATISFIED USING     @P5C
*                               AN ALREADY MOUNTED REMOVEABLE
*                               MEDIA VOLUME (TAPE OR OPTICAL).
*                               ONLY VALID
*                               IF BIT 1 OR 2 IS ON.
ST2FLGS8 EQU    X'80'           WHEN ON, AN IMMEDIATE BACKUP COPY
*                               WAS SCHEDULED FOR THIS OBJECT.  @L5A
ST2FLGS9 EQU    X'40'           WHEN ON, THE OBJECT IS STORED
*                               TO LOB STORAGE STRUCTURE        @L5A
ST2FLGS10 EQU   X'20'           WHEN ON, OBJECT STORED ON SUBLEVEL 1
*                               VOLUME.                         @L6A
ST2FLGS11 EQU   X'10'           WHEN ON, OBJECT STORED ON SUBLEVEL 2
*                               VOLUME.                         @L6A
ST2FLGS12 EQU   X'08'           UNUSED                          @L8A
ST2FLGS13 EQU   X'04'           DELHOLD=HOLD                    @L8A
ST2FLGS14 EQU   X'02'           Retention-protected             @L8A
ST2FLGS15 EQU   X'01'           Deletion-protected              @L8A
ST2FLGS16 EQU   X'80'           Event-based-retention           @L8A
```

Figure A-4 SMF record type 85 subtype 1 -10 from CBRSMF macro (2 of 6)

```
*********************************************************************
*                                                                   *
* SUBTYPE 3 - OSREQ RETRIEVE FLAGS                                  *
*                                                                   *
*********************************************************************
ST3FLGS0 EQU    X'80'           WHEN ON, PRIMARY COPY OF OBJECT
*                               RETRIEVED FROM Disk.           @L9C
ST3FLGS1 EQU    X'40'           WHEN ON, PRIMARY COPY OF OBJECT
*                               RETRIEVED FROM OPTICAL.
ST3FLGS2 EQU    X'20'           WHEN ON, PRIMARY COPY OF OBJECT
*                               RETRIEVED FROM TAPE.
ST3FLGS3 EQU    X'10'           WHEN ON, EITHER THE FIRST OR THE
*                               SECOND BACKUP COPY OF THE OBJECT WAS
*                               RETRIEVED FROM OPTICAL AS RESULT OF
*                               VIEW=BACKUP OR VIEW=BACKUP2 BEING
*                               SPECIFIED ON THE OSREQ MACRO.
*                               REFER TO BIT 10 TO INDICATE WHICH
*                               BACKUP COPY WAS RETRIEVED.     @L2C
ST3FLGS4 EQU    X'08'           WHEN ON, EITHER THE FIRST OR THE
*                               SECOND BACKUP COPY OF THE OBJECT WAS
*                               RETRIEVED FROM TAPE AS RESULT OF
*                               VIEW=BACKUP OR VIEW=BACKUP2 BEING
*                               SPECIFIED ON THE OSREQ MACRO.
*                               REFER TO BIT 10 TO INDICATE WHICH
*                               BACKUP COPY WAS RETRIEVED.     @P1C
ST3FLGS5 EQU    X'04'           WHEN ON, EITHER THE FIRST OR THE
*                               SECOND BACKUP COPY OF THE OBJECT WAS
*                               RETRIEVED FROM OPTICAL AS A RESULT
*                               OF THE PRIMARY COPY OF THE OBJECT
*                               BEING UNAVAILABLE AND THE AUTOMATIC
*                               ACCESS TO BACKUP BEING ACTIVE. @P9C
*                               REFER TO BIT 10 TO INDICATE WHICH
*                               BACKUP COPY WAS RETRIEVED.     @P1C
ST3FLGS6 EQU    X'02'           WHEN ON, EITHER THE FIRST OR THE
*                               SECOND BACKUP COPY OF THE OBJECT WAS
*                               RETRIEVED FROM TAPE AS A RESULT
*                               OF THE PRIMARY COPY OF THE OBJECT
*                               BEING UNAVAILABLE AND THE AUTOMATIC
*                               ACCESS TO BACKUP BEING ACTIVE. @P9C
*                               REFER TO BIT 10 TO INDICATE WHICH
*                               BACKUP COPY WAS RETRIEVED.     @P1C
ST3FLGS7 EQU    X'01'           WHEN ON, THE OSREQ RETRIEVE
*                               REQUEST RESULTED IN THE MOUNTING
*                               OF A SHELF-RESIDENT REMOVABLE
*                               MEDIA VOLUME (TAPE OR OPTICAL)
*                               BY A HUMAN OPERATOR. ONLY VALID
*                               IF BIT 1, 2, 3, 4, 5 OR 6 IS ON. @PAC
```

Figure A-5 SMF record type 85 subtype 1-10 from CBRSMF macro (3 of 6)

```
ST3FLGS8 EQU    X'80'            WHEN ON, THE OSREQ RETRIEVE
*                                REQUEST RESULTED IN THE MOUNTING
*                                OF A LIBRARY-RESIDENT REMOVABLE
*                                MEDIA VOLUME (TAPE OR OPTICAL)
*                                BY A HUMAN OPERATOR. ONLY VALID
*                                IF BIT 1, 2, 3, 4, 5 OR 6 IS ON. @PAC
ST3FLGS9 EQU    X'40'            WHEN ON, THE OSREQ RETRIEVE
*                                REQUEST WAS SATISFIED USING      @P5C
*                                AN ALREADY MOUNTED REMOVEABLE
*                                MEDIA VOLUME (TAPE OR OPTICAL).
*                                ONLY VALID
*                                IF BIT 1, 2, 3, 4, 5 OR 6 IS ON. @PAC
ST3FLGS10 EQU    X'20'           WHEN ON, THE SECOND BACKUP COPY OF
*                                THE OBJECT WAS RETRIEVED         @L4C
ST3FLGS11 EQU    X'10'           WHEN ON, A RECALL WAS SCHEDULED
*                                FOR THIS OBJECT                  @L4A
ST3FLGS12 EQU    X'08'           WHEN ON, A RECALL WAS EXPLICITLY @P5C
*                                SPECIFIED ON THE OSREQ RETRIEVE
*                                REQUEST                          @L4A
ST3FLGS13 EQU    X'04'           WHEN ON, THE PRIMARY COPY OF
*                                THE OBJECT WAS RETRIEVED FROM
*                                A LOB STORAGE STRUCTURE          @L5A
ST3FLGS14 EQU    X'02'           WHEN ON, OBJECT RETR. FROM SUBLEVEL 1
*                                VOLUME.                          @L6A
ST3FLGS15 EQU    X'01'           WHEN ON, OBJECT RETR. FROM SUBLEVEL 2
*                                VOLUME.                          @L6A
*
***********************************************************************
*                                                                     *
* SUBTYPE 4 - OSREQ QUERY FLAGS                                       *
*                                                                     *
***********************************************************************
ST4FLGS0 EQU    X'80'            WHEN ON, the QUERY BACKUP OPTION has
*                                been disabled by specifying QB=N in
*                                the IEFSSNxx PARMLIB member.
*                                When OFF, the QUERY BACKUP OPTION is
*                                enabled, either by default or by
*                                specifying QB=Y in the IEFSSNxx
*                                PARMLIB member.                  @L7A

***********************************************************************
```

Figure A-6 SMF record type 85 subtype 1-10 from CBRSMF macro (4 of 6)

```
*                                                                     *
* SUBTYPE 5 - OSREQ CHANGE FLAGS                                      *
*                                                                     *
***********************************************************************
ST5FLG0  EQU    X'80'           WHEN ON, MANAGEMENT CLASS
*                               SPECIFIED ON OSREQ CHANGE.
ST5FLG1  EQU    X'40'           WHEN ON, STORAGE CLASS
*                               SPECIFIED ON OSREQ CHANGE.
ST5FLG2  EQU    X'20'           WHEN ON, RETENTION PERIOD
*                               SPECIFIED ON OSREQ CHANGE.
ST5FLG3  EQU    X'10'           RETPD=-1                        @L8A
ST5FLG4  EQU    X'08'           RETPD=-2                        @L8A
ST5FLG5  EQU    X'04'           RETPD=X'7FFFFFFF'               @L8A
ST5FLG6  EQU    X'02'           EVENTEXP supplied               @L8A
ST5FLG7  EQU    X'01'           DELHOLD=HOLD                    @L8A
ST5FLG8  EQU    X'80'           DELHOLD=NOHOLD                  @L8A
***********************************************************************
*                                                                     *
* SUBTYPE 6 - OSREQ DELETE FLAGS                                      *
*                                                                     *
***********************************************************************
ST6FLG0  EQU    X'80'           WHEN ON, PRIMARY COPY OF OBJECT
*                               DELETED FROM Disk.                @L9C
ST6FLG1  EQU    X'40'           WHEN ON, PRIMARY COPY OF OBJECT
*                               DELETED FROM OPTICAL.
ST6FLG2  EQU    X'20'           WHEN ON, PRIMARY COPY OF OBJECT
*                               DELETED FROM TAPE.
ST6FLG3  EQU    X'10'           WHEN ON, BACKUP COPY OF OBJECT
*                               DELETED FROM OPTICAL.
ST6FLG4  EQU    X'08'           WHEN ON, BACKUP COPY OF OBJECT
*                               DELETED FROM TAPE.
ST6FLG5  EQU    X'04'           WHEN ON, 2ND BACKUP COPY OF OBJECT
*                               DELETED FROM OPTICAL.             @L2A
ST6FLG6  EQU    X'02'           WHEN ON, 2ND BACKUP COPY OF OBJECT
*                               DELETED FROM TAPE.                @L2A
ST6FLG7  EQU    X'01'           WHEN ON, THE PRIMARY COPY OF
*                               THE OBJECT WAS DELETED FROM
*                               A LOB STORAGE STRUCTURE           @P6C
ST6FLG8 EQU    X'80'           WHEN ON, OBJECT DELETED FROM SUBLEVEL 1
*                              VOLUME.                            @L6A
ST6FLG9 EQU    X'40'           WHEN ON, OBJECT DELETED FROM SUBLEVEL 2
*                              VOLUME.                            @L6A
***********************************************************************
*                                                                     *
* SUBTYPE 7 - OSREQ UNACCESS FLAGS                                    *
*                                                                     *
***********************************************************************
***********************************************************************
```

Figure A-7 SMF record type 85 subtype 1 significant fields from CBRSMF macro (5 of 6)

```
***********************************************************************
***********************************************************************
*                                                                     *
* SUBTYPE 8 - OSREQ STOREBEG FLAGS                               @L7A*
*                                                                     *
***********************************************************************
***********************************************************************
*                                                                     *
* SUBTYPE 9 - OSREQ STOREPRT FLAGS                               @L7A*
*                                                                     *
***********************************************************************
***********************************************************************
*                                                                     *
* SUBTYPE 10 - OSREQ STOREEND FLAGS                              @L7A*
*                                                                     *
***********************************************************************
ST10FLGS0 EQU   X'80'           When on, the object is stored to Disk.
*                                                                  @L9C
ST10FLGS1 EQU   X'40'           UNUSED                             @L8A
ST10FLGS2 EQU   X'20'           Object store to Tape               @L8A
ST10FLGS3 EQU   X'10'           UNUSED
ST10FLGS4 EQU   X'08'           UNUSED
ST10FLGS5 EQU   X'04'           Volume on shelf                    @L8A
ST10FLGS6 EQU   X'02'           Volume in library                  @L8A
ST10FLGS7 EQU   X'01'           Volume on drive                    @L8A
ST10FLGS8 EQU   X'80'           Immediate backup copy was sched    @L8A
ST10FLGS9 EQU   X'40'           WHEN ON, THE OBJECT IS STORED
*                               TO LOB STORAGE STRUCTURE
ST10FLGS10 EQU  X'20'           Obj. store to sublvl1              @L8A
ST10FLGS11 EQU  X'10'           Obj. store to sublvl2              @L8A
ST10FLGS12 EQU  X'08'           When on, the CANCEL=YES keyword was
*                               specified indicating the store
*                               sequence was successfully cancelled. ST10FLGS13 EQU
X'04'            DELHOLD=HOLD                    @L8A
ST10FLGS14 EQU  X'02'           Retention-protected                @L8A
ST10FLGS15 EQU  X'01'           Deletion-protected                 @L8A
ST10FLGS16 EQU  X'80'           Event-based-retention              @L8A
```

Figure A-8 SMF record type 85 subtype 1-10 from CBRSMF macro (6 of 6)

A.2.2 Program SMF85TH SMF record type 85 subtypes 32-35

OAM writes SMF Record type 85 subtypes 32 - 35 to document the output that results from the execution of OSMC macros used in batch jobs. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TH that scans the SMF type 85 subtypes 32-35 records and formats the records of activity. The program itself and how to construct it is documented in A.3.3, “SMF record type 85 subtype 32-35 data display program SMF85TH” on page 274.

Figure A-9 show the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

If you do not want output from all the types that the program can process, change the SMF type and subtype selection statement to include only those that you want. For example, change OUTDD(OUTDD,TYPE(85(32:35))) to OUTDD(OUTDD,TYPE(85(32))) to exclude subtypes 33-35.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1      <- change as needed
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(32:35)))
/*
//  EXEC PGM=SMF85TH
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TH.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-9 SMF85TH SMF data listing JCL

Figure A-11 on page 229 through Figure A-17 on page 235 show parts of the current content of the CRBSMF macro (assembler portion). The figures show how it relates to SMF Record type 85 subtypes 32-35. To reduce the number of lines, selected parts of this section of the macro are omitted. However, the full contents can be seen by looking at the actual macro. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. In this case you need fields prefixed with ST32 for the bulk of the fields, then ST33, ST34, and ST35 for differences from ST32 if applicable.

One of the significant changes introduced with V1R13 is a different way of recording some data. Some fields, for example ST32PDWK, were defined as a binary length 4 field that contains a number in KBytes. This process has been found to be imprecise. Therefore, new fields have been created, such as ST32PDWB (the replacement for ST32PDWK) defined as binary length 8, and containing a number in bytes. Now both ST32PDWK and ST32PDWB are provided. If ST32PDWK overflows, it would be set to X’FFFFFFFF’. The CRBSMF details show which fields expressed in KBytes have been replaced with version expressed in bytes.

Figure A-10 shows selected lines of output from running program SMF85TH. These SMF records relate to Storage Group processing so no objects are separately identified. In the sample shown, field ST32PDWK shows X'00000023', which is 35 decimal KB. Field ST32PDWB shows X'0000000000008A80', which is decimal 35456 bytes and therefore more precise than the rounded KB value.

```
SMF85TH SMFDTE/SMFTME:              2011286 22:43:16.007
STYPE/FUNC/SGN/VSNO/VSN1/MT:         32/(STORAGE GROUP PROCESSING)       /SG3     /       /       /
PDWO/PDWK/PDRO/PDRK/PDDO/PDDK:       00000000    /00000000    /00000000    /00000000    /00000000    /00000000    /
PDWB/PDRB/PDDB:                     0000000000000000 /0000000000000000 /0000000000000000
POWO/POWK/PORO/PORK/PODO/PODK:       00000000    /00000000    /00000000    /00000000    /00000000    /00000000
POWB/PORB/PODB:                     000000000000000000000000000000000000000000
PTWO/PTWK/PTRK/PTRD/PTDO/PTDK:       00000000    /00000000    /00000000    /00000000    /00000000    /00000000
PTWB/PTRB/PTDB:                     0000000000000000 /0000000000000000 /0000000000000000
BOWO/BOWK/BORO/BORK/BODO/BODK:       00000000    /00000000    /00000000    /00000000    /00000000    /00000000
BOWB/BORB/BODB:                     0000000000000000 /0000000000000000 /0000000000000000
BTWO/BTWK/BTRO/BTRK/BTDO/BTDK:       00000001    /00000023    /00000000    /00000000    /00000000    /00000000
BTWB/BTRB/BTDB:                     0000000000008A80 /0000000000000000 /0000000000000000
B2OWO/B2OWK/B2ORO/B2ORK/B2ODO/B2ODK: 00000000    /00000000    /00000000    /00000000    /00000000    /00000000
B2OWB/B2ORB/B2ODK:                  0000000000000000 /0000000000000000 /0000000000000000
B2TWO/B2TWK/B2TRO/B2TRK/B2TDO/B2TDK: 00000000    /00000000    /00000000    /00000000    /00000000    /00000000
B2TWK/B2TRK/B2TDK:                  0000000000000000 /0000000000000000 /0000000000000000
DTUP/DTDE/4KIN/4KDE/32KI/32KD/NCE:   00000002     /00000000     /00000000000 /00000000000 /00000000000 00000000000..
FLGS/NTE/RCLD/RCLK/LOBI/LOBD:        41000000 /00000000    /00000000    /00000000    /00000000    /00000000
RCLB:                               0000000000000000
PUWO/PUWK/PURO/PURK/PUDO/PUDK:       00000000    /00000000    /00000000    /00000000    /00000000    /00000000
PUWB/PURB/PUDB:                       0000000000000000 /0000000000000000 /0000000000000000
```

Figure A-10 Selected output examples from running the SMF85TH program

Figure A-11 through Figure A-17 on page 235 show parts of the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 32-35.

```
***********************************************************************
*   OAM SUBTYPE SECTION, FOR SUBTYPES 32 - 35                         *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------       *
*        32 OSMC STORAGE GROUP PROCESSING                             *
*        33 OSMC DASD SPACE MANAGEMENT PROCESSING                     *
*        34 OSMC OPTICAL DISK RECOVERY UTILITY                        *
*        35 OSMC MOVE VOLUME (MOVEVOL) UTILITY                        *
***********************************************************************
ST32     DSECT                   SUBTYPES 32 - 35
ST32SGN  DS     CL8' '           STORAGE GROUP NAME
ST32VSN0 DS     CL6' '           VOLUME SERIAL NUMBER OF A TAPE OR
*                                OPTICAL VOLUME.  ONLY VALID FOR
*                                SUBTYPES 34 AND 35, THIS FIELD
*                                CONTAINS BLANKS FOR ALL OTHER
*                                SUBTYPES.                      @P8C
ST32VSN1 DS     CL6' '           VOLUME SERIAL NUMBER OF OPPOSITE
*                                SIDE OF AN OPTICAL VOLUME.
*                                ONLY VALID FOR SUBTYPES 34 and 35.
*                                IF THE VOLUME SERIAL NUMBER CONTAINED
*                                IN FIELD ST32VSN0 IS THE VOLUME
*                                SERIAL NUMBER OF A TAPE VOLUME, THIS
*                                FIELD WILL BE 'N/A'.  THIS FIELD
*                                CONTAINS BLANKS FOR ALL OTHER
*                                SUBTYPES.                      @P8C
... omitted lines
***********************************************************************
* COUNTS OF PRIMARY OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM DISK SUBLEVEL 1 (DSL1).                            @LAC
***********************************************************************
ST32PDWO DS     BL4'0'           NUMBER OF PRIMARY OBJECTS
*                                WRITTEN TO DSL1.                 @LAC
ST32PDWK DS     BL4'0'           NUMBER OF KILOBYTES PRIMARY
*                                OBJECTS WRITTEN TO DSL1.
*                                SUPERSEDED BY ST32PDWB.
*                                SEE NOTE 1.                      @LAC
ST32PDRO DS     BL4'0'           NUMBER OF PRIMARY OBJECTS
*                                READ FROM DSL1.                  @LAC
ST32PDRK DS     BL4'0'           NUMBER OF KILOBYTES PRIMARY
*                                OBJECTS READ FROM DSL1.
*                                SUPERSEDED BY ST32PDRB.
*                                SEE NOTE 1.                      @LAC
ST32PDDO DS     BL4'0'           NUMBER OF PRIMARY OBJECTS
*                                DELETED FROM DSL1.               @LAC
ST32PDDK DS     BL4'0'           NUMBER OF KILOBYTES OF PRIMARY
*                                OBJECTS DELETED FROM DSL1.
*                                SUPERSEDED BY ST32PDDB.
*                                SEE NOTE 1.                      @LAC
```

Figure A-11 SMF type 85 subtypes 32-35 significant fields from CBRSMF macro (1 of 7)

```
... lines omitted
***********************************************************************
* COUNTS OF PRIMARY OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM TAPE.
***********************************************************************
ST32PTWO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               WRITTEN TO TAPE.
ST32PTWK DS     BL4'0'          NUMBER OF KILOBYTES OF PRIMARY
*                               OBJECTS WRITTEN TO TAPE.
*                               SUPERSEDED BY ST32PTWB.
*                               SEE NOTE 1.                     @LAC
ST32PTRO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               READ FROM TAPE.
ST32PTRK DS     BL4'0'          NUMBER OF KILOBYTES OF PRIMARY
*                               OBJECTS READ FROM TAPE.
*                               SUPERSEDED BY ST32PTRB.
*                               SEE NOTE 1.                     @LAC
ST32PTDO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               LOGICALLY DELETED FROM TAPE.
ST32PTDK DS     BL4'0'          NUMBER OF KILOBYTES OF PRIMARY
*                               OBJECTS LOGICALLY DELETED FROM
*                               TAPE. SUPERSEDED BY ST32PTDB.
*                               SEE NOTE 1.                     @LAC
... omitted lines
***********************************************************************
* COUNTS OF BACKUP OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM TAPE.
***********************************************************************
ST32BTWO DS     BL4'0'          NUMBER OF BACKUP  OBJECTS
*                               WRITTEN TO TAPE.
ST32BTWK DS     BL4'0'          NUMBER OF KILOBYTES OF BACKUP
*                               OBJECTS WRITTEN TO TAPE.
*                               SUPERSEDED BY ST32BTWB.
*                               SEE NOTE 1.                     @LAC
ST32BTRO DS     BL4'0'          NUMBER OF BACKUP  OBJECTS
*                               READ FROM TAPE.
ST32BTRK DS     BL4'0'          NUMBER OF KILOBYTES OF BACKUP
*                               OBJECTS READ FROM TAPE.
*                               SUPERSEDED BY ST32BTRB.
*                               SEE NOTE 1.                     @LAC
ST32BTDO DS     BL4'0'          NUMBER OF BACKUP  OBJECTS
*                               LOGICALLY DELETED FROM TAPE.
ST32BTDK DS     BL4'0'          NUMBER OF KILOBYTES OF BACKUP
*                               OBJECTS LOGICALLY DELETED FROM
*                               TAPE. SUPERSEDED BY ST32BTDB.
*                               SEE NOTE 1.                     @LAC
... omitted lines
```

Figure A-12 SMF type 85 subtypes 32-35 significant fields from CBRSMF macro (2 of 7)

```
***********************************************************************
* COUNTS OF BACKUP2 OBJECTS (AND KILOBYTES) WRITTEN, READ AND    @L2A
* DELETED FROM TAPE.                                             @L2A
***********************************************************************
ST32B2TWO DS     BL4'0'         NUMBER OF BACKUP2 OBJECTS        @L2A
*                               WRITTEN TO TAPE.                 @L2A
ST32B2TWK DS     BL4'0'         NUMBER OF KILOBYTES OF BACKUP2   @L2A
*                               OBJECTS WRITTEN TO TAPE.         @L2A
*                               SUPERSEDED BY ST32B2TWB.         @LAC
*                               SEE NOTE 1.                      @LAC
ST32B2TRO DS     BL4'0'         NUMBER OF BACKUP2 OBJECTS        @L2A
*                               READ FROM TAPE.                  @L2A
ST32B2TRK DS     BL4'0'         NUMBER OF KILOBYTES OF BACKUP2   @L2A
*                               OBJECTS READ FROM TAPE.          @L2A
*                               SUPERSEDED BY ST32B2TRB.         @LAC
*                               SEE NOTE 1.                      @LAC
ST32B2TDO DS     BL4'0'         NUMBER OF BACKUP2 OBJECTS        @L2A
*                               LOGICALLY DELETED FROM TAPE.     @L2A
ST32B2TDK DS     BL4'0'         NUMBER OF KILOBYTES OF BACKUP2   @L2A
*                               OBJECTS LOGICALLY DELETED FROM   @L2A
*                               TAPE. SUPERSEDED BY ST32B2TDB.   @LAC
*                               SEE NOTE 1.                      @LAC
***********************************************************************
* COUNTS OF ACTIVITY AGAINST THE OBJECT STORAGE DATABASE
* (OBJECT DIRECTORY TABLE, 4K OBJECT STORAGE TABLE AND 32K
* OBJECT STORAGE TABLE).
***********************************************************************
ST32DTUP DS     BL4'0'          NUMBER OF ROWS UPDATED IN THE
*                               OBJECT DIRECTORY TABLE.
ST32DTDE DS     BL4'0'          NUMBER OF ROWS DELETED FROM THE
*                               OBJECT DIRECTORY TABLE.
ST324KIN DS     BL4'0'          NUMBER OF ROWS INSERTED INTO THE
*                               4K OBJECT STORAGE TABLE.
ST324KDE DS     BL4'0'          NUMBER OF ROWS DELETED FROM THE
*                               4K OBJECT STORAGE TABLE.
ST3232KI DS     BL4'0'          NUMBER OF ROWS INSERTED INTO THE
*                               32K OBJECT STORAGE TABLE.
ST3232KD DS     BL4'0'          NUMBER OF ROWS DELETED FROM THE
*                               32K OBJECT STORAGE TABLE.
ST32NCE  DS     BL4'0'          NUMBER OF OPTICAL CARTRIDGES
*                               EXPIRED.  VALID ONLY FOR
*                               SUBTYPE 32.
```

Figure A-13 SMF type 85 subtypes 32-35 significant fields from CBRSMF macro (3 of 9)

```
ST32FLGS DS     BL4'0'          PROCESSING FLAGS
ST32FLG0 EQU    X'80'           WHEN ON, THIS PROCESS WAS
*                               INVOKED AUTOMATICALLY UNDER
*                               SOFTWARE CONTROL.
ST32FLG1 EQU    X'40'           WHEN ON, THIS PROCESS WAS
*                               INVOKED BY A MODIFY OAM,START
*                               COMMAND.
ST32FLG2 EQU    X'20'           WHEN ON, THIS PROCESS WAS
*                               INVOKED USING AN ISMF LINE
*                               OPERATOR.
ST32FLG3 EQU    X'10'           WHEN ON, VOL RECOVERY WAS
*                               INVOKED W/ BACKUP1 KEYWORD OR
*                               DEFAULTED TO BACKUP1            @L2A
ST32FLG4 EQU    X'08'           WHEN ON, VOL RECOVERY WAS
*                               INVOKED W/ BACKUP2 KEYWORD      @L2A
ST32FLG5 EQU    X'04'           WHEN ON, VOL RECOVERY OR MOVEVOL WAS
*                               SPECIFIED WITH DELETE OPTION    @L3A
ST32FLG6 EQU    X'02'           WHEN ON, VOL RECOVERY OR MOVEVOL WAS
*                               SPECIFIED WITH RECYCLE OPTION   @L3A
ST32FLG7 EQU    X'01'           WHEN ON, INDICATED PROCESSING OBJECT
*                               STORAGE GROUP                   @L3A
ST32FLG8 EQU    X'80'           WHEN ON, INDICATED PROCESSING BACKUP
*                               OBJECT STORAGE GROUP            @L3A
ST32FLG9 EQU    X'40'           WHEN ON, CYCLE ENDTIME EXCEEDED @L3A
*
ST32NTE  DS     BL4'0'          NUMBER OF TAPE VOLUMES EXPIRED
*                               VALID ONLY FOR SUBTYPE 32       @L3A
ST32RCLD DS     BL4'0'          NUMBER OF RECALLED OBJECTS
*                               VALID ONLY FOR SUBTYPE 32       @P4A
ST32RCLK DS     BL4'0'          NUMBER OF KILOBYTES OF
*                               RECALLED OBJECTS
*                               SUPERSEDED BY ST32RCLB.
*                               SEE NOTE 1.                     @LAC
*                               VALID ONLY FOR SUBTYPE 32       @P4A
ST32LOBI DS     BL4'0'          NUMBER OF LOB ROWS INSERTED
*                               VALID ONLY FOR SUBTYPE 32       @P4A
ST32LOBD DS     BL4'0'          NUMBER OF LOB ROWS DELETED
*                               VALID ONLY FOR SUBTYPE 32       @P4A
```

Figure A-14 SMF type 85 subtypes 32-35 significant fields from CBRSMF macro (4 of 7)

```
***********************************************************************
* COUNTS OF PRIMARY OBJECTS (AND KILOBYTES) WRITTEN, READ AND
* DELETED FROM TAPE SUBLEVEL 2 (TSL2).                            @L6A
***********************************************************************
ST32PUWO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               WRITTEN TO TSL2.                  @L6A
ST32PUWK DS     BL4'0'          NUMBER OF KILOBYTES OF PRIMARY
*                               OBJECTS WRITTEN TO TSL2.
*                               SUPERSEDED BY ST32PUWB.
*                               SEE NOTE 1.                       @LAC
ST32PURO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               READ FROM TSL2.                   @L6A
ST32PURK DS     BL4'0'          NUMBER OF KILOBYTES OF PRIMARY
*                               OBJECTS READ FROM TSL2.
*                               SUPERSEDED BY ST32PURB.
*                               SEE NOTE 1.                       @LAC
ST32PUDO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               LOGICALLY DELETED FROM TSL2.      @L6A
ST32PUDK DS     BL4'0'          NUMBER OF KILOBYTES OF PRIMARY
*                               OBJECTS LOGICALLY DELETED FROM
*                               TSL2. SUPERSEDED BY ST32PUDB.
*                               SEE NOTE 1.                       @LAC
***********************************************************************
* COUNTS OF PRIMARY OBJECTS WRITTEN, READ, AND DELETED FROM
* DISK SUBLEVEL 2 (DSL2).                                         @LAA
***********************************************************************
ST32PEWO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               WRITTEN TO DSL2.                  @LAA
ST32PERO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               READ FROM DSL2.                   @LAA
ST32PEDO DS     BL4'0'          NUMBER OF PRIMARY OBJECTS
*                               LOGICALLY DELETED FROM DSL2.      @LAA
***********************************************************************
* BYTE COUNTS OF PRIMARY OBJECTS WRITTEN, READ AND
* DELETED FROM DISK SUBLEVEL 1 (DSL1).                            @LAA
***********************************************************************
ST32PDWB DS     BL8'0'          NUMBER OF BYTES OF PRIMARY
*                               OBJECTS WRITTEN TO DSL1.          @LAA
ST32PDRB DS     BL8'0'          NUMBER OF BYTES OF PRIMARY
*                               OBJECTS READ FROM DSL1.           @LAA
ST32PDDB DS     BL8'0'          NUMBER OF BYTES OF PRIMARY
*                               OBJECTS DELETED FROM DSL1.        @LAA
... omitted lines
```

Figure A-15 SMF type 85 subtypes 32-35 significant fields from CBRSMF macro (5 of 7)

```
***********************************************************************
* BYTE COUNTS OF PRIMARY OBJECTS WRITTEN, READ AND
* DELETED FROM TAPE.                                             @LAA
***********************************************************************
ST32PTWB DS     BL8'0'           NUMBER OF BYTES OF PRIMARY
*                                OBJECTS WRITTEN TO TAPE.        @LAA
ST32PTRB DS     BL8'0'           NUMBER OF BYTES OF PRIMARY
*                                OBJECTS READ FROM TAPE.         @LAA
ST32PTDB DS     BL8'0'           NUMBER OF BYTES OF PRIMARY
*                                OBJECTS LOGICALLY DELETED FROM
*                                TAPE.                           @LAA
... omitted lines
***********************************************************************
* BYTE COUNTS OF BACKUP OBJECTS WRITTEN, READ AND
* DELETED FROM TAPE.                                             @LAA
***********************************************************************
ST32BTWB DS     BL8'0'           NUMBER OF BYTES OF BACKUP
*                                OBJECTS WRITTEN TO TAPE.        @LAA
ST32BTRB DS     BL8'0'           NUMBER OF BYTES OF BACKUP
*                                OBJECTS READ FROM TAPE.         @LAA
ST32BTDB DS     BL8'0'           NUMBER OF BYTES OF BACKUP
*                                OBJECTS LOGICALLY DELETED FROM
*                                TAPE.                           @LAA
... omitted lines
***********************************************************************
* BYTE COUNTS OF BACKUP2 OBJECTS WRITTEN, READ AND
* DELETED FROM TAPE.                                             @LAA
***********************************************************************
ST32B2TWB DS     BL8'0'          NUMBER OF BYTES OF BACKUP2
*                                OBJECTS WRITTEN TO TAPE.        @LAA
ST32B2TRB DS     BL8'0'          NUMBER OF BYTES OF BACKUP2
*                                OBJECTS READ FROM TAPE.         @LAA
ST32B2TDB DS     BL8'0'          NUMBER OF BYTES OF BACKUP2
*                                OBJECTS LOGICALLY DELETED FROM
*                                TAPE.                           @LAA
***********************************************************************
* BYTE COUNTS OF RECALLED OBJECTS                                @LAA
***********************************************************************
ST32RCLB DS      BL8'0'          NUMBER OF BYTES OF
*                                RECALLED OBJECTS.               @LAA
```

Figure A-16 SMF type 85 subtypes 32-35 significant fields from CBRSMF macro (6 of 7)

```
*********************************************************************
* BYTE COUNTS OF PRIMARY OBJECTS WRITTEN, READ AND
* DELETED FROM TAPE SUBLEVEL 2 (TSL2).
*********************************************************************
ST32PUWB DS      BL8'0'         NUMBER OF BYTES OF PRIMARY
*                               OBJECTS WRITTEN TO TSL2.        @LAA
ST32PURB DS      BL8'0'         NUMBER OF BYTES OF PRIMARY
*                               OBJECTS READ FROM TSL2.         @LAA
ST32PUDB DS      BL8'0'         NUMBER OF BYTES OF PRIMARY
*                               OBJECTS LOGICALLY DELETED FROM
*                               TSL2.                           @LAA
*********************************************************************
* BYTE COUNTS OF PRIMARY OBJECTS WRITTEN, READ AND
* DELETED FROM DISK SUBLEVEL 2 (DSL2).                          @LAA
*********************************************************************
ST32PEWB DS      BL8'0'         NUMBER OF BYTES OF PRIMARY
*                               OBJECTS WRITTEN TO DSL2.        @LAA
ST32PERB DS      BL8'0'         NUMBER OF BYTES OF PRIMARY
*                               OBJECTS READ FROM DSL2.         @LAA
ST32PEDB DS      BL8'0'         NUMBER OF BYTES OF PRIMARY
*                               OBJECTS LOGICALLY DELETED FROM
*                               DSL2.                           @LAA
```

Figure A-17 SMF type 85 subtypes 32-35 significant fields from CBRSMF macro (7 of 7)

A.2.3 Program SMF85TQ SMF record type 85 subtype 36

OAM writes SMF Record type 85 subtypes 36 to document the output that results from the execution of OSMC SINGLE OBJECT RECOVERY UTILITY macros used in batch jobs. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TQ that scans the SMF type 85 subtype 36 records and formats the records of activity. The program itself and how to construct it is documented in A.3.4, “SMF record type 85 subtype 36 data display program SMF85TQ” on page 284

Figure A-18 shows the JCL to extract the SMF records and run the program.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1       <-change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(36)))
/*
//  EXEC PGM=SMF85TQ
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TQ.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-18 SMF85TQ SMF data listing JCL

Figure A-20 on page 237 through Figure A-21 on page 238 show parts of the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtype 36. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST36.

One of the significant changes introduced with V1R13 is the option to store objects on what is termed disk sublevel 2. The disk sublevel 1 is conventional DASD, and sublevel 2 uses zFS or NFS file structures.

Figure A-19 shows selected lines of output from running program SMF85TQ. The following fields are highlighted:

- OLEN: Object length shown as decimal 35456
- DSL shown as 02: From the CBRSMF record definitions, the DSL field has the following meaning: Disk sublevel associated with the recovered object. This indicates that the object was recovered to sublevel 2, which is the file system, as expected.

```
SMF TYPE 85 SUBTYPE 36 RECORDS
SMF85TQ SMFDTE/TME:    2011287 15:03:20.067
COLN/CNID:            ANDRECS.SG3.OAMTES9                   /00000000026
OBJN/SGN/OLEN:          MHLRES2.SMF85TP.PDS.OAMTES9                /SG3
/00000035456
BVSN/BMT/BTKN/TVSN/TMT: THS025/07/00000000006       /      /
OVSN/OMT/FLGS/DSL:           /  /80000000 /02
```

Figure A-19 Selected output examples from running the SMF85TQ program

Figure A-20 through Figure A-21 on page 238 show the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 36.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPE 36                               *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------       *
*        36 OSMC SINGLE OBJECT RECOVERY UTILITY                       *
*                                                                     *
***********************************************************************
ST36     DSECT                  SUBTYPE 36
ST36COLN DS     CL44' '         COLLECTION NAME
ST36CNID DS     BL4'0'          COLLECTION ID
ST36OBJN DS     CL44' '         OBJECT NAME
ST36SGN  DS     CL8' '          STORAGE GROUP NAME
ST36OLEN DS     BL4'0'          OBJECT LENGTH
ST36BVSN DS     CL6' '          VOLUME SERIAL NUMBER OF OPTICAL
*                               VOLUME OR TAPE VOLUME FROM WHICH
*                               THE BACKUP COPY OF THE OBJECT
*                               WAS READ.
ST36BMT  DS     CL2' '          MEDIA TYPE OF THE VOLUME FROM
*                               WHICH THE BACKUP COPY OF THE
*                               BACKUP COPY OF THE OBJECT WAS
*                               READ.
ST36BTKN DS     BL4'0'          VOLUME LOCATION TOKEN ASSOCIATED
*                               WITH BACKUP COPY OF THE OBJECT
*                               ON THE VOLUME SPECIFIED IN THE
*                               ST36BVSN FIELD.
ST36TVSN DS     CL6' '          VOLUME SERIAL NUMBER OF OPTICAL
*                               VOLUME OR TAPE VOLUME TO WHICH
*                               THE NEW PRIMARY COPY OF THE
*                               OBJECT WAS WRITTEN.  THIS FIELD
*                               CONTAINS BLANKS IF THE NEW
*                               PRIMARY COPY WAS WRITTEN TO
*                               A DISK SUBLEVEL.                @L9C
ST36TMT  DS     CL2' '          MEDIA TYPE OF THE VOLUME TO
*                               WHICH THE NEW PRIMARY COPY OF
*                               OBJECT WAS WRITTEN.
```

Figure A-20 SMF type 85 subtypes 36 significant fields from CBRSMF macro (1 of 2)

```
ST36OVSN DS     CL6' '          VOLUME SERIAL NUMBER OF THE      @P5C
*                               ORIGINAL OPTICAL OR TAPE VOLUME
*                               ON WHICH THE PRIMARY COPY OF THE
*                               OBJECT RESIDED PRIOR TO THE
*                               START OF THE SINGLE OBJECT
*                               RECOVERY UTILITY.  THIS FIELD
*                               CONTAINS BLANKS, IF THE PRIMARY
*                               COPY OF THE OBJECT CURRENTLY
*                               RESIDES ON A DISK SUBLEVEL       @L9C
ST36OMT  DS     CL2' '          MEDIA TYPE OF THE ORIGINAL
*                               OPTICAL OR TAPE VOLUME ON WHICH
*                               THE PRIMARY COPY OF THE OBJECT
*                               RESIDED PRIOR TO THE START OF
*                               THE SINGLE OBJECT RECOVERY
*                               UTILITY.  THIS FIELD
*                               CONTAINS BLANKS, IF THE PRIMARY
*                               COPY OF THE OBJECT CURRENTLY
*                               RESIDES ON A DISK SUBLEVEL       @L9C
ST36FLGS DS     BL4'0'          PROCESSING FLAGS             @L2C
*
ST36FLG0 EQU    X'80'           WHEN ON, OBJECT RECOVERY
*                               INVOKED W/ BACKUP1 KEYWORD
*                               OR DEFAULTED TO BACKUP1      @L2A
ST36FLG1 EQU    X'40'           WHEN ON, OBJECT RECOVERY
*                               INVOKED W/ BACKUP2 KEYWORD   @L2A
ST36DSL  DS     CL2' '          Disk sublevel associated with the
*                               recovered object.  This field is only
*                               valid when the ST36TVSN field contains
*                               blanks.                          @L9A
```

Figure A-21 SMF type 85 subtypes 36 significant fields from CBRSMF macro (2 of 2)

A.2.4 Program SMF85TO SMF record type 85 subtype 38

OAM writes SMF Record type 85 subtype 38 to document the output that results from the execution of OSMC SINGLE OBJECT RECALL UTILITY macros used in batch jobs. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TO that scans the SMF type 85 subtype 38 records and formats the records of activity. The program itself and how to construct it is documented in A.3.5, “SMF Record type 85 subtype 38 data display program SMF85TO” on page 289

Figure A-22 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1      < --- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(38)))
/*
//  EXEC PGM=SMF85TO
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TO.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-22 SMF85TO SMF data listing JCL

Figure A-24 on page 240 shows parts of the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 38. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST38.

Figure A-23 shows selected lines of output from running program SMF85TO.

```
No relevant OAM activity occurred during the test time frame
```

Figure A-23 Selected output examples from running the SMF85TQ program

Figure A-24 shows the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 38.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPE 38                               *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- -----------------------------------------------------    *
*        38 OSMC SINGLE OBJECT RECALL UTILITY                @P2A    *
*                                                                     *
***********************************************************************
ST38     DSECT                  SUBTYPE 38                   @P2A
ST38COLN DS     CL44' '         COLLECTION NAME              @P2A
ST38CNID DS     BL4'0'          COLLECTION ID                @P2A
ST38OBJN DS     CL44' '         OBJECT NAME                  @P2A
ST38SGN  DS     CL8' '          STORAGE GROUP NAME           @P2A
ST38OLEN DS     BL4'0'          OBJECT LENGTH                @P2A
ST38VSN  DS     CL6' '          VOLUME SERIAL NUMBER OF OPTICAL
*                               VOLUME OR TAPE VOLUME FROM WHICH
*                               THE COPY OF THE OBJECT
*                               WAS READ.                    @P2A
ST38MT   DS     CL2' '          MEDIA TYPE OF THE VOLUME FROM
*                               WHICH THE COPY OF THE OBJECT
*                               WAS READ.                    @P2A
ST38TKN  DS     BL4'0'          VOLUME LOCATION TOKEN ASSOCIATED
*                               WITH THE COPY OF THE OBJECT
*                               ON THE VOLUME SPECIFIED IN THE
*                               ST38VSN FIELD.               @01C
ST38RCLD DS     BL4'0'          NUMBER OF DAYS THIS OBJECT
*                               IS TO REMAIN IN RECALLED
*                               MODE.                        @01A
ST38VT   DS     CL1' '          VOLUME TYPE OF THE VOLUME
*                               SPECIFIED IN THE ST38VSN FIELD
*                               G = VOLUME IS A GROUPED VOLUME
*                               BELONGING TO AN OBJECT STORAGE
*                               GROUP.
*                               B = VOLUME IS A GROUPED VOLUME
*                               BELONGING TO AN OBJECT BACKUP
*                               STORAGE GROUP                @01C
ST38BT   DS     CL1' '          BACKUP TYPE OF THE VOLUME
*                               SPECIFIED IN THE ST38VSN FIELD
*                               1 = BACKUP1 VOLUME
*                               2 = BACKUP2 VOLUME           @01C
ST38DSL  DS     CL2' '          Disk sublevel associated with the
*                               recalled object.                 @L9C
ST38FLGS DS     BL4'0'          PROCESSING FLAGS             @P2A
*
ST38FLG0 EQU    X'80'           WHEN ON, OBJECT RECALL
*                               WAS SUCCESSFUL.              @P2A
```

Figure A-24 SMF type 85 subtypes 38 significant fields from CBRSMF macro (1 of 1)

A.2.5 Program SMF85TI SMF record type 85 subtype 39

OAM writes SMF Record type 85 subtypes 39 to document the output that results from the specification that OSMC IMMEDIATE BACKUP COPY function is to be used.

The Immediate Backup specification is part of the SMS Management Class settings. Specifying Immediate Backup causes an OSMC subtask to be scheduled to make a backup copy to tape when an object is STORED. This process requires a tape to be mounted.

The more standard setup allows the OSMC subtask to create a backup during a scheduled run where several objects are processed together.

The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TI that scans the SMF type 85 subtypes 39 records and formats the records of activity. The program itself and how to construct it is documented in A.3.6, "SMF Record type 85 subtype 39 data display program SMF85TI" on page 294

Figure A-25 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1     <-- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
          INDD(DUMPIN,OPTIONS(DUMP))
          OUTDD(OUTDD,TYPE(85(39)))
/*
//  EXEC PGM=SMF85TI
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TI.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-25 SMF85TI SMF data listing JCL

Figure A-27 on page 243 though Figure A-28 on page 244 show the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 39. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST39.

DFSMS V1R13 introduced the ability to store objects on File Systems zFS or NFS. The file system storage type is called Disk sublevel 2 and conventional DASD storage is called Disk sublevel 1.

- ST39FLG10 EQU X'20' indicates that the object is on sublevel 1
- ST39FLG11 EQU X'10' indicates that the object is on sublevel 2 (that is, zFS or NFS)

Figure A-26 shows selected lines of output from running program SMF85TI.

In this sample, apart from the collection name (COLN) and the object name (OBJN) the point of interest is in the FLGS field which is displayed as X'8290000'. The flags have the following meanings:

- ST39FLG0 indicates that the primary copy of the object is on DASD
- ST39FLG6 indicates that the backup is on tape
- ST39FLG8 indicates that the backup write to tape was successful
- ST39FLG11 indicates that the primary copy is stored on disk sublevel 2 (the file system)

```
SMF85TI SMFDTE/TME:         2011287/18:23:49.096
COLN/CNID:                  ANDRECS.SG3.OB9.D141011                     00000000029
OBJN/SGN/MCN/OLEN:          ANDRECS.SG3.OB9.D141011                    /SG3     /MCOBDASD/00000000099
SVSN/SMT/TVSN/TMT/BTKN/FLGS:      /  /THS025/07/00000000012        /82900000
```

Figure A-26 Selected output examples from running the SMF85TI program

Figure A-27 through Figure A-28 on page 244 show the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 39.

```
**********************************************************************
*                                                                    *
*   OAM SUBTYPE SECTION, FOR SUBTYPE 39                          @L5A*
*                                                                    *
*   SUBTYPE FUNCTION                                             @L5A*
*   ------- ---------------------------------------------------      *
*        39 OSMC IMMEDIATE BACKUP COPY                           @L5A*
*                                                                    *
**********************************************************************
ST39     DSECT                  SUBTYPE 39                        @L5A
ST39COLN DS     CL44' '         COLLECTION NAME                   @L5A
ST39CNID DS     BL4'0'          COLLECTION ID                     @L5A
ST39OBJN DS     CL44' '         OBJECT NAME                       @L5A
ST39SGN  DS     CL8' '          STORAGE GROUP NAME                @L5A
ST39MCN  DS     CL8' '          MANAGEMENT CLASS NAME             @L5A
ST39OLEN DS     BL4'0'          OBJECT LENGTH                     @L5A
ST39SVSN DS     CL6' '          SOURCE VOLUME SERIAL NUMBER OF
*                                OPTICAL VOLUME OR TAPE VOLUME
*                                FROM WHICH THE PRIMARY OBJECT
*                                WAS READ.
*                                ONLY VALID IF THE BIT 1 OR 2
*                                IS ON IN FIELD ST39FLGS          @L5A
ST39SMT  DS     CL2' '          SOURCE MEDIA TYPE OF THE VOLUME
*                                FROM WHICH THE PRIMARY OBJECT
*                                WAS READ.
*                               ONLY VALID IF THE BIT 1 OR 2
*                                IS ON IN FIELD ST39FLGS          @L5A
ST39TVSN DS     CL6' '          TARGET VOLUME SERIAL NUMBER OF
*                                OPTICAL VOLUME OR TAPE VOLUME
*                                ON WHICH THE BACKUP COPY OF
*                                THE OBJECT WAS WRITTEN.
*                                ONLY VALID IF THE BIT 1 OR 2
*                                IS ON IN FIELD ST39FLGS          @L5A
ST39TMT  DS     CL2' '          TARGET MEDIA TYPE OF THE VOLUME
*                                ON WHICH THE BACKUP COPY OF
*                                THE OBJECT WAS WRITTEN.
*                                ONLY VALID IF THE BIT 1 OR 2
*                                IS ON IN FIELD ST39FLGS          @L5A
ST39BTKN DS     BL4'0'         VOLUME LOCATION TOKEN ASSOCIATED
*                                WITH THE COPY OF THE OBJECT
*                                ON THE VOLUME SPECIFIED IN THE
*                                ST39TVSN FIELD.                  @L5A
```

Figure A-27 SMF type 85 subtypes 39 significant fields from CBRSMF macro (1 of 2)

```
ST39FLGS DS     BL4'0'          PROCESSING FLAGS                    @L5A
ST39FLG0 EQU    X'80'           WHEN ON, THE PRIMARY COPY
*                                 IS STORED TO DISK.                @L9C
ST39FLG1 EQU    X'40'           WHEN ON, THE PRIMARY COPY
*                                 IS STORED TO OPTICAL.             @L5A
ST39FLG2 EQU    X'20'           WHEN ON, THE PRIMARY COPY
*                                 IS STORED TO TAPE.                @L5A
ST39FLG3 EQU    X'10'           RESERVED                            @L5A
ST39FLG4 EQU    X'08'           RESERVED                            @L5A
ST39FLG5 EQU    X'04'           WHEN ON, THE BACKUP COPY
*                                 IS STORED TO OPTICAL.             @L5A
ST39FLG6 EQU    X'02'           WHEN ON, THE BACKUP COPY
*                                 IS STORED TO TAPE.                @L5A
ST39FLG7 EQU    X'01'           RESERVED                            @L5A
ST39FLG8 EQU    X'80'           WHEN ON, WRITE TO BACKUP COPY
*                                 WAS SUCCESSFUL.                   @L5A
ST39FLG9 EQU    X'00'           RESERVED                            @L9A
ST39FLG10 EQU   X'20'           When on, primary is stored on SL1  @L9A
ST39FLG11 EQU   X'10'           When on, primary is stored on SL2  @L9A
```

Figure A-28   SMF type 85 subtypes 39 significant fields from CBRSMF macro (2 of 2)

A.2.6 Program SMF85TJ SMF record type 85 subtype 40

OAM writes SMF Record type 85 subtypes 40 to document the output that results from the execution of OSMC COMMAND RECYCLE macros used in batch jobs. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TJ that scans the SMF type 85 subtype 40 records and formats the records of activity. The program itself and how to construct it is documented in A.3.7, “SMF Record type 85 subtype 40 data display program SMF85TJ” on page 299

Figure A-29 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1     <-- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(40)))
/*
//  EXEC PGM=SMF85TJ
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TJ.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-29 SMF85TJ SMF data listing JCL

Figure A-31 on page 246 shows the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 40. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST39.

Figure A-30 shows selected lines of output from running program SMF85TJ if available.

```
No relevant OAM activity occurred during the test time frame
```

Figure A-30 Selected output examples from running the SMF85TJ program

Figure A-31 shows the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 40.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPE 40                               *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------       *
*        40 OSMC COMMAND RECYCLE                                 @L5A *
*                                                                     *
***********************************************************************
ST40     DSECT                  SUBTYPE 40
ST40STRD DS     CL10' '         DATE RECYCLE COMMAND STARTED     @L5A
ST40ENDD DS     CL10' '         DATE RECYCLE COMMAND ENDED       @L5A
ST40VOLN DS     BL2'0'          NUMBER OF VOLSERS COMPLETED      @L5A
ST40PCTV DS     BL2'0'          PERCENT VALID USED FOR COMMAND   @L5A
ST40LIM  DS     BL2'0'          LIMIT USED FOR COMMAND           @L5A
ST40SUBL DS     CL1' '          TAPE SUBLEVEL USED FOR COMMAND:
*                               A=ALL,
*                               -=TSL NOT SPECIFIED ON CMD,
*                               1=TAPE SUBLEVEL 1,
*                               2=TAPE SUBLEVEL 2.
*                                                                @L6A
         DS     BL1'0'          RESERVED                         @L6C
ST40END  DS     0C              END OF BASE SECTION              @L5A
***********************************************************************
* SUBTYPE 40 VOLUME ARRAY SECTION                                     *
***********************************************************************
ST40VOLD DSECT                  ARRAY OF VOLS COMPLETED RECYCLE @L5A
ST40VSN  DS     40CL6' '        VOLSER                          @L5A
```

Figure A-31 SMF type 85 subtypes 40 significant fields from CBRSMF macro (1 of 1)

A.2.7 Program SMF85TP SMF record type 85 subtype 90-93

OAM writes SMF Record type 85 subtypes 90-93 to document the output that results from the execution of LCS function with File Systems. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TP that scans the SMF type 85 subtypes 90-93 records and formats the records of activity. The program itself and how to construct it is documented in A.3.8, “SMF Record type 85 subtypes 90-93 data display program SMF85TP” on page 304.

Figure A-32 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

If you do not want output from all the types that the program can process, change the SMF selection statement to include only those that you want. For example, change `OUTDD(OUTDD,TYPE(85(90:93))) to` OUTDD(OUTDD,TYPE(85(92,93))) to exclude subtypes 90 and 91.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1         <-- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(90:93)))
/*
//  EXEC PGM=SMF85TP
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TP.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-32 SMF85TP SMF data listing JCL

Figure A-33 shows selected lines of output from running program SMF85TP. This example shows activity that occurred managing an example sent to a zFS file system. These SMF records are new with DFSMS V1.13, and relate only to zFS storage groups.

In this case, the instance shown on line `OBJN/FST/INST/FLGS` is the same as when the object was first stored, *06070E0B.*

```
SMF85TP SMFDTE/TME:            2011286/21:26:09.035
STYPE/FUNC/DIR/:              90/(LCS FILE SYSTEM WRITE REQUEST)   //myoam/sg3
SGN/COLN/:                    SG3      ANDRECS.SG3.OAMTES9
OBJN/FST/INST/FLGS:           MHLRES2.SMF85TP.PDS.OAMTES9              /ZFS  06070E0B 40000000
OLEN/OOFF/LIQT/LDQT/LEQT/RC/RS: 00008A80 00000000 00000000 00000000 000001DF 00000000 00000000

SMF85TP SMFDTE/TME:            2011286/21:26:14.055
STYPE/FUNC/DIR/:              91/(LCS FILE SYSTEM READ REQUEST)    //myoam/sg3
SGN/COLN/:                    SG3      ANDRECS.SG3.OAMTES9
OBJN/FST/INST/FLGS:           MHLRES2.SMF85TP.PDS.OAMTES9              /ZFS  06070E0B 40000000
OLEN/OOFF/LIQT/LDQT/LEQT/RC/RS: 00008A80 00000000 00000000 00000000 0000000E 00000000 00000000
```

Figure A-33 Selected output examples from running the SMF85TP program

Figure A-34 shows the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 90-93. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST39.

DFSMS V1R13 introduced SMF type 85 subtypes 90-93 to track what is happening in the File Systems so all fields are new.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPE 90,91,92,93                      *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------      *
*        90 LCS File System Write Request                             *
*        91 LCS File System Read  Request                             *
*        92 LCS File System Physical Delete Request                   *
*        93 LCS File System Physical Delete Request for uncommitted   *
*           Store cleanup                                             *
***********************************************************************
ST90     DSECT                  SUBTYPE 90                       @L9A
ST90DIR  DS     CL30' '         File system directory            @L9A
ST90SGN  DS     CL8' '          OBJECT storage group name        @L9A
ST90COLN DS     CL44' '         Collection name                  @L9A
ST90OBJN DS     CL44' '         Object name                      @L9A
ST90FST  DS     CL2' '          File system type                 @L9A
*                                 Only valid for subtypes 90 and 91
ST90INST DS     BL4'0'          Instance ID                      @L9A
ST90FLGS DS     BL4'0'          Processing Flags                 @L9A
ST90FLG0 EQU    X'80'           When on, request originated within@L9A
*                               the OAM address space
ST90FLG1 EQU    X'40'           When on, request originated      @L9A
*                               outside of the OAM address space
ST90OLEN DS     BL4'0'          Object length                    @L9A
*                                 Only valid for subtypes 90 and 91
ST90OOFF DS     BL4'0'          Object offset                    @L9A
*                                 Valid for a subtype 91 partial read
ST90LIQT DS     BL4'0'          LCS input-work-queue time        @PBC
*                                (in milliseconds, only for 90 and 91)
ST90LDQT DS     BL4'0'          LCS dispatcher-queue time        @PBC
*                                (in milliseconds, only for 90 and 91)
ST90LEQT DS     BL4'0'          LCS execution-queue time         @PBC
*                                (in milliseconds, only for 90 and 91)
ST90RC   DS     BL4'0'          LCS Return code                  @L9A
ST90RS   DS     BL4'0'          LCS Reason code                  @L9A
```

Figure A-34 SMF type 85 subtypes 90-93 significant fields from CBRSMF macro (1 of 1)

A.2.8 Program SMF85TR SMF record type 85 subtype 80-81

OAM writes SMF Record type 85 subtypes 80-81 to document the output that results from the execution of LCS LIBRARY VARY ONLINE or OFFLINE. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TR that scans the SMF type 85 subtypes 80-81 records and formats the records of activity. The program itself and how to construct it is documented in A.3.9, "SMF Record type 85 subtypes 80-81 data display program SMF85TR" on page 310/

Figure A-35 shows the JCL to extract the SMF records and run the program.

If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

If you do not want output from all the types that the program can process, change the SMF selection statement to include only those you want. For example, change `OUTDD(OUTDD,TYPE(85(80-81))) to` OUTDD(OUTDD,TYPE(85(81))) to exclude subtype 80.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1     <-- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(80:81)))
/*
//  EXEC PGM=SMF85TR
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TR.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-35 SMF85TR SMF data listing JCL

Figure A-37 on page 250 shows the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 80-81. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST80.

Figure A-36 shows selected lines of output from running program SMF85TR if available.

```
No appropriate OAM activity occurred during the test time frame
```

Figure A-36 Selected output examples from running the SMF85TR program

Figure A-37 on page 250 shows the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 40.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPES 80 - 81                         *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------       *
*        80 LCS TAPE LIBRARY VARY ONLINE                              *
*        81 LCS TAPE LIBRARY VARY OFFLINE                             *
*                                                                     *
***********************************************************************
ST80     DSECT                  SUBTYPES 80 AND 81
ST80TLN  DS     CL8' '          TAPE LIBRARY NAME
ST80TLDT DS     CL8' '          TAPE LIBRARY DEVICE TYPE
ST80LTQT DS     BL4'0'          LCS LIBRARY TASK QUEUE TIME.
*                               THE AMOUNT OF TIME (IN
*                               MILLISECONDS) THAT THIS REQUEST
*                               SPENT ON THE LCS LIBRARY QUEUE
*                               (LQE QUEUE) WAITING TO BE
*                               PROCESSED.
ST80LTPT DS     BL4'0'          LCS LIBRARY TASK PROCESSING TIME.
*                               THE AMOUNT OF TIME (IN
*                               MILLISECONDS) THAT THIS REQUEST
*                               TOOK TO BE PROCESSED BY THE
*                               LIBRARY TASK.
ST80RC   DS     BL4'0'          LCS RETURN CODE
ST80RS   DS     BL4'0'          LCS REASON CODE
ST80FLGS DS     BL4'0'          PROCESSING FLAGS, RESERVED.
```

Figure A-37 SMF type 85 subtypes 80-81 significant fields from CBRSMF macro (1 of 1)

A.2.9 Program SMF85TS SMF record type 85 subtype 87

OAM writes SMF Record type 85 subtype 87 to document the output that results from the execution of OAM OWNED TAPE VOLUME DEMOUNT. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TS that scans the SMF type 85 subtype 87 records and formats the records of activity. The program itself and how to construct it is documented in A.3.10, “SMF Record type 85 subtype 87 data display program SMF85TS” on page 315.

Figure A-38 on page 251 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1      <- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
          INDD(DUMPIN,OPTIONS(DUMP))
          OUTDD(OUTDD,TYPE(85(87)))
/*
//  EXEC PGM=SMF85TS
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TS.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-38 SMF85TS SMF data listing JCL

Figure A-40 on page 252 through Figure A-41 on page 253 show the current content of the CRBSMF macro (assembler portion). The figures show how it relates to SMF Record type 85 subtypes 87. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST87.

One of the significant changes introduced with V1R13 is a different way of recording some data. Some fields, for example ST87NKBW, were defined as a binary length 4 field that contains a number in Kbytes. This process is imprecise, so new fields are created, such as ST87NBW (the replacement for ST87NKBW) defined as binary length 8 that contains a number in bytes. Now both ST87NKBW and ST87NBW are provided. If ST87NKBW overflows, it would be set to X'FFFFFFFF'. The CRBSMF details show which fields expressed in KB are duplicated with the version expressed in bytes.

Figure A-39 on page 252 shows selected lines of output from running program SMF85TS. This record shows the details that concern tape volume named THS025. By reference to the description of the fields in the CRBSMF macro, you can see the following messages:

- NKBW (NUMBER OF KILOBYTES OF OBJECT DATA WRITTEN TO THIS TAPE VOLUME WHILE IT WAS MOUNTED. SUPERSEDED BY ST87NBW) is X'23'(decimal 35)
- NBW(NUMBER OF BYTES OF OBJECT DATA WRITTEN TO THIS TAPE VOLUME WHILE IT WAS MOUNTED.) is X'8A80' (decimal 35456)

These values show that the metrics are consistent. If a problem arose with a previous WRITE to tape, these values probably would not be consistent.

```
SMF TYPE 85 SUBTYPE 87 RECORDS
SMF85TS SMFDTE/TME:          2011286/22:44:53.023
STYPE/FUNC:                87/(OAM OWNED TAPE VOLUME DEMOUNT)
TDDN/TDDT/TVUN/VSN/TMT/TVT:   0B23/00000000 /3590-1  /THS025/07/B
SGN/RC/RS/FLGS:             TOAMBK1 /00000000 /00000000 /00000000
TMNT/NOW/NKBW/NOR/NKBR/NBW/NBR: 00017BCA /00000001 /00000023 /00000000 /00000000
/0000000000008A80 ...
```

Figure A-39 Selected output examples from running the SMF85TS program

Figure A-40 through Figure A-41 on page 253 show the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtype 87.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPE 87                               *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------       *
*        87 OAM OWNED TAPE VOLUME DEMOUNT                             *
*                                                                     *
***********************************************************************
ST87     DSECT                  SUBTYPE 87
ST87TDDN DS     CL4' '          MVS DEVICE NUMBER CORRESPONDING
*                               TO TAPE DRIVE.
ST87TDDT DS     CL4' '          TAPE DRIVE UCB DEVICE TYPE
ST87TVUN DS     CL8' '          TAPE VOLUME UNIT NAME
ST87VSN  DS     CL6' '          TAPE VOLUME SERIAL NUMBER
ST87TMT  DS     CL2' '          TAPE MEDIA TYPE
*                               02 = IBM 3480 STANDARD CAPACITY
*                                    CARTRIDGE SYSTEM TAPE.
*                               04 = IBM 3480 ENHANCED CAPACITY
*                                    CARTRIDGE SYSTEM TAPE.
*                               05 = IBM 3590 HIGH PERFORMANCE
*                                    CARTRIDGE TAPE.
ST87TVT  DS     CL1' '          TAPE VOLUME TYPE.
*                               G = TAPE VOLUME IS A GROUPED
*                               VOLUME BELONGING TO AN OBJECT
*                               STORAGE GROUP.
*                               B = TAPE VOLUME IS A GROUPED
*                               VOLUME BELONGING TO AN OBJECT
*                               BACKUP STORAGE GROUP.
         DS     CL3' '          RESERVED.
ST87SGN  DS     CL8' '          STORAGE GROUP NAME.
ST87RC   DS     BL4'0'          LCS RETURN CODE
ST87RS   DS     BL4'0'          LCS REASON CODE
```

Figure A-40 SMF type 85 subtypes 87 significant fields from CBRSMF macro (1 of 2)

```
**********************************************************************
* SUBTYPE 87 PROCESSING FLAGS                                        *
**********************************************************************
ST87FLGS DS     BL4'0'          PROCESSING FLAGS.
ST87FLG0 EQU    X'80'           THIS REQUEST WAS PROCESSED
*                               USING A TAPE VOLUME ASSOCIATED
*                               WITH SUBLEVEL1           @L6A
ST87FLG1 EQU    X'40'           THIS REQUEST WAS PROCESSED
*                               USING A TAPE VOLUME ASSOCIATED
*                               WITH SUBLEVEL2           @L6A
ST87TMNT DS     BL4'0'          ELAPSED TIME (IN MILLISECONDS) @P5C
*                               THAT TAPE VOLUME WAS MOUNTED,
*                               MEASURED FROM COMPLETION OF THE
*                               DFSMS OPEN MACRO TO COMPLETION
*                               OF THE DFSMS CLOSE MACRO.
ST87NOW  DS     BL4'0'          NUMBER OF OBJECT WRITTEN TO THIS
*                               TAPE VOLUME WHILE IT
*                               WAS MOUNTED.
ST87NKBW DS     BL4'0'          NUMBER OF KILOBYTES OF OBJECT
*                               DATA WRITTEN TO THIS TAPE
*                               VOLUME WHILE IT WAS MOUNTED.
*                               SUPERSEDED BY ST87NBW.
*                               SEE NOTE 1.                    @LAC
ST87NOR  DS     BL4'0'          NUMBER OF OBJECT READ FROM THIS
*                               TAPE VOLUME WHILE IT
*                               WAS MOUNTED.
ST87NKBR DS     BL4'0'          NUMBER OF KILOBYTES OF OBJECT
*                               DATA READ FROM THIS TAPE
*                               VOLUME WHILE IT WAS MOUNTED.
*                               SUPERSEDED BY ST87NBR.
*                               SEE NOTE 1.                    @LAC
         DS     BL4'0'          RESERVED                       @LAA
ST87NBW  DS     BL8'0'          NUMBER OF BYTES OF OBJECT
*                               DATA WRITTEN TO THIS TAPE
*                               VOLUME WHILE IT WAS MOUNTED.   @LAA
ST87NBR  DS     BL8'0'          NUMBER OF BYTES OF OBJECT
*                               DATA READ FROM THIS TAPE
*                               VOLUME WHILE IT WAS MOUNTED.   @LAA
**********************************************************************
*                                                                    *
* NOTE 1: 4 BYTE FIELDS REPORTING KILOBYTE VALUES HAVE BEEN          *
*         SUPERSEDED BY CORRESPONDING 8 BYTE FIELDS REPORTING VALUES *
*         IN BYTES. THE EXISTING 4 BYTE FIELDS WILL CONTAIN          *
*         X'FFFF FFFF' IF AN OVERFLOW CONDITION OCCURS.              *
```

Figure A-41   SMF type 85 subtypes 87 significant fields from CBRSMF macro (2 of 2)

A.2.10 Program SMF85TU SMF record type 85 subtype 82-86

OAM writes SMF Record type 85 subtype 82-86 to document the output that results from the execution of LCS TAPE LIBRARY VOLUME activities. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TU that scans the SMF type 85 subtypes 82-86 records and formats the records of activity. The program itself and how to construct it is documented in A.3.11, “SMF Record type 85 subtype 82-86 data display program SMF85TU” on page 321.

Figure A-42 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

If you do not want output from all the types that the program can process, change the SMF selection statement to include only those you want. For example, change `OUTDD(OUTDD,TYPE(85(82-86))) to` OUTDD(OUTDD,TYPE(85(83,84))) to exclude subtypes 82, 85, and 86.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1     <-- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
          INDD(DUMPIN,OPTIONS(DUMP))
          OUTDD(OUTDD,TYPE(85(82:86)))
/*
//  EXEC PGM=SMF85TU
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TU.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-42 SMF85TU SMF data listing JCL

Figure A-43 shows selected lines of output from running program SMF85TU if available.

```
No relevant OAM activity occurred during the test time frame
```

Figure A-43 Selected output example from running the SMF85TU program

Figure A-44 through Figure A-45 on page 256 show the current content of the CRBSMF macro (assembler portion) as it relates to SMF Record type 85 subtypes 82-86.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. This example needs fields prefixed with ST87.

```
 ***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPES 82 - 86                         *
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- ---------------------------------------------------       *
*        82 LCS TAPE LIBRARY VOLUME ENTRY                             *
*        83 LCS TAPE LIBRARY VOLUME EJECT                             *
*        84 LCS TAPE LIBRARY VOLUME AUDIT                             *
*        85 LCS TAPE LIBRARY VOLUME MOUNT                             *
*        86 LCS TAPE LIBRARY VOLUME DEMOUNT                           *
*                                                                     *
***********************************************************************
ST82     DSECT                    SUBTYPES 82 - 86
ST82TLN  DS     CL8' '            TAPE LIBRARY NAME
ST82TLDT DS     CL8' '            TAPE LIBRARY DEVICE TYPE
ST82TDDT DS     CL4' '            TAPE DRIVE UCB DEVICE TYPE.
*                                 VALID FOR SUBTYPES 85 AND 86,
*                                 CONTAINS ZEROS FOR OTHERS.
ST82TDDN DS     CL4' '            MVS DEVICE NUMBER CORRESPONDING
*                                 TO TAPE DRIVE.
*                                 VALID FOR SUBTYPES 85 AND 86,
*                                 CONTAINS ZEROS FOR OTHERS.
ST82VSN  DS     CL6' '            TAPE VOLUME SERIAL NUMBER
ST82TMT  DS     CL2' '            TAPE MEDIA TYPE
*                                 02 = IBM 3480 STANDARD CAPACITY
*                                      CARTRIDGE SYSTEM TAPE.
*                                 04 = IBM 3480 ENHANCED CAPACITY
*                                      CARTRIDGE SYSTEM TAPE.
*                                 05 = IBM 3590 HIGH PERFORMANCE
*                                      CARTRIDGE TAPE.
ST82LIQT DS     BL4'0'            AMOUNT OF TIME (IN MILLISECONDS)
*                                 THIS REQUEST SPENT ON THE LCS
*                                 INPUT-WORK-QUEUE WAITING TO
*                                 PROCESSED.
ST82LDQT DS     BL4'0'            AMOUNT OF TIME (IN MILLISECONDS)
*                                 THIS REQUEST SPENT ON THE LCS
*                                 DISPATCHER-QUEUE WAITING TO
*                                 PROCESSED.
```

Figure A-44 SMF type 85 subtypes 82-86 significant fields from CBRSMF macro (1 of 2)

```
ST82LTQT DS     BL4'0'           LIBRARY TASK QUEUE TIME.
*                                AMOUNT OF TIME (IN MILLISECONDS)
*                                THIS REQUEST SPENT ON THE LCS
*                                LIBRARY TASK QUEUE (LQE QUEUE)
*                                WAITING TO BE PROCESSED.
*                                NORMALLY, THIS FIELD REPRESENTS
*                                CARTRIDGE TRANSPORT MECHANISM
*                                WAIT TIME.  THAT IS, THE AMOUNT
*                                OF TIME WAITING FOR THE CARTRIDGE
*                                TRANSPORT MECHANISM (ROBOT) TO
*                                BECOME AVAILABLE WITHIN AN
*                                AUTOMATED STORAGE LIBRARY.

ST82LTPT DS     BL4'0'           LIBRARY TASK PROCESSING TIME.
*                                AMOUNT OF TIME (IN MILLISECONDS)
*                                THIS REQUEST TOOK TO BE
*                                PROCESSED BY THE LIBRARY TASK.
*                                WAITING TO BE PROCESSED.
*                                NORMALLY, THIS FIELD REPRESENTS
*                                CARTRIDGES TRANSPORT MECHANISM
*                                SERVICE TIME.  THAT IS, THE
*                                TIM SPENT BY THE
*                                CARTRIDGE TRANSPORT MECHANISM
*                                (ROBOT) TO MOVE CARTRIDGES WITHIN
*                                THE OPTICAL DISK LIBRARY.
ST82RC   DS     BL4'0'           LCS RETURN CODE
ST82RS   DS     BL4'0'           LCS REASON CODE
ST82FLGS DS     BL4'0'           PROCESSING FLAGS, RESERVED.
```

Figure A-45 SMF type 85 subtypes 82-86 significant fields from CBRSMF macro (2 of 2)

A.2.11 Program SMF85TW SMF record type 85 subtype 78, 79, 88

OAM writes SMF Record type 85 subtype 87 to document the output that results from the execution of LCS TAPE LIBRARY VOLUME activities. The functions that generate the SMF records are documented in Table A-1 on page 214.

This book includes a simple program called SMF85TW that scans the SMF type 85 subtypes 78, 79, 88 records and formats the records of activity. The program itself and how to construct it is documented in A.3.12, “SMF Record type 85 subtypes 78,79,88 data display program SMF85TW” on page 326

Figure A-46 shows the JCL to extract the SMF records and run the program. If this job is obtained by copy and paste from the PDF manual, the leading blanks on the lines that begin with INDD and OUTDD are not copied. You must insert at least one blank at the start of these lines.

If you do not want output from all the types that the program can process, change the SMF selection statement to include only those you want. For example, change `OUTDD(OUTDD,TYPE(85(78,79,88)))` to OUTDD(OUTDD,TYPE(85(79,88))) to exclude subtypes 78.

```
//MHLRES2S JOB (999,POK),MSGLEVEL=1,NOTIFY=MHLRES2
// EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//DUMPIN DD DISP=SHR,DSN=SYS1.SC64.MAN1       <-- change as necessary
//OUTDD  DD DSN=&SMFT85,SPACE=(CYL,(10,5)),RECFM=VBS,LRECL=32760,
// DISP=(,PASS,DELETE),UNIT=SYSDA
//SYSIN DD *
           INDD(DUMPIN,OPTIONS(DUMP))
           OUTDD(OUTDD,TYPE(85(78,79,88)))
/*
//  EXEC PGM=SMF85TW
//STEPLIB DD DISP=SHR,DSN=MHLRES2.SMF85TW.DFSMS13.LOAD
//SYSUDUMP DD SYSOUT=A
//SMFIN   DD DISP=SHR,DCB=BFTEK=A,DSN=&SMFT85
//PRINT   DD SYSOUT=A,RECFM=UA
```

Figure A-46 SMF85TW SMF data listing JCL

Figure A-48 on page 259 through Figure A-51 on page 262 show the current content of the CRBSMF macro (assembler portion). The content is shown as it relates to SMF Record type 85 subtypes 78, 79, and 88. The macro definitions are needed to interpret the program output.

The field meanings are documented in the CBRSMF macro, which has two forms of the macro. One is used by the assembler, and the other by the compiler when generating modules that create SMF type 85 records. Although the field names are the same in both sections, in some cases there is more explanation on one than the other. In this case you need fields prefixed with ST78, and if differences are required for the other subtypes, ST79 and ST88.

Figure A-47 shows selected lines of output from running program SMF85TU. This record shows that the details of an object named MHLRES2.SMF85TP.PDS.OAMTES9 as it is written to tape. By reference to the description of the fields in the CRBSMF macro, you can see the following messages:

- NOBJ (TOTAL NUMBER OF OBJECTS PROCESSED IN THIS REQUEST) is 1
- SOBJ (TOTAL NUMBER OF OBJECTS IN THIS REQUEST PROCESSED SUCCESSFULLY)
- NKBP (NUMBER OF KILOBYTES OF OBJECT DATA IN THIS REQUEST) is X'23'(decimal 35)
- SKBP (NUMBER OF KILOBYTES OF OBJECT DATA SUCCESSFULLY PROCESSED IN THIS REQUEST) and is also X'23'(decimal 35). These values are in KB.
- OLEN is X'8A80' (decimal 35456)

These values show that the metrics are consistent. If a problem arises with the WRITE to tape, these values probably would not be consistent with each other.

```
SMF85TW SMFDTE/TME:            2011286/22:43:16.003
STYPE/FUNC/ORMN/OTMN:          78/(LCS TAPE WRITE REQUEST     )   /            /
TDUN/TDDN/TVT/SGN/LIQT/LDQT/LEQT: 3590-1 /0B23/B/TOAMBK1 /00000000 /00000005 /00006780
TXQT/LMAT/LMDT                00000000 /00000027 /00000000
LDCT/LDOT/LDPT/LBRT/LBWT/LBCT:  00000000 /00005934 /00000000 /00000000 /00000000 /00000000
FLGS/NOBJ/NKBP/SOBJ/SKBP:      50000000 /00000001 /00000023 /00000001 /00000023
COLN/OBJN/OLEN:                   ANDRECS.SG3.OAMTES9                 MHLRES2.SMF85TP.PDS.OAMTES9
00008A80
OOFF/VSN/TMT/OTKN/RC/RS:          00000000 /THS025/07/00000006 /00000000 /00000000
```

Figure A-47 Selected output examples from running the SMF85TW program

Figure A-48 through Figure A-51 on page 262 show the current content of the CRBSMF macro (assembler portion). The content is shown as it relates to SMF Record type 85 subtype 78, 79, and 88.

```
***********************************************************************
*                                                                     *
*   OAM SUBTYPE SECTION, FOR SUBTYPES 78, 79 AND 88              @P0C*
*                                                                     *
*   SUBTYPE FUNCTION                                                  *
*   ------- --------------------------------------------------        *
*        78 LCS TAPE WRITE REQUEST                                    *
*        79 LCS TAPE READ REQUEST                                @L1A*
*        88 LCS TAPE LOGICAL DELETE REQUEST                      @P0A*
*                                                                     *
***********************************************************************
ST78     DSECT                    SUBTYPES 78, 79 AND 88
ST78ORMN DS     CL16' '           OAM REQUEST MEMBER NAME        @L1A
ST78OTMN DS     CL16' '           OAM TARGET  MEMBER NAME        @L1A
ST78TDUN DS     CL8' '            TAPE DRIVE UNIT NAME
ST78TDDN DS     CL4' '            MVS DEVICE NUMBER OF TAPE DRIV
ST78TVT  DS     CL1' '            TAPE VOLUME TYPE.
*                                 G = OBJECT STORAGE GROUP,
*                                 B = OBJECT BACKUP STOR GROUP,
*                                 S = SCRATCH OPTICAL VOLUME.
         DS     CL3' '            RESERVED
ST78SGN  DS     CL8' '            STORAGE GROUP NAME.
ST78LIQT DS     BL4'0'            LCS INPUT-WORK-QUEUE TIME.
*                                 AMOUNT OF TIME (IN MILLISECONDS)
*                                 THIS REQUEST SPENT ON THE LCS
*                                 INPUT-WORK-QUEUE WAITING TO
*                                 PROCESSED.
ST78LDQT DS     BL4'0'            LCS DISPATCHER-QUEUE TIME.
*                                 AMOUNT OF TIME (IN MILLISECONDS)
*                                 THIS REQUEST SPENT ON THE LCS
*                                 DISPATCHER-QUEUE WAITING TO
*                                 PROCESSED.
ST78LEQT DS     BL4'0'            LCS EXECUTION-QUEUE TIME.
*                                 AMOUNT OF TIME (IN MILLISECONDS)
*                                 THIS REQUEST SPENT ON THE LCS
*                                 EXECUTION-QUEUE BEING
*                                 PROCESSED.
ST78LXQT DS     BL4'0'            AMOUNT OF TIME (IN
*                                 MILLISECONDS) THIS REQUEST
*                                 SPENT ON THE XCF CROSS SYSTEM-
*                                 QUEUE BEING PROCESSED.        @L1A
ST78LMAT DS     BL4'0'            MVS DYNAMIC ALLOCATION (SVC 99)
*                                 TIME.  THIS IS THE AMOUNT OF
*                                 TIME (IN MILLISECONDS) THAT WAS
*                                 REQUIRED BY MVS DYNAMIC
*                                 ALLOCATION TO DYNAMICALLY
*                                 ALLOCATE THE TAPE DRIVE.  THIS
*                                 FIELD IS ONLY VALID IF BIT 1 IN
*                                 FIELD ST78FLGS IS ON.
```

Figure A-48 SMF type 85 subtypes 78, 79, 88 significant fields from CBRSMF macro (1 of 4)

```
ST78LMDT DS     BL4'0'          MVS DYNAMIC DEALLOCATION (SVC 99)
*                               TIME.  THIS IS THE AMOUNT OF
*                               TIME (IN MILLISECONDS) THAT WAS
*                               REQUIRED BY MVS DYNAMIC
*                               DEALOCATION TO DYNAMICALLY
*                               DEALLOCATE THE TAPE DRIVE.  THIS
*                               FIELD IS ONLY VALID IF BIT 2 IN
*                               FIELD ST78FLGS IS ON.
ST78LDCT DS     BL4'0'          DFSMS CLOSE TIME.
*                               THIS IS THE AMOUNT OF
*                               TIME (IN MILLISECONDS) THAT WAS
*                               REQUIRED BY DFSMS TO CLOSE AN
*                               ALREADY-OPENED TAPE DATA SET.
*                               THIS FIELD IS ONLY VALID IF BIT 2
*                               IN FIELD ST78FLGS IS ON.
ST78LDOT DS     BL4'0'          DFSMS OPEN TIME.
*                               THIS IS THE AMOUNT OF
*                               TIME (IN MILLISECONDS) THAT WAS
*                               REQUIRED BY DFSMS OPEN PROCESSING
*                               TO OPEN THE TAPE DATA SET.
*                               THIS FIELD IS ONLY VALID IF BIT 1
*                               OR BIT 2 IN FIELD ST78FLGS IS ON.
ST78LDPT DS     BL4'0'          DFSMS POINT TIME.  THIS IS
*                               THE AMOUNT OF TIME (IN
*                               MILLISECONDS) THAT WAS
*                               REQUIRED BY DFSMS POINT
*                               PROCESSING TO POSITION TO THE
*                               CORRECT BLOCK-ID ON THE MEDIA.
*                               THIS FIELD
*                               IS ONLY VALID IF BIT 1 OR
*                               BIT 2 IN FIELD ST78FLGS IS
*                               ON.
ST78LBRT DS     BL4'0'          BSAM READ TIME.  THIS IS
*                               THE AMOUNT OF TIME (IN
*                               MILLISECONDS) THAT LCS SPENT
*                               IN BSAM READ PROCESSING READING
*                               DATA FROM THE TAPE VOLUME.  ONLY
*                               VALID FOR SUBTYPE 79.
ST78LBWT DS     BL4'0'          BSAM WRITE TIME.  THIS IS
*                               THE AMOUNT OF TIME (IN
*                               MILLISECONDS) THAT LCS SPENT
*                               IN BSAM WRITE PROCESSING WRITING
*                               DATA TO THE TAPE VOLUME.  ONLY
*                               VALID FOR SUBTYPE 78.
ST78LBCT DS     BL4'0'          BSAM CHECK TIME.  THIS IS
*                               THE AMOUNT OF TIME (IN
*                               MILLISECONDS) THAT LCS SPENT
*                               IN BSAM CHECK PROCESSING WAITING
*                               FOR I/O OPERATIONS TO COMPLETED.
*                               VALID FOR SUBTYPES 78 AND 79.
```

Figure A-49 SMF type 85 subtypes 78, 79, 88 significant fields from CBRSMF macro (2 of 4)

```
***********************************************************************
* SUBTYPE 78 PROCESSING FLAGS                                         *
***********************************************************************
ST78FLGS DS     BL4'0'          PROCESSING FLAGS, RESERVED.
ST78FLG0 EQU    X'80'           WHEN ON, THIS REQUEST WAS
*                               PROCESSED USING A CURRENTLY
*                               MOUNTED TAPE VOLUME AND DID
*                               NOT REQUIRE AN UNMOUNTED TAPE
*                               VOLUME TO BE MOUNTED.           @P6C
ST78FLG1 EQU    X'40'           WHEN ON, THIS REQUEST REQUIRED
*                               AN UNMOUNTED TAPE VOLUME
*                               AND THE TAPE DRIVE USED
*                               TO PROCESS THIS REQUEST WAS
*                               EMPTY AT THE START OF PROCESSING
*                               THIS REQUEST.  THEREFORE, THIS
*                               REQUEST DID NOT REQUIRE A
*                               CURRENTLY MOUNTED TAPE
*                               VOLUME TO BE DEMOUNTED PRIOR
*                               TO MOUNTING THE REQUIRED TAPE
*                               VOLUME.
ST78FLG2 EQU    X'20'           WHEN ON, THIS REQUEST REQUIRED AN
*                               UNMOUNTED TAPE VOLUME TO BE
*                               MOUNTED AND THE TAPE DRIVE TASK
*                               USED TO PROCESS THIS REQUEST
*                               ALREADY HAD A TAPE DRIVE
*                               ALLOCATED AND A DIFFERENT TAPE
*                               VOLUME MOUNTED.  THEREFORE, THIS
*                               REQUEST REQUIRED A CURRENTLY
*                               MOUNTED TAPE VOLUME TO BE
*                               DEMOUNTED PRIOR TO MOUNTING THE
*                               REQUIRED TAPE VOLUME.
ST78FLG3 EQU    X'10'           THIS REQUEST WAS PROCESSED
*                               USING A TAPE DRIVE INSIDE AN
*                               IBM AUTOMATED TAPE LIBRARY
*                               DATASERVER.
ST78FLG4 EQU    X'08'           THIS REQUEST WAS PROCESSED
*                               USING A TAPE VOLUME ASSOCIATED
*                               WITH SUBLEVEL1            @L6A
ST78FLG5 EQU    X'04'           THIS REQUEST WAS PROCESSED
*                               USING A TAPE VOLUME ASSOCIATED
*                               WITH SUBLEVEL2            @L6A
```

Figure A-50   SMF type 85 subtypes 78, 79, 88 significant fields from CBRSMF macro (3 of 4)

```
ST78NOBJ DS     BL4'0'          TOTAL NUMBER OF OBJECTS
*                               PROCESSED IN THIS REQUEST
ST78NKBP DS     BL4'0'          NUMBER OF KILOBYTES OF OBJECT
*                               DATA IN THIS REQUEST
ST78SOBJ DS     BL4'0'          TOTAL NUMBER OF OBJECTS
*                               IN THIS REQUEST PROCESSED
*                               SUCCESSFULLY.                   @P5C
ST78SKBP DS     BL4'0'          NUMBER OF KILOBYTES OF OBJECT
*                               DATA SUCCESSFULLY               @P5C
*                               PROCESSED IN THIS REQUEST
         DS     CL16' '         RESERVED.
ST78END  DS     0C              END OF SUBTYPE 78 BASE SECTION
***********************************************************************
* SUBTYPE 78,79 & 88 OBJECT ARRAY SECTION                       @P0C*
***********************************************************************
ST78OBJD DSECT                  ARRAY OF OBJECTS IN THIS REQUEST
ST78COLN DS     CL44' '         COLLECTION NAME
ST78OBJN DS     CL44' '         OBJECT NAME
ST78OLEN DS     BL4'0'          OBJECT LENGTH
ST78OOFF DS     BL4'0'          OBJECT OFFSET, ONLY VALID
*                               ON SUBTYPE 79 FOR A
*                               PARTIAL OBJECT READ.
ST78VSN  DS     CL6' '          OPTICAL VOLUME SERIAL NUMBER
ST78TMT  DS     CL2' '          TAPE MEDIA TYPE.
*                               02 = IBM 3480 STANDARD CAPACITY
*                                    CARTRIDGE SYSTEM TAPE.
*                               04 = IBM 3480 ENHANCED CAPACITY
*                                    CARTRIDGE SYSTEM TAPE.
*                               05 = IBM 3590 HIGH PERFORMANCE
*                                    CARTRIDGE TAPE.
ST78OTKN DS     BL4'0'          OBJECT LOCATION TOKEN
ST78RC   DS     BL4'0'          LCS RETURN CODE
ST78RS   DS     BL4'0'          LCS REASON CODE
```

Figure A-51 SMF type 85 subtypes 78, 79, 88 significant fields from CBRSMF macro (4 of 4)

A.3 Building the SMF85 programs

Each sample program is documented separately so that they can be used individually as required. However, the process for building them is the same in each case.

The books are published in the proprietary PDF format. This format means that a simple copy from the book and paste into a workstation document does not always preserve blanks (spaces). These blanks might be necessary for correct use. If you use copy and paste and there are spaces that must be preserved, additional steps might be needed.

If you have a PDF to DOC (MS WORD) converter, the resulting document probably has fewer spaces than in the original document, but usually at least one. One is usually all that is required for the purposes of the source reconstruction.

If copy and paste from the book is used, certain spaces must be preserved. These spaces are documented as necessary. If the source files are retrieved from the ITSO FTP site, the blanks are in place as required.

A.3.1 OAM SMF85 analysis program common preparation steps

The process for building each of the supplied programs, and looking up the field references is the same.

- ► For more information about data set allocation, see A.3.1.2, "OAM SMF85 Analysis program common data set creation" on page 264.
- ► For more information about CRBSMF macro reference data, see A.3.1.3, "OAM CRBSMF macro reference data" on page 264.
- ► For more information about the sample JCL for building the programs, see A.3.1.4, "SMF85 program assembly and link" on page 265.
- ► (Optional) For more information about locate source materials on the ITSO FTP site, see A.3.1.1, "OAM SMF85 analysis source materials" on page 263

A.3.1.1 OAM SMF85 analysis source materials

This section describes how to locate, download, install, and use this material.

Locating the web material

1. The web material associated with this book is available in softcopy on the Internet from the IBM Redbooks web server at:

   `ftp://www.redbooks.ibm.com/redbooks/SG247961`

   Alternatively, you can go to the IBM Redbooks website at:

   ibm.com/redbooks

2. Select the **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG247961.

Using the web material

The additional web material that accompanies this book includes the following files:

File name	*Description*
SMF85JCL.zip	Compressed code samples for SMF type 85 record analysis

System requirements for downloading the web material

The web material requires the following system configuration:

Hard disk space: 1MB minimum
Operating System: Current Windows

Downloading and extracting the web material

Create a subdirectory (folder) on your workstation, and extract the contents of the web material compressed file into this folder.

When expanded into a directory on your workstation, you find files that contain JCL. Use these jobs to build each program. These jobs also run the resulting program and assembler source files that are referred to in the sections related to each program.

A.3.1.2 OAM SMF85 Analysis program common data set creation

Each of the sample sets needs a data set to store the source code and the JCL, and a data set to store the load module. These data sets can be created by any means that you prefer, but an example is provided in Figure A-52. If this job needs to be rerun, existing data sets must be deleted or renamed. This JCL is also available as member ALLOC from the ITSO FTP site.

```
//MHLRES2A JOB (1234567,COMMENT),MHLRES2,TIME=10,
// MSGLEVEL=1,CLASS=A,
// NOTIFY=&SYSUID
/*JOBPARM S=SC64
//ALLOCDS  PROC
//PDS      EXEC PGM=IEFBR14
//DDPDS    DD  DISP=(,CATLG),UNIT=SYSDA,
//             SPACE=(TRK,(10,2,5)),DCB=(RECFM=FB,LRECL=80),
//             DSN=&USER..&SET..DFSMS13.PDS
//LOAD     EXEC PGM=IEFBR14
//DDLOAD   DD  DISP=(,CATLG),UNIT=SYSDA,
//             SPACE=(TRK,(10,2,5)),DCB=(RECFM=U,BLKSIZE=32760),
//             DSN=&USER..&SET..DFSMS13.LOAD
//  PEND
// EXEC ALLOCDS,
//      USER=MHLRES2,      <---------- CHANGE AS REQUIRED
//      SET=SMF85TA        <---------- CHANGE TO THE PARTICULAR PROGRAM SET
```

Figure A-52 Sample JCL to create required data sets for SMF85 programs

A.3.1.3 OAM CRBSMF macro reference data

Locate member CBRSMF in SYS1.SAMPLIB on the running system by using BROWSE mode. This information is needed when the analysis programs are run and the output is being reviewed.

A.3.1.4 SMF85 program assembly and link

The same basic JCL is used to build each program. Copy and paste the JCL in Figure A-53 into the PDS data set for each program. The result should contain 27 lines.

Attention: At least one blank (space) must precede the SETSSI statement

This JCL is also available from the ITSO FTP site as documented in A.3.1.1, "OAM SMF85 analysis source materials" on page 263 as member BUILDJCL.

```
//MHLRES20 JOB (1234567,COMMENT),MHLRES2,TIME=10,MSGCLASS=J,
// MSGLEVEL=1,CLASS=A,NOTIFY=&SYSUID
/*JOBPARM S=*
//ASMHCL PROC
//ASM      EXEC PGM=ASMA90,REGION=0M,
//              PARM='OBJECT,NODECK'
//SYSLIN   DD  DSN=&&OBJ,DISP=(NEW,PASS),UNIT=SYSDA,
//              SPACE=(TRK,(10,2)),DCB=BLKSIZE=3120
//SYSLIB   DD  DISP=SHR,DSN=SYS1.MACLIB
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
//SYSIN DD DISP=SHR,DSN=&USER..&SET..DFSMS13.PDS(&SET.A)
//*
//LKED     EXEC PGM=HEWL,REGION=2048K,COND=(8,LE,ASM),
//         PARM='XREF,LIST,LET'
//SYSLIN   DD  DSN=&&OBJ,DISP=(OLD,DELETE)
//         DD  DDNAME=SYSIN
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(5,5))
//LKED.SYSLMOD  DD DISP=SHR,DSN=&USER..&SET..DFSMS13.LOAD(&SET.)
// PEND
//  EXEC ASMHCL,
//  USER=MHLRES2,   <----------- CHANGE AS REQUIRED
//  SET=SMF85TA     <----------- CHANGE TO THE PARTICULAR PROGRAM SET
/*
//LKED.SYSIN   DD *
  SETSSI 00001D00
```

Figure A-53 Sample JCL to assemble and link the SMF85 programs

A.3.1.5 SMF85 program restore process

The source code for these program samples is stored inline in the book, and also in the ITSO Additional Materials repository. For more information, see A.3.1.1, "OAM SMF85 analysis source materials" on page 263. For each program, the process is the same.

1. Copy and paste process

   These instructions relate to PCOMM setup. If another terminal emulator is used, the setup is similar.

   In the PCOMM configuration setup, define a window emulation with rows longer than 80, such as 27X132 or 62X160. It is not always possible to define 60X132, but this configuration is preferable because more rows of text can be pasted in each cycle.

   Set the TSO ISPF EDIT profile to NUMBER OFF and to reflect the FIXED-80 attribute, which is DATA (FIXED-80).

Copy a number of lines from this document and paste into the TSO/ISPF data set as many times as necessary.

2. FTP process

   See A.3.1.1, “OAM SMF85 analysis source materials” on page 263 for the overall process for retrieving the source files.

   After they are stored on a workstation, the text files need to be transferred to the mainframe TSO/ISPF session. Use the standard process for your work station. Building each program requires transfer of the source code for that program to the data sets.

A.3.1.6 SMF85 Program sample job to run all programs

On the ITSO FTP repository (but not in the inline code in this book) is a job in file RUNALL. This job runs all the SMF85 programs by using the same SMF data extract. This job can be used to get a picture of what happened in OAM in a specified time frame.

A.3.2 SMF Record type 85 subtype 1-10 data display program SMF85TA

Program SMF85TA displays the contents of selected fields of SMF record Type 85 subtypes 1/2/3/4/5/6/7/8/9/10 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that immediate backup is occurring.

There are two steps to build the program. The build is done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.2.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, “OAM SMF85 Analysis program common data set creation” on page 264. If the sample JCL is used, the SET to be specified is SMF85TA.

When USER=MHLRES2 and SET=SMF85TA, the following data sets result:

- MHLRES2.SMF85TA.DFSMS13.PDS
- MHLRES2.SMF85TA.DFSMS13.LOAD

A.3.2.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the Interactive System Productivity Facility (ISPF) session. To use the FTP process see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

> **Attention:** The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-3 on page 267 into member SMF85TAA of data set MHLRES2.SMF85TA.DFSMS13.PDS. The result should contain 367 lines.

> **Tip:** In the code in Example A-3 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-3 shows the source code for the SMF85TA program.

Example A-3 SMF85TA program source code

```
 MACRO                                                          00010000
&NAME    SEGSTART                                                       00020000
&NAME    STM   14,12,12(13)           SAVE HIS REGS IN HIS SAVE AREA    00030000
R0       EQU   0                                                        00040000
R1       EQU   1                                                        00050000
R2       EQU   2                                                        00060000
R3       EQU   3                                                        00070000
R4       EQU   4                                                        00080000
R5       EQU   5                                                        00090000
R6       EQU   6                                                        00100000
R7       EQU   7                                                        00110000
R8       EQU   8                                                        00120000
R9       EQU   9                                                        00130000
R10      EQU   10                                                       00140000
R11      EQU   11                                                       00150000
RB       EQU   12                                                       00160000
R13      EQU   13                                                       00170000
R14      EQU   14                                                       00180000
R15      EQU   15                                                       00190000
         BALR  12,0                   SET UP ADDRESSABILITY             00200000
         USING *,12                   USE REG 12 AS BASE REG            00210000
         ST    13,SAVEREGS+4          SAVE @ OF HIS SAVEAREA IN MINE    00220000
         LA    03,SAVEREGS            LOAD @ OF MY SAVE AREA IN REG 3   00230000
         ST    03,8(13)               SAVE @ OF MY SAVE AREA IN HIS     00240000
         LR    13,03                  LOAD @ OF MY SAVE AREA IN REG 13  00250000
         MEND                                                           00260000
         MACRO                                                          00270000
&NAME    SEGEND                                                         00280000
&NAME    L     13,SAVEREGS+4          LOAD REG13 WITH @ OF HIS SAVE     00290000
         LM    14,12,12(13)           RESTORE REGS FROM HIS SAVEAREA    00300000
         XR    R15,R15                                                  00310000
         BR    14                     RETURN TO CALLING RTN VIA REG 14  00320000
SAVEREGS DC    18F'0'                 SET UP SAVE AREA                  00330000
         MEND                                                           00340000
SMFR85TA  SEGSTART                                                      00350000
*  THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS PARTS OF 00360000
*  THE SMF TYPE 85 SUBTYPE 1-10 RECORDS.                                00370003
*  IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED        00380000
*  TO SELECT ANY OR ALL OF TYPE 85 SUBTYPES 1-10                        00390003
*  RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR                 00400000
*  OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.               00410000
*                                                                       00420000
*  THE STANDARD SMF RECORD MAPPING MACROS ARE USED.                     00430000
*  REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD                  00440000
*  R3    START OF WHOLE RECORD                                          00450000
*  THERE IS 1 DSECTS TO BE MAPPED                                       00460000
*  R4    START OF SUBTYPE RECORDS                                       00470000
*  R5    SPARE                                                          00480000
*  R6    SPARE                                                          00490000
*  R7    SPARE                                                          00500000
*  OTHER REGISTER USES                                                  00510000
*  R12   OVERALL BASE REGISTER                                          00520000
```

```
*  R8    RECORD TYPE/SUBTYPE CHECKING/WORKING                           00530000
*  R9    LENGTH OF PARTICULAR DSECT                                     00540000
*  R10   NUMBER OF ENTRIES IN THE TRIPLET                               00550000
*                                                                       00560000
*  QSAM GET LOCATE PROCESSING IS USED                                   00570000
*                                                                       00580000
         OPEN SMFIN                                                     00590000
         OPEN (PRINTDCB,(OUTPUT))                                       00600000
         PUT  PRINTDCB,PRINTHDR                                         00610000
READ     GET SMFIN                                                      00620000
* COPY PARAMETER POINTER                                                00630000
         LR    R3,R1                                                    00640000
* R3 -> SMF RECORD                                                      00650000
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION                         00660000
         USING CBRSMF85,R3                                              00670000
*   CHECK IF TYPE 85                                                    00680000
         CLI   SMF85RTY,X’55’                                           00690000
         BNE   IGNORE                                                   00700000
*        DC    F’0’  CREATE AN ABEND TO LOOK AT THE RECORDS             00710000
CHKSTYP1 DS    0H                                                       00720000
*   CHECK IF ANY OF SUBTYPE 1-10                                        00730003
         CLI   SMF85STY+1,X’01’                                         00740000
         BNE   *+18                                                     00750000
         MVI   STYPE,C’1’                                               00760000
         MVC   FUNC,=CL15’OSREQ ACCESS’                                 00770000
         B     STISOK                                                   00780005
         CLI   SMF85STY+1,X’02’                                         00790000
         BNE   *+18                                                     00800000
         MVI   STYPE,C’2’                                               00810000
         MVC   FUNC,=CL15’OSREQ STORE’                                  00820000
         B     STISOK                                                   00830005
         CLI   SMF85STY+1,X’03’                                         00840000
         BNE   *+18                                                     00850000
         MVI   STYPE,C’3’                                               00860000
         MVC   FUNC,=CL15’OSREQ RETRIEVE’                               00870000
         B     STISOK                                                   00880005
         CLI   SMF85STY+1,X’04’                                         00890000
         BNE   *+18                                                     00900000
         MVI   STYPE,C’4’                                               00910000
         MVC   FUNC,=CL15’OSREQ QUERY’                                  00920000
         B     STISOK                                                   00930005
         CLI   SMF85STY+1,X’05’                                         00940000
         BNE   *+18                                                     00950000
         MVI   STYPE,C’5’                                               00960000
         MVC   FUNC,=CL15’OSREQ CHANGE’                                 00970000
         B     STISOK                                                   00980005
         CLI   SMF85STY+1,X’06’                                         00990000
         BNE   *+18                                                     01000000
         MVI   STYPE,C’6’                                               01010000
         MVC   FUNC,=CL15’OSREQ DELETE’                                 01020000
         B     STISOK                                                   01030005
         CLI   SMF85STY+1,X’07’                                         01040000
         BNE   *+18                                                     01050000
         MVI   STYPE,C’7’                                               01060000
         MVC   FUNC,=CL15’OSREQ UNACCESS’                               01070000
```

```
         B     STISOK                                                   01080005
         CLI   SMF85STY+1,X'08'                                         01090003
         BNE   *+18                                                     01100003
         MVI   STYPE,C'8'                                               01110003
         MVC   FUNC,=CL15'OSREQ STOREBEG'                               01120003
         B     STISOK                                                   01130005
         CLI   SMF85STY+1,X'09'                                         01140003
         BNE   *+18                                                     01150003
         MVI   STYPE,C'9'                                               01160003
         MVC   FUNC,=CL15'OSREQ STOREPRT'                               01170003
         B     STISOK                                                   01180005
         CLI   SMF85STY+1,X'0A'                                         01190003
         BNE   *+18                                                     01200003
         MVI   STYPE,C'A'                                               01210003
         MVC   FUNC,=CL15'OSREQ STOREEND'                               01220003
         B     STISOK                                                   01230005
         B     IGNORE       OTHERWISE IGNORE                            01240027
STISOK   EQU   *                                                        01250005
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             01260000
* IS ONE OF TYPE 85 SUBTYPE 1-10 SO EXTRACT DATA                        01270003
* R3 IS THE START OF THE WHOLE RECORD                                   01280000
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.               01290000
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION        01300000
* ADD R8 TO R3                                                          01310000
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS                           01320000
         LA    R4,SMF85END                                              01330000
         USING ST1,R4                                                   01340000
         L     R8,SMF85OSO                                              01350000
         LH    R9,SMF85OSL                                              01360000
         LH    R10,SMF85OSN                                             01370000
*  PROCESS THE SUMMARY ENTRIES TRIPLET.                                 01380000
*  FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START                 01390000
*  SECOND HW IS THE LENGTH OF EACH TRIPLET                              01400000
*  THIRD HW IS THE NUMBER OF TRIPLETS                                   01410000
*  FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM       01420000
*  THE SMF RECORD TYPE 85 SUBTYPE 1-10 RECORDS.                         01430003
*  COLN COMES FROM ST1COLN                                              01440000
*  OBJN COMES FROM ST1OBJN                                              01450000
*  ETC                                                                  01460000
*  ST1FLGS IS NOT INTERPRETED - EACH BIT JUST SHOWN AS 1 OR 0           01470000
*                                                                       01480000
SCOTRIP  DS    0H                                                       01490000
         LA    R4,0(R3,R8)                                              01500000
         LA    R4,0(R3,R8)                                              01510000
         UNPK  YYDDD(7),SMF85DTE                                        01520000
         CLI   YYDDD+1,C'0'                                             01530000
         BE   SETD0                                                     01540000
         CLI   YYDDD+1,C'1'                                             01550000
         BE   SETD1                                                     01560000
         DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                  01570017
SETD0    MVC   YYDDD(2),=C'19'                                          01580000
         B     SETDZ                                                    01590000
SETD1    MVC   YYDDD(2),=C'20'                                          01600000
*                                                                       01610000
SETDZ    EQU   *                                                        01620000
```

```
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT
         LA    R5,100 PREPARE TO DIVIDE BY 100
         LA    R6,0
         L     R7,SMF85TME  GET THE TIME
         DR    R6,R5   -> SECS IN R7, HUNS IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  HUS,DWORD+6(2)
*        DC    F'0'
* NOW GET THE SECS
         LA    R5,60 PREPARE TO DIVIDE BY 60
         LA    R6,0
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  SS,DWORD+6(2)
* NOW GET THE MINS
         LA    R6,0
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  MM,DWORD+6(2)
         CVD   R7,DWORD     DO HOURS
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  HH,DWORD+6(2)
*
         PUT   PRINTDCB,PRINTL0
*
         PUT   PRINTDCB,PRINTL1
*  COPY    COLN
         MVC   COLN,ST1COLN
*  COPY    OBJN
         MVC   OBJN,ST1OBJN
         PUT   PRINTDCB,PRINTL2
*
*  COPY    SGN
         MVC   SGN,ST1SGN
*  COPY    SCN
         MVC   SCN,ST1SCN
*  COPY    MCN
         MVC   MCN,ST1MCN
*
*  CONVERT LEN
         L     R1,ST1LEN
         CVD   R1,DWORD
         OI    DWORD+7,X'0F'
         UNPK  LEN(11),DWORD+2(6)
         PUT   PRINTDCB,PRINTL3
*  CONVERT      TTOK  & TOK
* DO TOK FIRST
*        TOK IS 8 BYTES BINARY -> 16 BYTES PRINTABLE
*        MVC   TOK,ST1TOK
         UNPK  TRWORK(15),ST1TOK+1(8)  15 BYTES (ONE REDUNDANT BYTE)
         NC    TRWORK(15),=15X'0F'
         TR    TRWORK(15),TRTAB
```

```
         MVC   TOK+2(14),TRWORK                                         02180000
         UNPK  TRWORK(3),ST1TOK(2)  LAST BYTE + ONE REDUNDANT BYTE      02190000
         NC    TRWORK(3),=3X'0F'                                        02200000
         TR    TRWORK(3),TRTAB                                          02210000
         MVC   TOK(2),TRWORK                                            02220000
*                                                                       02230000
*        TTOK IS 16 BYTES BINARY -> 32 BYTES PRINTABLE                  02240000
*        HAVE TO UNPACK THIS AS TWO SETS OF 16 AS PER TOK               02250000
*        FIRST DO 16 BYTES, THE REPEAT FOR THE NEXT TWO                 02260000
         UNPK  TRWORK(15),ST1TTOK+1(8)  15 BYTES  (ONE REDUNDANT BYTE)  02270000
         NC    TRWORK(15),=15X'0F'                                      02280000
         TR    TRWORK(15),TRTAB                                         02290000
         MVC   TTOK+2(14),TRWORK                                        02300000
         UNPK  TRWORK(3),ST1TTOK(2)  LAST BYTES + ONE REDUNDANT BYTE    02310000
         NC    TRWORK(3),=3X'0F'                                        02320000
         TR    TRWORK(3),TRTAB                                          02330000
         MVC   TTOK(2),TRWORK                                           02340000
* NOW DO IT ALL AGAIN WITH OFFSET OF 8 ON ST1TTOK AND OFSET OF 16 ON    02350000
*        TTOK                                                           02360000
         UNPK  TRWORK(15),ST1TTOK+1+8(8) 15 BYTES (ONE REDUNDANT BYTE)  02370000
         NC    TRWORK(15),=15X'0F'                                      02380000
         TR    TRWORK(15),TRTAB                                         02390000
         MVC   TTOK+2+16(14),TRWORK                                     02400000
         UNPK  TRWORK(3),ST1TTOK+8(2)  LAST BYTES + ONE REDUNDANT BYTE  02410000
         NC    TRWORK(3),=3X'0F'                                        02420000
         TR    TRWORK(3),TRTAB                                          02430000
         MVC   TTOK+16(2),TRWORK                                        02440000
*                                                                       02450000
         PUT   PRINTDCB,PRINTL3A                                        02460028
*                                                                       02470000
* COPY VSN & VMT                                                        02480000
         MVC   VSN,ST1VSN                                               02490000
         MVC   MT,ST1VMT                                                02500000
*  CONVERT RC & RS                                                      02510000
*  CONVERT RC                                                           02520000
         UNPK  RC(09),ST1RC(5)                                          02530011
         TR    RC(08),HEXTAB-240                                        02540011
         MVI   RC+8,X'40'                                               02550012
*  CONVERT RS                                                           02560000
         UNPK  RS(09),ST1RS(5)                                          02570015
         TR    RS(08),HEXTAB-240                                        02580013
         MVI   RS+8,X'40'                                               02590013
*  PRINT   FLAGS                                                        02600000
         UNPK  FLGS(09),ST1FLGS(5)  UNPK 1 MORE THAN NEEDED             02610000
         MVI   FLGS+8,C' ' BLANK OUT THE EXTRA BYTE                     02620000
         NC    FLGS(08),=8X'0F'                                         02630000
         TR    FLGS(8),TRTAB                                            02640000
         PUT   PRINTDCB,PRINTL4                                         02650000
*  COPY    STOK                                                         02660004
         MVC   STOK,ST1STOK                                             02670004
*  CONVERT RC2                                                          02680004
         L     R1,ST1RC2                                                02690004
         CVD   R1,DWORD                                                 02700004
         OI    DWORD+7,X'0F'                                            02710004
         UNPK  RC2(08),DWORD+4(4)                                       02720004
```

```
         PUT   PRINTDCB,PRINTL5                                          02730009
*  CONVERT STOUT                                                         02740004
         L     R1,ST1STOUT                                               02750004
         CVD   R1,DWORD                                                  02760004
         OI    DWORD+7,X'0F'                                             02770004
         UNPK  STOUT(08),DWORD+4(4)                                      02780004
*  COPY    OLRD                                                          02790004
         MVC   OLRD,ST1OLRD                                              02800004
*  COPY    NLRD                                                          02810005
         MVC   NLRD,ST1NLRD                                              02820005
*  CONVERT INST                                                          02830004
         UNPK  INST(09),ST1INST(5)                                       02840016
         TR    INST(08),HEXTAB-240                                       02850016
         MVI   INST+8,X'40'                                              02860016
*        DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                   02870018
         PUT   PRINTDCB,PRINTL6                                          02880009
WRITEIT  DS    0H                                                        02890000
         PUT   PRINTDCB,PRINTBLK                                         02900000
*  LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS                02910000
*                                                                        02920000
*  WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                           02930000
         LA    R8,0(R8,R9)                                               02940000
         BCT   R10,SCOTRIP                                               02950000
         B     READ                                                      02960000
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS         02970000
         B     READ                                                      02980000
FINISH   DS    0H                                                        02990000
         SEGEND                                                          03000000
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,              X03010001
               EODAD=FINISH                                              03020000
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133                03030000
DWORD    DS    D                                                         03040000
         ORG   DWORD                                                     03050000
         DC    C'12345678'                                               03060000
TRWORK   DS    CL33                                                      03070000
HEXTAB   DC    C'0123456789ABCDEF'                                       03080011
TRTAB    DC    C'0123456789ABCDEF'                                       03090011
PRINTBLK DC    CL133' '                                                  03100000
PRINTHDR DC    CL133'1SMF TYPE 85 SUBTYPE 1-10 RECORDS'                  03110010
PRINTL0  DC    CL133' SMF85TA  SMFDTE/TME:'                              03120029
         ORG   PRINTL0+27                                                03130000
YYDDD    DC    CL7' ',CL1'/'                                             03140022
HH       DC    CL2' ',C':'                                               03150023
MM       DC    CL2' ',C':'                                               03160023
SS       DC    CL2' ',C'.'                                               03170023
HUS      DC    CL3' ',CL1' '                                             03180023
         ORG                                                             03190000
PRINTL1  DC    CL133' STYPE/FUNC:'                                       03200003
         ORG   PRINTL1+27                                                03210000
STYPE    DC    CL1' ',CL1'/' CONVERTED                                   03220023
FUNC     DC    CL15' ',C' '                                              03230023
         ORG                                                             03240000
PRINTL2  DC    CL133' COLN/OBJN:'                                        03250000
         ORG   PRINTL2+27                                                03260000
COLN     DC    CL44' ',C'/'                                              03270024
```

```
OBJN     DC    CL44’ ‘                                                03280021
         ORG                                                          03290000
*                                                                     03300000
PRINTL3  DC    CL133’ SGN/SCN/MCN/LEN/:’                              03310028
         ORG   PRINTL3+27                                             03320000
SGN      DC    CL8’ ‘,CL1’/’                                          03330024
SCN      DC    CL8’ ‘,CL1’/’                                          03340024
MCN      DC    CL8’ ‘,CL1’/’                                          03350024
LEN      DC    CL12’ ‘,CL1’ ‘                                         03360028
         ORG                                                          03370000
PRINTL3A DC    CL133’ TTOK/TOK:’                                      03380028
         ORG   PRINTL3A+27                                            03390028
TTOK     DC    CL32’ ‘,CL1’/’  CONVERTED                              03400028
TOK      DC    CL16’ ‘,CL1’ ‘  CONVERTED                              03410028
         ORG                                                          03420028
*                                                                     03430000
PRINTL4  DC    CL133’ VSN/VMT/RC/RS/FLGS:’                            03440004
         ORG   PRINTL4+27                                             03450000
VSN      DC    CL6’ ‘,CL1’/’                                          03460024
MT       DC    CL2’ ‘,CL1’/’                                          03470024
RC       DC    CL09’ ‘,CL1’/’                                         03480024
RS       DC    CL09’ ‘,CL1’/’  CONVERTED                              03490024
FLGS     DC    CL20’ ‘      AS-IS                                     03500000
         ORG                                                          03510000
*                                                                     03520004
PRINTL5  DC    CL133’ STOK/RC2:’                                      03530009
         ORG   PRINTL5+27                                             03540009
STOK     DC    CL16’ ‘,CL1’/’                                         03550024
RC2      DC    CL8’ ‘,CL1’ ‘ CONVERTED                                03560026
         ORG                                                          03570009
PRINTL6  DC    CL133’ STOUT/OLRD/NLRD/INST:’                          03580009
         ORG   PRINTL6+27                                             03590009
STOUT    DC    CL8’ ‘,CL1’/’                                          03600024
OLRD     DC    CL10’ ‘,CL1’/’                                         03610024
NLRD     DC    CL10’ ‘,CL1’/’                                         03620024
INST     DC    CL08’ ‘     CONVERTED BL4                              03630004
         ORG                                                          03640004
SMFDSECT DSECT                                                        03650000
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                     03660000
         END                                                          03670000
```

A.3.2.3 Storing and running the JCL to build the LOAD module SMF85TA

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-53 on page 265 into your PDS MHLRES2.SMF85TA.DFSMS13.PDS as member SMF85TAJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, “OAM SMF85 analysis source materials” on page 263 as member BUILDJCL
2. Update the JCL to refer to the USER you want to use. Update the SET statement to refer to the appropriate program that is being built, in this case SMF85TA.
3. Run the job after the member is created. The return code for both steps should be 0. Program SMF85TA can now be run by using the JCL shown in Figure A-1 on page 216.

A.3.3 SMF record type 85 subtype 32-35 data display program SMF85TH

Program SMG85TH displays the contents of selected fields of SMF record Type 85 subtypes 32/33/34/35 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that immediate backup is occurring.

The steps to build the program must be done once, after which they can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.3.1 Creating PDS/PDSE data sets

The sample JCL to create these data sets is shown in Figure A-52 on page 264. If the sample JCL is used, the SET to be specified is SMF85TH.

When USER=MHLRES2 and SET=SMF85TH, the following data sets result:

- MHLRES2.SMF85TH.DFSMS13.PDS
- MHLRES2.SMF85TH.DFSMS13.LOAD

A.3.3.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session. To use the FTP process, see A.3.1.5, "SMF85 program restore process" on page 265. If using the copy and paste process, see A.3.1.5, "SMF85 program restore process" on page 265.

> **Attention:** The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, "SMF85 program restore process" on page 265. Copy and paste the contents of Example A-4 into member SMF85THA of data set MHLRES2.SMF85TH.DFSMS13.PDS. The result should contain 525 lines.

> **Tip:** In the code in Example A-4 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

There is a line of code reading ST32BORB EQU CT32BORB in the source. This code corrects a field definition error where the field is labeled CT32BORB instead of ST32BORB. When the CRBSMF is corrected to use the label ST32BORB, an error due to a duplicate label occurs. If this error occurs, remove the line of code reading ST32BORB EQU CT32BORB from the SMF85TH assembler source code.

Example A-4 SMF85TH program source code

```
 MACRO                                                          00010000
&NAME    SEGSTART                                                       00020000
&NAME    STM   14,12,12(13)           SAVE HIS REGS IN HIS SAVE AREA    00030000
R0       EQU   0                                                        00040000
R1       EQU   1                                                        00050000
R2       EQU   2                                                        00060000
R3       EQU   3                                                        00070000
R4       EQU   4                                                        00080000
R5       EQU   5                                                        00090000
R6       EQU   6                                                        00100000
R7       EQU   7                                                        00110000
```

```
R8       EQU   8                                                        00120000
R9       EQU   9                                                        00130000
R10      EQU   10                                                       00140000
R11      EQU   11                                                       00150000
RB       EQU   12                                                       00160000
R13      EQU   13                                                       00170000
R14      EQU   14                                                       00180000
R15      EQU   15                                                       00190000
         BALR  12,0                   SET UP ADDRESSABILITY             00200000
         USING *,12                   USE REG 12 AS BASE REG            00210000
         USING *+4096,R11             USE REG 11 AS ADDITIONAL BASE     00220018
         LA    R11,4095(RB)                                             00230014
         LA    R11,1(R11)                                               00240013
         ST    13,SAVEREGS+4          SAVE @ OF HIS SAVEAREA IN MINE    00250000
         LA    03,SAVEREGS            LOAD @ OF MY SAVE AREA IN REG 3   00260000
         ST    03,8(13)               SAVE @ OF MY SAVE AREA IN HIS     00270000
         LR    13,03                  LOAD @ OF MY SAVE AREA IN REG 13  00280000
         MEND                                                           00290000
         MACRO                                                          00300000
&NAME    SEGEND                                                         00310000
&NAME    L     13,SAVEREGS+4          LOAD REG13 WITH @ OF HIS SAVE     00320000
         LM    14,12,12(13)           RESTORE REGS FROM HIS SAVEAREA    00330000
         XR    R15,R15                                                  00340000
         BR    14                     RETURN TO CALLING RTN VIA REG 14  00350000
SAVEREGS DC    18F'0'                 SET UP SAVE AREA                  00360000
         MEND                                                           00370000
        MACRO                                                           00380000
        HEXTEXT4 &KEY                                                   00390005
        UNPK  &KEY.(9),ST32&KEY.(5)                                     00400007
        TR    &KEY.(08),HEXTAB-240                                      00410028
        MVI   &KEY.+8,X'40' BLANK THE EXTRA BYTE                        00420005
        MEND                                                            00430000
        MACRO                                                           00440006
        HEXTEXT8 &KEY                                                   00450005
        UNPK  &KEY.(09),ST32&KEY.(5)                                    00460008
        TR    &KEY.(08),HEXTAB-240                                      00470011
        UNPK  &KEY.+8(09),ST32&KEY.+4(5)                                00480008
        TR    &KEY.+8(08),HEXTAB-240                                    00490011
        MVI   &KEY.+16,X'40' BLANK THE EXTRA BYTE                       00500005
        MEND                                                            00510005
SMFR85TH  SEGSTART                                                      00520000
*  THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS PARTS OF 00530000
*  THE SMF TYPE 85 SUBTYPE 32-35 RECORDS.                               00540000
*  IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED        00550000
*  TO SELECT ANY OR ALL OF TYPE 85 SUBTYPES 32-35                       00560000
*  RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR                 00570000
*  OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.               00580000
*                                                                       00590000
*  THE STANDARD SMF RECORD MAPPING MACROS ARE USED.                     00600000
*  REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD                  00610000
*  R3    START OF WHOLE RECORD                                          00620000
*  THERE IS 1 DSECTS TO BE MAPPED                                       00630000
*  R4    START OF SUBTYPE RECORDS                                       00640000
*  R5    FOR DIVIDING (DIVISOR)                                         00650000
*  R6    FOR DIVIDING - EVEN-ODD PAIR WITH R7  (DIVIDEND)               00660000
```

```
*  R7    FOR DIVIDING
*  OTHER REGISTER USES
*  R12   OVERALL BASE REGISTER
*  R8    RECORD TYPE/SUBTYPE CHECKING/WORKING
*  R9    LENGTH OF PARTICULAR DSECT
*  R10   NUMBER OF ENTRIES IN THE TRIPLET
*
*  QSAM GET LOCATE PROCESSING IS USED
*
         OPEN SMFIN
         OPEN (PRINTDCB,(OUTPUT))
         PUT  PRINTDCB,PRINTHDR
READ     GET SMFIN
* COPY PARAMETER POINTER
         LR    R3,R1
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
         USING CBRSMF85,R3
*   CHECK IF TYPE 85
         CLI   SMF85RTY,X'55'
         BNE   IGNORE
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS
CHKSTYP1 DS    0H
*   CHECK IF ANY OF SUBTYPE 32-35
         CLI   SMF85STY+1,X'20'
         BNE   *+18
         MVI   STYPE,C'2'
         MVC   FUNC,=CL34'(STORAGE GROUP PROCESSING)'
         B     STOK
         CLI   SMF85STY+1,X'21'
         BNE   *+18
         MVI   STYPE,C'3'
         MVC   FUNC,=CL34'(DASD SPACE MANAGEMENT PROCESSING)'
         B     STOK
         CLI   SMF85STY+1,X'22'
         BNE   *+18
         MVI   STYPE,C'4'
         MVC   FUNC,=CL34'(OPTICAL DISK RECOVERY UTILITY)'
         B     STOK
         CLI   SMF85STY+1,X'22'
         BNE   *+18
         MVI   STYPE,C'4'
         MVC   FUNC,=CL34'(OPTICAL DISK RECOVERY UTILITY)'
         B     STOK
         CLI   SMF85STY+1,X'23'
         BNE   *+18
         MVI   STYPE,C'5'
         MVC   FUNC,=CL34'(MOVE VOLUME (MOVEVOL) UTILITY)'
         B     STOK
*        OTHERWISE IGNORE
         B     IGNORE
STOK     EQU   *
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS
* IS ONE OF TYPE 85 SUBTYPE 32-35 SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
```

```
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
         LA    R4,SMF85END
         USING ST32,R4
         L     R8,SMF85OSO
         LH    R9,SMF85OSL
         LH    R10,SMF85OSN
*  PROCESS THE SUMMARY ENTRIES TRIPLET.
*  FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START
*  SECOND HW IS THE LENGTH OF EACH TRIPLET
*  THIRD HW IS THE NUMBER OF TRIPLETS
*  FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM
*  THE SMF RECORD TYPE 85 SUBTYPE 32-35 RECORDS.
*  SGN  COMES FROM ST32SGN
*  VSN1 COMES FROM ST32VSN1
*  MT   COMES FROM ST32OMT
*  ETC
*  STXXFLGS IS NOT INTERPRETED - EACH BIT JUST SHOWN AS 1 OR 0
*
SCOTRIP  DS    0H
         LA    R4,0(R3,R8)
         UNPK  YYDDD(7),SMF85DTE
         CLI   YYDDD+1,C'0'
         BE   SETD0
         CLI   YYDDD+1,C'1'
         BE   SETD1
*        OTHERWISE ABEND AS SOMETHING HAS GONE WRONG
         DC    F'0'
SETD0    MVC   YYDDD(2),=C'19'
         B     SETDZ
SETD1    MVC   YYDDD(2),=C'20'
*
SETDZ    EQU   *
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT
         LA    R5,100 PREPARE TO DIVIDE BY 100
         LA    R6,0
         L     R7,SMF85TME  GET THE TIME
         DR    R6,R5   -> SECS IN R7, HUNS IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  HUS,DWORD+6(2)
*        DC    F'0'
* NOW GET THE SECS
         LA    R5,60 PREPARE TO DIVIDE BY 60
         LA    R6,0
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  SS,DWORD+6(2)
* NOW GET THE MINS
         LA    R6,0
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6
         CVD   R6,DWORD
```

```
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING               01770000
         UNPK  MM,DWORD+6(2)                                          01780000
         CVD   R7,DWORD     DO HOURS                                  01790000
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING               01800000
         UNPK  HH,DWORD+6(2)                                          01810000
*                                                                     01820000
         PUT   PRINTDCB,PRINTL0                                       01830000
*  COPY    SGN                                                        01840000
         MVC   SGN,ST32SGN                                            01850000
*  COPY    VSM0                                                       01860000
         MVC   VSN0,ST32VSN0                                          01870000
*  COPY    VSM1                                                       01880000
         MVC   VSN1,ST32VSN1                                          01890000
*  COPY    MT                                                         01900000
         MVC   MT,ST32OMT                                             01910000
         PUT   PRINTDCB,PRINTL1                                       01920000
*                                                                     01930000
         HEXTEXT4 PDWO CONVERT                                        01940005
         HEXTEXT4 PDWK CONVERT                                        01950005
         HEXTEXT4 PDRO CONVERT                                        01960005
         HEXTEXT4 PDRK CONVERT                                        01970005
         HEXTEXT4 PDDO CONVERT                                        01980005
         HEXTEXT4 PDDK CONVERT                                        01990005
         PUT   PRINTDCB,PRINTL2                                       02000000
*                                                                     02010005
         HEXTEXT8 PDWB CONVERT                                        02020005
*        DC F'0' CAUSE ABEND                                          02030019
         HEXTEXT8 PDRB CONVERT                                        02040005
         HEXTEXT8 PDDB CONVERT                                        02050005
         PUT   PRINTDCB,PRINTL2A                                      02060005
*                                                                     02070000
         HEXTEXT4 POWO CONVERT                                        02080005
         HEXTEXT4 POWK CONVERT                                        02090005
         HEXTEXT4 PORO CONVERT                                        02100005
         HEXTEXT4 PORK CONVERT                                        02110005
         HEXTEXT4 PODO CONVERT                                        02120005
         HEXTEXT4 PODK CONVERT                                        02130005
         PUT   PRINTDCB,PRINTL3                                       02140000
         HEXTEXT8 POWB CONVERT                                        02150015
         HEXTEXT8 PORB CONVERT                                        02160015
         HEXTEXT8 PODB CONVERT                                        02170015
         PUT   PRINTDCB,PRINTL3A                                      02180015
*                                                                     02190000
         HEXTEXT4 PTWO CONVERT                                        02200005
         HEXTEXT4 PTWK CONVERT                                        02210005
         HEXTEXT4 PTRO CONVERT                                        02220005
         HEXTEXT4 PTRK CONVERT                                        02230005
         HEXTEXT4 PTDO CONVERT                                        02240005
         HEXTEXT4 PTDK CONVERT                                        02250005
         PUT   PRINTDCB,PRINTL4                                       02260000
*                                                                     02270015
         HEXTEXT8 PTWB CONVERT                                        02280015
         HEXTEXT8 PTRB CONVERT                                        02290015
         HEXTEXT8 PTDB CONVERT                                        02300015
         PUT   PRINTDCB,PRINTL4A                                      02310015
```

```
*                                                                       02320000
         HEXTEXT4 BOWO CONVERT                                          02330005
         HEXTEXT4 BOWK CONVERT                                          02340005
         HEXTEXT4 BORO CONVERT                                          02350005
         HEXTEXT4 BORK CONVERT                                          02360005
         HEXTEXT4 BODO CONVERT                                          02370005
         HEXTEXT4 BODK CONVERT                                          02380005
         PUT   PRINTDCB,PRINTL5                                         02390000
*                                                                       02400000
         HEXTEXT8 BOWB CONVERT                                          02410015
ST32BORB EQU CT32BORB                                                   02420016
         HEXTEXT8 BORB CONVERT                                          02430015
         HEXTEXT8 BODB CONVERT                                          02440015
         PUT   PRINTDCB,PRINTL5A                                        02450015
*                                                                       02460015
         HEXTEXT4 BTWO CONVERT                                          02470005
         HEXTEXT4 BTWK CONVERT                                          02480005
         HEXTEXT4 BTRO CONVERT                                          02490005
         HEXTEXT4 BTRK CONVERT                                          02500005
         HEXTEXT4 BTDO CONVERT                                          02510005
         HEXTEXT4 BTDK CONVERT                                          02520005
         PUT   PRINTDCB,PRINTL6                                         02530000
*                                                                       02540015
         HEXTEXT8 BTWB CONVERT                                          02550015
         HEXTEXT8 BTRB CONVERT                                          02560015
         HEXTEXT8 BTDB CONVERT                                          02570015
         PUT   PRINTDCB,PRINTL6A                                        02580015
*                                                                       02590000
         HEXTEXT4 B2OWO CONVERT                                         02600005
         HEXTEXT4 B2OWK CONVERT                                         02610005
         HEXTEXT4 B2ORO CONVERT                                         02620005
         HEXTEXT4 B2ORK CONVERT                                         02630005
         HEXTEXT4 B2ODO CONVERT                                         02640005
         HEXTEXT4 B2ODK CONVERT                                         02650005
         PUT   PRINTDCB,PRINTL7                                         02660000
*                                                                       02670000
         HEXTEXT8 B2OWB CONVERT                                         02680015
         HEXTEXT8 B2ORB CONVERT                                         02690015
         HEXTEXT8 B2ODB CONVERT                                         02700015
         PUT   PRINTDCB,PRINTL7A                                        02710015
*                                                                       02720015
         HEXTEXT4 B2TWO CONVERT                                         02730015
         HEXTEXT4 B2TWK CONVERT                                         02740015
         HEXTEXT4 B2TRO CONVERT                                         02750015
         HEXTEXT4 B2TRK CONVERT                                         02760015
         HEXTEXT4 B2TDO CONVERT                                         02770015
         HEXTEXT4 B2TDK CONVERT                                         02780015
         PUT   PRINTDCB,PRINTL8                                         02790021
*                                                                       02800015
         HEXTEXT8 B2TWB CONVERT                                         02810015
         HEXTEXT8 B2TRB CONVERT                                         02820015
         HEXTEXT8 B2TDB CONVERT                                         02830015
         PUT   PRINTDCB,PRINTL8A                                        02840015
         HEXTEXT4 DTUP CONVERT                                          02850005
         HEXTEXT4 DTDE CONVERT                                          02860005
```

```
*  CONVERT 4KIN                                                         02870000
         L     R1,ST324KIN                                              02880000
         CVD   R1,DWORD                                                 02890000
         OI    DWORD+7,X'0F'                                            02900000
         UNPK  N4KIN(11),DWORD+2(6)                                     02910000
*  CONVERT 4KDE                                                         02920000
         L     R1,ST324KDE                                              02930000
         CVD   R1,DWORD                                                 02940000
         OI    DWORD+7,X'0F'                                            02950000
         UNPK  N4KDE(11),DWORD+2(6)                                     02960000
*  CONVERT 32KI                                                         02970000
         L     R1,ST3232KI                                              02980000
         CVD   R1,DWORD                                                 02990000
         OI    DWORD+7,X'0F'                                            03000000
         UNPK  N32KI(11),DWORD+2(6)                                     03010000
*  CONVERT 32KD                                                         03020000
         L     R1,ST3232KD                                              03030000
         CVD   R1,DWORD                                                 03040000
         OI    DWORD+7,X'0F'                                            03050000
         UNPK  N32KD(11),DWORD+2(6)                                     03060000
         HEXTEXT4 NCE  CONVERT                                          03070005
         PUT   PRINTDCB,PRINTL9                                         03080000
*  INTERPRET THE FLAGS                                                  03090000
         UNPK  FLGS(09),ST32FLGS(5)  UNPK 1 MORE THAN NEEDED            03100000
         MVI   FLGS+8,C' ' BLANK OUTTHE EXTRA BYTE                      03110000
         NC    FLGS(08),=8X'0F'                                         03120000
         TR    FLGS(8),TRTAB                                            03130000
         HEXTEXT4 NTE  CONVERT                                          03140005
         HEXTEXT4 RCLD CONVERT                                          03150005
         HEXTEXT4 RCLK CONVERT                                          03160005
         HEXTEXT4 LOBI CONVERT                                          03170005
         HEXTEXT4 LOBD CONVERT                                          03180005
         PUT   PRINTDCB,PRINTLA                                         03190015
*                                                                       03200015
         HEXTEXT8 RCLB CONVERT                                          03210015
         PUT   PRINTDCB,PRINTLAA                                        03220015
*                                                                       03230015
         HEXTEXT4 PUWO CONVERT                                          03240015
         HEXTEXT4 PUWK CONVERT                                          03250015
         HEXTEXT4 PURO CONVERT                                          03260015
         HEXTEXT4 PURK CONVERT                                          03270015
         HEXTEXT4 PUDO CONVERT                                          03280015
         HEXTEXT4 PUDK CONVERT                                          03290015
*                                                                       03300000
         PUT   PRINTDCB,PRINTLB                                         03310021
*                                                                       03320015
         HEXTEXT8 PUWB CONVERT                                          03330015
         HEXTEXT8 PURB CONVERT                                          03340015
         HEXTEXT8 PUDB CONVERT                                          03350015
*                                                                       03360015
         PUT   PRINTDCB,PRINTLBA                                        03370015
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             03380000
         PUT   PRINTDCB,PRINTBLK                                        03390000
*  LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS               03400000
*  WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                          03410000
```

```
         LA    R8,0(R8,R9)                                              03420000
         BCT   R10,SCOTRIP                                              03430000
         B     READ                                                     03440000
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        03450000
         B     READ                                                     03460000
FINISH   DS    0H                                                       03470000
         SEGEND                                                         03480000
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C03490000
               EODAD=FINISH                                             03500000
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133               03510000
DWORD    DS    D                                                        03520000
         ORG   DWORD                                                    03530000
         DC    C’12345678’                                              03540000
TRWORK   DS    CL33                                                     03550000
TRTAB    DC    C’0123456789ABCDEF’                                      03560000
HEXTAB   DC    C’0123456789ABCDEF’ TRANSLATE TABLE                      03570012
PRINTBLK DC    CL133’ ‘                                                 03580000
PRINTHDR DC    CL133’1SMF TYPE 85 SUBTYPE 32-35 RECORDS’                03590000
PRINTL0  DC    CL133’ SMF85TH SMFDTE/SMFTME:’                           03600041
         ORG   PRINTL0+38                                               03610000
YYDDD    DC    CL7’ ‘,CL1’ ‘                                            03620036
HH       DC    CL2’ ‘,C’:’                                              03630036
MM       DC    CL2’ ‘,C’:’                                              03640036
SS       DC    CL2’ ‘,C’.’                                              03650036
HUS      DC    CL3’ ‘,CL1’ ‘                                            03660036
         ORG                                                            03670000
PRINTL1  DC    CL133’ STYPE/FUNC/SGN/VSN0/VSN1/MT:’                     03680035
         ORG   PRINTL1+38                                               03690000
STYPE3   DC    CL1’3’  PREFIX TO SUBTYPES 32-35                         03700000
STYPE    DC    CL1’ ‘,CL1’/’                                            03710036
FUNC     DC    CL34’ ‘,CL1’/’                                           03720036
SGN      DC    CL8’ ‘,C’/’                                              03730036
VSN0     DC    CL6’ ‘,C’/’                                              03740036
VSN1     DC    CL6’ ‘,C’/’                                              03750036
MT       DC    CL2’ ‘,C’ ‘                                              03760036
         ORG                                                            03770000
*                                                                       03780000
PRINTL2  DC    CL133’ PDWO/PDWK/PDRO/PDRK/PDDO/PDDK:’                   03790000
         ORG   PRINTL2+38                                               03800000
PDWO     DC    CL12’ ‘,CL1’/’  CONVERTED                                03810036
PDWK     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED TO PDWB                 03820036
PDRO     DC    CL12’ ‘,CL1’/’  CONVERTED                                03830036
PDRK     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED TO PDRB                 03840036
PDDO     DC    CL12’ ‘,CL1’/’  CONVERTED                                03850036
PDDK     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED TO PDDB                 03860036
         ORG                                                            03870000
*                                                                       03880000
PRINTL2A DC    CL133’ PDWB/PDRB/PDDB:’                                  03890005
         ORG   PRINTL2A+38                                              03900025
PDWB     DC    CL17’ ‘,CL1’/’  CONVERTED  MOVED FR PDWK                 03910036
PDRB     DC    CL17’ ‘,CL1’/’  CONVERTED  MOVED FR PDRK                 03920036
PDDB     DC    CL17’ ‘,CL1’ ‘  CONVERTED MOVED FR PDDK                  03930036
         ORG                                                            03940005
*                                                                       03950005
PRINTL3  DC    CL133’ POWO/POWK/PORO/PORK/PODO/PODK:’                   03960000
```

```
         ORG   PRINTL3+38                                              03970027
POWO     DC    CL12’ ‘,CL1’/’  CONVERTED                               03980036
POWK     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED TO POWB                03990036
PORO     DC    CL12’ ‘,CL1’/’  CONVERTED                               04000036
PORK     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED TO PORB                04010036
PODO     DC    CL12’ ‘,CL1’/’  CONVERTED                               04020036
PODK     DC    CL12’ ‘,CL1’ ‘  CONVERTED MOVED TO PODB                 04030036
         ORG                                                           04040000
PRINTL3A DC    CL133’ POWB/PORB/PODB:’                                 04050015
         ORG   PRINTL3A+38                                             04060027
POWB     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED FR POWK                04070036
PORB     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED FR PORK                04080036
PODB     DC    CL12’ ‘,CL1’ ‘  CONVERTED  MOVED FR PODK                04090036
         ORG                                                           04100015
PRINTL4  DC    CL133’ PTWO/PTWK/PTRK/PTRD/PTDO/PTDK:’                  04110018
         ORG   PRINTL4+38                                              04120000
PTWO     DC    CL12’ ‘,CL1’/’  CONVERTED                               04130036
PTWK     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED TO PTWB                04140036
PTRO     DC    CL12’ ‘,CL1’/’  CONVERTED                               04150036
PTRK     DC    CL12’ ‘,CL1’/’  CONVERTED  MOVED TO PTRB                04160036
PTDO     DC    CL12’ ‘,CL1’/’  CONVERTED                               04170036
PTDK     DC    CL12’ ‘,CL1’ ‘  CONVERTED MOVED TO PTDB                 04180036
         ORG                                                           04190000
PRINTL4A DC    CL133’ PTWB/PTRB/PTDB:’                                 04200015
         ORG   PRINTL4A+38                                             04210015
PTWB     DC    CL17’ ‘,CL1’/’  CONVERTED                               04220036
PTRB     DC    CL17’ ‘,CL1’/’  CONVERTED  MOVED FR PTRK                04230036
PTDB     DC    CL17’ ‘,CL1’ ‘  CONVERTED MOVED FR PTDK                 04240036
         ORG                                                           04250015
PRINTL5  DC    CL133’ BOWO/BOWK/BORO/BORK/BODO/BODK:’                  04260015
         ORG   PRINTL5+38                                              04270015
BOWO     DC    CL12’ ‘,CL1’/’ CONVERTED                                04280036
BOWK     DC    CL12’ ‘,CL1’/’ CONVERTED MOVED TO BOWB                  04290036
BORO     DC    CL12’ ‘,CL1’/’ CONVERTED                                04300036
BORK     DC    CL12’ ‘,CL1’/’ CONVERTED MOVED TO BORB                  04310036
BODO     DC    CL12’ ‘,CL1’/’ CONVERTED                                04320036
BODK     DC    CL12’ ‘,CL1’ ‘ CONVERTED MOVED TO BODB                  04330036
         ORG                                                           04340015
PRINTL5A DC    CL133’ BOWB/BORB/BODB:’                                 04350024
         ORG   PRINTL5A+38                                             04360015
BOWB     DC    CL17’ ‘,CL1’/’  CONVERTED MOVED FR BOWK                 04370036
BORB     DC    CL17’ ‘,CL1’/’  CONVERTED MOVED FR BORK                 04380036
BODB     DC    CL17’ ‘,CL1’ ‘  CONVERTED MOVED FR BODK                 04390036
         ORG                                                           04400015
PRINTL6  DC    CL133’ BTWO/BTWK/BTRO/BTRK/BTDO/BTDK:’                  04410000
         ORG   PRINTL6+38                                              04420000
BTWO     DC    CL12’ ‘,CL1’/’   CONVERTED                              04430036
BTWK     DC    CL12’ ‘,CL1’/’   CONVERTED  MOVED TO BTWB               04440036
BTRO     DC    CL12’ ‘,CL1’/’   CONVERTED                              04450036
BTRK     DC    CL12’ ‘,CL1’/’   CONVERTED  MOVED TO BTRB               04460036
BTDO     DC    CL12’ ‘,CL1’/’   CONVERTED                              04470036
BTDK     DC    CL12’ ‘,CL1’ ‘   CONVERTED  MOVED TO BTDB               04480036
         ORG                                                           04490000
PRINTL6A DC    CL133’ BTWB/BTRB/BTDB:’                                 04500015
         ORG   PRINTL6A+38                                             04510015
```

```
BTWB     DC    CL17’ ‘,CL1’/’   CONVERTED  MOVED FR BTWK              04520036
BTRB     DC    CL17’ ‘,CL1’/’   CONVERTED  MOVED FR BTRK              04530036
BTDB     DC    CL17’ ‘,CL1’ ‘   CONVERTED  MOVED FR BTDK              04540036
         ORG                                                          04550015
PRINTL7  DC    CL133’ B2OWO/B2OWK/B2ORO/B2ORK/B2ODO/B2ODK:’           04560000
         ORG   PRINTL7+38                                             04570000
B2OWO    DC    CL12’ ‘,CL1’/’  CONVERTED                              04580036
B2OWK    DC    CL12’ ‘,CL1’/’  CONVERTED    MOVED TO B2OWB            04590036
B2ORO    DC    CL12’ ‘,CL1’/’ CONVERTED                               04600036
B2ORK    DC    CL12’ ‘,CL1’/’ CONVERTED    MOVED TO B2ORB             04610036
B2ODO    DC    CL12’ ‘,CL1’/’ CONVERTED                               04620036
B2ODK    DC    CL12’ ‘,CL1’ ‘  CONVERTED    MOVED TO B2ODB            04630036
         ORG                                                          04640000
PRINTL7A DC    CL133’ B2OWB/B2ORB/B2ODK:’                             04650015
         ORG   PRINTL7A+38                                            04660015
B2OWB    DC    CL17’ ‘,CL1’/’  CONVERTED    MOVED FR B2OWK            04670036
B2ORB    DC    CL17’ ‘,CL1’/’  CONVERTED    MOVED FR B2ORK            04680036
B2ODB    DC    CL17’ ‘,CL1’ ‘  CONVERTED    MOVED TO B2ODK            04690036
         ORG                                                          04700015
PRINTL8  DC    CL133’ B2TWO/B2TWK/B2TRO/B2TRK/B2TDO/B2TDK:’           04710000
         ORG   PRINTL8+38                                             04720000
B2TWO    DC    CL12’ ‘,CL1’/’   CONVERTED                             04730036
B2TWK    DC    CL12’ ‘,CL1’/’   CONVERTED  MOVED TO B2TWB             04740036
B2TRO    DC    CL12’ ‘,CL1’/’   CONVERTED                             04750036
B2TRK    DC    CL12’ ‘,CL1’/’   CONVERTED  MOVED TO B2TRB             04760036
B2TDO    DC    CL12’ ‘,CL1’/’   CONVERTED                             04770036
B2TDK    DC    CL12’ ‘,CL1’ ‘   CONVERTED  MOVED TO B2TDB             04780036
         ORG                                                          04790000
PRINTL8A DC    CL133’ B2TWK/B2TRK/B2TDK:’                             04800015
         ORG   PRINTL8A+38                                            04810015
B2TWB    DC    CL17’ ‘,CL1’/’  CONVERTED  MOVED FR B2TWK              04820036
B2TRB    DC    CL17’ ‘,CL1’/’  CONVERTED  MOVED FR B2TRK              04830036
B2TDB    DC    CL17’ ‘,CL1’ ‘  CONVERTED  MOVED TO B2TDK              04840036
         ORG                                                          04850015
PRINTL9  DC    CL133’ DTUP/DTDE/4KIN/4KDE/32KI/32KD/NCE:’             04860000
         ORG   PRINTL9+38                                             04870000
DTUP      DC    CL12’ ‘,CL1’/’  CONVERTED                             04880036
DTDE      DC    CL12’ ‘,CL1’/’  CONVERTED                             04890036
N4KIN     DC    CL12’ ‘,CL1’/’  CONVERTED                             04900036
N4KDE     DC    CL12’ ‘,CL1’/’  CONVERTED                             04910036
N32KI     DC    CL12’ ‘ CL1’/’  CONVERTED                             04920036
N32KD     DC    CL12’ ‘,CL1’/’  CONVERTED                             04930036
NCE       DC    CL12’ ‘,CL1’ ‘  CONVERTED                             04940036
         ORG                                                          04950000
PRINTLA  DC    CL133’ FLGS/NTE/RCLD/RCLK/LOBI/LOBD:’                  04960015
         ORG   PRINTLA+38                                             04970015
FLGS     DC    CL9’ ‘,CL1’/’    INTERPRETED AS 0 OR 1                 04980037
NTE       DC    CL12’ ‘,CL1’/’  CONVERTED                             04990037
RCLD      DC    CL12’ ‘,CL1’/’  CONVERTED                             05000037
RCLK      DC    CL12’ ‘,CL1’/’  CONVERTED MOVED TO RCLB               05010037
LOBI      DC    CL12’ ‘,CL1’/’  CONVERTED                             05020037
LOBD      DC    CL12’ ‘,CL1’ ‘  CONVERTED                             05030037
         ORG                                                          05040000
PRINTLAA DC    CL133’ RCLB:’                                          05050024
         ORG   PRINTLAA+38                                            05060015
```

```
RCLB      DC    CL17' '  CONVERTED  MOVED FR RCLK                    05070020
         ORG                                                         05080015
PRINTLB  DC    CL133' PUWO/PUWK/PURO/PURK/PUDO/PUDK:'                05090015
         ORG   PRINTLB+38                                            05100015
PUWO      DC    CL12' ',CL1'/'   CONVERTED                           05110037
PUWK      DC    CL12' ',CL1'/'   CONVERTED   MOVED TO PUWB           05120037
PURO      DC    CL12' ',CL1'/'   CONVERTED                           05130037
PURK      DC    CL12' ',CL1'/'   CONVERTED   MOVED TO PURB           05140037
PUDO      DC    CL12' ',CL1'/'   CONVERTED                           05150037
PUDK      DC    CL12' ',CL1' '  CONVERTED   MOVED TO PUDB            05160037
         ORG                                                         05170022
PRINTLBA DC    CL133' PUWB/PURB/PUDB:'                               05180015
         ORG   PRINTLBA+38                                           05190020
PUWB      DC    CL17' ',CL1'/'   CONVERTED   MOVED FR PUWK           05200037
PURB      DC    CL17' ',CL1'/'   CONVERTED   MOVED FR PURK           05210037
PUDB      DC    CL17' ',CL1' '   CONVERTED   MOVED FR PUDK           05220037
SMFDSECT DSECT                                                       05230000
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                    05240000
         END                                                         05250000
```

A.3.3.3 Storing and running the JCL to build the LOAD module SMF85TH

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-53 on page 265 into your PDS MHLRES1.SMF85TH.DFSMS13.PDS as member SMF85THJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, "OAM SMF85 analysis source materials" on page 263 as member BUILDJCL.
2. Update the JCL to refer to the USER you want to use. Also update the SET statement to refer to the appropriate program that is being built, in this case SMF85TH.
3. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TH can now be run by using the JCL shown in Figure A-9 on page 227.

A.3.4 SMF record type 85 subtype 36 data display program SMF85TQ

Program SMG85TQ displays the contents of selected fields of SMF record Type 85 subtype 36 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that immediate backup is occurring.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.4.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, "OAM SMF85 Analysis program common data set creation" on page 264. If the sample JCL is used, the SET to be specified is SMF85TQ.

When USER=MHLRES2 and SET=SMF85TQ, the following data sets result:

- MHLRES2.SMF85TQ.DFSMS13.PDS
- MHLRES2.SMF85TQ.DFSMS13.LOAD

A.3.4.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session. To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process see A.3.1.5, “SMF85 program restore process” on page 265.

> **Attention:** The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-5 into member SMF85TQA of data set MHLRES2.SMF85TQ.DFSMS13.PDS. The result should contain 226 lines.

> **Tip:** In the code in Example A-5 on the line that starts with SMFIN there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-5 SMF85TQ program source code

```
 MACRO                                                         00010006
&NAME    SEGSTART                                                      00020006
&NAME    STM   14,12,12(13) SAVE HIS REGS IN HIS SAVE AREA             00030006
R0       EQU   0                                                       00040006
R1       EQU   1                                                       00050006
R2       EQU   2                                                       00060006
R3       EQU   3                                                       00070006
R4       EQU   4                                                       00080006
R5       EQU   5                                                       00090006
R6       EQU   6                                                       00100006
R7       EQU   7                                                       00110006
R8       EQU   8                                                       00120006
R9       EQU   9                                                       00130006
R10      EQU   10                                                      00140006
R11      EQU   11                                                      00150006
RB       EQU   12                                                      00160006
R13      EQU   13                                                      00170006
R14      EQU   14                                                      00180006
R15      EQU   15                                                      00190006
         BALR  12,0 SET UP ADDRESSABILITY                              00200006
         USING *,12 USE REG 12 AS BASE REG                             00210006
         ST    13,SAVEREGS+4 SAVE @ OF HIS SAVEAREA IN MINE            00220006
         LA    03,SAVEREGS LOAD @ OF MY SAVE AREA IN REG 3             00230006
         ST    03,8(13) SAVE @ OF MY SAVE AREA IN HIS                  00240006
         LR    13,03 LOAD @ OF MY SAVE AREA IN REG 13                  00250006
         MEND                                                          00260006
         MACRO                                                         00270006
&NAME    SEGEND                                                        00280006
&NAME    L     13,SAVEREGS+4 LOAD REG13 WITH @ OF HIS SAVE             00290006
         LM    14,12,12(13) RESTORE REGS FROM HIS SAVEAREA             00300006
         XR    R15,R15                                                 00310006
         BR    14 RETURN TO CALLING RTN VIA REG 14                     00320006
SAVEREGS DC    18F'0' SET UP SAVE AREA                                 00330006
         MEND                                                          00340006
SMFR85TI SEGSTART                                                      00350006
```

```
* THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF
* THE SMF TYPE 85 SUBTYPE 36 RECORDS, WHICH ARE THE OSMC
* SINGLE OBJECT RECOVERY.
* IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
* TO SELECT TYPE 85 SUBTYPE 36
* RECORDS FROM EITHER THE ACTIVE SMF ‘MAN’ DATASETS OR
* OFF A PREVIOUSLY EXTRACTED COPY OF THE ‘MAN’ DATASETS.
*
* THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
* REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
* R3 START OF WHOLE RECORD
* THERE IS 1 DSECTS TO BE MAPPED
* R4 START OF ST36 OSMC  SINGLE OBJECT RECOVERY
* R5 SPARE
* R6 SPARE
* R7 SPARE
* OTHER REGISTER USES
* R12 OVERALL BASE REGISTER
* R8 RECORD TYPE/SUBTYPE CHECKING/WORKING
* R9 LENGTH OF PARTICULAR DSECT
* R10 NUMBER OF ENTRIES IN THE TRIPLET
*
* QSAM GET LOCATE PROCESSING IS USED
*
         OPEN  SMFIN
         OPEN  (PRINTDCB,(OUTPUT))
         PUT   PRINTDCB,PRINTHDR
READ     GET   SMFIN
         LR    R3,R1  * COPY PARAMETER POINTER
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
         USING CBRSMF85,R3
* CHECK IF TYPE 85
         CLI   SMF85RTY,X’55’
         BNE   IGNORE
CHKSTYP1 DS    0H
* CHECK IF SUBTYPE 39
         CLI   SMF85STY+1,X’24’
         BNE   IGNORE
* DC F’0’ CREATE AN ABEND TO LOOK AT THE RECORDS
* IS TYPE 85 SUBTYPE 36, SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
         LA    R4,SMF85END
         USING ST36,R4
         L     R8,SMF85OSO
         LH    R9,SMF85OSL
         LH    R10,SMF85OSN
* PROCESS THE SUMMARY ENTRIES TRIPLET.
* FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START
* SECOND HW IS THE LENGTH OF EACH TRIPLET
* THIRD HW IS THE NUMBER OF TRIPLETS
```

```
* FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM
* THE SMF RECORD TYPE 85 SUBTYPE 39 RECORDS.
* COLN COMES FROM ST36COLN
* CNID COMES FROM ST36CNID
* ETC
* ST36FLGS IS NOT INTERPRETED - EACH BIT JUST SHOWN AS 1 OR 0
*
SCOTRIP  DS    0H
         LA    R4,0(R3,R8)
         UNPK  YYDDD(7),SMF85DTE
         CLI   YYDDD+1,C’0’
         BE   SETD0
         CLI   YYDDD+1,C’1’
         BE   SETD1
*        OTHERWISE ABEND AS SOMETHING HAS GONE WRONG
         DC    F’0’
SETD0    MVC   YYDDD(2),=C’19’
         B     SETDZ
SETD1    MVC   YYDDD(2),=C’20’
*
SETDZ    EQU   *
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT
         LA    R5,100 PREPARE TO DIVIDE BY 100
         LA    R6,0
         L     R7,SMF85TME  GET THE TIME
         DR    R6,R5   -> SECS IN R7, HUNS IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X’0F’  FIX THE SIGN FOR PRINTING
         UNPK  HUS,DWORD+6(2)
* NOW GET THE SECS
         LA    R5,60 PREPARE TO DIVIDE BY 60
         LA    R6,0
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X’0F’  FIX THE SIGN FOR PRINTING
         UNPK  SS,DWORD+6(2)
* NOW GET THE MINS
         LA    R6,0
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X’0F’  FIX THE SIGN FOR PRINTING
         UNPK  MM,DWORD+6(2)
         CVD   R7,DWORD     DO HOURS
         OI    DWORD+7,X’0F’  FIX THE SIGN FOR PRINTING
         UNPK  HH,DWORD+6(2)
         PUT   PRINTDCB,PRINTL0
         LA    R4,0(R3,R8)
         MVC   COLN,ST36COLN
* CONVERT CNID
         L     R1,ST36CNID
         CVD   R1,DWORD
         OI    DWORD+7,X’0F’
         UNPK  CNID(11),DWORD+2(6)
         PUT   PRINTDCB,PRINTL1
         MVC   OBJN,ST36OBJN
```

```
         MVC   SGN,ST36SGN                                              01460006
* CONVERT OLEN                                                          01470006
         L     R1,ST36OLEN                                              01480006
         CVD   R1,DWORD                                                 01490006
         OI    DWORD+7,X'0F'                                            01500006
         UNPK  OLEN(11),DWORD+2(6)                                      01510006
         PUT   PRINTDCB,PRINTL2                                         01520006
         MVC   BVSN,ST36BVSN                                            01530006
         MVC   BMT,ST36BMT                                              01540006
* CONVERT BTKN                                                          01550006
         L     R1,ST36BTKN                                              01560006
         CVD   R1,DWORD                                                 01570006
         OI    DWORD+7,X'0F'                                            01580006
         UNPK  BTKN(11),DWORD+2(6)                                      01590006
         MVC   TVSN,ST36TVSN                                            01600006
         MVC   TMT,ST36TMT                                              01610006
         PUT   PRINTDCB,PRINTL3                                         01620006
         MVC   OVSN,ST36OVSN                                            01630006
         MVC   OMT,ST36OMT                                              01640006
* PRINT FLAGS                                                           01650006
         UNPK  FLGS(09),ST36FLGS(5) UNPK 1 MORE THAN NEEDED             01660006
         MVI   FLGS+8,C' ' BLANK OUTTHE EXTRA BYTE                      01670006
         MVC   DSL,ST36DSL                                              01680006
         PUT   PRINTDCB,PRINTL4                                         01690006
WRITEIT  DS    0H                                                       01700006
         PUT   PRINTDCB,PRINTBLK                                        01710006
* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS                01720006
* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                           01730006
         LA    R8,0(R8,R9)                                              01740006
         BCT   R10,SCOTRIP                                              01750006
         B     READ                                                     01760006
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        01770006
         B     READ                                                     01780006
FINISH   DS    0H                                                       01790006
         SEGEND                                                         01800006
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C01810006
               EODAD=FINISH                                             01820006
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133               01830006
DWORD    DS    D                                                        01840006
         ORG   DWORD                                                    01850006
         DC    C'12345678'                                              01860006
PRINTBLK DC    CL133' '                                                 01870006
PRINTHDR DC    CL133'1SMF TYPE 85 SUBTYPE 36 RECORDS'                   01880006
PRINTL0  DC    CL133' SMF85TQ SMFDTE/TME:'                              01890007
         ORG   PRINTL0+25                                               01900010
YYDDD    DC    CL7' ',CL1' '                                            01910006
HH       DC    CL2' ',C':'                                              01920006
MM       DC    CL2' ',C':'                                              01930006
SS       DC    CL2' ',C'.'                                              01940006
HUS      DC    CL3' ',CL1' '                                            01950006
         ORG                                                            01960006
PRINTL1  DC    CL133' COLN/CNID:'                                       01970006
         ORG   PRINTL1+25                                               01980010
COLN     DC    CL44' ',C'/'                                             01990006
CNID     DC    CL20' ',C' ' CONVERTED FROM BL4                          02000006
```

```
         ORG                                                       02010006
PRINTL2  DC    CL133' OBJN/SGN/OLEN:'                              02020006
         ORG   PRINTL2+25                                          02030010
OBJN     DC    CL44' ',CL1'/'                                      02040006
SGN      DC    CL8' ',CL1'/'                                       02050006
OLEN     DC    CL20' ',C' ' CONVERTED FROM BL4                     02060006
         ORG                                                       02070006
*                                                                  02080006
PRINTL3  DC    CL133' BVSN/BMT/BTKN/TVSN/TMT:'                     02090006
         ORG   PRINTL3+25                                          02100010
BVSN     DC    CL6' ',CL1'/'                                       02110006
BMT      DC    CL2' ',CL1'/'                                       02120006
BTKN     DC    CL20' ',CL1'/'      CONVERTED FROM BL4              02130006
TVSN     DC    CL6' ',CL1'/'                                       02140006
TMT      DC    CL2' ',CL1' '                                       02150009
         ORG                                                       02151009
PRINTL4  DC    CL133' OVSN/OMT/FLGS/DSL:'                          02160006
         ORG   PRINTL4+25                                          02170010
OVSN     DC    CL6' ',CL1'/'                                       02180006
OMT      DC    CL2' ',CL1'/'                                       02190006
FLGS     DC    CL09' ',CL1'/'       AS-IS                          02200010
DSL      DC    CL2' ',C' '                                         02210006
         ORG                                                       02220006
SMFDSECT DSECT                                                     02230006
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                  02240006
         END                                                       02250006
```

A.3.4.3 Storing and running the JCL to build the LOAD module SMF85TQ

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-53 on page 265 into your PDS MHLRES1.SMF85TQ.DFSMS13.PDS as member SMF85TQJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, "OAM SMF85 analysis source materials" on page 263 as member BUILDJCL
2. Update the JCL to refer to the USER you want to use. Then update the SET statement to refer to the appropriate program that is being built, in this case SMF85TQ.
3. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TQ can now be run by using the JCL shown in Figure A-9 on page 227.

A.3.5 SMF Record type 85 subtype 38 data display program SMF85TO

Program SMG85TO displays the contents of selected fields of SMF record Type 85 subtype 38 data. It is not intended to provide a comprehensive report on OAM activity. Instead, it verifies that retrieval from tape to DASD is occurring when an object is recalled.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.5.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, "OAM SMF85 Analysis program common data set creation" on page 264. If the sample JCL is used, the 'SET' to be specified is SMF85TQ.

When USER=MHLRES2 and SET=SMF85TO, the following data sets result:

- MHLRES2.SMF85TO.DFSMS13.PDS
- MHLRES2.SMF85TO.DFSMS13.LOAD

A.3.5.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session. To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

> **Attention:** The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-6 into member SMF85TOA of data set MHLRES2.SMF85TO.DFSMS13.PDS. The result should contain 218 lines.

> **Tip:** In the code in Example A-6 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-6 SMF85TO program source code

```
 MACRO                                                          00010000
&NAME    SEGSTART                                                       00020000
&NAME    STM   14,12,12(13)            SAVE HIS REGS IN HIS SAVE AREA   00030000
R0       EQU   0                                                        00040000
R1       EQU   1                                                        00050000
R2       EQU   2                                                        00060000
R3       EQU   3                                                        00070000
R4       EQU   4                                                        00080000
R5       EQU   5                                                        00090000
R6       EQU   6                                                        00100000
R7       EQU   7                                                        00110000
R8       EQU   8                                                        00120000
R9       EQU   9                                                        00130000
R10      EQU   10                                                       00140000
R11      EQU   11                                                       00150000
RB       EQU   12                                                       00160000
R13      EQU   13                                                       00170000
R14      EQU   14                                                       00180000
R15      EQU   15                                                       00190000
         BALR  12,0                    SET UP ADDRESSABILITY            00200000
         USING *,12                    USE REG 12 AS BASE REG           00210000
         ST    13,SAVEREGS+4           SAVE @ OF HIS SAVEAREA IN MINE   00220000
         LA    03,SAVEREGS             LOAD @ OF MY SAVE AREA IN REG 3  00230000
         ST    03,8(13)                SAVE @ OF MY SAVE AREA IN HIS    00240000
         LR    13,03                   LOAD @ OF MY SAVE AREA IN REG 13 00250000
         MEND                                                           00260000
         MACRO                                                          00270000
         SEGEND                                                         00280000
         L     13,SAVEREGS+4           LOAD REG13 WITH @ OF HIS SAVE    00290000
         LM    14,12,12(13)            RESTORE REGS FROM HIS SAVEAREA   00300000
```

```
         XR    R15,R15
         BR    14                      RETURN TO CALLING RTN VIA REG 14
SAVEREGS DC    18F'0'                  SET UP SAVE AREA
         MEND
SMFR85TO  SEGSTART
*  THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF
*  THE SMF TYPE 85 SUBTYPE 38 RECORDS, WHICH ARE THE OAM DATA SET
*  RECALL SUMMARY RECORDS
*  IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
*  TO SELECT TYPE 85 SUBTYPE 38
*  RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR
*  OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.
*
*  THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
*  REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
*  R3    START OF WHOLE RECORD
*  THERE IS 1 DSECTS TO BE MAPPED
*  R4    START OF ST38 SINGLE OBJECT RECALL SECTION
*  R5    SPARE
*  R6    SPARE
*  R7    SPARE
*  OTHER REGISTER USES
*  R12   OVERALL BASE REGISTER
*  R8    RECORD TYPE/SUBTYPE CHECKING/WORKING
*  R9    LENGTH OF PARTICULAR DSECT
*  R10   NUMBER OF ENTRIES IN THE TRIPLET
*
*  QSAM GET LOCATE PROCESSING IS USED
*
         OPEN SMFIN
         OPEN (PRINTDCB,(OUTPUT))
         PUT  PRINTDCB,PRINTHDR
READ     GET SMFIN
         LR    R3,R1     * COPY PARAMETER POINTER
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
         USING CBRSMF85,R3
         CLI   SMF85RTY,X'55'    *   CHECK IF TYPE 85
         BNE   IGNORE
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS
CHKSTYP1 DS    0H
         CLI   SMF85STY+1,X'26'  *   CHECK IF SUBTYPE 38
         BNE   IGNORE
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS
* IS TYPE 85 SUBTYPE 38, SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
         LA    R4,SMF85END
         USING ST38,R4
         L     R8,SMF85OSO
         LH    R9,SMF85OSL
         LH    R10,SMF85OSN
```

```
*  PROCESS THE SUMMARY ENTRIES TRIPLET.                                 00860000
*  FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START                 00870000
*  SECOND HW IS THE LENGTH OF EACH TRIPLET                              00880000
*  THIRD HW IS THE NUMBER OF TRIPLETS                                   00890000
*  FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM       00900000
*  THE SMF RECORD TYPE 85 SUBTYPE 38 RECORDS.                           00910000
*  COLN COMES FROM ST38COLN                                             00920000
*  CNID COMES FROM ST38CNID                                             00930000
*  ETC                                                                  00940000
*  ST38FLGS IS INTERPRETED AS FLGO ON OR OFF                            00950000
*                                                                       00960000
SCOTRIP  DS    0H                                                       00970000
         LA    R4,0(R3,R8)                                              00980045
         UNPK  YYDDD(7),SMF85DTE                                        00990045
         CLI   YYDDD+1,C'0'                                             01000045
         BE   SETD0                                                     01010045
         CLI   YYDDD+1,C'1'                                             01020045
         BE   SETD1                                                     01030045
*        DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                  01040045
SETD0    MVC   YYDDD(2),=C'19'                                          01050045
         B     SETDZ                                                    01060045
SETD1    MVC   YYDDD(2),=C'20'                                          01070045
*                                                                       01080045
SETDZ    EQU   *                                                        01090045
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT             01100045
         LA    R5,100 PREPARE TO DIVIDE BY 100                          01110045
         LA    R6,0                                                     01120045
         L     R7,SMF85TME  GET THE TIME                                01130045
         DR    R6,R5   -> SECS IN R7, HUNS IN R6                        01140045
         CVD   R6,DWORD                                                 01150045
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01160045
         UNPK  HUS,DWORD+6(2)                                           01170045
*        DC    F'0'                                                     01180045
* NOW GET THE SECS                                                      01190045
         LA    R5,60 PREPARE TO DIVIDE BY 60                            01200045
         LA    R6,0                                                     01210045
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6              01220045
         CVD   R6,DWORD                                                 01230045
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01240045
         UNPK  SS,DWORD+6(2)                                            01250045
* NOW GET THE MINS                                                      01260045
         LA    R6,0                                                     01270045
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6               01280045
         CVD   R6,DWORD                                                 01290045
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01300045
         UNPK  MM,DWORD+6(2)                                            01310045
         CVD   R7,DWORD     DO HOURS                                    01320045
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01330045
         UNPK  HH,DWORD+6(2)                                            01340045
         LA    R4,0(R3,R8)                                              01350000
         MVC   COLN,ST38COLN                                            01360000
*  CONVERT CNID                                                         01370000
         UNPK  CNID(09),ST38CNID                                        01380037
         TR    CNID(08),HEXTAB-240                                      01390042
         MVI   CNID+8,X'40'                                             01400039
```

```
         PUT   PRINTDCB,PRINTL1
         MVC   OBJN,ST38OBJN
         MVC   SGN,ST38SGN
*  CONVERT OLEN
         UNPK  OLEN(09),ST38OLEN(5)
         TR    OLEN(08),HEXTAB-240
         MVI   OLEN+8,X’40’
*        DC    F’0’  CREATE AND ABEND TO LOOK AT THE RECORDS
         PUT   PRINTDCB,PRINTL2
         MVC   VSN,ST38VSN
         MVC   MT,ST38MT
*  CONVERT TKN
         UNPK  TKN(09),ST38TKN(5)
         TR    TKN(08),HEXTAB-240
         MVI   TKN+8,X’40’
         MVC   VT,ST38VT
         MVC   BT,ST38BT
*  CONVERT FLAGS
         MVC   FLGS,=CL20’FLG0 OFF’
         TM    0(R1),ST38FLG0   IS THE FLAG ON?
         BNO   FLG0OFF
         MVC   FLGS(08),=C’FLG0 ON ‘
FLG0OFF  EQU *
         PUT   PRINTDCB,PRINTL3
*        DC    F’0’  CREATE AN ABEND TO LOOK AT THE RECORDS
WRITEIT  DS    0H
         PUT   PRINTDCB,PRINTBLK
*  LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS
*  WHEN BCT REACHES ZERO GO GET ANOTHER RECORD
         LA    R8,0(R8,R9)
         BCT   R10,SCOTRIP
         B     READ
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS
         B     READ
FINISH   DS    0H
         SEGEND
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C
               EODAD=FINISH
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133
DWORD    DS    D
         ORG   DWORD
         DC    C’12345678’
HEXTAB   DC    C’0123456789ABCDEF’ TRANSLATE TABLE
PRINTBLK DC    CL133’ ‘
PRINTHDR DC    CL133’1SMF TYPE 85 SUBTYPE 38 RECORDS’
PRINTL0  DC    CL133’ SMF85TO SMFDTE/TME:’
         ORG   PRINTL0+35
YYDDD    DC    CL7’ ‘,CL1’/’
HH       DC    CL2’ ‘,C’:’
MM       DC    CL2’ ‘,C’:’
SS       DC    CL2’ ‘,C’.’
HUS      DC    CL3’ ‘,CL1’ ‘
         ORG
PRINTL1  DC    CL133’ COLN/CNID(IN HEX):’
         ORG   PRINTL1+31
```

```
COLN     DC    CL44' ',C' '                                             01960044
CNID     DC    CL20' ',C' ' CONVERTED FROM BL4                          01970044
         ORG                                                            01980000
*                                                                       01990000
PRINTL2  DC    CL133' OBJN/SGN/OLEN(KB IN HEX):'                        02000037
         ORG   PRINTL2+33                                               02010036
OBJN     DC    CL44' ',CL1' '                                           02020044
SGN      DC    CL8' ',CL1' '                                            02030044
OLEN     DC    CL20' ',C' ' CONVERTED FROM BL4                          02040044
         ORG                                                            02050000
*                                                                       02060000
PRINTL3  DC    CL133' VSN/MT/TKN(IN HEX)/VT/BT/FLGS:'                   02070037
         ORG   PRINTL3+31                                               02080037
VSN      DC    CL6' ',CL1' '                                            02090044
MT       DC    CL2' ',CL1' '                                            02100044
TKN      DC    CL20' ',CL1' '      CONVERTED FROM BL4                   02110044
VT       DC    CL2' ',CL1' '                                            02120044
BT       DC    CL2' ',CL1' '                                            02130044
FLGS     DC    CL20' ',CL1' '      INTERPRETED                          02140044
         ORG                                                            02150000
SMFDSECT DSECT                                                          02160000
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                       02170000
         END                                                            02180000
```

A.3.5.3 Storing and running the JCL to build the LOAD module SMF85TO

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-52 on page 264 into your PDS MHLRES1.SMF85TO.DFSMS13.PDS as member SMF85TOJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, “OAM SMF85 analysis source materials” on page 263 as member BUILDJCL.
2. Update the JCL to refer to the USER you want to use. Then update the SET statement to refer to the appropriate program that is being built, in this case SMF85TO.
3. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TO can now be run by using the JCL shown in Figure A-22 on page 239.

A.3.6 SMF Record type 85 subtype 39 data display program SMF85TI

Program SMG85TI displays the contents of selected fields of SMF record Type 85 subtype 39 data. It is not intended to provide a comprehensive report on OAM activity. Rather, it verifies that retrieval from tape to DASD is occurring when an object is recalled.

There are two steps to build the program. This process needs to be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.6.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, “OAM SMF85 Analysis program common data set creation” on page 264. If the sample JCL is used, the ‘SET’ to be specified is SMF85TI.

When USER=MHLRES2 and SET=SMF85TI, the following data sets result:

- MHLRES2.SMF85TI.DFSMS13.PDS
- MHLRES2.SMF85TI.DFSMS13.LOAD

A.3.6.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by copying from the document and pasting into the ISPF session.

To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

Attention: The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as described in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-7 into member SMF85TIA of data set MHLRES2.SMF85TI.DFSMS13.PDS. The result should contain 223 lines.

Tip: In the code in Example A-7 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-7 SMF85TI program source code

```
 MACRO                                                          00010005
&NAME    SEGSTART                                                       00020005
&NAME    STM   14,12,12(13) SAVE HIS REGS IN HIS SAVE AREA              00030005
R0       EQU   0                                                        00040005
R1       EQU   1                                                        00050005
R2       EQU   2                                                        00060005
R3       EQU   3                                                        00070005
R4       EQU   4                                                        00080005
R5       EQU   5                                                        00090005
R6       EQU   6                                                        00100005
R7       EQU   7                                                        00110005
R8       EQU   8                                                        00120005
R9       EQU   9                                                        00130005
R10      EQU   10                                                       00140005
R11      EQU   11                                                       00150005
RB       EQU   12                                                       00160005
R13      EQU   13                                                       00170005
R14      EQU   14                                                       00180005
R15      EQU   15                                                       00190005
         BALR  12,0 SET UP ADDRESSABILITY                               00200005
         USING *,12 USE REG 12 AS BASE REG                              00210005
         ST    13,SAVEREGS+4 SAVE @ OF HIS SAVEAREA IN MINE             00220005
         LA    03,SAVEREGS LOAD @ OF MY SAVE AREA IN REG 3              00230005
         ST    03,8(13) SAVE @ OF MY SAVE AREA IN HIS                   00240005
         LR    13,03 LOAD @ OF MY SAVE AREA IN REG 13                   00250005
         MEND                                                           00260005
         MACRO                                                          00270005
&NAME    SEGEND                                                         00280005
&NAME    L     13,SAVEREGS+4 LOAD REG13 WITH @ OF HIS SAVE              00290005
         LM    14,12,12(13) RESTORE REGS FROM HIS SAVEAREA              00300005
         XR    R15,R15                                                  00310005
         BR    14 RETURN TO CALLING RTN VIA REG 14                      00320005
SAVEREGS DC    18F'0' SET UP SAVE AREA                                  00330005
         MEND                                                           00340005
```

```
SMFR85TI SEGSTART
* THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF
* THE SMF TYPE 85 SUBTYPE 39 RECORDS, WHICH ARE THE OAM DATA SET
* IMMEDIATE BACKUP RECORDS.
* IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
* TO SELECT TYPE 85 SUBTYPE 39
* RECORDS FROM EITHER THE ACTIVE SMF ‘MAN’ DATASETS OR
* OFF A PREVIOUSLY EXTRACTED COPY OF THE ‘MAN’ DATASETS.
*
* THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
* REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
* R3 START OF WHOLE RECORD
* THERE IS 1 DSECTS TO BE MAPPED
* R4 START OF ST39 OSMC IMMEDIATE BACKUP COPY
* R5 SPARE
* R6 SPARE
* R7 SPARE
* OTHER REGISTER USES
* R12 OVERALL BASE REGISTER
* R8 RECORD TYPE/SUBTYPE CHECKING/WORKING
* R9 LENGTH OF PARTICULAR DSECT
* R10 NUMBER OF ENTRIES IN THE TRIPLET
*
* QSAM GET LOCATE PROCESSING IS USED
*
         OPEN  SMFIN
         OPEN  (PRINTDCB,(OUTPUT))
         PUT   PRINTDCB,PRINTHDR
READ     GET   SMFIN
* COPY PARAMETER POINTER
         LR    R3,R1
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
         USING CBRSMF85,R3
* CHECK IF TYPE 85
         CLI   SMF85RTY,X’55’
         BNE   IGNORE
* DC F’0’ CREATE AN ABEND TO LOOK AT THE RECORDS
CHKSTYP1 DS    0H
* CHECK IF SUBTYPE 39
         CLI   SMF85STY+1,X’27’
         BNE   IGNORE
* DC F’0’ CREATE AN ABEND TO LOOK AT THE RECORDS
* IS TYPE 85 SUBTYPE 39, SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
         LA    R4,SMF85END
         USING ST39,R4
         L     R8,SMF85OSO
         LH    R9,SMF85OSL
         LH    R10,SMF85OSN
* PROCESS THE SUMMARY ENTRIES TRIPLET.
```

```
* FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START
* SECOND HW IS THE LENGTH OF EACH TRIPLET
* THIRD HW IS THE NUMBER OF TRIPLETS
* FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM
* THE SMF RECORD TYPE 85 SUBTYPE 39 RECORDS.
* COLN COMES FROM ST39COLN
* CNID COMES FROM ST39CNID
* ETC
* ST39FLGS IS NOT INTERPRETED - EACH BIT JUST SHOWN AS 1 OR 0
*
SCOTRIP  DS    0H
         LA    R4,0(R3,R8)
         UNPK  YYDDD(7),SMF85DTE
         CLI   YYDDD+1,C'0'
         BE   SETD0
         CLI   YYDDD+1,C'1'
         BE   SETD1
*        DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG
SETD0    MVC   YYDDD(2),=C'19'
         B     SETDZ
SETD1    MVC   YYDDD(2),=C'20'
*
SETDZ    EQU   *
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT
         LA    R5,100 PREPARE TO DIVIDE BY 100
         LA    R6,0
         L     R7,SMF85TME  GET THE TIME
         DR    R6,R5   -> SECS IN R7, HUNS IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  HUS,DWORD+6(2)
*        DC    F'0'
* NOW GET THE SECS
         LA    R5,60 PREPARE TO DIVIDE BY 60
         LA    R6,0
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  SS,DWORD+6(2)
* NOW GET THE MINS
         LA    R6,0
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6
         CVD   R6,DWORD
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  MM,DWORD+6(2)
         CVD   R7,DWORD     DO HOURS
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING
         UNPK  HH,DWORD+6(2)
         PUT   PRINTDCB,PRINTLO
         LA    R4,0(R3,R8)
         MVC   COLN,ST39COLN
* CONVERT CNID
         L     R1,ST39CNID
         CVD   R1,DWORD
         OI    DWORD+7,X'0F'
```

```
         UNPK  CNID(11),DWORD+2(6)                                      01440005
         PUT   PRINTDCB,PRINTL1                                         01450005
         MVC   OBJN,ST39OBJN                                            01460005
         MVC   SGN,ST39SGN                                              01470005
         MVC   MCN,ST39MCN                                              01480005
* CONVERT OLEN                                                          01490005
         L     R1,ST39OLEN                                              01500005
         CVD   R1,DWORD                                                 01510005
         OI    DWORD+7,X'0F'                                            01520005
         UNPK  OLEN(11),DWORD+2(6)                                      01530005
         PUT   PRINTDCB,PRINTL2                                         01540005
*                                                                       01550005
         MVC   SVSN,ST39SVSN                                            01560005
         MVC   SMT,ST39SMT                                              01570005
         MVC   TVSN,ST39TVSN                                            01580005
         MVC   TMT,ST39TMT                                              01590005
* CONVERT BTKN                                                          01600005
         L     R1,ST39BTKN                                              01610005
         CVD   R1,DWORD                                                 01620005
         OI    DWORD+7,X'0F'                                            01630005
         UNPK  BTKN(11),DWORD+2(6)                                      01640005
* PRINT FLAGS                                                           01650005
         UNPK FLGS(09),ST39FLGS(5) UNPK 1 MORE THAN NEEDED              01660005
         MVI FLGS+8,C' ' BLANK OUTTHE EXTRA BYTE                        01670005
         PUT PRINTDCB,PRINTL3                                           01680005
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS                        01690005
WRITEIT  DS 0H                                                          01700005
         PUT PRINTDCB,PRINTBLK                                          01710005
* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS                01720005
* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                           01730005
         LA    R8,0(R8,R9)                                              01740005
         BCT   R10,SCOTRIP                                              01750005
         B     READ                                                     01760005
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        01770005
         B     READ                                                     01780005
FINISH   DS    0H                                                       01790005
         SEGEND                                                         01800005
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C01810005
               EODAD=FINISH                                             01820005
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133               01830005
DWORD    DS    D                                                        01840005
         ORG   DWORD                                                    01850005
         DC    C'12345678'                                              01860005
PRINTBLK DC    CL133' '                                                 01870005
PRINTHDR DC    CL133'1SMF TYPE 85 SUBTYPE 39 RECORDS'                   01880005
PRINTL0  DC    CL133' SMF85TI SMFDTE/TME:'                              01890000
         ORG   PRINTL0+30                                               01900007
YYDDD    DC    CL7' ',CL1'/'                                            01910000
HH       DC    CL2' ',C':'                                              01920000
MM       DC    CL2' ',C':'                                              01930000
SS       DC    CL2' ',C'.'                                              01940000
HUS      DC    CL3' ',CL1' '                                            01950000
         ORG                                                            01960000
PRINTL1  DC    CL133' COLN/CNID:'                                       01970005
         ORG PRINTL1+30                                                 01980007
```

```
COLN     DC    CL44’ ‘,C’ ‘                                               01990005
CNID     DC    CL20’ ‘,C’ ‘ CONVERTED FROM BL4                            02000005
         ORG                                                              02010005
*                                                                         02020005
PRINTL2  DC    CL133’ OBJN/SGN/MCN/OLEN:’                                 02030005
         ORG   PRINTL2+30                                                 02040007
OBJN     DC    CL44’ ‘,CL1’/’                                             02050005
SGN      DC    CL8’ ‘,CL1’/’                                              02060005
MCN      DC    CL8’ ‘,CL1’/’                                              02070005
OLEN     DC    CL20’ ‘,C’ ‘ CONVERTED FROM BL4                            02080005
         ORG                                                              02090005
*                                                                         02100005
PRINTL3  DC    CL133’ SVSN/SMT/TVSN/TMT/BTKN/FLGS:’                       02110005
         ORG   PRINTL3+30                                                 02120005
SVSN     DC    CL6’ ‘,CL1’/’                                              02130005
SMT      DC    CL2’ ‘,CL1’/’                                              02140005
TVSN     DC    CL6’ ‘,CL1’/’                                              02150005
TMT      DC    CL2’ ‘,CL1’/’                                              02160005
BTKN     DC    CL20’ ‘,CL1’/’    CONVERTED FROM BL4                       02170005
FLGS     DC    CL20’ ‘,CL1’ ‘ AS-IS                                       02180005
         ORG                                                              02190005
SMFDSECT DSECT                                                            02200005
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                         02210005
         END                                                              02220005
```

A.3.6.3 Storing and running the JCL to build the LOAD module SMF85TIJ

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-53 on page 265 into your PDS MHLRES1.SMF85TI.DFSMS13.PDS as member SMF85TIJ. The result should contain 27 lines.
2. Update the JCL to refer to the USER you want to use. Update the SET statement to refer to the appropriate program that is being built, in this case SMF85TI.
3. Run the job when the member is created. The return code for each step should be 0. Program SMF85TI can now be run by using the JCL shown in Figure A-25 on page 241.

A.3.7 SMF Record type 85 subtype 40 data display program SMF85TJ

Program SMG85TJ displays the contents of selected fields of SMF record Type 85 subtype 40 data. It is not intended to provide a comprehensive report on OAM activity. Rather, it verifies that retrieval from tape to DASD is occurring when an object is recalled.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.7.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, “OAM SMF85 Analysis program common data set creation” on page 264. If the sample JCL is used, the ‘SET’ to be specified is SMF85TJ.

When USER=MHLRES2 and SET=SMF85TJ, the following data sets result:

- MHLRES2.SMF85TJ.DFSMS13.PDS
- MHLRES2.SMF85TJ.DFSMS13.LOAD

A.3.7.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository, or reconstructed by cutting from the document and pasting into the ISPF session. To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

Attention: The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-8 into member SMF85TJA of data set MHLRES2.SMF85TJ.DFSMS13.PDS. The result should contain 199 lines.

Tip: In the code in Example A-8 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-8 SMF85TJ program source code

```
 MACRO                                                          00010014
&NAME    SEGSTART                                                       00020014
&NAME    STM   14,12,12(13) SAVE HIS REGS IN HIS SAVE AREA              00030014
R0       EQU   0                                                        00040014
R1       EQU   1                                                        00050014
R2       EQU   2                                                        00060014
R3       EQU   3                                                        00070014
R4       EQU   4                                                        00080014
R5       EQU   5                                                        00090014
R6       EQU   6                                                        00100014
R7       EQU   7                                                        00110014
R8       EQU   8                                                        00120014
R9       EQU   9                                                        00130014
R10      EQU   10                                                       00140014
R11      EQU   11                                                       00150014
RB       EQU   12                                                       00160014
R13      EQU   13                                                       00170014
R14      EQU   14                                                       00180014
R15      EQU   15                                                       00190014
         BALR  12,0 SET UP ADDRESSABILITY                               00200014
         USING *,12 USE REG 12 AS BASE REG                              00210014
         ST    13,SAVEREGS+4 SAVE @ OF HIS SAVEAREA IN MINE             00220014
         LA    03,SAVEREGS LOAD @ OF MY SAVE AREA IN REG 3              00230014
         ST    03,8(13) SAVE @ OF MY SAVE AREA IN HIS                   00240014
         LR    13,03 LOAD @ OF MY SAVE AREA IN REG 13                   00250014
         MEND                                                           00260014
         MACRO                                                          00270014
&NAME    SEGEND                                                         00280014
&NAME    L     13,SAVEREGS+4 LOAD REG13 WITH @ OF HIS SAVE              00290014
         LM    14,12,12(13) RESTORE REGS FROM HIS SAVEAREA              00300014
         XR    R15,R15                                                  00310014
         BR    14 RETURN TO CALLING RTN VIA REG 14                      00320014
SAVEREGS DC    18F'0' SET UP SAVE AREA                                  00330014
         MEND                                                           00340014
         MACRO                                                          00350014
```

```
&NAME    BINDEC &KEY
         LH    R1,ST40&KEY.
         CVD   R1,DWORD
         OI    DWORD+7,X'0F'
         UNPK  &KEY.(11),DWORD+2(6)
         MEND
SMFR85TJ SEGSTART
* THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF
* THE SMF TYPE 85 SUBTYPE 40 RECORDS, WHICH ARE THE
* OAM COMMAND RECYCLE RECORDS.
* IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
* TO SELECT TYPE 85 SUBTYPE 40
* RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR
* OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.
*
* THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
* REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
* R3 START OF WHOLE RECORD
* THERE IS 1 DSECTS TO BE MAPPED
* R4 START OF ST40 OSMC COMMAND RECYCLE
* R5 START OF VOLUME ARRAY - ASSUMED TO START AT THE END OF BASE
* R6 NUMBER OF VOLUMES
* R7 SPARE
* OTHER REGISTER USES
* R12 OVERALL BASE REGISTER
* R8 RECORD TYPE/SUBTYPE CHECKING/WORKING
* R9 LENGTH OF PARTICULAR DSECT
* R10 NUMBER OF ENTRIES IN THE TRIPLET
*
* QSAM GET LOCATE PROCESSING IS USED
*
         OPEN  SMFIN
         OPEN  (PRINTDCB,(OUTPUT))
         PUT   PRINTDCB,PRINTHDR
READ     GET   SMFIN
         LR    R3,R1  * COPY PARAMETER POINTER
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
         USING CBRSMF85,R3
         CLI   SMF85RTY,X'55'  * CHECK IF TYPE 85
         BNE   IGNORE
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
CHKSTYP1 DS    0H
         CLI   SMF85STY+1,X'28'     * CHECK IF SUBTYPE 40
         BNE   IGNORE
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
* IS TYPE 85 SUBTYPE 40, SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
         LA    R4,SMF85END
         USING ST40,R4
         L     R8,SMF85OSO
```

```
         LH    R9,SMF85OSL                                              00910014
         LH    R10,SMF85OSN                                             00920014
* PROCESS THE SUMMARY ENTRIES TRIPLET.                                  00930014
* FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START                  00940014
* SECOND HW IS THE LENGTH OF EACH TRIPLET                               00950014
* THIRD HW IS THE NUMBER OF TRIPLETS                                    00960014
* FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM        00970014
* THE SMF RECORD TYPE 85 SUBTYPE 40 RECORDS.                            00980014
* STRD COMES FROM ST40STRD                                              00990014
* ENDD COMES FROM ST40ENDD                                              01000014
* ETC                                                                   01010014
*                                                                       01020014
SCOTRIP  DS    0H                                                       01030014
         LA    R4,0(R3,R8)                                              01040016
         UNPK  YYDDD(7),SMF85DTE                                        01050016
         CLI   YYDDD+1,C'0'                                             01060016
         BE   SETD0                                                     01070016
         CLI   YYDDD+1,C'1'                                             01080016
         BE   SETD1                                                     01090016
*        DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                  01100016
SETD0    MVC   YYDDD(2),=C'19'                                          01110016
         B     SETDZ                                                    01120016
SETD1    MVC   YYDDD(2),=C'20'                                          01130016
*                                                                       01140016
SETDZ    EQU   *                                                        01150016
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT             01160016
         LA    R5,100 PREPARE TO DIVIDE BY 100                          01170016
         LA    R6,0                                                     01180016
         L     R7,SMF85TME  GET THE TIME                                01190016
         DR    R6,R5   -> SECS IN R7, HUNS IN R6                        01200016
         CVD   R6,DWORD                                                 01210016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01220016
         UNPK  HUS,DWORD+6(2)                                           01230016
*        DC    F'0'                                                     01240016
* NOW GET THE SECS                                                      01250016
         LA    R5,60 PREPARE TO DIVIDE BY 60                            01260016
         LA    R6,0                                                     01270016
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6              01280016
         CVD   R6,DWORD                                                 01290016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01300016
         UNPK  SS,DWORD+6(2)                                            01310016
* NOW GET THE MINS                                                      01320016
         LA    R6,0                                                     01330016
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6               01340016
         CVD   R6,DWORD                                                 01350016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01360016
         UNPK  MM,DWORD+6(2)                                            01370016
         CVD   R7,DWORD     DO HOURS                                    01380016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01390016
         UNPK  HH,DWORD+6(2)                                            01400016
         LA    R4,0(R3,R8)                                              01410014
         LA    R5,ST40END POINT TO THE VOLUME ARRAY                     01420014
         USING ST40VOLD,R5                                              01430014
         MVC   STRD,ST40STRD    * COPY STRD                             01440014
         MVC   ENDD,ST40ENDD    * COPY ENDD                             01450014
```

```
         BINDEC VOLN           * CONVERT VOLN                           01460014
         BINDEC PCTV           * CONVERT PCTV                           01470014
         BINDEC LIM            * CONVERT LIM                            01480014
         MVC   SUBL,ST40SUBL   * COPY    SUBL                           01490014
         PUT   PRINTDCB,PRINTL1                                         01500014
         LH     R6,ST40VOLN    * GET NUMBER OF VOLUMES                  01510015
* DC F’0’ CREATE AN ABEND TO LOOK AT THE RECORDS                        01520014
VLOOP    MVC   VSN,ST40VSN                                              01530014
         PUT   PRINTDCB,PRINTL2                                         01540014
         LA    R5,6(R5)                                                 01550014
         BCT   R6,VLOOP                                                 01560014
WRITEIT  DS    0H                                                       01570014
         PUT   PRINTDCB,PRINTBLK                                        01580014
* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS                01590014
* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                           01600014
         LA    R8,0(R8,R9)                                              01610014
         BCT   R10,SCOTRIP                                              01620014
         B     READ                                                     01630014
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        01640014
         B     READ                                                     01650014
FINISH   DS    0H                                                       01660014
         SEGEND                                                         01670014
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C01680014
               EODAD=FINISH                                             01690014
PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133                 01700014
DWORD    DS  D                                                          01710014
         ORG DWORD                                                      01720014
         DC  C’12345678’                                                01730014
PRINTBLK DC  CL133’ ‘                                                   01740014
PRINTHDR DC  CL133’1SMF TYPE 85 SUBTYPE 40 RECORDS’                     01750014
PRINTL0  DC    CL133’ SMF85TJ SMFDTE/TME:’                              01760016
         ORG   PRINTL0+35                                               01770016
YYDDD    DC    CL7’ ‘,CL1’/’                                            01780016
HH       DC    CL2’ ‘,C’:’                                              01790016
MM       DC    CL2’ ‘,C’:’                                              01800016
SS       DC    CL2’ ‘,C’.’                                              01810016
HUS      DC    CL3’ ‘,CL1’ ‘                                            01820016
         ORG                                                            01830016
PRINTL1  DC  CL133’ STRD/ENDD/VOLN/PCTV/LIM/SUBL:’                      01840014
         ORG PRINTL1+31                                                 01850014
STRD     DC  CL10’ ‘,C’/’                                               01860014
ENDD     DC  CL10’ ‘,C’/’                                               01870014
VOLN     DC  CL12’ ‘,C’/’      CONVERTED FROM BINARY                    01880014
PCTV     DC  CL12’ ‘,C’/’      CONVERTED FROM BINARY                    01890014
LIM      DC  CL12’ ‘,C’/’      CONVERTED FROM BINARY                    01900014
SUBL     DC  CL1’ ‘,C’ ‘       COPIED                                   01910014
         ORG                                                            01920014
PRINTL2  DC    CL133’ VSN:’                                             01930014
         ORG   PRINTL2+5                                                01940014
VSN      DC    CL6’ ‘                                                   01950014
         ORG                                                            01960014
SMFDSECT DSECT                                                          01970014
         IFASMFR (85) THIS INCLUDES CBRSMF MACRO                        01980014
         END                                                            01990014
```

A.3.7.3 Storing and running the JCL to build the LOAD module SMF85TJ

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-53 on page 265 into your PDS MHLRES1.SMF85TJ.DFSMS13.PDS as member SMF85TJJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, “OAM SMF85 analysis source materials” on page 263 as member BUILDJCL.
2. Update the JCL to refer to the USER you want to use. Update the SET statement to refer to the appropriate program that is being built, in this case SMF85TJ.
3. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TJ can now be run by using the JCL shown in Figure A-29 on page 245.

A.3.8 SMF Record type 85 subtypes 90-93 data display program SMF85TP

Program SMG85TP displays the contents of selected fields of SMF record Type 85 subtypes 90-93 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that immediate backup is occurring.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.8.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, “OAM SMF85 Analysis program common data set creation” on page 264. If the sample JCL is used, the ‘SET’ to be specified is SMF85TP.

When USER=MHLRES2 and SET=SMF85TP, the following data sets result:

- MHLRES2.SMF85TP.DFSMS13.PDS
- MHLRES2.SMF85TP.DFSMS13.LOAD

A.3.8.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session. To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

> **Attention:** The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-9 into member SMF85TPA of data set MHLRES2.SMF85TP.DFSMS13.PDS. The result should contain 270 lines.

> **Tip:** In the code in Example A-9 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-9 SMF85TP program source code

```
 MACRO                                                         00010000
&NAME    SEGSTART                                                   00020000
&NAME    STM   14,12,12(13)           SAVE HIS REGS IN HIS SAVE AREA   00030000
R0       EQU   0                                                    00040000
```

```
R1       EQU   1                                                00050000
R2       EQU   2                                                00060000
R3       EQU   3                                                00070000
R4       EQU   4                                                00080000
R5       EQU   5                                                00090000
R6       EQU   6                                                00100000
R7       EQU   7                                                00110000
R8       EQU   8                                                00120000
R9       EQU   9                                                00130000
R10      EQU   10                                               00140000
R11      EQU   11                                               00150000
RB       EQU   12                                               00160000
R13      EQU   13                                               00170000
R14      EQU   14                                               00180000
R15      EQU   15                                               00190000
         BALR  12,0                SET UP ADDRESSABILITY        00200000
         USING *,12                USE REG 12 AS BASE REG       00210000
         ST    13,SAVEREGS+4       SAVE @ OF HIS SAVEAREA IN MINE   00220000
         LA    03,SAVEREGS         LOAD @ OF MY SAVE AREA IN REG 3  00230000
         ST    03,8(13)            SAVE @ OF MY SAVE AREA IN HIS    00240000
         LR    13,03               LOAD @ OF MY SAVE AREA IN REG 13 00250000
         MEND                                                   00260000
         MACRO                                                  00270000
         SEGEND                                                 00280000
         L     13,SAVEREGS+4       LOAD REG13 WITH @ OF HIS SAVE    00290000
         LM    14,12,12(13)        RESTORE REGS FROM HIS SAVEAREA   00300000
         XR    R15,R15                                          00310000
         BR    14                  RETURN TO CALLING RTN VIA REG 14 00320000
SAVEREGS DC    18F'0'              SET UP SAVE AREA             00330000
         MEND                                                   00340000
SMFR85T0  SEGSTART                                              00350000
*  THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF   00360000
*  THE SMF TYPE 85 SUBTYPE 90, 91, 92, 93                       00370044
*  WHICH ARE THE OAM LCS FILE SYSTEM RECORDS                    00380044
*  IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED    00390000
*  TO SELECT TYPE 85 SUBTYPE 90,91,92 OR 93                     00400044
*  RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR         00410000
*  OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.       00420000
*                                                               00430000
*  THE STANDARD SMF RECORD MAPPING MACROS ARE USED.             00440000
*  REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD          00450000
*  R3    START OF WHOLE RECORD                                  00460000
*  THERE IS 1 DSECTS TO BE MAPPED                               00470000
*  R4    START OF ST90 MAPPING FOR 90,91,92,93                  00480044
*  R5    SPARE                                                  00490000
*  R6    SPARE                                                  00500000
*  R7    SPARE                                                  00510000
*  OTHER REGISTER USES                                          00520000
*  R12   OVERALL BASE REGISTER                                  00530000
*  R8    RECORD TYPE/SUBTYPE CHECKING/WORKING                   00540000
*  R9    LENGTH OF PARTICULAR DSECT                             00550000
*  R10   NUMBER OF ENTRIES IN THE TRIPLET                       00560000
*                                                               00570000
*  QSAM GET LOCATE PROCESSING IS USED                           00580000
*                                                               00590000
```

```
         OPEN SMFIN
         OPEN (PRINTDCB,(OUTPUT))
         PUT  PRINTDCB,PRINTHDR
READ     GET SMFIN
         LR    R3,R1  * COPY PARAMETER POINTER
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
         USING CBRSMF85,R3
         CLI   SMF85RTY,X'55'  *   CHECK IF TYPE 85
         BNE   IGNORE
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS
CHKSTYP1 DS    0H
*   CHECK IF ANY OF SUBTYPE 90-93
         CLI   SMF85STY+1,X'5A'
         BNE   *+18
         MVI   STYPE,C'0'
         MVC   FUNC,=CL34'(LCS FILE SYSTEM WRITE REQUEST)'
         B     STOK
         CLI   SMF85STY+1,X'5B'
         BNE   *+18
         MVI   STYPE,C'1'
         MVC   FUNC,=CL34'(LCS FILE SYSTEM READ REQUEST)'
         B     STOK
         CLI   SMF85STY+1,X'5C'
         BNE   *+18
         MVI   STYPE,C'2'
         MVC   FUNC,=CL34'(LCS FILE SYSTEM PHYSICAL DELETE)'
         B     STOK
         CLI   SMF85STY+1,X'5D'
         BNE   *+18
         MVI   STYPE,C'3'
         MVC   FUNC,=CL34'(LCS FILE SYS DELETE-STORE CLEANUP)'
         B     STOK
         B     IGNORE    *        OTHERWISE IGNORE
STOK     EQU   *
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS
* IS TYPE 85 SUBTYPE 90/91/92/93 SO EXTRACT THE DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
         LA    R4,SMF85END
         USING ST90,R4
         L     R8,SMF85OSO
         LH    R9,SMF85OSL
         LH    R10,SMF85OSN
*  PROCESS THE SUMMARY ENTRIES TRIPLET.
*  FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START
*  SECOND HW IS THE LENGTH OF EACH TRIPLET
*  THIRD HW IS THE NUMBER OF TRIPLETS
*  FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM
*  THE SMF RECORD TYPE 85 SUBTYPE 90/91/92/93 RECORDS
*  COLN COMES FROM ST90COLN
*  CNID COMES FROM ST90CNID
```

```
*  ETC                                                                  01150000
*  ST90FLGS IS INTERPRETED AS FLG0 ON OR OFF                            01160044
*                                                                       01170000
SCOTRIP  DS    0H                                                       01180000
         LA    R4,0(R3,R8)                                              01190062
         LA    R4,0(R3,R8)                                              01200062
         UNPK  YYDDD(7),SMF85DTE                                        01210062
         CLI   YYDDD+1,C'0'                                             01220062
         BE   SETD0                                                     01230062
         CLI   YYDDD+1,C'1'                                             01240062
         BE   SETD1                                                     01250062
         DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                  01260062
SETD0    MVC   YYDDD(2),=C'19'                                          01270062
         B     SETDZ                                                    01280062
SETD1    MVC   YYDDD(2),=C'20'                                          01290062
*                                                                       01300062
SETDZ    EQU   *                                                        01310062
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT             01320062
         LA    R5,100 PREPARE TO DIVIDE BY 100                          01330062
         LA    R6,0                                                     01340062
         L     R7,SMF85TME  GET THE TIME                                01350062
         DR    R6,R5   -> SECS IN R7, HUNS IN R6                        01360062
         CVD   R6,DWORD                                                 01370062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01380062
         UNPK  HUS,DWORD+6(2)                                           01390062
*        DC    F'0'                                                     01400062
* NOW GET THE SECS                                                      01410062
         LA    R5,60 PREPARE TO DIVIDE BY 60                            01420062
         LA    R6,0                                                     01430062
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6              01440062
         CVD   R6,DWORD                                                 01450062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01460062
         UNPK  SS,DWORD+6(2)                                            01470062
* NOW GET THE MINS                                                      01480062
         LA    R6,0                                                     01490062
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6               01500062
         CVD   R6,DWORD                                                 01510062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01520062
         UNPK  MM,DWORD+6(2)                                            01530062
         CVD   R7,DWORD     DO HOURS                                    01540062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01550062
         UNPK  HH,DWORD+6(2)                                            01560062
         PUT   PRINTDCB,PRINTL0                                         01570062
         LA    R4,0(R3,R8)                                              01580000
         MVC   COLN,ST90COLN                                            01590044
         MVC   DIR,ST90DIR                                              01600049
         PUT   PRINTDCB,PRINTL1                                         01610056
         MVC   SGN,ST90SGN                                              01620065
         MVC   COLN,ST90COLN                                            01630049
         PUT   PRINTDCB,PRINTL2                                         01640065
         MVC   OBJN,ST90OBJN                                            01650065
*  CONVERT FST FLAG                                                     01660049
         MVC   FST,=CL4'NFS'                                            01670057
         CLC   ST90FST,=C'01' IS IT NFS                                 01680051
         BE    FLGNFS                                                   01690050
```

```
         MVC   FST,=CL4'ZFS'   OTHERWISE CHANGE TO ZFS                  01700057
FLGNFS   EQU *                                                          01710050
*  CONVERT INST                                                         01720049
         UNPK  INST(09),ST90INST(5)                                     01730049
         TR    INST(08),HEXTAB-240                                      01740049
         MVI   INST+8,X'40'                                             01750049
*  CONVERT FLAGS                                                        01760049
         UNPK  FLGS(09),ST90FLGS(5)                                     01770065
         TR    FLGS(08),HEXTAB-240                                      01780065
         MVI   FLGS+8,X'40'                                             01790065
         PUT   PRINTDCB,PRINTL3                                         01800067
*  CONVERT OLEN                                                         01810049
         UNPK  OLEN(09),ST90OLEN(5)                                     01820049
         TR    OLEN(08),HEXTAB-240                                      01830049
         MVI   OLEN+8,X'40'                                             01840049
*  CONVERT OOFF                                                         01850049
         UNPK  OOFF(09),ST90OOFF(5)                                     01860049
         TR    OOFF(08),HEXTAB-240                                      01870049
         MVI   OOFF+8,X'40'                                             01880049
*  CONVERT LIQT                                                         01890049
         UNPK  LIQT(09),ST90LIQT(5)                                     01900049
         TR    LIQT(08),HEXTAB-240                                      01910049
         MVI   LIQT+8,X'40'                                             01920049
*  CONVERT LDQT                                                         01930049
         UNPK  LDQT(09),ST90LDQT(5)                                     01940049
         TR    LDQT(08),HEXTAB-240                                      01950049
         MVI   LDQT+8,X'40'                                             01960049
*  CONVERT LEQT                                                         01970049
         UNPK  LEQT(09),ST90LEQT(5)                                     01980049
         TR    LEQT(08),HEXTAB-240                                      01990049
         MVI   LEQT+8,X'40'                                             02000049
*  CONVERT RC                                                           02010049
         UNPK  RC(09),ST90RC(5)                                         02020049
         TR    RC(08),HEXTAB-240                                        02030049
         MVI   RC+8,X'40'                                               02040049
*        DC    H'0'                                                     02050070
*  CONVERT RS                                                           02060049
         UNPK  RS(09),ST90RS(5)                                         02070049
         TR    RS(08),HEXTAB-240                                        02080049
         MVI   RS+8,X'40'                                               02090049
         PUT   PRINTDCB,PRINTL4                                         02100054
WRITEIT  DS    0H                                                       02110000
         PUT   PRINTDCB,PRINTBLK                                        02120000
*  LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS               02130000
*  WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                          02140000
         LA    R8,0(R8,R9)                                              02150000
         BCT   R10,SCOTRIP                                              02160000
         B     READ                                                     02170000
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        02180000
         B     READ                                                     02190000
FINISH   DS    0H                                                       02200000
         SEGEND                                                         02210000
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C02220000
               EODAD=FINISH                                             02230000
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133               02240000
```

```
DWORD    DS    D                                                        02250000
         ORG   DWORD                                                    02260000
         DC    C’12345678’                                              02270000
HEXTAB   DC    C’0123456789ABCDEF’ TRANSLATE TABLE                      02280042
PRINTBLK DC    CL133’ ‘                                                 02290000
PRINTHDR DC    CL133’1SMF TYPE 85 SUBTYPE 90-93 RECORDS’                02300045
PRINTL0  DC    CL133’ SMF85TP SMFDTE/TME:’                              02310071
         ORG   PRINTL0+33                                               02320068
YYDDD    DC    CL7’ ‘,CL1’/’                                            02330064
HH       DC    CL2’ ‘,C’:’                                              02340064
MM       DC    CL2’ ‘,C’:’                                              02350064
SS       DC    CL2’ ‘,C’.’                                              02360064
HUS      DC    CL3’ ‘,CL1’ ‘                                            02370064
         ORG                                                            02380062
PRINTL1  DC    CL133’ STYPE/FUNC/DIR/:’                                 02390065
         ORG   PRINTL1+33                                               02400068
STYPE9   DC    CL1’9’ PREFIX TO SUBTYPES 90-93                          02410045
STYPE    DC    CL1’ ‘,C’/’                                              02420064
FUNC     DC    CL34’ ‘,C’/’                                             02430064
DIR      DC    CL30’ ‘,C’ ‘                                             02440066
         ORG                                                            02450000
PRINTL2  DC    CL133’ SGN/COLN/:’                                       02460065
         ORG   PRINTL2+33                                               02470068
SGN      DC    CL8’ ‘,C’ ‘                                              02480065
COLN     DC    CL44’ ‘,C’ ‘                                             02490066
         ORG                                                            02500052
PRINTL3  DC    CL133’ OBJN/FST/INST/FLGS:’                              02510067
         ORG   PRINTL3+33                                               02520068
OBJN     DC    CL44’ ‘,C’/’                                             02530065
FST      DC    CL4’ ‘,C’ ‘                                              02540065
INST     DC    CL8’ ‘,CL1’/’   CONVERTED FROM BL4                       02550064
FLGS     DC    CL8’ ‘,CL1’/’   CONVERTED FROM BL4                       02560064
         ORG                                                            02570000
PRINTL4  DC    CL133’ OLEN/OOFF/LIQT/LDQT/LEQT/RC/RS:’                  02580067
         ORG   PRINTL4+33                                               02590068
OLEN     DC    CL8’ ‘,CL1’/’   CONVERTED FROM BL4                       02600067
OOFF     DC    CL8’ ‘,CL1’/’ CONVERTED FROM BL4                         02610068
LIQT     DC    CL8’ ‘,CL1’/’                                            02620064
LDQT     DC    CL8’ ‘,CL1’/’                                            02630064
LEQT     DC    CL8’ ‘,CL1’/’     CONVERTED FROM BL4                     02640064
RC       DC    CL8’ ‘,CL1’/’                                            02650064
RS       DC    CL8’ ‘,C’ ‘                                              02660064
         ORG                                                            02670000
SMFDSECT DSECT                                                          02680000
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                       02690000
         END                                                            02700000
```

A.3.8.3 Storing and running the JCL to build the LOAD module SMF85TP

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-9 on page 227 into your PDS MHLRES2.SMF85TP.DFSMS13.PDS as member SMFT85PJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, “OAM SMF85 analysis source materials” on page 263 as member BUILDJCL.
2. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TP can now be run by using the JCL shown in Figure A-32 on page 247.

A.3.9 SMF Record type 85 subtypes 80-81 data display program SMF85TR

Program SMG85TR displays the contents of selected fields of SMF record Type 85 subtypes 80-81 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that immediate backup is occurring.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.9.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, “OAM SMF85 Analysis program common data set creation” on page 264. If the sample JCL is used, the SET to be specified is SMF85TR.

When USER=MHLRES2 and SET=SMF85TR, the following data sets result:

- MHLRES2.SMF85TR.DFSMS13.PDS
- MHLRES2.SMF85TR.DFSMS13.LOAD

A.3.9.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session.

To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

> **Attention:** The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-10 into member SMF85TRA of data set MHLRES2.SMF85TR.DFSMS13.PDS. The result should contain 217 lines.

> **Tip:** In the code in Example A-10, on the line that starts with SMFIN there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-10 SMF85TR program source code

```
 MACRO                                                         00010000
&NAME    SEGSTART                                                      00020000
&NAME    STM   14,12,12(13)            SAVE HIS REGS IN HIS SAVE AREA   00030000
R0       EQU   0                                                       00040000
R1       EQU   1                                                       00050000
R2       EQU   2                                                       00060000
```

```
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11
RB       EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         BALR  12,0                   SET UP ADDRESSABILITY
         USING *,12                   USE REG 12 AS BASE REG
         ST    13,SAVEREGS+4          SAVE @ OF HIS SAVEAREA IN MINE
         LA    03,SAVEREGS            LOAD @ OF MY SAVE AREA IN REG 3
         ST    03,8(13)               SAVE @ OF MY SAVE AREA IN HIS
         LR    13,03                  LOAD @ OF MY SAVE AREA IN REG 13
         MEND
         MACRO
         SEGEND
         L     13,SAVEREGS+4          LOAD REG13 WITH @ OF HIS SAVE
         LM    14,12,12(13)           RESTORE REGS FROM HIS SAVEAREA
         XR    R15,R15
         BR    14                     RETURN TO CALLING RTN VIA REG 14
SAVEREGS DC    18F'0'                 SET UP SAVE AREA
         MEND
SMFR85TO  SEGSTART
*  THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF
*  THE SMF TYPE 85 SUBTYPES 80 AND 81
*  WHICH ARE THE OAM LCS FILE SYSTEM RECORDS
*  IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
*  TO SELECT TYPE 85 SUBTYPE 80 AND/OR 81
*  RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR
*  OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.
*
*  THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
*  REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
*  R3    START OF WHOLE RECORD
*  THERE IS 1 DSECTS TO BE MAPPED
*  R4    START OF ST80 MAPPING FOR 80,81
*  R5    SPARE
*  R6    SPARE
*  R7    SPARE
*  OTHER REGISTER USES
*  R12   OVERALL BASE REGISTER
*  R8    RECORD TYPE/SUBTYPE CHECKING/WORKING
*  R9    LENGTH OF PARTICULAR DSECT
*  R10   NUMBER OF ENTRIES IN THE TRIPLET
*
*  QSAM GET LOCATE PROCESSING IS USED
*
         OPEN SMFIN
         OPEN (PRINTDCB,(OUTPUT))
```

```
         PUT  PRINTDCB,PRINTHDR                                         00620000
READ     GET SMFIN                                                      00630000
         LR    R3,R1  * COPY PARAMETER POINTER                          00640064
* R3 -> SMF RECORD                                                      00650000
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION                         00660000
         USING CBRSMF85,R3                                              00670046
         CLI   SMF85RTY,X'55'  *   CHECK IF TYPE 85                     00680064
         BNE   IGNORE                                                   00690000
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             00700000
CHKSTYP1 DS    0H                                                       00710045
*   CHECK IF ANY OF SUBTYPE 80-91                                       00720065
         CLI   SMF85STY+1,X'50'                                         00730065
         BNE   *+18                                                     00740045
         MVI   STYPE,C'0'                                               00750045
         MVC   FUNC,=CL34'(LCS TAPE LIBRARY VARY ONLINE)'               00760065
         B     STOK                                                     00770045
         CLI   SMF85STY+1,X'51'                                         00780065
         BNE   *+18                                                     00790045
         MVI   STYPE,C'1'                                               00800045
         MVC   FUNC,=CL34'(LCS TAPE LIBRARY VARY OFFLINE)'              00810065
         B     STOK                                                     00820045
         B     IGNORE    *        OTHERWISE IGNORE                      00830064
STOK     EQU   *                                                        00840045
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             00850000
* IS TYPE 85 SUBTYPE 90/91/92/93 SO EXTRACT THE DATA                    00860044
* R3 IS THE START OF THE WHOLE RECORD                                   00870000
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.               00880000
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION        00890000
* ADD R8 TO R3                                                          00900000
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS                           00910000
         LA    R4,SMF85END                                              00920047
         USING ST80,R4                                                  00930065
         L     R8,SMF85OSO                                              00940047
         LH    R9,SMF85OSL                                              00950047
         LH    R10,SMF85OSN                                             00960047
*  PROCESS THE SUMMARY ENTRIES TRIPLET.                                 00970000
*  FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START                 00980000
*  SECOND HW IS THE LENGTH OF EACH TRIPLET                              00990000
*  THIRD HW IS THE NUMBER OF TRIPLETS                                   01000000
*  FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM       01010000
*  THE SMF RECORD TYPE 85 SUBTYPE 90/91/92/93 RECORDS                   01020044
*  COLN COMES FROM ST80COLN                                             01030065
*  CNID COMES FROM ST80CNID                                             01040065
*  ETC                                                                  01050000
*  ST80FLGS IS INTERPRETED AS FLG0 ON OR OFF                            01060065
*                                                                       01070000
SCOTRIP  DS    0H                                                       01080000
         LA    R4,0(R3,R8)                                              01090062
         LA    R4,0(R3,R8)                                              01100062
         UNPK  YYDDD(7),SMF85DTE                                        01110062
         CLI   YYDDD+1,C'0'                                             01120062
         BE   SETD0                                                     01130062
         CLI   YYDDD+1,C'1'                                             01140062
         BE   SETD1                                                     01150062
         DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                  01160062
```

```
SETD0    MVC   YYDDD(2),=C'19'                                          01170062
         B     SETDZ                                                    01180062
SETD1    MVC   YYDDD(2),=C'20'                                          01190062
*                                                                       01200062
SETDZ    EQU   *                                                        01210062
*     CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT            01220062
         LA    R5,100 PREPARE TO DIVIDE BY 100                          01230062
         LA    R6,0                                                     01240062
         L     R7,SMF85TME  GET THE TIME                                01250062
         DR    R6,R5   -> SECS IN R7, HUNS IN R6                        01260062
         CVD   R6,DWORD                                                 01270062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01280062
         UNPK  HUS,DWORD+6(2)                                           01290062
*        DC    F'0'                                                     01300062
* NOW GET THE SECS                                                      01310062
         LA    R5,60 PREPARE TO DIVIDE BY 60                            01320062
         LA    R6,0                                                     01330062
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6              01340062
         CVD   R6,DWORD                                                 01350062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01360062
         UNPK  SS,DWORD+6(2)                                            01370062
* NOW GET THE MINS                                                      01380062
         LA    R6,0                                                     01390062
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6               01400062
         CVD   R6,DWORD                                                 01410062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01420062
         UNPK  MM,DWORD+6(2)                                            01430062
         CVD   R7,DWORD     DO HOURS                                    01440062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01450062
         UNPK  HH,DWORD+6(2)                                            01460062
         PUT   PRINTDCB,PRINTL0                                         01470062
         LA    R4,0(R3,R8)                                              01480000
         MVC   TLN,ST80TLN                                              01490066
         MVC   TLDT,ST80TLDT                                            01500067
         PUT   PRINTDCB,PRINTL1                                         01510056
*  CONVERT LTQT                                                         01520066
         UNPK  LTQT(09),ST80LTQT(5)                                     01530066
         TR    LTQT(08),HEXTAB-240                                      01540066
         MVI   LTQT+8,X'40'                                             01550066
*  CONVERT LTPT                                                         01560066
         UNPK  LTPT(09),ST80LTPT(5)                                     01570066
         TR    LTPT(08),HEXTAB-240                                      01580066
         MVI   LTPT+8,X'40'                                             01590066
*  CONVERT RC                                                           01600066
         UNPK  RC(09),ST80RC(5)                                         01610066
         TR    RC(08),HEXTAB-240                                        01620066
         MVI   RC+8,X'40'                                               01630066
*  CONVERT RS                                                           01640066
         UNPK  RS(09),ST80RS(5)                                         01650066
         TR    RS(08),HEXTAB-240                                        01660066
         MVI   RS+8,X'40'                                               01670066
         PUT   PRINTDCB,PRINTL2                                         01680056
*  CONVERT FLAGS                                                        01690049
         MVC   FLGS,=CL20'FLGS RSV'                                     01700067
WRITEIT  DS    0H                                                       01710000
```

```
         PUT   PRINTDCB,PRINTBLK                                      01720000
*  LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS             01730000
*  WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                        01740000
         LA    R8,0(R8,R9)                                            01750000
         BCT   R10,SCOTRIP                                            01760000
         B     READ                                                   01770000
IGNORE   DS    OH EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS      01780000
         B     READ                                                   01790000
FINISH   DS    OH                                                     01800000
         SEGEND                                                       01810000
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,           C01820000
               EODAD=FINISH                                           01830000
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133             01840000
DWORD    DS    D                                                      01850000
         ORG   DWORD                                                  01860000
         DC    C’12345678’                                            01870000
HEXTAB   DC    C’0123456789ABCDEF’ TRANSLATE TABLE                    01880042
PRINTBLK DC    CL133’ ‘                                               01890000
PRINTHDR DC    CL133’1SMF TYPE 85 SUBTYPE 80-81 RECORDS’              01900065
PRINTL0  DC    CL133’ SMF85TR SMFDTE/TME:’                            01910068
         ORG   PRINTL0+31                                             01920063
YYDDD    DC    CL7’ ‘,CL1’/’                                          01930064
HH       DC    CL2’ ‘,C’:’                                            01940064
MM       DC    CL2’ ‘,C’:’                                            01950064
SS       DC    CL2’ ‘,C’.’                                            01960064
HUS      DC    CL3’ ‘,CL1’ ‘                                          01970064
         ORG                                                          01980062
PRINTL1  DC    CL133’ STYPE/FUNC/TLN/TLDT:’                           01990065
         ORG   PRINTL1+31                                             02000037
STYPE9   DC    CL1’8’ PREFIX TO SUBTYPES 80-81                        02010065
STYPE    DC    CL1’ ‘,C’/’                                            02020064
FUNC     DC    CL34’ ‘,C’/’                                           02030064
TLN      DC    CL08’ ‘,C’/’                                           02040065
TLDT     DC    CL8’ ‘,C’ ‘                                            02050065
         ORG                                                          02060000
PRINTL2  DC    CL133’ COLN/OBJN/FST:’                                 02070057
         ORG   PRINTL2+33                                             02080060
LTQT     DC    CL8’ ‘,CL1’/’                                          02090065
LTPT     DC    CL8’ ‘,CL1’/’                                          02100065
RC       DC    CL8’ ‘,CL1’/’                                          02110064
RS       DC    CL8’ ‘,C’ ‘                                            02120064
FLGS     DC    CL8’ ‘,C’ ‘ RESERVED                                   02130065
         ORG                                                          02140000
SMFDSECT DSECT                                                        02150000
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                     02160000
         END                                                          02170000
```

A.3.9.3 Storing and running the JCL to build the LOAD module SMF85TR

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-9 on page 227 into your PDS MHLRES2.SMF85TR.DFSMS13.PDS as member SMFT85RJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, "OAM SMF85 analysis source materials" on page 263 as member BUILDJCL.
2. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TR can now be run by using the JCL shown in Figure A-35 on page 249.

A.3.10 SMF Record type 85 subtype 87 data display program SMF85TS

Program SMG85TS displays the contents of selected fields of SMF record Type 85 subtype 87 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that immediate backup is occurring.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.10.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, "OAM SMF85 Analysis program common data set creation" on page 264. If the sample JCL is used, the SET to be specified is SMF85TS.

When USER=MHLRES2 and SET=SMF85TS, the following data sets result:

- MHLRES2.SMF85TS.DFSMS13.PDS
- MHLRES2.SMF85TS.DFSMS13.LOAD

A.3.10.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session. To use the FTP process, see A.3.1.5, "SMF85 program restore process" on page 265. If using the copy and paste process, see A.3.1.5, "SMF85 program restore process" on page 265.

> **Attention:** The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, "SMF85 program restore process" on page 265. Copy and paste the contents of Example A-11 into member SMF85TSA of data set MHLRES2.SMF85TS.DFSMS13.PDS. The result should contain 273 lines.

> **Tip:** In the code in Example A-11 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-11 SMF85TS program source code

```
 MACRO                                                          00010000
&NAME    SEGSTART                                                       00020000
&NAME    STM   14,12,12(13)            SAVE HIS REGS IN HIS SAVE AREA   00030000
R0       EQU   0                                                        00040000
R1       EQU   1                                                        00050000
R2       EQU   2                                                        00060000
R3       EQU   3                                                        00070000
```

```
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11
RB       EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         BALR  12,0                         SET UP ADDRESSABILITY
         USING *,12                         USE REG 12 AS BASE REG
         ST    13,SAVEREGS+4                SAVE @ OF HIS SAVEAREA IN MINE
         LA    03,SAVEREGS                  LOAD @ OF MY SAVE AREA IN REG 3
         ST    03,8(13)                     SAVE @ OF MY SAVE AREA IN HIS
         LR    13,03                        LOAD @ OF MY SAVE AREA IN REG 13
         MEND
         MACRO
         SEGEND
         L     13,SAVEREGS+4                LOAD REG13 WITH @ OF HIS SAVE
         LM    14,12,12(13)                 RESTORE REGS FROM HIS SAVEAREA
         XR    R15,R15
         BR    14                           RETURN TO CALLING RTN VIA REG 14
SAVEREGS DC    18F'0'                       SET UP SAVE AREA
         MEND
SMFR85TO  SEGSTART
*  THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF
*  THE SMF TYPE 85 SUBTYPE 87
*  WHICH ARE THE OAM OWNED TAPE VOLUME DEMOUNT RECORDS
*  IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
*  TO SELECT TYPE 85 SUBTYPE 87
*  RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR
*  OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.
*
*  THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
*  REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
*  R3    START OF WHOLE RECORD
*  THERE IS 1 DSECTS TO BE MAPPED
*  R4    START OF ST87 MAPPING FOR 87
*  R5    SPARE
*  R6    SPARE
*  R7    SPARE
*  OTHER REGISTER USES
*  R12   OVERALL BASE REGISTER
*  R8    RECORD TYPE/SUBTYPE CHECKING/WORKING
*  R9    LENGTH OF PARTICULAR DSECT
*  R10   NUMBER OF ENTRIES IN THE TRIPLET
*
*  QSAM GET LOCATE PROCESSING IS USED
*
         OPEN SMFIN
         OPEN (PRINTDCB,(OUTPUT))
         PUT  PRINTDCB,PRINTHDR
```

```
READ     GET SMFIN                                                      00630000
         LR    R3,R1  * COPY PARAMETER POINTER                          00640064
* R3 -> SMF RECORD                                                      00650000
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION                         00660000
         USING CBRSMF85,R3                                              00670046
         CLI   SMF85RTY,X'55'  *   CHECK IF TYPE 85                     00680064
         BNE   IGNORE                                                   00690000
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             00700000
CHKSTYP1 DS    0H                                                       00710045
*   CHECK IF ANY OF SUBTYPE 87                                          00720068
         CLI   SMF85STY+1,X'57'                                         00730068
         BNE   *+18                                                     00740045
         MVI   STYPE,C'7'                                               00750082
         MVC   FUNC,=CL34'(OAM OWNED TAPE VOLUME DEMOUNT)'              00760082
         B     STOK                                                     00770045
         B     IGNORE    *        OTHERWISE IGNORE                      00780064
STOK     EQU   *                                                        00790045
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             00800000
* IS TYPE 85 SUBTYPE 87 SO EXTRACT THE DATA                             00810068
* R3 IS THE START OF THE WHOLE RECORD                                   00820000
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.               00830000
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION        00840000
* ADD R8 TO R3                                                          00850000
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS                           00860000
         LA    R4,SMF85END                                              00870047
         USING ST87,R4                                                  00880068
         L     R8,SMF85OSO                                              00890047
         LH    R9,SMF85OSL                                              00900047
         LH    R10,SMF85OSN                                             00910047
*  PROCESS THE SUMMARY ENTRIES TRIPLET.                                 00920000
*  FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START                 00930000
*  SECOND HW IS THE LENGTH OF EACH TRIPLET                              00940000
*  THIRD HW IS THE NUMBER OF TRIPLETS                                   00950000
*  FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM       00960000
*  THE SMF RECORD TYPE 85 SUBTYPE 90/91/92/93 RECORDS                   00970044
*  COLN COMES FROM ST87COLN                                             00980068
*  CNID COMES FROM ST87CNID                                             00990068
*  ETC                                                                  01000000
*  ST87FLGS IS INTERPRETED AS FLGO ON OR OFF                            01010068
*                                                                       01020000
SCOTRIP  DS    0H                                                       01030000
         LA    R4,0(R3,R8)                                              01040062
         LA    R4,0(R3,R8)                                              01050062
         UNPK  YYDDD(7),SMF85DTE                                        01060062
         CLI   YYDDD+1,C'0'                                             01070062
         BE   SETD0                                                     01080062
         CLI   YYDDD+1,C'1'                                             01090062
         BE   SETD1                                                     01100062
         DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                  01110062
SETD0    MVC   YYDDD(2),=C'19'                                          01120062
         B     SETDZ                                                    01130062
SETD1    MVC   YYDDD(2),=C'20'                                          01140062
*                                                                       01150062
SETDZ    EQU   *                                                        01160062
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT             01170062
```

```
         LA    R5,100 PREPARE TO DIVIDE BY 100                          01180062
         LA    R6,0                                                     01190062
         L     R7,SMF85TME  GET THE TIME                                01200062
         DR    R6,R5   -> SECS IN R7, HUNS IN R6                        01210062
         CVD   R6,DWORD                                                 01220062
         OI    DWORD+7,X’OF’  FIX THE SIGN FOR PRINTING                 01230062
         UNPK  HUS,DWORD+6(2)                                           01240062
*        DC    F’0’                                                     01250062
* NOW GET THE SECS                                                      01260062
         LA    R5,60 PREPARE TO DIVIDE BY 60                            01270062
         LA    R6,0                                                     01280062
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6              01290062
         CVD   R6,DWORD                                                 01300062
         OI    DWORD+7,X’OF’  FIX THE SIGN FOR PRINTING                 01310062
         UNPK  SS,DWORD+6(2)                                            01320062
* NOW GET THE MINS                                                      01330062
         LA    R6,0                                                     01340062
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6               01350062
         CVD   R6,DWORD                                                 01360062
         OI    DWORD+7,X’OF’  FIX THE SIGN FOR PRINTING                 01370062
         UNPK  MM,DWORD+6(2)                                            01380062
         CVD   R7,DWORD     DO HOURS                                    01390062
         OI    DWORD+7,X’OF’  FIX THE SIGN FOR PRINTING                 01400062
         UNPK  HH,DWORD+6(2)                                            01410062
         PUT   PRINTDCB,PRINTL0                                         01420062
         LA    R4,0(R3,R8)                                              01430000
         PUT   PRINTDCB,PRINTL1                                         01440077
         MVC   TDDN,ST87TDDN                                            01450068
*        MVC   TDDT,ST87TDDT                                            01460079
*  CONVERT TDDT                                                         01470079
         UNPK  TDDT(09),ST87TDDT(5)                                     01480079
         TR    TDDT(08),HEXTAB-240                                      01490079
         MVI   TDDT+8,X’40’                                             01500079
         MVC   TVUN,ST87TVUN                                            01510068
         MVC   VSN,ST87VSN                                              01520069
         MVC   TMT,ST87TMT                                              01530068
         MVC   TVT,ST87TVT                                              01540068
         PUT   PRINTDCB,PRINTL2                                         01550077
         MVC   SGN,ST87SGN                                              01560068
*  CONVERT RC                                                           01570066
         UNPK  RC(09),ST87RC(5)                                         01580068
         TR    RC(08),HEXTAB-240                                        01590066
         MVI   RC+8,X’40’                                               01600066
*  CONVERT RS                                                           01610066
         UNPK  RS(09),ST87RS(5)                                         01620068
         TR    RS(08),HEXTAB-240                                        01630066
         MVI   RS+8,X’40’                                               01640066
*  CONVERT FLAGS                                                        01650049
          UNPK  FLGS(09),ST87FLGS(5)  UNPK 1 MORE THAN NEEDED           01660072
          MVI   FLGS+8,C’ ‘ BLANK OUT THE EXTRA BYTE                    01670072
          NC    FLGS(08),=8X’0F’                                        01680072
          TR    FLGS(8),HEXTAB                                          01690074
         PUT   PRINTDCB,PRINTL3                                         01700077
*  CONVERT TMNT                                                         01710072
         UNPK  TMNT(09),ST87TMNT(5)                                     01720072
```

```
         TR    TMNT(08),HEXTAB-240                                      01730072
         MVI   TMNT+8,X’40’                                             01740072
*  CONVERT NOW                                                          01750072
         UNPK  NOW(09),ST87NOW(5)                                       01760072
         TR    NOW(08),HEXTAB-240                                       01770072
         MVI   NOW+8,X’40’                                              01780072
*  CONVERT NKBW      MOVED TO NBW                                       01790072
         UNPK  NKBW(09),ST87NKBW(5)                                     01800072
         TR    NKBW(08),HEXTAB-240                                      01810072
         MVI   NKBW+8,X’40’                                             01820072
*  CONVERT NOR                                                          01830072
         UNPK  NOR(09),ST87NOR(5)                                       01840072
         TR    NOR(08),HEXTAB-240                                       01850072
         MVI   NOR+8,X’40’                                              01860072
*  CONVERT NKBR      MOVED TO NBR                                       01870072
         UNPK  NKBR(09),ST87NKBR(5)                                     01880072
         TR    NKBR(08),HEXTAB-240                                      01890072
         MVI   NKBR+8,X’40’                                             01900072
*  CONVERT NBW                                                          01910072
        UNPK  TRWORK(15),ST87NBW+1(8) 15 BYTES (ONE REDUNDANT BYTE)     01920073
        NC    TRWORK(15),=15X’0F’                                       01930072
        TR    TRWORK(15),HEXTAB                                         01940074
        MVC   NBW+2(14),TRWORK                                          01950072
        UNPK  TRWORK(3),ST87NBW(2) LAST BYTE + ONE REDUNDANT BYTE       01960073
        NC    TRWORK(3),=3X’0F’                                         01970072
        TR    TRWORK(3),HEXTAB                                          01980074
        MVC   NBW(2),TRWORK                                             01990072
*  CONVERT NBR                                                          02000072
        UNPK  TRWORK(15),ST87NBR+1(8) 15 BYTES (ONE REDUNDANT BYTE)     02010073
        NC    TRWORK(15),=15X’0F’                                       02020072
        TR    TRWORK(15),HEXTAB                                         02030074
        MVC   NBR+2(14),TRWORK                                          02040072
        UNPK  TRWORK(3),ST87NBR(2) LAST BYTE + ONE REDUNDANT BYTE       02050073
        NC    TRWORK(3),=3X’0F’                                         02060072
        TR    TRWORK(3),HEXTAB                                          02070074
        MVC   NBR(2),TRWORK                                             02080072
         PUT   PRINTDCB,PRINTL4                                         02090077
WRITEIT  DS    0H                                                       02100000
         PUT   PRINTDCB,PRINTBLK                                        02110000
*  LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS               02120000
*  WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                          02130000
         LA    R8,0(R8,R9)                                              02140000
         BCT   R10,SCOTRIP                                              02150000
         B     READ                                                     02160000
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        02170000
         B     READ                                                     02180000
FINISH   DS    0H                                                       02190000
         SEGEND                                                         02200000
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C02210000
               EODAD=FINISH                                             02220000
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133               02230000
DWORD    DS    D                                                        02240000
         ORG   DWORD                                                    02250000
         DC    C’12345678’                                              02260073
TRWORK   DS    CL33                                                     02270073
```

```
HEXTAB   DC    C’0123456789ABCDEF’ TRANSLATE TABLE                      02280042
PRINTBLK DC    CL133’ ‘                                                 02290000
PRINTHDR DC    CL133’1SMF TYPE 85 SUBTYPE 87 RECORDS’                   02300071
PRINTL0  DC    CL133’ SMF85TS SMFDTE/TME:’                              02310076
         ORG   PRINTL0+33                                               02320078
YYDDD    DC    CL7’ ‘,CL1’/’                                            02330064
HH       DC    CL2’ ‘,C’:’                                              02340064
MM       DC    CL2’ ‘,C’:’                                              02350064
SS       DC    CL2’ ‘,C’.’                                              02360064
HUS      DC    CL3’ ‘,CL1’ ‘                                            02370064
         ORG                                                            02380062
PRINTL1  DC    CL133’ STYPE/FUNC:’                                      02390077
         ORG   PRINTL1+33                                               02400078
STYPE9   DC    CL1’8’ PREFIX TO SUBTYPES 87                             02410068
STYPE    DC    CL1’ ‘,C’/’                                              02420064
FUNC     DC    CL34’ ‘,C’ ‘                                             02430080
         ORG                                                            02440000
PRINTL2  DC    CL133’ TDDN/TDDT/TVUN/VSN/TMT/TVT:’                      02450070
         ORG   PRINTL2+33                                               02460060
TDDN     DC    CL4’ ‘,CL1’/’                                            02470068
TDDT     DC    CL9’ ‘,CL1’/’                                            02480080
TVUN     DC    CL8’ ‘,CL1’/’                                            02490068
VSN      DC    CL6’ ‘,CL1’/’                                            02500077
TMT      DC    CL2’ ‘,CL1’/’                                            02510068
TVT      DC    CL1’ ‘,CL1’ ‘                                            02520077
         ORG                                                            02530075
PRINTL3  DC    CL133’ SGN/RC/RS/FLGS:’                                  02540068
         ORG   PRINTL3+33                                               02550068
SGN      DC    CL8’ ‘,CL1’/’                                            02560068
RC       DC    CL9’ ‘,CL1’/’                                            02570080
RS       DC    CL9’ ‘,C’/’                                              02580080
FLGS     DC    CL8’ ‘,C’ ‘                                              02590068
         ORG                                                            02600075
PRINTL4  DC    CL133’ TMNT/NOW/NKBW/NOR/NKBR/NBW/NBR:’                  02610072
         ORG   PRINTL4+33                                               02620078
TMNT     DC    CL9’ ‘,CL1’/’ CONVERTED                                  02630080
NOW      DC    CL9’ ‘,CL1’/’ CONVERTED                                  02640080
NKBW     DC    CL9’ ‘,CL1’/’ CONVERTED                                  02650080
NOR      DC    CL9’ ‘,CL1’/’ CONVERTED                                  02660080
NKBR     DC    CL9’ ‘,CL1’/’ CONVERTED                                  02670080
NBW      DC    CL17’ ‘,CL1’/’ CONVERTED                                 02680080
NBR      DC    CL16’ ‘,CL1’ ‘ CONVERTED                                 02690077
         ORG                                                            02700000
SMFDSECT DSECT                                                          02710000
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                       02720000
         END                                                            02730000
```

A.3.10.3 Storing and running the JCL to build the LOAD module SMF85TS

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-9 on page 227 into your PDS MHLRES2.SMF85TS.DFSMS13.PDS as member SMFT85SJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, “OAM SMF85 analysis source materials” on page 263 as member BUILDJCL.
2. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TS can now be run by using the JCL shown in Figure A-38 on page 251.

A.3.11 SMF Record type 85 subtype 82-86 data display program SMF85TU

Program SMG85TU displays the contents of selected fields of SMF record Type 85 subtypes 82-86 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that LCS Tape Library volume activity is occurring.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.11.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, “OAM SMF85 Analysis program common data set creation” on page 264. If the sample JCL is used, the SET to be specified is SMF85TU.

When USER=MHLRES2 and SET=SMF85TU, the following data sets result:

- MHLRES2.SMF85TU.DFSMS13.PDS
- MHLRES2.SMF85TU.DFSMS13.LOAD

A.3.11.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session.

To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

Attention: The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-12 into member SMF85TUA of data set MHLRES2.SMF85TU.DFSMS13.PDS. The result should contain 260 lines.

Tip: In the code in Example A-12 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-12 SMF85TU program source code

```
 MACRO                                                      00010000
&NAME    SEGSTART                                                   00020000
&NAME    STM   14,12,12(13)           SAVE HIS REGS IN HIS SAVE AREA   00030000
R0       EQU   0                                                    00040000
R1       EQU   1                                                    00050000
R2       EQU   2                                                    00060000
```

```
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11
RB       EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         BALR  12,0                    SET UP ADDRESSABILITY
         USING *,12                    USE REG 12 AS BASE REG
         ST    13,SAVEREGS+4           SAVE @ OF HIS SAVEAREA IN MINE
         LA    03,SAVEREGS             LOAD @ OF MY SAVE AREA IN REG 3
         ST    03,8(13)                SAVE @ OF MY SAVE AREA IN HIS
         LR    13,03                   LOAD @ OF MY SAVE AREA IN REG 13
         MEND
         MACRO
         SEGEND
         L     13,SAVEREGS+4           LOAD REG13 WITH @ OF HIS SAVE
         LM    14,12,12(13)            RESTORE REGS FROM HIS SAVEAREA
         XR    R15,R15
         BR    14                      RETURN TO CALLING RTN VIA REG 14
SAVEREGS DC    18F'0'                  SET UP SAVE AREA
         MEND
SMFR85TO  SEGSTART
*  THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF
*  THE SMF TYPE 85 SUBTYPES 82 TO 86
*  WHICH ARE THE OAM LCS FILE SYSTEM RECORDS
*  IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED
*  TO SELECT TYPE 85 SUBTYPE 82/83/84/85/86
*  RECORDS FROM EITHER THE ACTIVE SMF 'MAN' DATASETS OR
*  OFF A PREVIOUSLY EXTRACTED COPY OF THE 'MAN' DATASETS.
*
*  THE STANDARD SMF RECORD MAPPING MACROS ARE USED.
*  REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD
*  R3    START OF WHOLE RECORD
*  THERE IS 1 DSECTS TO BE MAPPED
*  R4    START OF ST82 MAPPING FOR 82/3/4/5/6
*  R5    SPARE
*  R6    SPARE
*  R7    SPARE
*  OTHER REGISTER USES
*  R12   OVERALL BASE REGISTER
*  R8    RECORD TYPE/SUBTYPE CHECKING/WORKING
*  R9    LENGTH OF PARTICULAR DSECT
*  R10   NUMBER OF ENTRIES IN THE TRIPLET
*
*  QSAM GET LOCATE PROCESSING IS USED
*
         OPEN SMFIN
         OPEN (PRINTDCB,(OUTPUT))
```

```
         PUT  PRINTDCB,PRINTHDR                                         00620000
READ     GET SMFIN                                                      00630000
         LR    R3,R1  * COPY PARAMETER POINTER                          00640064
* R3 -> SMF RECORD                                                      00650000
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION                         00660000
         USING CBRSMF85,R3                                              00670046
         CLI   SMF85RTY,X'55'  *   CHECK IF TYPE 85                     00680064
         BNE   IGNORE                                                   00690000
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             00700000
CHKSTYP1 DS    0H                                                       00710045
*   CHECK IF ANY OF SUBTYPE 82-86                                       00720075
         CLI   SMF85STY+1,X'52'                                         00730075
         BNE   *+18                                                     00740045
         MVI   STYPE,C'2'                                               00750075
         MVC   FUNC,=CL34'(LCS TAPE LIBRARY VOLUME ENTRY)'              00760075
         B     STOK                                                     00770045
         CLI   SMF85STY+1,X'53'                                         00780075
         BNE   *+18                                                     00790075
         MVI   STYPE,C'3'                                               00800075
         MVC   FUNC,=CL34'(LCS TAPE LIBRARY VOLUME EJECT)'              00810075
         B     STOK                                                     00820075
         CLI   SMF85STY+1,X'54'                                         00830075
         BNE   *+18                                                     00840075
         MVI   STYPE,C'4'                                               00850075
         MVC   FUNC,=CL34'(LCS TAPE LIBRARY VOLUME AUDIT)'              00860075
         B     STOK                                                     00870075
         CLI   SMF85STY+1,X'55'                                         00880075
         BNE   *+18                                                     00890075
         MVI   STYPE,C'5'                                               00900075
         MVC   FUNC,=CL34'(LCS TAPE LIBRARY VOLUME MOUNT)'              00910075
         B     STOK                                                     00920075
         CLI   SMF85STY+1,X'56'                                         00930075
         BNE   *+18                                                     00940075
         MVI   STYPE,C'6'                                               00950075
         MVC   FUNC,=CL34'(LCS TAPE LIBRARY VOLUME DEMOUNT)'            00960075
         B     STOK                                                     00970075
         B     IGNORE    *        OTHERWISE IGNORE                      00980064
STOK     EQU   *                                                        00990045
*        DC    F'0'  CREATE AN ABEND TO LOOK AT THE RECORDS             01000000
* IS TYPE 85 SUBTYPE 82-86 SO EXTRACT THE DATA                          01010075
* R3 IS THE START OF THE WHOLE RECORD                                   01020000
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.               01030000
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION        01040000
* ADD R8 TO R3                                                          01050000
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS                           01060000
         LA    R4,SMF85END                                              01070047
         USING ST82,R4                                                  01080075
         L     R8,SMF85OSO                                              01090047
         LH    R9,SMF85OSL                                              01100047
         LH    R10,SMF85OSN                                             01110047
*  PROCESS THE SUMMARY ENTRIES TRIPLET.                                 01120000
*  FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START                 01130000
*  SECOND HW IS THE LENGTH OF EACH TRIPLET                              01140000
*  THIRD HW IS THE NUMBER OF TRIPLETS                                   01150000
*  FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM       01160000
```

```
*  THE SMF RECORD TYPE 85 SUBTYPE 90/91/92/93 RECORDS                      01170044
*  COLN COMES FROM ST82COLN                                                01180075
*  CNID COMES FROM ST82CNID                                                01190075
*  ETC                                                                     01200000
*  ST82FLGS IS INTERPRETED AS FLGO ON OR OFF                               01210075
*                                                                          01220000
SCOTRIP  DS    0H                                                          01230000
         LA    R4,0(R3,R8)                                                 01240062
         LA    R4,0(R3,R8)                                                 01250062
         UNPK  YYDDD(7),SMF85DTE                                           01260062
         CLI   YYDDD+1,C'0'                                                01270062
         BE   SETD0                                                        01280062
         CLI   YYDDD+1,C'1'                                                01290062
         BE   SETD1                                                        01300062
         DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                     01310062
SETD0    MVC   YYDDD(2),=C'19'                                             01320062
         B     SETDZ                                                       01330062
SETD1    MVC   YYDDD(2),=C'20'                                             01340062
*                                                                          01350062
SETDZ    EQU   *                                                           01360062
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT                01370062
         LA    R5,100 PREPARE TO DIVIDE BY 100                             01380062
         LA    R6,0                                                        01390062
         L     R7,SMF85TME  GET THE TIME                                   01400062
         DR    R6,R5   -> SECS IN R7, HUNS IN R6                           01410062
         CVD   R6,DWORD                                                    01420062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                    01430062
         UNPK  HUS,DWORD+6(2)                                              01440062
*        DC    F'0'                                                        01450062
* NOW GET THE SECS                                                         01460062
         LA    R5,60 PREPARE TO DIVIDE BY 60                               01470062
         LA    R6,0                                                        01480062
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6                 01490062
         CVD   R6,DWORD                                                    01500062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                    01510062
         UNPK  SS,DWORD+6(2)                                               01520062
* NOW GET THE MINS                                                         01530062
         LA    R6,0                                                        01540062
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6                  01550062
         CVD   R6,DWORD                                                    01560062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                    01570062
         UNPK  MM,DWORD+6(2)                                               01580062
         CVD   R7,DWORD     DO HOURS                                       01590062
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                    01600062
         UNPK  HH,DWORD+6(2)                                               01610062
         PUT   PRINTDCB,PRINTLO                                            01620062
         LA    R4,0(R3,R8)                                                 01630000
         MVC   TLN,ST82TLN                                                 01640076
         MVC   TLDT,ST82TLDT                                               01650076
*        MVC   TDDT,ST82TDDT                                               01660079
*  CONVERT TDDT                                                            01670079
         UNPK  TDDT(09),ST82TDDT(5)                                        01680079
         TR    TDDT(08),HEXTAB-240                                         01690079
         MVI   TDDT+8,X'40'                                                01700079
         MVC   VSN,ST82VSN                                                 01710075
```

```
         MVC   TMT,ST82TMT                                              01720075
*  CONVERT LIQT                                                         01730075
         UNPK  LIQT(09),ST82LIQT(5)                                     01740075
         TR    LIQT(08),HEXTAB-240                                      01750075
         MVI   LIQT+8,X'40'                                             01760075
*  CONVERT LDQT                                                         01770075
         UNPK  LDQT(09),ST82LDQT(5)                                     01780075
         TR    LDQT(08),HEXTAB-240                                      01790075
         MVI   LDQT+8,X'40'                                             01800075
         PUT   PRINTDCB,PRINTL1                                         01810075
*  CONVERT LTQT                                                         01820075
         UNPK  LTQT(09),ST82LTQT(5)                                     01830075
         TR    LTQT(08),HEXTAB-240                                      01840075
         MVI   LTQT+8,X'40'                                             01850075
*  CONVERT LTPT                                                         01860075
         UNPK  LTPT(09),ST82LTPT(5)                                     01870075
         TR    LTPT(08),HEXTAB-240                                      01880075
         MVI   LTPT+8,X'40'                                             01890075
*  CONVERT RC                                                           01900075
         UNPK  RC(09),ST82RC(5)                                         01910075
         TR    RC(08),HEXTAB-240                                        01920075
         MVI   RC+8,X'40'                                               01930075
*  CONVERT RS                                                           01940066
         UNPK  RS(09),ST82RS(5)                                         01950075
         TR    RS(08),HEXTAB-240                                        01960066
         MVI   RS+8,X'40'                                               01970066
*  CONVERT FLAGS                                                        01980049
          UNPK  FLGS(09),ST82FLGS(5)  UNPK 1 MORE THAN NEEDED           01990075
          MVI   FLGS+8,C' ' BLANK OUT THE EXTRA BYTE                    02000072
          NC    FLGS(08),=8X'0F'                                        02010072
          TR    FLGS(8),HEXTAB                                          02020074
         PUT   PRINTDCB,PRINTL3                                         02030075
WRITEIT  DS    0H                                                       02040000
         PUT   PRINTDCB,PRINTBLK                                        02050000
*  LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS               02060000
*  WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                          02070000
         LA    R8,0(R8,R9)                                              02080000
         BCT   R10,SCOTRIP                                              02090000
         B     READ                                                     02100000
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        02110000
         B     READ                                                     02120000
FINISH   DS    0H                                                       02130000
         SEGEND                                                         02140000
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C02150000
               EODAD=FINISH                                             02160000
PRINTDCB DCB   DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133               02170000
DWORD    DS    D                                                        02180000
         ORG   DWORD                                                    02190000
         DC    C'12345678'                                              02200073
TRWORK   DS    CL33                                                     02210073
HEXTAB   DC    C'0123456789ABCDEF' TRANSLATE TABLE                      02220042
PRINTBLK DC    CL133' '                                                 02230000
PRINTHDR DC    CL133'1SMF TYPE 85 SUBTYPES 82-86 RECORDS'               02240075
PRINTL0  DC    CL133' SMF85TU SMFDTE/TME:'                              02250078
         ORG   PRINTL0+31                                               02260063
```

```
YYDDD    DC    CL7’ ‘,CL1’/’                                           02270064
HH       DC    CL2’ ‘,C’:’                                             02280064
MM       DC    CL2’ ‘,C’:’                                             02290064
SS       DC    CL2’ ‘,C’.’                                             02300064
HUS      DC    CL3’ ‘,CL1’ ‘                                           02310064
         ORG                                                           02320062
PRINTL1  DC    CL133’ STYPE/FUNC/TLN/TLDT:’                            02330065
         ORG   PRINTL1+31                                              02340037
STYPE9   DC    CL1’8’ PREFIX TO SUBTYPES 87                            02350068
STYPE    DC    CL1’ ‘,C’/’                                             02360064
FUNC     DC    CL34’ ‘,C’/’                                            02370064
TLN      DC    CL09’ ‘,C’/’                                            02380079
TLDT     DC    CL8’ ‘,C’ ‘                                             02390065
         ORG                                                           02400000
PRINTL2  DC    CL133’ TDDT/TDDN/VSN/TMT/LIQT/LDQT:’                    02410075
         ORG   PRINTL2+33                                              02420060
TDDT     DC    CL9’ ‘,CL1’/’                                           02430079
TDDN     DC    CL4’ ‘,CL1’/’                                           02440079
VSN      DC    CL6’ ‘,CL1’/’                                           02450075
TMT      DC    CL2’ ‘,CL1’/’                                           02460068
LIQT     DC    CL9’ ‘,CL1’/’ CONVERTED                                 02470079
LDQT     DC    CL8’ ‘,CL1’ ‘ CONVERTED                                 02480079
         ORG                                                           02490077
PRINTL3  DC    CL133’ LIQT/LTPT/RC/RS/FLGS:’                           02500075
         ORG   PRINTL3+33                                              02510075
LTQT     DC    CL9’ ‘,CL1’/’ CONVERTED                                 02520079
LTPT     DC    CL9’ ‘,CL1’/’ CONVERTED                                 02530079
RC       DC    CL9’ ‘,CL1’/’                                           02540079
RS       DC    CL9’ ‘,C’ ‘                                             02550079
FLGS     DC    CL8’ ‘,C’ ‘                                             02560068
         ORG                                                           02570077
SMFDSECT DSECT                                                         02580000
         IFASMFR  (85) THIS INCLUDES CBRSMF MACRO                      02590000
         END                                                           02600000
```

A.3.11.3 Storing and running the JCL to build the LOAD module SMF85TU

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-9 on page 227 into your PDS MHLRES2.SMF85TU.DFSMS13.PDS as member SMFT85UJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, “OAM SMF85 analysis source materials” on page 263 as member BUILDJCL.
2. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TU can now be run by using the JCL shown in Figure A-38 on page 251.

A.3.12 SMF Record type 85 subtypes 78,79,88 data display program SMF85TW

Program SMG85TW displays the contents of selected fields of SMF record Type 85 subtypes 78,79,88 data. It is not intended to provide a comprehensive report on OAM activity, but rather to verify that immediate backup is occurring.

The steps to build the program must be done once, after which it can be run several times. It is not necessary to have in-depth assembler experience, but familiarity with JCL is required.

A.3.12.1 Creating PDS/PDSE data sets

To create these data sets, see A.3.1.2, “OAM SMF85 Analysis program common data set creation” on page 264. If the sample JCL is used, the SET to be specified is SMF85TW.

When USER=MHLRES2 and SET=SMF85TW, the following data sets result:

- MHLRES2.SMF85TW.DFSMS13.PDS
- MHLRES2.SMF85TW.DFSMS13.LOAD

A.3.12.2 Storing the program source in the PDS

The source can be retrieved from the ITSO FTP repository or reconstructed by cutting from the document and pasting into the ISPF session. To use the FTP process, see A.3.1.5, “SMF85 program restore process” on page 265. If using the copy and paste process, see A.3.1.5, “SMF85 program restore process” on page 265.

Attention: The resulting source code must have at least one blank where a blank is shown in the listing.

Make sure that your mainframe terminal environment is set up as in A.3.1.5, “SMF85 program restore process” on page 265. Copy and paste the contents of Example A-13 into member SMF85TWA of data set MHLRES2.SMF85TW.DFSMS13.PDS. The result should contain 303 lines.

Tip: In the code in Example A-13 on the line that starts with SMFIN, there is a continuation indicator that must be in column 72. The text on the following line must start in column 16.

Example A-13 SMF85TW program source code

```
 MACRO                                                          00010014
&NAME    SEGSTART                                                       00020014
&NAME    STM   14,12,12(13) SAVE HIS REGS IN HIS SAVE AREA              00030014
R0       EQU   0                                                        00040014
R1       EQU   1                                                        00050014
R2       EQU   2                                                        00060014
R3       EQU   3                                                        00070014
R4       EQU   4                                                        00080014
R5       EQU   5                                                        00090014
R6       EQU   6                                                        00100014
R7       EQU   7                                                        00110014
R8       EQU   8                                                        00120014
R9       EQU   9                                                        00130014
R10      EQU   10                                                       00140014
R11      EQU   11                                                       00150014
RB       EQU   12                                                       00160014
R13      EQU   13                                                       00170014
R14      EQU   14                                                       00180014
R15      EQU   15                                                       00190014
         BALR  12,0 SET UP ADDRESSABILITY                               00200014
         USING *,12 USE REG 12 AS BASE REG                              00210014
         ST    13,SAVEREGS+4 SAVE @ OF HIS SAVEAREA IN MINE             00220014
         LA    03,SAVEREGS LOAD @ OF MY SAVE AREA IN REG 3              00230014
         ST    03,8(13) SAVE @ OF MY SAVE AREA IN HIS                   00240014
         LR    13,03 LOAD @ OF MY SAVE AREA IN REG 13                   00250014
         MEND                                                           00260014
```

```
         MACRO                                                          00270014
&NAME    SEGEND                                                         00280014
&NAME    L     13,SAVEREGS+4 LOAD REG13 WITH @ OF HIS SAVE              00290014
         LM    14,12,12(13) RESTORE REGS FROM HIS SAVEAREA              00300014
         XR    R15,R15                                                  00310014
         BR    14 RETURN TO CALLING RTN VIA REG 14                      00320014
SAVEREGS DC    18F’0’ SET UP SAVE AREA                                  00330014
         MEND                                                           00340014
         MACRO                                                          00350014
&NAME    BINDEC &KEY                                                    00360014
         LH    R1,ST78&KEY.                                             00370016
         CVD   R1,DWORD                                                 00380014
         OI    DWORD+7,X’0F’                                            00390014
         UNPK  &KEY.(11),DWORD+2(6)                                     00400014
         MEND                                                           00410014
         MACRO                                                          00420032
&NAME    HEXTEXT4 &KEY                                                  00430032
         UNPK  &KEY.(9),ST78&KEY.(5)                                    00440034
         TR    &KEY.(08),HEXTAB-240                                     00450032
         MVI   &KEY.+8,X’40’ BLANK THE EXTRA BYTE                       00460032
         MEND                                                           00470032
         MACRO                                                          00480032
&NAME    HEXTEXT8 &KEY                                                  00490032
         UNPK  &KEY.(09),ST78&KEY.(5)                                   00500034
         TR    &KEY.(08),HEXTAB-240                                     00510032
         UNPK  &KEY.+8(09),ST78&KEY.+4(5)                               00520034
         TR    &KEY.+8(08),HEXTAB-240                                   00530032
         MVI   &KEY.+16,X’40’ BLANK THE EXTRA BYTE                      00540032
         MEND                                                           00550032
SMFR85TJ SEGSTART                                                       00560014
* THIS IS A SIMPLE PROGRAM TO DISPLAY THE CONTENTS OF VARIOUS OF        00570014
* THE SMF TYPE 85 SUBTYPE 78,79 AND 88 RECORDS, WHICH ARE THE           00580016
* OAM COMMAND RECYCLE RECORDS.                                          00590014
* IT IS ASSUMED THAT THE IFASMFDP PROGRAM HAS ALREADY BEEN USED         00600014
* TO SELECT TYPE 85 SUBTYPE 78/79/88                                    00610016
* RECORDS FROM EITHER THE ACTIVE SMF ‘MAN’ DATASETS OR                  00620014
* OFF A PREVIOUSLY EXTRACTED COPY OF THE ‘MAN’ DATASETS.                00630014
*                                                                       00640014
* THE STANDARD SMF RECORD MAPPING MACROS ARE USED.                      00650014
* REGISTER EQUATES TO PARTS OF THE SMF TYPE 85 RECORD                   00660014
* R3 START OF WHOLE RECORD                                              00670014
* THERE IS 1 DSECTS TO BE MAPPED                                        00680014
* R4 START OF ST78 LCS TAPE WRITE/READ/LOGICAL DELETE                   00690016
* R5 START OF VOLUME ARRAY - ASSUMED TO START AT THE END OF BASE        00700014
* R6 NUMBER OF VOLUMES                                                  00710014
* R7 SPARE                                                              00720014
* OTHER REGISTER USES                                                   00730014
* R12 OVERALL BASE REGISTER                                             00740014
* R8 RECORD TYPE/SUBTYPE CHECKING/WORKING                               00750014
* R9 LENGTH OF PARTICULAR DSECT                                         00760014
* R10 NUMBER OF ENTRIES IN THE TRIPLET                                  00770014
*                                                                       00780014
* QSAM GET LOCATE PROCESSING IS USED                                    00790014
*                                                                       00800014
         OPEN  SMFIN                                                    00810014
```

```
         OPEN  (PRINTDCB,(OUTPUT))
         PUT   PRINTDCB,PRINTHDR
READ     GET   SMFIN
         LR    R3,R1  * COPY PARAMETER POINTER
* R3 -> SMF RECORD
* USE SMF R3 RECORD MAPPING FOR INITIAL VERSION
         USING CBRSMF85,R3
         CLI   SMF85RTY,X'55'  * CHECK IF TYPE 85
         BNE   IGNORE
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
CHKSTYP1 DS    0H
         CLI   SMF85STY+1,X'4E'     * CHECK IF SUBTYPE 78
         BNE   *+20
         MVC   STYPE,=C'78'
         MVC   FUNC,=CL34'(LCS TAPE WRITE REQUEST       )'
         B     STOK
         CLI   SMF85STY+1,X'4F'     * CHECK IF SUBTYPE 78
         BNE   *+20
         MVC   STYPE,=C'79'
         MVC   FUNC,=CL34'(LCS TAPE READ  REQUEST       )'
         B     STOK
         CLI   SMF85STY+1,X'58'     * CHECK IF SUBTYPE 78
         BNE   *+20
         MVC   STYPE,=C'88'
         MVC   FUNC,=CL34'(LCS TAPE LOGICAL DELETE REQ. )'
         B     STOK
         B     IGNORE
STOK     DS    0H
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS
* IS TYPE 85 SUBTYPE 78, 79 OR 88 SO EXTRACT DATA
* R3 IS THE START OF THE WHOLE RECORD
* FIRST ESTABLISH ADDRESSIBILITY TO THE VARIOUS SECTIONS.
* GENERAL PROCESS IS LOAD R8 WITH OFFSET TO THE RELEVANT SECTION
* ADD R8 TO R3
* THEN THE DSECTS SHOULD ADDRESS THE SECTIONS
         LA    R4,SMF85END
         USING ST78,R4
         L     R8,SMF85OSO
         LH    R9,SMF85OSL
         LH    R10,SMF85OSN
* PROCESS THE SUMMARY ENTRIES TRIPLET.
* FIRST FULLWORD IS OFFSET TO WHERE THE TRIPLETS START
* SECOND HW IS THE LENGTH OF EACH TRIPLET
* THIRD HW IS THE NUMBER OF TRIPLETS
* FIELDS USED IN THE REPORT CORRESPOND TO THE RECORDS TAKEN FROM
* THE SMF RECORD TYPE 85 SUBTYPE 40 RECORDS.
* STRD COMES FROM ST78STRD
* ENDD COMES FROM ST78ENDD
* ETC
*
SCOTRIP  DS    0H
         LA    R4,0(R3,R8)
         UNPK  YYDDD(7),SMF85DTE
         CLI   YYDDD+1,C'0'
         BE   SETD0
```

```
         CLI   YYDDD+1,C'1'                                             01370016
         BE   SETD1                                                     01380016
*        DC    F'0'  ABEND AS SOMETHING HAS GONE WRONG                  01390028
SETD0    MVC   YYDDD(2),=C'19'                                          01400016
         B     SETDZ                                                    01410016
SETD1    MVC   YYDDD(2),=C'20'                                          01420016
*                                                                       01430016
SETDZ    EQU   *                                                        01440016
*    CONVERT THE TIME FROM HUNDREDTHS OF SEC SINCE MIDNIGHT             01450016
         LA    R5,100 PREPARE TO DIVIDE BY 100                          01460016
         LA    R6,0                                                     01470016
         L     R7,SMF85TME  GET THE TIME                                01480016
         DR    R6,R5   -> SECS IN R7, HUNS IN R6                        01490016
         CVD   R6,DWORD                                                 01500016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01510016
         UNPK  HUS,DWORD+6(2)                                           01520016
*        DC    F'0'                                                     01530016
* NOW GET THE SECS                                                      01540016
         LA    R5,60 PREPARE TO DIVIDE BY 60                            01550016
         LA    R6,0                                                     01560016
         DR    R6,R5   -> MINS IN R7, SECS REMAINDER IN R6              01570016
         CVD   R6,DWORD                                                 01580016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01590016
         UNPK  SS,DWORD+6(2)                                            01600016
* NOW GET THE MINS                                                      01610016
         LA    R6,0                                                     01620016
         DR    R6,R5   -> HRS IN R7, MINS REMAINDER IN R6               01630016
         CVD   R6,DWORD                                                 01640016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01650016
         UNPK  MM,DWORD+6(2)                                            01660016
         CVD   R7,DWORD     DO HOURS                                    01670016
         OI    DWORD+7,X'0F'  FIX THE SIGN FOR PRINTING                 01680016
         UNPK  HH,DWORD+6(2)                                            01690016
         PUT   PRINTDCB,PRINTL0                                         01700016
         LA    R5,ST78END POINT TO THE VOLUME ARRAY                     01710016
         USING ST78OBJD,R5                                              01720017
         MVC   ORMN,ST78ORMN    * COPY                                  01730016
         MVC   OTMN,ST78OTMN    * COPY                                  01740016
         MVC   TDUN,ST78TDUN      * COPY                                01750017
         PUT   PRINTDCB,PRINTL1                                         01760016
         MVC   TDDN,ST78TDDN    * COPY                                  01770016
         MVC   TVT,ST78TVT      * COPY                                  01780016
         MVC   SGN,ST78SGN      * COPY                                  01790016
         HEXTEXT4 LIQT          * CONVERT                               01800033
         HEXTEXT4 LDQT          * CONVERT                               01810033
         HEXTEXT4 LEQT          * CONVERT                               01820033
         PUT   PRINTDCB,PRINTL2                                         01830016
         HEXTEXT4 LXQT          * CONVERT                               01840033
         HEXTEXT4 LMAT          * CONVERT                               01850033
         HEXTEXT4 LMDT          * CONVERT                               01860033
         PUT   PRINTDCB,PRINTL3                                         01870016
         HEXTEXT4 LDCT          * CONVERT                               01880033
         HEXTEXT4 LDOT          * CONVERT                               01890033
         HEXTEXT4 LDPT          * CONVERT                               01900033
         HEXTEXT4 LBRT          * CONVERT                               01910033
```

```
         HEXTEXT4 LBWT         * CONVERT                                01920033
         HEXTEXT4 LBCT         * CONVERT                                01930033
         PUT   PRINTDCB,PRINTL4                                         01940016
         HEXTEXT4 FLGS         * CONVERT                                01950033
         HEXTEXT4 NOBJ         * CONVERT                                01960033
         HEXTEXT4 NKBP         * CONVERT                                01970033
         HEXTEXT4 SOBJ         * CONVERT                                01980033
         HEXTEXT4 SKBP         * CONVERT                                01990033
         PUT   PRINTDCB,PRINTL5                                         02000016
         L      R6,ST78NOBJ    * GET NUMBER OF VOLUMES                  02010037
* DC F'0' CREATE AN ABEND TO LOOK AT THE RECORDS                        02020029
VLOOP    MVC   COLN,ST78COLN                                            02030016
         MVC   OBJN,ST78OBJN                                            02040016
         HEXTEXT4 OLEN         * CONVERT                                02050033
         PUT   PRINTDCB,PRINTL6                                         02060016
         HEXTEXT4 OOFF         * CONVERT                                02070033
         MVC   VSN,ST78VSN                                              02080016
         MVC   TMT,ST78TMT                                              02090016
         HEXTEXT4 OTKN         * CONVERT                                02100033
         HEXTEXT4 RC           * CONVERT                                02110033
         HEXTEXT4 RS           * CONVERT                                02120033
*        DC    H'0'                                                     02130040
         PUT   PRINTDCB,PRINTL7                                         02140016
ST78VARR EQU   ST78RS+4-ST78OBJD                                        02150022
         LA    R5,ST78VARR(R5)                                          02160021
         BCT   R6,VLOOP                                                 02170014
WRITEIT  DS    0H                                                       02180014
         PUT   PRINTDCB,PRINTBLK                                        02190014
* LOOP BACK AT THIS POINT IF THERE ARE ANY MORE TRIPLETS                02200014
* WHEN BCT REACHES ZERO GO GET ANOTHER RECORD                           02210014
         LA    R8,0(R8,R9)                                              02220014
         BCT   R10,SCOTRIP                                              02230014
         B     READ                                                     02240014
IGNORE   DS    0H EXIT WITH OUT WRITING IF NOT THE RIGHT RECORDS        02250014
         B     READ                                                     02260014
FINISH   DS    0H                                                       02270014
         SEGEND                                                         02280014
SMFIN    DCB   DDNAME=SMFIN,DSORG=PS,MACRF=(GL),EROPT=SKP,             C02290014
               EODAD=FINISH                                             02300014
PRINTDCB DCB DDNAME=PRINT,DSORG=PS,MACRF=(PM),LRECL=133                 02310014
DWORD    DS  D                                                          02320014
         ORG DWORD                                                      02330014
         DC  C'12345678'                                                02340014
HEXTAB   DC    C'0123456789ABCDEF' TRANSLATE TABLE                      02350035
PRINTBLK DC  CL133' '                                                   02360014
PRINTHDR DC  CL133'1SMF TYPE 85 SUBTYPE 78, 79 OR 88 RECORDS'           02370016
PRINTL0  DC    CL133' SMF85TW SMFDTE/TME:'                              02380041
         ORG   PRINTL0+35                                               02390025
YYDDD    DC    CL7' ',CL1'/'                                            02400016
HH       DC    CL2' ',C':'                                              02410016
MM       DC    CL2' ',C':'                                              02420016
SS       DC    CL2' ',C'.'                                              02430016
HUS      DC    CL3' ',CL1' '                                            02440016
         ORG                                                            02450023
PRINTL1  DC  CL133' STYPE/FUNC/ORMN/OTMN:'                              02460016
```

```
         ORG PRINTL1+35                                                 02470025
STYPE    DC  CL2’ ‘,C’/’                                                02480016
FUNC     DC  CL34’ ‘,C’/’                                               02490016
ORMN     DC  CL16’ ‘,C’/’                                               02500016
OTMN     DC  CL16’ ‘,C’ ‘                                               02510026
         ORG                                                            02520019
PRINTL2  DC  CL133’ TDUN/TDDN/TVT/SGN/LIQT/LDQT/LEQT:’                  02530025
         ORG PRINTL2+35                                                 02540025
TDUN     DC  CL8’ ‘,C’/’                                                02550016
TDDN     DC  CL4’ ‘,C’/’                                                02560016
TVT      DC  CL1’ ‘,C’/’        COPIED                                  02570029
SGN      DC  CL9’ ‘,C’/’                                                02580036
LIQT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02590036
LDQT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02600036
LEQT     DC  CL9’ ‘,C’ ‘        CONVERTED FROM BINARY                   02610043
         ORG                                                            02620019
PRINTL3  DC  CL133’ TXQT/LMAT/LMDT’                                     02630016
         ORG PRINTL3+35                                                 02640025
LXQT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02650036
LMAT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02660036
LMDT     DC  CL9’ ‘,C’ ‘        CONVERTED FROM BINARY                   02670036
         ORG                                                            02680019
PRINTL4  DC  CL133’ LDCT/LDOT/LDPT/LBRT/LBWT/LBCT:’                     02690016
         ORG PRINTL4+35                                                 02700025
LDCT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02710036
LDOT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02720036
LDPT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02730036
LBRT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02740036
LBWT     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02750036
LBCT     DC  CL9’ ‘,C’ ‘        CONVERTED FROM BINARY                   02760036
         ORG                                                            02770019
PRINTL5  DC  CL133’ FLGS/NOBJ/NKBP/SOBJ/SKBP:’                          02780023
         ORG PRINTL5+35                                                 02790025
FLGS     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02800036
NOBJ     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02810036
NKBP     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02820036
SOBJ     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02830036
SKBP     DC  CL9’ ‘,C’ ‘        CONVERTED FROM BINARY                   02840036
         ORG                                                            02850014
PRINTL6  DC  CL133’ COLN/OBJN/OLEN:’                                    02860016
         ORG PRINTL6+35                                                 02870025
COLN     DC  CL44’/’                                                    02880026
OBJN     DC  CL44’/’                                                    02890026
OLEN     DC  CL8’ ‘,C’ ‘        CONVERTED FROM BINARY                   02900016
         ORG                                                            02910030
PRINTL7  DC  CL133’ OOFF/VSN/TMT/OTKN/RC/RS:’                           02920042
         ORG PRINTL7+35                                                 02930025
OOFF     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02940036
VSN      DC  CL6’ ‘,C’/’                                                02950026
TMT      DC  CL2’ ‘,C’/’                                                02960026
OTKN     DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02970036
RC       DC  CL9’ ‘,C’/’        CONVERTED FROM BINARY                   02980036
RS       DC  CL9’ ‘,C’ ‘        CONVERTED FROM BINARY                   02990036
         ORG                                                            03000014
SMFDSECT DSECT                                                          03010014
```

```
IFASMFR (85) THIS INCLUDES CBRSMF MACRO                  03020014
END                                                      03030014
```

A.3.12.3 Storing and running the JCL to build the LOAD module SMF85TW

Perform these steps to store and run the JCL:

1. Copy and paste the contents of Figure A-9 on page 227 into your PDS MHLRES2.SMF85TW.DFSMS13.PDS as member SMFT85WJ. The result should contain 27 lines. This JCL is also available from the ITSO FTP site as documented in A.3.1.1, “OAM SMF85 analysis source materials” on page 263 as member BUILDJCL.
2. Run the job when the member is created. The return code for both steps should be 0. Program SMF85TW can now be run by using the JCL shown in Figure A-46 on page 257.

B

Additional material

This book refers to additional material that can be downloaded from the Internet as described in the following sections.

Locating the web material

The web material associated with this book is available in softcopy on the Internet from the IBM Redbooks web server at:

`ftp://www.redbooks.ibm.com/redbooks/SG247961`

Alternatively, you can go to the IBM Redbooks website at:

`ibm.com/redbooks`

Select **Additional materials** and open the directory that corresponds with IBM Redbooks form number SG247961.

Using the web material

The additional web material that accompanies this book includes the following files:

File name	*Description*
SMF85JCL.zip	Zipped code samples for SMF type 85 record analysis

System requirements for downloading the web material

The web material requires the following system configuration:

Hard disk space:	1 MB minimum
Operating System:	Current Windows

Downloading and extracting the web material

Create a subdirectory (folder) on your workstation, and extract the contents of the web material.zip file into this folder.

When expanded into a directory on your workstation, you will find files containing JCL. Use these jobs to build each program. Run the resulting program and assembler source files that are referred to in the text in the sections related to each program.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 338. Note that some of the documents referenced here might be available in softcopy only.

- *DFSMShsm Fast Replication Technical Guide*, SG24-7069
- *VSAM Demystified*, SG24-6105
- *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580
- *GDPS Family: An Introduction to Concepts and Capabilities*, SG24-6374
- *ABCs of z/OS System Programming Volume 7*, SG24-6987

Other publications

These publications are also relevant as further information sources:

- *z/OS DFSMS Using Data Sets,* SC26-7410
- *DFSMS Access Method Services for Catalogs,* SC26-73942
- *z/OS MVS System Management Facilities (SMF),* SA22-7630
- *MVS System Command Manual,* SA22-7627
- *DFSMSdfp Diagnosis*, GY27-7618
- *IBM Health Checker for z/OS V1R12.0 User's Guide,* SA22-7994
- *TSO/E Customization*, SA22-7783
- *z/OS MVS Initialization and Tuning Reference,* SA22-7592
- *Managing Catalogs,* SC26-7409
- *z/OS V1.13.0 ISPF User's Guide Vol II,* SC34-4823
- *z/OS DFSMS OAM Planning, Installation, and Storage Administration Guide for Object Support*, SC35-0426
- *z/OS V1R13 Migration*, GA22-7499
- *Device Support Facilities (ICKDSF) User's Guide and Reference*, GC35-0033
- *z/OS Distributed File Service zSeries File System Implementation z/OS V1R11*, SG24-6580
- *z/OS MVS System Messages & Codes, Volume 4*, SA22-7634
- *z/OS MVS System Messages & Codes, Volume 7 (IEB - IEE),* SA22-7637

Online resources

These websites are also relevant as further information sources:

- z/OS V1.13 DFSMS online publications

  `http://www-03.ibm.com/systems/z/os/zos/bkserv/r13pdf/`

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and additional materials, as well as order hardcopy Redbooks publications, at this website:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Symbols

E

F

G

H

I

J

L

M

N

O

IBM

Redbooks

z/OS V1.13 DFSMS Technical Update

(0.5" spine)
0.475"<->0.873"
250 <-> 459 pages

z/OS V1.13 DFSMS Technical Update

Features and functions of DFSMS V1.13

Implementation hints and tips

Code and JCL samples

Each release of IBM Data Facility Storage Management Subsystem (DFSMS) builds on the previous version. The latest release, IBM z/OS V1.13 DFSMS, provides enhancements in these areas for the z/OS platform in a system-managed storage environment:

- Storage management
- Data access
- Device support
- Program management
- Distributed data access

This IBM Redpaper Redbooks publication provides a summary of the functions and enhancements in z/OS V1.13 DFSMS. It provides information that you need to understand and evaluate the content of this DFSMS release, along with practical implementation hints and tips. This book also includes enhancements that are available by enabling PTFs that have been integrated into z/OS DFSMS V1.13.

This book was written for storage professionals and system programmers who have experience with the components of DFSMS. It provides sufficient information so that you can start prioritizing the implementation of new functions and evaluating their applicability in your DFSMS environment.

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

SG24-7961-00 ISBN 073843695X